Springer Handbook of Robotics

Siciliano
Khatib
Editors

2nd Edition

Kröger
Multimedia Editor

Springer

Springer Handbook of Robotics

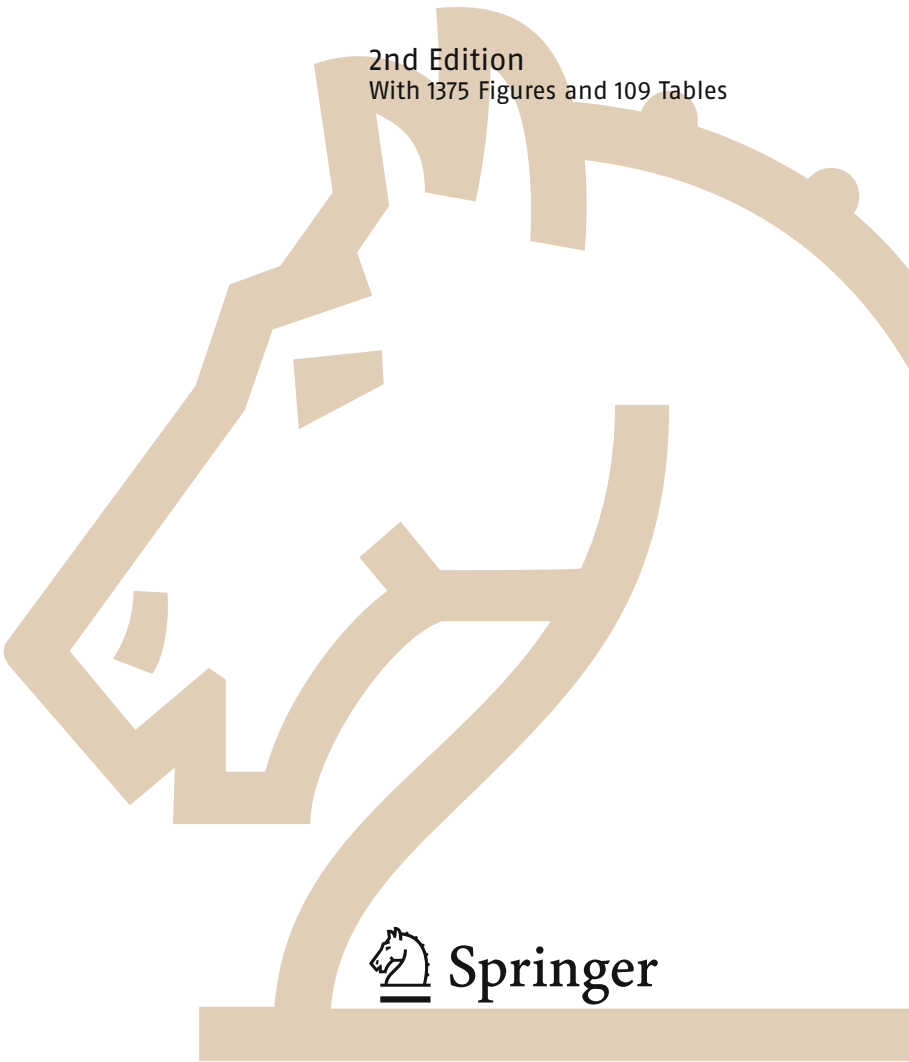
Springer Handbooks provide a concise compilation of approved key information on methods of research, general principles, and functional relationships in physical and applied sciences. The world's leading experts in the fields of physics and engineering will be assigned by one or several renowned editors to write the chapters comprising each volume. The content is selected by these experts from Springer sources (books, journals, online content) and other systematic and approved recent publications of scientific and technical information.

The volumes are designed to be useful as readable desk reference book to give a fast and comprehensive overview and easy retrieval of essential reliable key information, including tables, graphs, and bibliographies. References to extensive sources are provided.

Springer Handbook of Robotics

Bruno Siciliano, Oussama Khatib (Eds.)

2nd Edition
With 1375 Figures and 109 Tables



Springer

Editors
Bruno Siciliano
University of Naples Federico II
Department of Electrical Engineering and Information Technology
Naples, Italy
siciliano@unina.it

Oussama Khatib
Stanford University
Department of Computer Science
Artificial Intelligence Laboratory
Stanford, USA
khatib@cs.stanford.edu

ISBN: 978-3-319-32550-7 e-ISBN: 978-3-319-32552-1
DOI 10.1007/978-3-319-32552-1
Library of Congress Control Number: 2016937424

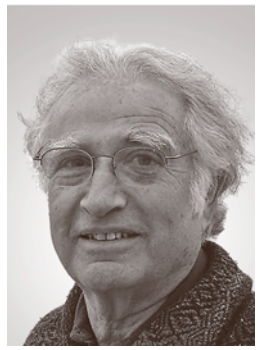
© Springer-Verlag Berlin Heidelberg 2016
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herin or for any errors or omissions that may have been made.

Production and typesetting: le-tex publishing services GmbH, Leipzig
Handbook Coordination: Dr. J. Hinterberg, Dr. W. Skolaut, Heidelberg
Typography and layout: schreiberVIS, Seeheim
Illustrations: Hippmann GbR, Schwarzenbruck
Cover design: eStudio Calamar Steinen, Barcelona
Cover production: WMXDesign GmbH, Heidelberg
Printing and binding: PHOENIX PRINT, Würzburg

Printed on acid free paper

This Springer imprint is published by Springer Nature
The registered company is Springer-Verlag GmbH Berlin Heidelberg

Foreword



Bernard Roth
Professor of
Mechanical Engineering
Stanford University

My first introduction to robotics came via a phone call in 1964. The caller was Fred Terman, the author of the world-famous *Radio Engineer's Handbook*, who was at the time Provost of Stanford University. Dr. Terman informed me that a computer science professor, John McCarthy, had just been awarded a large research grant, part of which required the development of computer-controlled manipulators. Someone had suggested to Terman that it would be prudent if the mathematically oriented McCarthy had some contact with mechanical designers. Since I was the only one on the Stanford faculty whose specialty was mechanism design, Terman decided to phone me, even though we had never met and I was a young assistant professor fresh out of graduate school with only 2 years at Stanford.

Dr. Terman's phone call led me to a close association with John McCarthy and the Stanford Artificial Intelligence Laboratory (SAIL) that he founded. Robotics became one of the pillars of my entire academic career, and I have maintained my interest in teaching and researching the subject through to the present day.

The modern history of robotic manipulation dates from the late 1940s when servoed arms were developed in connection with master–slave manipulator systems used to protect technicians handling nuclear materials. Developments in this area have continued to the present day. However, in the early 1960s there was very little academic or commercial activity in robotics. The first academic activity was the thesis of H. A. Ernst, in 1961, at MIT. He used a slave arm equipped with touch sensors, and ran it under computer control. The idea in his study was to use the information from the touch sensors to guide the arm.

This was followed by the SAIL project and a similar project started by Professor Marvin Minsky at MIT, which were the only sizeable academic ventures into robotics at that time. There were a few attempts at commercial manipulators, primarily in connection with part production in the automotive industry. In the USA there were two different manipulator designs that were being experimented with in the auto industry; one came from American Machine and Foundry (AMF) and the other from Unimation, Inc.

There were also a few mechanical devices developed as hand, leg, and arm prosthetics, and, a bit later, some exoskeletal devices to enhance human performance. In those days there were no microprocessors. So, these devices were either without computer control, or tethered to a remote so-called minicomputer, or even a mainframe computer.

Initially, some in the computer science community felt that computers were powerful enough to control any mechanical device and make it perform satisfactorily. We quickly learned that this was not to be the case. We started on a twofold track. One was to develop particular devices for SAIL, so that hardware demonstrations and proof-of-concept systems were available for the fledgling robotics community to experiment with. The other track, which was more or less moonlighted from the work at SAIL, was the development of a basic mechanical science of robotics. I had a strong feeling that a meaningful science could be developed, and that it would be best to think in terms of general concepts rather than concentrate exclusively on particular devices.

Fortuitously, it turned out that the two tracks supported each other very naturally and, most importantly, the right students were interested in doing their research in this area. Hardware developments proved to be specific examples of more general concepts, and the students were able to develop both the hardware and the theory.

Originally, we purchased an arm in order to get started quickly. A group at Rancho Los Amigos Hospital, in Los Angeles, was selling a tongue-switch-controlled motor-driven exoskeleton arm to assist patients without muscular control of their arms. We purchased one of these, and connected it to a time-shared PDP-6 computer. The device was named *Butterfingers*; it was our first experimental robot. Several films demonstrating visual feedback control, block stacking tasks, and obstacle avoidance were made with *Butterfingers* as the star performer.

The first manipulator that we designed on our own was known simply as the *Hydraulic Arm*. As its name implies, it was powered by hydraulics. The idea was to build a very fast arm. We designed special rotary actuators, and the arm worked well. It became the experimental platform for testing the first ever dynamic analysis and time-optimal control of a robotic arm. However, its use was limited since the design speeds were much faster than required due to the limitations

of the computational, planning, and sensing capabilities that were common at that time.

We made an attempt to develop a truly digital arm. This led to a snake-like structure named the *Orm* (the Norwegian word for snake.) The *Orm* had several stages, each with an array of inflatable pneumatic actuators that were either fully extended or fully contracted. The basic idea was that, even though only a finite number of positions in the workspace could be reached, these would be sufficient if there were a large number of positions. A small prototype proof-of-concept *Orm* was developed. It led to the realization that this type of arm would not really serve the SAIL community.

The first truly functional arm from our group was designed by Victor Scheinman, who was a graduate student at the time. It was the very successful *Stanford Arm*, of which over ten copies were made as research tools to be used in various university, government, and industrial laboratories. The arm had six independently driven joints; all driven by computer-controlled servoed, DC electric motors. One joint was telescoping (prismatic) and the other five were rotary (revolute).

Whereas the geometry of *Butterfingers* required an iterative solution of the inverse kinematics, the geometric configuration of the *Stanford Arm* was chosen so that the inverse kinematics could be programmed in any easy-to-use time-efficient closed form. Furthermore, the mechanical design was specifically made to be compatible with the limitations inherent in time-share computer control. Various end-effectors could be attached to act as hands. On our version, the hand was in the form of a vise-grip jaw, with two sliding fingers driven by a servoed actuator (hence, a true seventh degree of freedom). It also had a specially designed six-axis wrist force sensor. Victor Scheinman went on to develop other important robots: the first was a small humanoid arm with six revolute joints. The original design was paid for by Marvin Minsky at the MIT AI Lab. Scheinman founded Vicarm, a small company, and produced copies of this arm and the *Stanford Arm* for other labs. Vicarm later became the West Coast Division of Unimation, Inc., where Scheinman designed the *PUMA* manipulator under General Motors sponsorship through Unimation. Later, for a company called Automatix, Scheinman developed the novel *Robot World* multirobot system. After Scheinman left Unimation, his colleagues Brian Carlisle and Bruce Shimano reorganized Unimation's West Coast Division into Adept, Inc., which to this day is the largest US manufacturer of assembly robots.

Quickly, the modern trend of carefully detailed mechanical and electronic design, optimized software, and complete system integration became the norm; to this day, this combination represents the hallmark of most highly regarded robotic devices. This is the basic concept behind *mechatronic*, a word conied in Japan as a concatenation of the words mechanics and electronics. Mechatronics that relies on computation is the essence of the technology inherent in robotics as we know it today.

As robotics developed around the world, a large number of people started working on various aspects, and specific subspecialties developed. The first big division was between people working on manipulators and those working on vision systems. Early on, vision systems seemed to hold more promise than any other method for giving robots information about their environment.

The idea was to have a television camera capture pictures of objects in the environment, and then use algorithms that allowed the computer images of the pictures to be analyzed, so as to infer required information about location, orientation, and other properties of objects. The initial successes with image systems were in problems dealing with positioning blocks, solving object manipulation problems, and reading assembly drawings. It was felt that vision held potential for use in robotic systems in connection with factory automation and space exploration. This led to research into software that would allow vision systems to recognize machine parts (particularly partially occluded parts, as occurred in the so-called *bin-picking* problems) and ragged-shaped rocks.

After the ability to *see* and move objects became established, the next logical need had to do with planning a sequence of events to accomplish a complex task. This led to the development of planning as an important branch in robotics. Making fixed plans for a known fixed environment is relatively straightforward. However, in robotics, one of the challenges is to let the robot discover its environment, and to modify its actions when the environment changes unexpectedly due to errors or unplanned events. Some early landmark studies in this area were carried out using a vehicle named *Shakey*, which, starting in 1966, was developed by Charlie Rosen's group at the Stanford Research Institute (now called SRI). *Shakey* had a TV camera, a triangulating range finder, bump sensors, and was connected to DEC PDP-10 and PDP-15 computers via radio and video links.

Shakey was the first mobile robot to reason about its actions. It used programs that gave it the ability for independent perception, world modeling, and action generation. Low-level action routines took care of simple moving, turning, and route planning. Intermediate-

level actions combined the low-level ones in ways that accomplished more complex tasks. The highest level programs could make and execute plans to achieve high-level goals supplied by a user.

Vision is very useful for navigation, locating objects, and determining their relative positions and orientation. However, it is usually not sufficient for assembling parts or working with robots where there are environmental constraining forces. This led to the need to measure the forces and torques generated by the environment, on a robot, and to use these measurements to control the robot's actions. For many years, force-controlled manipulation became one of the main topics of study at SAIL, and several other labs around the world. The use of force control in industrial practice has always lagged the research developments in this area. This seems to be due to the fact that, while a high level of force control is very useful for general manipulation issues, specific problems in very restricted industrial environments can often be handled with limited, or no, force control.

In the 1970s, specialized areas of study such as walking machines, hands, automated vehicles, sensor integration, and design for hostile environments began to develop rapidly. Today there are a large number of different specialties studied under the heading of robotics. Some of these specialties are classical engineering subject areas within which results have been developed that have been particularized to the types of machines called robots. Examples here are kinematics, dynamics, controls, machine design, topology, and trajectory planning. Each of these subjects has a long history predating the study of robotics; yet each has been an area of in-depth robotics research in order to develop its special character in regard to robotic-type systems and applications. In doing this specialized development, researchers have enriched the classical subjects by increasing both their content and scope.

At the same time that the theory was being developed, there was a parallel, although somewhat separate, growth of industrial robotics. Strong commercial development occurred in Japan and Europe, and there was also continued growth in the USA. Industrial associations were formed (the Japan Robot Association was formed in March 1971, and the Robotic Industries Association (RIA) was founded in 1974 in the USA) and trade shows, together with application-oriented technical sessions, were introduced and held on a regular basis. The most important were the International Symposium on Industrial Robots, the Conference on Industrial Robot Technology (now called the International Conference on Industrial Robot Technology), and the RIA annual trade show, which is now called the International Robots and Vision Show and Conference.

The first regular series of conferences emphasizing research, rather than the industrial, aspects of robotics, was inaugurated in 1973. It was sponsored jointly by the International Center for Mechanical Sciences (CISM), based in Udine, Italy, and the International Federation for the Theory of Mechanisms and Machines (IFToMM). (Although IFToMM is still used, its meaning has been changed to the International Federation for the Promotion of Mechanism and Machine Science.) It was named the Symposium on Theory and Practice of Robots and Manipulators (RoManSy). Its trademark was an emphasis on the mechanical sciences and the active participation of researchers from Eastern and Western Europe as well as North America and Japan. It is still held biannually. On a personal note, it is at RoManSy where I first met each of the editors of this Handbook: Dr. Khatib in 1978 and Dr. Siciliano in 1984. They were both students: Bruno Siciliano had been working on his PhD for about one year, and Oussama Khatib had just completed his PhD research. In both cases, it was love at first sight!

RoManSy was quickly joined by a host of other new conferences and workshops; today there are a large number of research oriented robotics meetings that take place through the year in many countries. Currently, the largest conference is the International Conference on Robotics and Automation (ICRA), which regularly draws well over 1000 participants.

In the beginning of the 1980s, the first real textbook on robotic manipulation in the USA was written by Richard *Lou* Paul (Richard P. Paul, *Robot Manipulators: Mathematics, Programming, and Control*, The MIT Press, Cambridge, MA, 1981). It used the idea of taking classical subjects in mechanics and applying them to robotics. In addition there were several topics developed directly from his thesis research at SAIL. (In the book, many examples are based on Scheinman's Stanford Arm.) Paul's book was a landmark event in the USA; it created a pattern for several influential future textbooks and also encouraged the creation of specialized robotics courses at a host of colleges and universities.

At about this same time, new journals were created to deal primarily with research papers in the areas related to robotics. The *International Journal of Robotics Research* was founded in the spring of 1982, and three years later the *IEEE Journal of Robotics and Automation* (now the *IEEE Transactions on Robotics*) was founded.

As microprocessors became ubiquitous, the question of what is or is not a robot came more into

play. This issue has, in my mind, never been successfully resolved. I do not think a definition will ever be universally agreed upon. There are of course the science fiction creatures-from-outer-space varieties, and the robots of the theater, literature, and the movies. There are examples of imaginary robot-like beings that predate the industrial revolution, but how about more down-to-Earth robots? In my view the definition is essentially a moving target that changes its character with technological progress. For example, when it was first developed, a ship's gyro auto-compass was considered a robot. Today, it is not generally included when we list the robots in our world. It has been demoted and is now considered an automatic control device.

For many, the idea of a robot includes the concept of multifunctionality, meaning the device is designed and built with the ability to be easily adapted or reprogrammed to do different tasks. In theory this idea is valid, but in practice it turns out that most robotic devices are multifunctional in only a very limited arena. In industry it was quickly discovered that a specialized machine, in general, performs much better than a general purpose machine. Furthermore, when the volume of production is high enough, a specialized machine can cost less to manufacture than a generalized one. So, specialized robots were developed for painting, riveting, quasiplanar parts assembly, press loading, circuit board stuffing, etc. In some cases robots are used in such specialized ways that it becomes difficult to draw the line between a so-called robot and an adjustable piece of *fixed* automation. Much of this practical unfolding is contrary to the dream of the pioneers in robotics, who had hoped for the development of general purpose machines that would do *everything*, and hence sell in great enough volume to be relatively inexpensive.

My view is that the notion of a robot has to do with which activities are, at a given time, associated with people and which are associated with machines. If a machine suddenly becomes able to do what we normally associate with people, the machine can be upgraded in classification and classified as a robot. After a while, people get used to the activity being done by machines, and the devices get downgraded from *robot* to *machine*. Machines that do not have fixed bases, and those that have arm- or leg-like appendages have the advantage of being more likely called robots, but it is hard to think of a consistent set of criteria that fits all the current naming conventions.

In actuality any machines, including familiar household appliances, which have microprocessors directing their actions can be considered as robots. In addition to vacuum cleaners, there are washing machines, refrigerators, and dishwashers that could be easily marketed as robotic devices. There are of course a wide range of possibilities, including those machines that have sensory environmental feedback and decision-making capabilities. In actual practice, in devices considered to be robotic, the amount of sensory and decision making capability may vary from a great deal to none.

In recent decades the study of robotics has expanded from a discipline centered on the study of mechatronic devices to a much broader interdisciplinary subject. An example of this is the area called human-centered robotics. Here one deals with the interactions between humans and intelligent machines. This is a growing area where the study of the interactions between robots and humans has enlisted expertise from outside the classical robotics domain. Concepts such as emotions in both robots and people are being studied, and older areas such as human physiology and biology are being incorporated into the mainstream of robotics research. These activities enrich the field of robotics, as they introduce new engineering and science dimensions into the research discourse.

Originally, the nascent robotics community was focused on getting things to work. Many early devices were remarkable in that they worked at all, and little notice was taken of their limited performance. Today, we have sophisticated, reliable devices as part of the modern array of robotic systems. This progress is the result of the work of thousands of people throughout the world. A lot of this work took place in universities, government research laboratories, and companies. It is a tribute to the worldwide engineering and scientific community that it has been able to create the vast amount of information that is contained in the 64 chapters of this Handbook. Clearly these results did not arise by any central planning or by an overall orderly scheme. So the editors of this handbook were faced with the difficult task of organizing the material into a logical and coherent whole.

The editors have accomplished this by organizing the contributions into a three-layer structure. The first layer deals with the *foundations* of the subject. This layer consists of a single part of nine chapters in which the authors lay out the root subjects: kinematics, dynamics, control, mechanisms, architecture, programming, reasoning, and sensing. These are the basic technological building blocks for robotics study and development.

The second layer has four parts. The first of these deals with *robot structures*; these are the arms, legs, hands, and other parts that most robots are made up of. At first blush, the hardware of legs, arms, and hands may look quite different from each other, yet they share a common set of attributes that allows them to all be treated with the same, or closely related, aspects of the fundamentals described in the first layer.

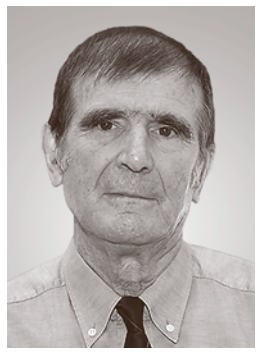
The second part of this layer deals with *sensing and perception*, which are basic abilities any truly autonomous robotic system must have. As was pointed out earlier, in practice, many so-called robotic devices have little of these abilities, but clearly the more advanced robots cannot exist without them, and the trend is very much toward incorporating such capabilities into robotic devices. The third part of this layer treats the subject areas associated with the technology of *manipulation and the interfacing of devices*. The fourth part of this layer is made up of eight chapters that treat *mobile robots and various forms of distributed robotics*.

The third layer consists of two separate parts (a total of 22 chapters) that deal with advanced applications at the forefront of today's research and development. There are two parts to this layer; one deals with *field and service robots*, and the other deals with *human-centered and lifelike robots*. To the uninitiated observer, these chapters are what advanced robotics is all about. However, it is important to realize that many of these extraordinary accomplishments would probably not exist without the previous developments introduced in the first two layers of this Handbook.

It is this intimate connection between theory and practice that has nurtured the growth of robotics and become a hallmark of modern robotics. These two complementary aspects have been a source of great personal satisfaction to those of us who have had the opportunity to both research and develop robotic devices. The contents of this Handbook admirably reflect this complementary aspect of the subject, and present a very useful bringing together of the vast accomplishments which have taken place in the last 50 years. Certainly, the contents of this Handbook will serve as a valuable tool and guide to those who will produce the even more capable and diverse next generations of robotic devices. The editors and authors have my congratulations and admiration.

Stanford, August 2007 Bernard Roth

Foreword



Georges Giralt
Emeritus Research Director
LAAS-CNRS Toulouse
(deceased)

To open this Handbook and unfold the richness of its 64 chapters, we here attempt a brief personal overview to sketch the evolution of robotics in its many aspects, concepts, trends, and central issues.

The modern story of Robotics began about half a century ago with developments in two different directions.

First, let us acknowledge the domain of mechanical arms, ranging from teleoperated tasks on radiation-contaminated products to industrial arms, with the landmark machine UNIMATE – standing for uni(versal)mate. The industrial development of products, mostly around the six-degree-of-freedom serial links paradigm and active research and development, associating mechanical engineering to the control specialism, was the main driving force here. Of particular note nowadays is the successfully pursued effort to design novel application-optimized structures, using powerful sophisticated mathematical tools. In a similar way, an important issue concerns the design and the actual building of arms and hands in the context of human-friendly robots for tomorrow's cognitive robot.

Second, and less well recognized, we should acknowledge the stream of work concerned with themes in artificial intelligence. A landmark project in this area was the mobile robot *Shakey* developed at Stanford International. This work, which aimed to bring together computer science, artificial intelligence, and applied mathematics to develop intelligent machines, remained a secondary area for quite some time. During the 1980s, building strength from many study cases encompassing a spectacular spectrum ranging from rovers for extreme environments (planet exploration, Antarctica, etc.), to service robots (hospitals, museum guides, etc.), a broad research domain arose in which machines could claim the status of intelligent robots.

Hence robotics researches could bring together these two different branches, with intelligent robots categorized in a solely computational way as bounded rationality machines, expanding on the 1980s third-generation robot definition:

> *(robot) . . . operating in the three-dimensional world as a machine endowed with the capacity to interpret and to reason about a task and about its execution, by intelligently relating perception to action.*

The field of autonomous robots, a widely recognized test-bed, has recently benefited from salient contributions in robot planning using the results of algorithmic geometry as well as of a stochastic framework approach applied both to environmental modeling and robot localization problems (SLAM, simultaneous localization and modeling), and further from the development of decisional procedures via Bayesian estimation and decision approaches.

For the last decade of the millennium, robotics largely dealt with the intelligent robot paradigm, blending together robots and machine-intelligence generic research within themes covering advanced sensing and perception, task reasoning and planning, operational and decisional autonomy, functional integration architectures, intelligent human–machine interfaces, safety, and dependability.

The second branch, for years referred to as non-manufacturing robotics, concerns a wide spectrum of research-driven real-world cases pertaining to field, service, assistive, and, later, personal robotics. Here, machine intelligence is, in its various themes, the central research direction, enabling the robot to act:

1. As a human surrogate, in particular for intervention tasks in remote and/or hostile environments
2. In close interaction with humans and operating in human environments in all applications encompassed by human-friendly robotics, also referred to as human-centered robotics
3. In tight synergy with the user, expanding from mechanical exoskeleton assistance, surgery, health care, and rehabilitation into human augmentation.

Consequently, at the turn of the millennium, robotics appears as a broad spectrum of research themes both supporting market products for well-engineered industrial workplaces, and a large number of domain-oriented application cases operating in hazardous and/or harsh environments (underwater robotics, rough-terrain rovers, health/rehabilitation care robotics, etc.) where robots exhibit meaningful levels of shared autonomy.

The evolution levels for robotics stress the role of theoretical aspects, moving from application domains to the technical and scientific area. The organization of this Handbook illustrates very well these different lev-

els. Furthermore, it rightly considers, besides a body of software systems, front-line matters on physical appearance and novel appendages, including legs, arms, and hands design in the context of human-friendly robots for tomorrow's cognitive robot.

Forefront robotics in the first decade of the current millennium is making outstanding progress, compounding the strength of two general directions:

- Short/mid-term application-oriented study cases
- Mid/long-term generic situated research.

For completeness, we should mention the large number of peripheral, robotics-inspired subjects, quite often concerning entertainment, advertising, and sophisticated toys.

The salient field of human-friendly robotics encompasses several front-line application domains where the robots operate in a human environment and in close interaction with humans (entertainment and education, public-oriented services, assistive and personal robots, etc.), which introduces the critical issue of human–robot interaction.

Right at the core of the field, emerges the forefront topic of personal robots for which three general characteristics should be emphasized:

1. They may be operated by a nonprofessional user;
2. They may be designed to share high-level decision making with the human user;
3. They may include a link to environment devices and machine appendages, remote systems, and operators; the shared decisional autonomy concept (co-autonomy) implied here unfolds into a large set of cutting-edge research issues and ethical problems.

The concept of the personal robot, expanding to robot assistant and universal companion, is a truly great challenge for robotics as a scientific and technical field, offering the mid/long-term perspective of achieving a paramount societal and economical impact. This introduces, and questions, front-line topics encompassing cognitive aspects: user-tunable human–machine intelligent interfaces, perception (scene analysis, category identification), open-ended learning (understanding the universe of action), skills acquisition, extensive robot-world data processing, decisional autonomy, and dependability (safety, reliability, communication, and operating robustness).

There is an obvious synergistic effort between the two aforementioned approaches, in spite of the necessary framework time differences. The scientific link not only brings together the problems and obtained results but also creates a synergistic exchange between the two sides and the benefits of technological progress.

Indeed, the corresponding research trends and application developments are supported by an explosive evolution of enabling technologies: computer processing power, telecommunications, networking, sensing devices, knowledge retrieval, new materials, micro- and nanotechnologies.

Today, looking to the mid- and long-term future, we are faced with very positive issues and perspectives but also having to respond to critical comments and looming dangers for machines that are in physical contact with the user and may also be capable of unwanted, unsafe behavior. Therefore, there is a clear need to include at the research level safety issues and the topic of multifaced dependability and the corresponding system constraints.

The *Handbook of Robotics* is an ambitious and timely endeavor. It summarizes a large number of problems, questions, and facets considered by 164 authors in 64 chapters. As such it not only provides an efficient display of basic topics and results obtained by researches around the world, but furthermore gives access to this variety of viewpoints and approaches to everyone. This is indeed an important tool for progress but, much more, is the central factor that will establish the two first decades of this millennium as the dawn of robotics, lifted to a scientific discipline at the core of machine intelligence.

Toulouse, December 2007 Georges Giralt

Foreword

The field of robotics was born in the middle of the last century when emerging computers were altering every field of science and engineering. Having gone through fast yet steady growth via a procession of stages from infancy, childhood, and adolescence to adulthood, robotics is now mature and is expected to enhance the quality of people's lives in society in the future.

In its infancy, the core of robotics consisted of pattern recognition, automatic control, and artificial intelligence. Taking on these new challenge, scientists and engineers in these fields gathered to investigate novel robotic sensors and actuators, planning and programming algorithms, and architectures to connect these components intelligently. In so doing, they created artifacts that could interact with humans in the real world. An integration of these early robotics studies yielded *hand–eye systems*, the test-bed of artificial intelligence research.

The playground for childhood robotics was the factory floor. Industrial robots were invented and introduced into the factory for automating spraying, spot welding, grinding, materials handling, and parts assembly. Machines with sensors and memories made the factory floor smarter, and its operations more flexible, reliable, and precise. Such robotic automation freed humans from heavy and tedious labor. The automobile, electric appliance, and semiconductor industries rapidly retooled their manufacturing lines into robot-integrated systems. In the late 1970s, the word *mechatronics*, originally coined by the Japanese, defined a new concept of machinery, one in which electronics was fused with mechanical systems, making a wide range of industrial products simpler, more functional, programmable, and intelligent. Robotics and mechatronics exerted an evolutionary impact on the design and operation of manufacturing processes as well as on manufactured products.

As robotics entered its adolescence, researchers were ambitious to explore new horizons. Kinematics, dynamics, and control system theory were refined and applied to real complex robot mechanisms. To plan and carry out real tasks, robots had to be made cognizant of their surroundings. Vision, the primary channel for external sensing, was exploited as the most general, effective, and efficient means for robots to understand their external situation. Advanced algorithms and powerful devices were developed to improve the speed and robustness of robot vision systems. Tactile and force sensing systems also needed to be developed for robots to manipulate objects. Studies on modeling, planning, knowledge, reasoning, and memorization expanded their intelligent properties. Robotics became defined as the study of intelligent connection of sensing to actuation. This definition covered all aspects of robotics: three scientific cores and one synthetic approach to integrate them. Indeed, system integration became a key aspect of robotic engineering as it allows the creation of lifelike machines. The fun of creating such robots attracted many students to the robotics field.

Hirochika Inoue
Professor Emeritus
The University of Tokyo

In advancing robotics further, scientific interest was directed at understanding humans. Comparative studies of humans and robots led to new approaches in scientific modeling of human functions. Cognitive robotics, lifelike behavior, biologically inspired robots, and a psychophysiological approach to robotic machines culminated in expanding the horizons of robotic potential. Generally speaking, an immature field is sparse in scientific understanding. Robotics in the 1980s and 1990s was in such a youthful stage, attracting a great many inquisitive researchers to this new frontier. Their continuous explorations into new realms form the rich scientific contents of this comprehensive volume.

Further challenges, along with expertise acquired on the cutting edge of robotics, opened the way to real-world applications for mature robotics. The early-stage playground gave way to a workshop for industrial robotics. Medical robotics, robot surgery, and in vivo imaging save patients from pain while providing doctors with powerful tools for conducting operations. New robots in such areas as rehabilitation, health care, and welfare are expected to improve quality of life in an aging society. It is the destiny of robots to go everywhere, in the air, under water, and into space. They are expected to work hand in hand with humans in such areas as agriculture, forestry, mining, construction, and hazardous environments and rescue operations, and to find utility both in domestic work and in providing services in shops, stores, restaurants, and hospitals. In a myriad of ways, robotic devices are expected to support our daily lives. At this point, however, robot applications are largely limited to structured environments, where they are separated from humans for safety sake.

In the next stage, their environment will be expanded to an unstructured world, one in which humans, as service takers, will always live and work beside robots. Improved sensing, more intelligence, enhanced safety, and better human understanding will be needed to prepare robots to function in such an environment. Not only technical but also social matters must be considered in finding solutions to issues impeding this progress.

Since my initial research to make a robot turn a crank, four decades have passed. I feel both lucky and happy to have witnessed the growth of robotics from its early beginnings. To give birth to robotics, fundamental technologies were imported from other disciplines. Neither textbooks nor handbooks were available. To reach the present stage, a great many scientists and engineers have challenged new frontiers; advancing robotics, they have enriched this body of knowledge from a variety of perspectives. The fruits of their endeavors are compiled in this *Handbook of Robotics*. More than 100 of the world's leading experts have collaborated in producing this publication. Now, people who wish to commit themselves to robotics research can find a firm foundation to build upon. This Handbook is sure to be used to further advance robotics science, reinforce engineering education, and systematically compile knowledge that will innovate both society and industry.

The roles of humans and robots in an aging society pose an important issue for scientists and engineers to consider. Can robotics contribute to securing peace, prosperity, and a greater quality of life? This is still an open question. However, recent advances in personal robots, robotic home appliances, and humanoids suggest a paradigm shift from the industrial to the service sector. To realize this, robotics must be addressed from such viewpoints as the working infrastructure within society, psychophysiology, law, economy, insurance, ethics, art, design, drama, and sports science. Future robotics should be studied as a subject that envelops both humanity and technology. This Handbook offers a selected technical foundation upon which to advance such newly emerging fields of robotics. I look forward to continuing progress adding page after page of robot-based prosperity to future society.

Tokyo, September 2007 Hirochika Inoue

Foreword

Rodney Brooks
Panasonic Professor of Robotics
Massachusetts Institute of Technology

Robots have fascinated people for thousands of years. Those automatons that were built before the 20th century did not connect sensing to action but rather operated through human agency or as repetitive machines. However, by the 1920s electronics had gotten to the stage that the first true robots that sensed the world and acted in it appropriately could be built. By 1950 we started to see descriptions of real robots appearing in popular magazines. By the 1960s industrial robots came onto the scene. Commercial pressures made them less and less responsive to their environments but faster and faster in what they did in their carefully engineered world. Then in the mid 1970s in France, Japan, and the USA we started to see robots rising again in a handful of research laboratories, and now we have arrived at a world-wide frenzy in research and the beginnings of large-scale deployment of intelligent robots throughout our world. This Handbook brings together the current state of robotics research in one place. It ranges from the mechanism of robots through sensing and perceptual processing, intelligence, action, and many application areas.

I have been more than fortunate to have lived with this revolution in robotics research over the last 30 years. As a teenager in Australia I built robots inspired by the tortoises of Walter described in the *Scientific American* in 1949 and 1950. When I arrived in Silicon Valley in 1977, just as the revolution in the personalization of computation was really coming into being, I instead turned to the much more obscure world of robots. In 1979 I was able to assist Hans Moravec at the Stanford Artificial Intelligence Lab (SAIL) as he coaxed his robot *The Cart* to navigate 20 m in 6 hours. Just 26 years later, in 2005, at the same laboratory, SAIL, Sebastian Thrun and his team coaxed their robot to autonomously drive 200 000 m in 6 hours: four orders of magnitude improvement in a mere 26 years, which is slightly better than a doubling every 2 years. However, robots have not just improved in speed, they have also increased in number. When I arrived at SAIL in 1977 we knew of three mobile robots operating in the world. Recently a company that I founded manufactured its 3 000 000th mobile robot, and the pace is increasing. Other aspects of robots have had similarly spectacular advances, although it is harder to provide such crisp numeric characterizations. In recent years we have gone from robots being too unaware of their surroundings that it was unsafe for people to share their workspace to robots that people can work with in close contact, and from robots that were totally unaware of people to robots that pick up on natural social cues from facial expressions to prosody in people's voices. Recently robotics has crossed the divide between flesh and machines so that now we are seeing neurorobotics ranging from prosthetic robotic extensions to rehabilitative robots for the disabled. And very recently robotics has become a respected contributor to research in cognitive science and neuroscience.

The research results chronicled in this volume give the key ideas that have enabled these spectacular advances. The editors, the part editors, and all the contributors have done a stellar job in bring this knowledge together in one place. Their efforts have produced a work that will provide a basis for much further research and development. Thank you, and congratulations to all who have labored on this pivotal book.

Some of the future robotics research will be incremental in nature, taking the state of the art and improving upon it. Other parts of future research will be more revolutionary, based on ideas that are antithetical to some of the ideas and current state of the art presented in this book.

As you study this volume and look for places to contribute to research through your own talents and hard work I want to alert you to capabilities or aspirations that I believe will make robots even more useful, more productive, and more accepted. I describe these capabilities in terms of the age at which a child has equivalent capabilities:

- The object-recognition capabilities of a 2-year-old child
- The language capabilities of a 4-year-old child
- The manual dexterity of a 6-year-old child
- The social understanding of an 8-year-old child.

Each of these is a very difficult goal. However even small amounts of progress towards any one of these goals will have immediate applications to robots out in the world. Good reading and best wishes as you contribute further to robotkind.

Cambridge, October 2007 Rodney Brooks

Preface to the Second Edition

The Springer Handbook of Robotics was a challenging six-year endeavour from 2002 to 2008. It mobilized a large number of active scientists and researchers to produce this unique comprehensive reference source combining basic and advanced developments. The handbook has been very successful and extremely well received in our community. New researchers have been attracted to robotics which in turn have contributed to further progress in this trans-disciplinary field.

The handbook soon established itself as a landmark in robotics publishing and beyond. It has been the bestseller of all Springer engineering books during the last seven years, the number one in chapter downloads (nearly forty thousand a year), and the fourth most downloaded over all Springer books in 2011. In February 2009, the handbook was recognized as the Winner of the American Association of Publishers (AAP) PROSE Award for Excellence in Physical Sciences & Mathematics as well as the Award for Engineering & Technology.

The rapid growth of our field as well as the birth of new research areas motivated us in 2011 to start pursuing a second edition with the intent to provide not only an update but also an expansion of the handbook's contents. Our editorial board (with David Orin, Frank Park, Henrik Christensen, Makoto Kaneko, Raja Chatila, Alex Zelinsky, and Daniela Rus) has been enthusiastically engaged during the last four years to coordinate the contributions of the authors to the seven parts of the handbook in its three-layer structure. The contents have been restructured to achieve four main objectives: the enlargement of foundational topics for robotics, the enlightenment of design of various types of robotic systems, the extension of the treatment on robots moving in the environment, and the enrichment of advanced robotics applications. Most previous chapters have been revised, fifteen new chapters have been introduced on emerging topics, and a new generation of authors have joined the handbook's team. The contents were finalized by the spring of 2015 after extensive review and feedback, and the project was completed by the fall of 2015 – generating, by that time, a record of over 12 000 additional emails in our folders to the 10 000 of the first edition. The result is an impressive collection of 80 chapters over the 7 parts, contributed by 229 authors, with more than 2300 pages, 1375 illustrations and 9411 references.

One of the major additions of the second edition of the handbook is the inclusion of multimedia material. An editorial team has been established under the leadership of Torsten Kröger and the contributions of Gianluca Antonelli, Dongjun Lee, Dezhen Song and Stefano Stramigioli. With the commitment of such a group of energetic young scholars, the multimedia project has been pursued in parallel to the handbook project. The multimedia editorial team has selected for each chapter video contributions, from those suggested by the authors, based on their quality and relevance to the chapter's contents. In addition, the handbook editors have produced tutorial videos that can be accessed directly from each part of the handbook. An openly accessible multimedia website, http://handbookofrobotics.org, has been established to host these videos with the sponsorship of IEEE Robotics and Automation Society and Google. The website has been conceived as a live dissemination project bringing the latest robotics contributions to the world community.

We are deeply grateful for the continuous commitment of our handbook extended team, particularly the newcomers to the project. We would like to express our gratitude and appreciation to Judith Hinterberg, Werner Skolaut and Thomas Ditzinger from Springer for their strong support, as well as to Anne Strohbach and the le-tex staff for their highly professional typesetting work in the production.

Eight years after the first appearance of the handbook, the second edition comes to light. Beyond its tutorial value for our community, it is our conviction that the handbook will continue to serve as a useful source to attract new researchers to robotics and inspire decades of vibrant progress in this fascinating field. The cooperative spirit inspiring our team since the inception of the first edition is amusingly illustrated in the video *The Handbook – A Short History* (VIDEO 844). The completion of the second edition has been inspired by that same spirit and the gradient has been kept :-) Our fellows in the robotics community are reminded now to … keep the Hessian ;-)

January 2016

Bruno Siciliano — Naples

Oussama Khatib — Stanford

Preface to the Multimedia Extension

Scientific and technical advancements in the domain of robotics have accelerated significantly over the past decade. Since the inception of the Second Edition of the Springer Handbook of Robotics in 2011, the Editors Bruno Siciliano and Oussama Khatib decided to add multimedia content and appointed an editorial team: Gianluca Antonelli, Dongjun Lee, Dezhen Song, Stefano Stramigioli, and myself as the Multimedia Editor.

Over the five years of the project, everyone on the team worked with all of the 229 authors, the Part Editors, and the Editors. Besides communicating with all 80 Authors' teams and reviewing, selecting, and improving all video contributions, we also scanned all the videos published at robotics conferences organized by the IEEE Robotics and Automation Society since 1991. A total of more than 5500 e-mails were sent back and forth to coordinate the project and to ensure the quality of the content. We implemented a video management system that allows authors to upload videos, editors to review videos, and readers to access videos. Videos were selected with the goal of helping convey content to all readers of the Second Edition. They may be relevant from a technical, scientific, educational, or historical perspective. All chapter and part videos are publicly accessible and can be found at

http://handbookofrobotics.org

In addition to the videos referenced in the chapters, each of the seven parts is accompanied by a part video giving an overview of each part. The storyboards of these videos were created by the Part Editors and then professionally produced.

The video content provided in the Multimedia Extension makes understanding the written content easier and was designed to be a comprehensive addition to the Handbook. Concepts, methods, experiments, and applications described in the book were animated, visually illustrated, or paired with sound and narration – giving readers a further dimension to comprehend the written content of the book.

Coordinating the work with more than 200 contributors cannot just be done by a small team, and we are deeply grateful for the support of many people and organizations. Judith Hinterberg and Thomas Ditzinger from the Springer Team in Heidelberg helped us tremendously with professional support during the entire production phase. The app for smartphones and tablets was implemented by Rob Baldwin from Studio Orb and allows readers easy access to multimedia content. The IEEE Robotics and Automation Society granted permissions to use all videos that have been published in the proceedings of conferences sponsored by the society. Google and X supported us by donating funds for the implementation of the website backend.

Following the Editors' inspiration, let us keep working and communicating as one community – and let us keep the Hessian all together . . . !

March 2016
Torsten Kröger Mountain View

Accessing Multimedia Contents

Multimedia contents are an integral part of the Second Edition of the Springer Handbook of Robotics. 69 chapters contain video icons like this one:

VIDEO 843

Each icon indicates a video ID that can be used to access individual videos in various simple and intuitive ways.

Using the Multimedia App

We recommend using the multimedia app for smartphone and tablet PCs. You can install the app on *iOS* and *Android* devices using the QR code below. The app allows you to simply scan the pages of the book and automatically play all videos on your device while reading the book.

Multimedia Contents

Using the Website: http://handbookofrobotics.org

All chapter videos and part videos can be accessed directly from the website of the multimedia extension. Just enter a video ID in the search field in the top right corner of the website. You may also use the website to browse through chapter and part videos.

Using PDF Files

If you read an electronic copy of the Handbook, each video icon contains a hyper link. Just click on the link to watch the corresponding video.

Using QR Codes

Each chapter starts with a QR code that contains a link to all videos of the chapter. Part videos can be accessed through the QR code at the beginning of each part.

About the Editors

Bruno Siciliano received his Doctorate degree in Electronic Engineering from the University of Naples, Italy, in 1987. He is Professor of Control and Robotics at University of Naples Federico II. His research focuses on methodologies and technologies in industrial and service robotics including force and visual control, cooperative robots, human-robot interaction, and aerial manipulation. He has co-authored 6 books and over 300 journal papers, conference papers and book chapters. He has delivered over 20 keynote presentations and over 100 colloquia and seminars at institutions around the world. He is a Fellow of IEEE, ASME and IFAC. He is Co-Editor of the Springer Tracts in Advanced Robotics (STAR) series and the Springer Handbook of Robotics, which received the PROSE Award for Excellence in Physical Sciences & Mathematics and was also the winner in the category Engineering & Technology. He has served on the Editorial Boards of prestigious journals, as well as Chair or Co-Chair for numerous international conferences. Professor Siciliano is the Past-President of the IEEE Robotics and Automation Society (RAS). He has been the recipient of several awards, including the IEEE RAS George Saridis Leadership Award in Robotics and Automation and the IEEE RAS Distinguished Service Award.

Oussama Khatib received his Doctorate degree in Electrical Engineering from Sup'Aero, Toulouse, France, in 1980. He is Professor of Computer Science at Stanford University. His research focuses on methodologies and technologies in human-centered robotics including humanoid control architectures, human motion synthesis, interactive dynamic simulation, haptics, and human-friendly robot design. He has co-authored over 300 journal papers, conference papers and book chapters. He has delivered over 100 keynote presentations and several hundreds of colloquia and seminars at institutions around the world. He is a Fellow of IEEE. He is Co-Editor of the Springer Tracts in Advanced Robotics (STAR) series and the Springer Handbook of Robotics, which received the PROSE Award for Excellence in Physical Sciences & Mathematics and was also the winner in the category Engineering & Technology. He has served on the Editorial Boards of prestigious journals, as well as Chair or Co-Chair for numerous international conferences. Professor Khatib is the President of the International Foundation of Robotics Research. He has been the recipient of several awards, including the IEEE RAS Pioneer Award in Robotics and Automation, the IEEE RAS George Saridis Leadership Award in Robotics and Automation, the IEEE RAS Distinguished Service Award, and the Japan Robot Association (JARA) Award in Research and Development.

About the Part Editors

David E. Orin

The Ohio State University
Department of Electrical and Computer Engineering
Columbus, USA
orin.1@osu.edu

Part A

David E. Orin received his PhD degree in Electrical Engineering from The Ohio State University in 1976. From 1976–1980, he taught at Case Western Reserve University. Since 1981, he has been at The Ohio State University, where he is currently a Professor Emeritus of Electrical and Computer Engineering. He was a sabbatical faculty at Sandia National Laboratories in 1996. His research interests center on humanoid and quadruped running and dynamic walking, dynamic maneuvers in legged locomotion, and robot dynamics. He has over 150 publications. His commitment to education has resulted in his receiving the Eta Kappa Nu Professor of the Year Award in the EE Department at OSU (1998–1999), and the MacQuigg Award for Outstanding Teaching in the College of Engineering (2003). He is a Fellow of the IEEE (1993) and was the President of the IEEE Robotics and Automation Society 2012–2013.

Frank C. Park

Seoul National University
Mechanical and Aerospace Engineering
Seoul, Korea
fcp@snu.ac.kr

Part B

Frank Chongwoo Park received his BS in Electrical Engineering from MIT in 1985, and PhD in Applied Mathematics from Harvard University in 1991. From 1991 to 1995 he was Assistant Professor of Mechanical and Aerospace Engineering at the University of California, Irvine. Since 1995 he has been Professor of Mechanical and Aerospace Engineering at Seoul National University, Korea. His research interests are in robot mechanics, planning and control, vision and image processing. In 2007–2008 he was an IEEE Robotics and Automation Society (RAS) Distinguished Lecturer. He has served on the editorial boards of the Springer Handbook of Robotics and Springer Tracts in Advanced Robotics (STAR), Robotica, and the ASME Journal of Mechanisms and Robotics. He is a fellow of the IEEE, and EiC of the IEEE Transactions on Robotics.

Henrik I. Christensen

Georgia Institute of Technology
Robotics and Intelligent Machines
Atlanta, USA
hic@cc.gatech.edu

Part C

Henrik I. Christensen is the KUKA Chair of Robotics and Director of Robotics at Georgia Institute of Technology, Atlanta, GA. He received MS and PhD degrees from Aalborg University in 1987 and 1990, respectively. He has held positions in Denmark, Sweden, and USA. He has published more than 300 contributions across vision, robotics, and AI. Results have been commercialized through major companies and 6 spin-off companies. He served as the founding coordinator of the European Robotics Research Network (EURON) and the US Robotics Virtual Organization. He was the editor of the US National Robotics Roadmap. He is a Fellow of the International Foundation of Robotics Research (IFRR), American Association of Advancement of Science (AAAS), and Institution of Electrical and Electronic Engineers (IEEE). He is an Editorial Board member of the Springer STAR series, and serves on the editorial board of several leading robotics journals.

Makoto Kaneko

Osaka University
Department of Mechanical Engineering
Suita, Japan
mk@mech.eng.osaka-u.ac.jp

Part D

Makoto Kaneko received the MS and PhD degrees in Mechanical Engineering from Tokyo University in 1978 and 1981, respectively. From 1981 to 1990 he was Researcher at the Mechanical Engineering Laboratory, from 1990 to 1993 an Associate Professor at Kyushu Institute of Technology, from 1993 to 2006 Professor at Hiroshima University, and in 2006 became a Professor at Osaka University. His research interests include tactile-based active sensing, grasping strategy, hyper human technology and its application to medical diagnosis, and his work has received 17 awards. He is an Editorial Board member of the STAR series and has served as chair or co-chair for several international conferences. He is an IEEE Fellow. He has served the IEEE Robotics and Automation Society as a Vice-President for Member Activities and as a Technical Editor of the IEEE Transactions on Robotics and Automation.

Raja Chatila

University Pierre et Marie Curie
Institute of Intelligent Systems and Robotics
Paris, France
raja.chatila@laas.fr

Part E

Raja Chatila, IEEE Fellow, is Director of Research at the French National Center of Scientific Research (CNRS), and Director of the Institute of Intelligent Systems and Robotics at Pierre and Marie Curie University in Paris. He is also Director of the Laboratory of Excellence SMART on human–machine interaction. He was Director of LAAS-CNRS, Toulouse France 2007–2010. His research covers aspects of robotics in navigation and SLAM, motion planning and control, cognitive and control architectures, human–robot interaction, and robot learning. He is author of over 140 publications. Current projects are Roboergosum on robot self-awareness and Spencer on human–robot interaction in populated environments. He is President of the IEEE Robotics and Automation Society for the term 2014–2015 and is member of the Ethics Committee on Research in Information Science and Technology of the Allistene. He received the IEEE Pioneer Award in Robotics and Automation and a Honorary Doctor of Örebro University (Sweden).

Alex Zelinsky

Department of Defence
DST Group Headquarters
Canberra, Australia
alexzelinsky@yahoo.com

Part F

Dr. Alex Zelinsky is a research leader in mobile robotics, computer vision and human–machine interaction. Dr. Zelinsky is Australia's Chief Defence Scientist and Chief Executive of the Defence Science and Technology Organisation (DSTO). Before joining DSTO in March 2012, Dr. Zelinsky was Group Executive, Information and Communication Sciences and Technology at CSIRO. Prior to joining CSIRO in July 2004, Dr. Zelinsky was CEO of Seeing Machines, a company dedicated to the commercialization of computer vision systems. The technology commercialized by Seeing Machines was developed at the Australian National University, where Dr. Zelinsky was Professor from 1996 to 2000. In 1997 he founded the Field and Services Robotics conference series. Dr. Zelinsky's contributions have been recognized by the Australian Engineering Excellence Awards (1999, 2002), Technology Pioneer at the World Economic Forum (2002-2004) and IEEE Robotics & Automation Society Inaba Technical Award for Innovation Leading to Production (2010), Pearcey Medal (2013). Dr. Zelinsky is an elected Fellow of the Australian Academy of Technological Sciences and Engineering (2002) and an elected Fellow of the IEEE (2008) and an Honorary Fellow of Institution of Engineers Australia (2013).

Daniela Rus

Massachusetts Institute of Technology
CSAIL Center for Robotics
Cambridge, USA
rus@csail.mit.edu

Part G

Daniela Rus is the Andrew and Erna Viterbi Professor of Electrical Engineering and Computer Science and Director of the Computer Science and Artificial Intelligence Laboratory (CSAIL) at MIT. Rus' research interests are in robotics, mobile computing, and data science. Rus is a Class of 2002 MacArthur Fellow, a fellow of ACM, AAAI and IEEE, and a member of the NAE. She earned her PhD in Computer Science from Cornell University. Prior to joining MIT, Rus was a Professor in the Computer Science Department at Dartmouth College.

About the Multimedia Editors

Torsten Kröger

Google Inc.
Mountain View, CA 94043, USA
t@kroe.org

Torsten Kroeger is a roboticist at Google and a visiting researcher at Stanford University. He received his Master's degree in Electrical Engineering from TU Braunschweig, Germany, in 2002. From 2003 to 2009, he was a research assistant at Robotics Research Institute at TU Braunschweig, from which he received his Doctorate degree in Computer Science in 2009 (summa cum laude). In 2010, he joined the Stanford AI Laboratory, where he worked on instantaneous trajectory generation, autonomous hybrid switched-control of robots, and distributed real-time hard- and software systems. He is the founder of Reflexxes GmbH, a spin-off of TU Braunschweig working on the development of deterministic real-time motion generation algorithms. In 2014, Reflexxes has joined Google. Torsten is an editor or an associate editor of multiple IEEE conference proceedings, books, and book series. He received the IEEE RAS Early Career Award, the Heinrich Büssing Award, the GFFT Award, two fellowships of the German Research Association, and he was a finalist of the IEEE/IFR IERA Award and the euRobotics TechTransfer Award.

Gianluca Antonelli

University of Cassino and Southern Lazio
Department of Electrical and Information Engineering
Cassino, Italy
antonelli@unicas.it

Gianluca Antonelli is an Associate Professor at the University of Cassino and Southern Lazio. His research interests include marine and industrial robotics as well as multiagent systems identification. He has published 32 international journal papers and more than 90 conference papers and is author of the book *Underwater Robots*. He is chair of the IEEE RAS Chapter of the IEEE-Italy section.

Dongjun Lee

Seoul National University
Department of Mechanical and Aerospace Engineering
Seoul, Korea
djlee@snu.ac.kr

Dr. Dongjun Lee currently directs the Interactive and Networked Robotics Lab (INRoL) at Seoul National University (SNU). He received the PhD degree from the University of Minnesota, and MS and BS degrees from KAIST. His main research interests are mechanics and control of robotic and mechatronic systems with emphasis on teleoperation, haptics, aerial robotics, and multi-robot systems.

Dezhen Song

Texas A&M University
Department of Computer Science
College Station, USA
dzsong@cs.tamu.edu

Dezhen Song received the PhD degree in Engineering from the University of California, Berkeley, in 2004. Currently, he is an Associate Professor with Texas A&M University, College Station. His research area is networked robotics, computer vision, optimization, and stochastic modeling. Dr. Song received the Kayamori Best Paper Award of the 2005 IEEE ICRA (with J. Yi and S. Ding) and the NSF Faculty Early Career Development (CAREER) Award in 2007.

Stefano Stramigioli

University of Twente
Control Laboratory, Faculty of Electrical Engineering, Mathematics & Computer Science
7500 AE, Enschede, The Netherlands
s.stramigioli@utwente.nl

Stefano Stramigioli received the MSc in 1992 and the PhD in 1998. Between the two degrees he worked as a researcher at the University of Twente. Since 1998 he has been faculty member and he is currently full Professor of Advanced Robotics and chair holder of the Robotics and Mechatronics group at the University of Twente. He is an officer and Senior Member of IEEE. He has about 200 publications including 4 books, book chapters, journal and conference contributions. He is currently the Vice President for Member Activities of the IEEE Robotics and Automation Society. He has been an AdCom member for IEEE RAS. Stefano is a member of the ESA Topical Team on Dynamics of Prehension in Micro-gravity and its application to Robotics and Prosthetics.

List of Authors

Markus W. Achtelik
ETH Zurich
Autonomous Systems Laboratory
Leonhardstrasse 21
8092 Zurich, Switzerland
markus@achtelik.net

Alin Albu-Schäffer
DLR Institute of Robotics and Mechatronics
Münchner Strasse 20
82230 Wessling, Germany
alin.albu-schaeffer@dlr.de

Kostas Alexis
ETH Zurich
Institute of Robotics and Intelligent Systems
Tannenstrasse 3
8092 Zurich, Switzerland
konstantinos.alexis@mavt.ethz.ch

Jorge Angeles
McGill University
Department of Mechanical Engineering and
Centre for Intelligent Machines
817 Sherbrooke Street West
Montreal, H3A 2K6, Canada
angeles@cim.mcgill.ca

Gianluca Antonelli
University of Cassino and Southern Lazio
Department of Electrical and Information
Engineering
Via G. Di Biasio 43
03043 Cassino, Italy
antonelli@unicas.it

Fumihito Arai
Nagoya University
Department of Micro-Nano Systems Engineering
Furo-cho, Chikusa-ku
464-8603 Nagoya, Japan
arai@mech.nagoya-u.ac.jp

Michael A. Arbib
University of Southern California
Computer Science, Neuroscience and ABLE Project
Los Angeles, CA 90089-2520, USA
arbib@usc.edu

J. Andrew Bagnell
Carnegie Mellon University
Robotics Institute
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
dbagnell@ri.cmu.edu

Randal W. Beard
Brigham Young University
Electrical and Computer Engineering
459 Clyde Building
Provo, UT 84602, USA
beard@byu.edu

Michael Beetz
University Bremen
Institute for Artificial Intelligence
Am Fallturm 1
28359 Bremen, Germany
ai-office@cs.uni-bremen.de

George Bekey
University of Southern California
Department of Computer Science
612 South Vis Belmonte Court
Arroyo Grande, CA 93420, USA
bekey@usc.edu

Maren Bennewitz
University of Bonn
Institute for Computer Science VI
Friedrich-Ebert-Allee 144
53113 Bonn, Germany
maren@cs.uni-bonn.de

Massimo Bergamasco
Sant'Anna School of Advanced Studies
Perceptual Robotics Laboratory
Via Alamanni 13
56010 Pisa, Italy
m.bergamasco@sssup.it

Marcel Bergerman
Carnegie Mellon University
Robotics Institute
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
marcel@cmu.edu

Antonio Bicchi
University of Pisa
Interdepartmental Research Center "E. Piaggio"
Largo Lucio Lazzarino 1
56122 Pisa, Italy
bicchi@ing.unipi.it

Aude G. Billard
Swiss Federal Institute of Technology (EPFL)
School of Engineering
EPFL-STI-I2S-LASA, Station 9
1015 Lausanne, Switzerland
aude.billard@epfl.ch

John Billingsley
University of Southern Queensland
Faculty of Engineering and Surveying
West Street
Toowoomba, QLD 4350, Australia
john.billingsley@usq.edu.au

Rainer Bischoff
KUKA Roboter GmbH
Technology Development
Zugspitzstrasse 140
86165 Augsburg, Germany
rainer.bischoff@kuka.com

Thomas Bock
Technical University Munich
Department of Architecture
Arcisstrasse 21
80333 Munich, Germany
thomas.bock@br2.ar.tum.de

Adrian Bonchis
CSIRO
Department of Autonomous Systems
1 Technology Court
Pullenvale, QLD 4069, Australia
adrian.bonchis@csiro.au

Josh Bongard
University of Vermont
Department of Computer Science
205 Farrell Hall
Burlington, VT 05405, USA
josh.bongard@uvm.edu

Wayne J. Book
Georgia Institute of Technology
G. W. Woodruff School of Mechanical Engineering
771 Ferst Drive
Atlanta, GA 30332-0405, USA
wayne.book@me.gatech.edu

Cynthia Breazeal
MIT Media Lab
Personal Robots Group
20 Ames Street
Cambridge, MA 02139, USA
cynthiab@media.mit.edu

Oliver Brock
Technical University Berlin
Robotics and Biology Laboratory
Marchstrasse 23
10587 Berlin, Germany
oliver.brock@tu-berlin.de

Alberto Broggi
University of Parma
Department of Information Technology
Viale delle Scienze 181A
43100 Parma, Italy
broggi@ce.unipr.it

Davide Brugali
University of Bergamo
Department of Computer Science and Mathematics
Viale Marconi 5
24044 Dalmine, Italy
brugali@unibg.it

Heinrich Bülthoff
Max-Planck-Institute for Biological Cybernetics
Human Perception, Cognition and Action
Spemannstrasse 38
72076 Tübingen, Germany
heinrich.buelthoff@tuebingen.mpg.de

Joel W. Burdick
California Institute of Technology
Department of Mechanical Engineering
1200 East California Boulevard
Pasadena, CA 9112, USA
jwb@robotics.caltech.edu

Wolfram Burgard
University of Freiburg
Institute of Computer Science
Georges-Koehler-Allee 79
79110 Freiburg, Germany
burgard@informatik.uni-freiburg.de

Fabrizio Caccavale
University of Basilicata
School of Engineering
Via dell'Ateneo Lucano 10
85100 Potenza, Italy
fabrizio.caccavale@unibas.it

Sylvain Calinon
Idiap Research Institute
Rue Marconi 19
1920 Martigny, Switzerland
sylvain.calinon@idiap.ch

Raja Chatila
University Pierre et Marie Curie
Institute of Intelligent Systems and Robotics
4 Place Jussieu
75005 Paris, France
raja.chatila@isir.upmc.fr

François Chaumette
Inria/Irisa
Lagadic Group
35042 Rennes, France
francois.chaumette@inria.fr

I-Ming Chen
Nanyang Technological University
School of Mechanical and Aerospace Engineering
50 Nanyang Avenue
639798 Singapore, Singapore
michen@ntu.edu.sg

Stefano Chiaverini
University of Cassino and Southern Lazio
Department of Electrical and Information Engineering
Via G. Di Biasio 43
03043 Cassino, Italy
chiaverini@unicas.it

Gregory S. Chirikjian
John Hopkins University
Department of Mechanical Engineering
3400 North Charles Street
Baltimore, MD 21218-2682, USA
gchirik1@jhu.edu

Kyu-Jin Cho
Seoul National University
Biorobotics Laboratory
1 Gwanak-ro, Gwanak-gu
Seoul, 151-744, Korea
kjcho@sun.ac.kr

Hyun-Taek Choi
Korea Research Institute of Ships & Ocean Engineering (KRISO)
Ocean System Engineering Research Division
32 Yuseong-daero 1312 Beon-gil, Yuseong-gu
Daejeon, 305-343, Korea
htchoiphd@gmail.com

Nak-Young Chong
Japan Advanced Institute of Science and Technology
Center for Intelligent Robotics
1-1 Asahidai, Nomi
923-1292 Ishikawa, Japan
nakyoung@jaist.ac.jp

Howie Choset
Carnegie Mellon University
Robotics Institute
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
choset@cs.cmu.edu

Henrik I. Christensen
Georgia Institute of Technology
Robotics and Intelligent Machines
801 Atlantic Drive NW
Atlanta, GA 30332-0280, USA
hic@cc.gatech.edu

Wendell H. Chun
University of Denver
Department of Electrical and Computer Engineering
2135 East Wesley Avenue
Denver, CO 80208, USA
wendell.chun@du.edu

Wan Kyun Chung
POSTECH
Robotics Laboratory
KIRO 410, San 31, Hyojadong
Pohang, 790-784, Korea
wkchung@postech.ac.kr

Woojin Chung
Korea University
Department of Mechanical Engineering
Anam-dong, Sungbuk-ku
Seoul, 136-701, Korea
smartrobot@korea.ac.kr

Peter Corke
Queensland University of Technology
Department of Electrical Engineering and Computer Science
2 George Street
Brisbane, QLD 4001, Australia
peter.corke@qut.edu.au

Elizabeth Croft
University of British Columbia
Department of Mechanical Engineering
6250 Applied Science Lanve
Vancouver, BC V6P 1K4, Canada
elizabeth.croft@ubc.ca

Mark R. Cutkosky
Stanford University
Department of Mechanical Engineering
450 Serra Mall
Stanford, CA 94305, USA
cutkosky@stanford.edu

Kostas Daniilidis
University of Pennsylvania
Department of Computer and Information Science
3330 Walnut Street
Philadelphia, PA 19104, USA
kostas@upenn.edu

Paolo Dario
Sant'Anna School of Advanced Studies
The BioRobotics Institute
Piazza Martiri della Libertà 34
56127 Pisa, Italy
paolo.dario@sssup.it

Kerstin Dautenhahn
University of Hertfordshire
School of Computer Science
College Lane
Hatfield, AL10 9AB, UK
k.dautenhahn@herts.ac.uk

Alessandro De Luca
Sapienza University of Rome
Department of Computer, Control, and
Management Engineering
Via Ariosto 25
00185 Rome, Italy
deluca@diag.uniroma1.it

Joris De Schutter
University of Leuven (KU Leuven)
Department of Mechanical Engineering
Celestijnenlaan 300
B-3001, Leuven-Heverlee, Belgium
joris.deschutter@kuleuven.be

Rüdiger Dillmann
Karlsruhe Institute of Technology
Institute for Technical Informatics
Haid-und-Neu-Strasse 7
76131 Karlsruhe, Germany
dillmann@ira.uka.de

Lixin Dong
Michigan State University
Department of Electrical and Computer
Engineering
428 South Shaw Lane
East Lansing, MI 48824-1226, USA
ldong@egr.msu.edu

Gregory Dudek
McGill University
Department of Computer Science
3480 University Street
Montreal, QC H3Y 3H4, Canada
dudek@cim.mcgill.ca

Hugh Durrant-Whyte
University of Sydney
Australian Centre for Field Robotics (ACFR)
Sydney, NSW 2006, Australia
hugh@acfr.usyd.edu.au

Roy Featherstone
The Australian National University
Department of Information Engineering
RSISE Building 115
Canberra, ACT 0200, Australia
roy.featherstone@anu.edu.au

Gabor Fichtinger
Queen's University
School of Computing
25 Union Street
Kingston, ON, K7L 2N8, Canada
gabor@cs.queensu.ca

Paolo Fiorini
University of Verona
Department of Computer Science
Strada le Grazie 15
37134 Verona, Italy
paolo.fiorini@univr.it

Paul Fitzpatrick
Italian Institute of Technology
Robotics, Brain, and Cognitive Sciences
Department
Via Morengo 30
16163 Genoa, Italy
paul.fitzpatrick@iit.it

Luke Fletcher
Boeing Research & Technology Australia
Brisbane, QLD 4001, Australia
luke.s.fletcher@gmail.com

Dario Floreano
Swiss Federal Institute of Technology (EPFL)
Laboratory of Intelligent Systems
LIS-IMT-STI, Station 9
1015 Lausanne, Switzerland
dario.floreano@epfl.ch

Thor I. Fossen
Norwegian University of Science and Technology
Department of Engineering Cyberentics
O.S. Bragstads plass 2D
7491 Trondheim, Norway
fossen@ieee.org

Li-Chen Fu
National Taiwan University
Department of Electrical Engineering
No. 1, Sec. 4, Roosevelt Road
106 Taipei, Taiwan
lichen@ntu.edu.tw

Maxime Gautier
University of Nantes
IRCCyN, ECN
1 Rue de la Noë
44321 Nantes, France
maxime.gautier@irccyn.ec-nantes.fr

Christos Georgoulas
Technical University Munich
Department of Architecture
Arcisstrasse 21
80333 Munich, Germany
christos.georgoulas@br2.ar.tum.de

Martin A. Giese
University Clinic Tübingen
Department for Cognitive Neurology
Otfried-Müller-Strasse 25
72076 Tübingen, Germany
martin.giese@uni-tuebingen.de

Ken Goldberg
University of California at Berkeley
Department of Industrial Engineering and
Operations Research
425 Sutardja Dai Hall
Berkeley, CA 94720-1758, USA
goldberg@ieor.berkeley.edu

Clément Gosselin
Laval University
Department of Mechanical Engineering
1065 Avenue de la Médecine
Quebec, QC G1K 7P4, Canada
gosselin@gmc.ulaval.ca

Eugenio Guglielmelli
University Campus Bio-Medico of Rome
Faculty Department of Engineering
Via Alvaro del Portillo 21
00128 Rome, Italy
e.guglielmelli@unicampus.it

Sami Haddadin
Leibniz University Hannover
Electrical Engineering and Computer Science
Appelstrasse 11
30167 Hannover, Germany
sami.haddadin@irt.uni-hannover.de

Martin Hägele
Fraunhofer IPA
Robot Systems
Nobelstrasse 12
70569 Stuttgart, Germany
mmh@ipa.fhg.de

Gregory D. Hager
Johns Hopkins University
Department of Computer Science
3400 North Charles Street
Baltimore, MD 21218, USA
hager@cs.jhu.edu

William R. Hamel
University of Tennessee
Mechanical, Aerospace, and Biomedical
Engineering
414 Dougherty Engineering Building
Knoxville, TN 37996-2210, USA
whamel@utk.edu

Blake Hannaford
University of Washington
Department of Electrical Engineering
Seattle, WA 98195-2500, USA
blake@ee.washington.edu

Kensuke Harada
National Institute of Advanced Industrial Science
and Technology
Intelligent Systems Research Institute
Tsukuba Central 2, Umezono, 1-1-1
305-8568 Tsukuba, Japan
kensuke.harada@aist.go.jp

Martial Hebert
Carnegie Mellon University
The Robotics Institute
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
hebert@ri.cmu.edu

Thomas C. Henderson
University of Utah
School of Computing
50 South Central Campus Drive
Salt Lake City, UT 84112, USA
tch@cs.utah.edu

Eldert van Henten
Wageningen University
Wageningen UR Greenhouse Horticulture
Droevendaalsesteeg 4
6708 PB, Wageningen, The Netherlands
eldert.vanhenten@wur.nl

Hugh Herr
MIT Media Lab
77 Massachusetts Avenue
Cambridge, MA 02139-4307, USA
hherr@media.mit.edu

Joachim Hertzberg
Osnabrück University
Institute for Computer Science
Albrechtstrasse 28
54076 Osnabrück, Germany
joachim.hertzberg@uos.de

Gerd Hirzinger
German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Münchner Strasse 20
82230 Wessling, Germany
gerd.hirzinger@dlr.de

John Hollerbach
University of Utah
School of Computing
50 South Central Campus Drive
Salt Lake City, UT 84112, USA
jmh@cs.utah.ledu

Kaijen Hsiao
Robert Bosch LLC
Research and Technology Center, Palo Alto
4005 Miranda Avenue
Palo Alto, CA 94304, USA
kaijenhsiao@gmail.com

Tian Huang
Tianjin University
Department of Mechanical Engineering
92 Weijin Road, Naukai
300072 Tianjin, China
tianhuang@tju.edu.cn

Christoph Hürzeler
Alstom Power Thermal Services
Automation and Robotics R&D
Brown Boveri Strasse 7
5401 Baden, Switzerland
christoph.huerzeler@power.alstom.com

Phil Husbands
University of Sussex
Department of Informatics
Brighton, BN1 9QH, UK
philh@sussex.ac.uk

Seth Hutchinson
University of Illinois
Department of Electrical and Computer Engineering
1308 West Main Street
Urbana-Champaign, IL 61801, USA
seth@illinois.edu

Karl Iagnemma
Massachusetts Institute of Technology
Laboratory for Manufacturing and Productivity
77 Massachusetts Avenue
Cambridge, MA 02139, USA
kdi@mit.edu

Fumiya Iida
University of Cambridge
Department of Engineering
Trumpington Street
Cambridge, CB2 1PZ, UK
fumiya.iida@eng.cam.ac.uk

Auke Jan Ijspeert
Swiss Federal Institute of Technology (EPFL)
School of Engineering
MED 1, 1226, Station 9
1015 Lausanne, Switzerland
auke.ijspeert@epfl.ch

Genya Ishigami
Keio University
Department of Mechanical Engineering
3-14-1 Hiyoshi
223-8522 Yokohama, Japan
ishigami@mech.keio.ac.jp

Michael Jenkin
York University
Department of Electrical Engineering and Computer Science
4700 Keele Street
Toronto, ON M3J 1P3, Canada
jenkin@cse.yorku.ca

Shuuji Kajita
National Institute of Advanced Industrial Science and Technology (AIST)
Intelligent Systems Research Institute
1-1-1 Umezono
305-8586 Tsukuba, Japan
s.kajita@aist.go.jp

Takayuki Kanda
Advanced Telecommunications Research (ATR) Institute International
Intelligent Robotics and Communication Laboratories
2-2-2 Hikaridai, Seikacho, Sorakugun
619-0288 Kyoto, Japan
kanda@atr.jp

Makoto Kaneko
Osaka University
Department of Mechanical Engineering
2-1 Yamadaoka
565-0871 Suita, Japan
mk@mech.eng.osaka-u.ac.jp

Sung-Chul Kang
Korea Institute of Science and Technology
Center for Bionics
39-1 Hawolgok-dong, Wolsong-gil 5
Seoul, Seongbuk-gu, Korea
kasch@kist.re.kr

Imin Kao
Stony Brook University
Department of Mechanical Engineering
167 Light Engineering
Stony Brook, NY 11794-2300, USA
imin.kao@stonybrook.edu

Lydia E. Kavraki
Rice University
Department of Computer Science
6100 Main Street
Houston, TX 77005, USA
kavraki@rice.edu

Charles C. Kemp
Georgia Institute of Technology and Emory University
313 Ferst Drive
Atlanta, GA 30332-0535, USA
charlie.kemp@bme.gatech.edu

Wisama Khalil
University of Nantes
IRCCyN, ECN
1 Rue de la Noë
44321 Nantes, France
wisama.khalil@irccyn.ec-nantes.fr

Oussama Khatib
Stanford University
Department of Computer Sciences,
Artificial Intelligence Laboratory
450 Serra Mall
Stanford, CA 94305, USA
khatib@cs.stanford.edu

Lindsay Kleeman
Monash University
Department of Electrical and Computer Systems Engineering
Melbourne, VIC 3800, Australia
kleeman@eng.monash.edu.au

Alexander Kleiner
Linköping University
Department of Computer Science
58183 Linköping, Sweden
alexander.kleiner@liu.se

Jens Kober
Delft University of Technology
Delft Center for Systems and Control
Mekelweg 2
2628 CD, Delft, The Netherlands
j.kober@tudelft.nl

Kurt Konolige
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043, USA
konolige@gmail.com

David Kortenkamp
TRACLabs Inc
1012 Hercules Drive
Houston, TX 77058, USA
korten@traclabs.com

Kazuhiro Kosuge
Tohoku University
System Robotics Laboratory
Aoba 6-6-01, Aramaki
980-8579 Sendai, Japan
kosuge@irs.mech.tohoku.ac.jp

Danica Kragic
Royal Institute of Technology (KTH)
Centre for Autonomous Systems
CSC-CAS/CVAP
10044 Stockholm, Sweden
dani@kth.se

Torsten Kröger
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043, USA
t@kroe.org

Roman Kuc
Yale University
Department of Electrical Engineering
10 Hillhouse Avenue
New Haven, CT 06520-8267, USA
kuc@yale.edu

James Kuffner
Carnegie Mellon University
The Robotics Institute
5000 Forbes Avenue
Pittsburgh, PA 15213-3891, USA
kuffner@cs.cmu.edu

Scott Kuindersma
Harvard University
Maxwell-Dworkin 151, 33 Oxford Street
Cambridge, MA 02138, USA
scottk@seas.harvard.edu

Vijay Kumar
University of Pennsylvania
Department of Mechanical Engineering and Applied Mechanics
220 South 33rd Street
Philadelphia, PA 19104-6315, USA
kumar@seas.upenn.edu

Steven M. LaValle
University of Illinois
Department of Computer Science
201 North Goodwin Avenue, 3318 Siebel Center
Urbana, IL 61801, USA
lavalle@cs.uiuc.edu

Florant Lamiraux
LAAS-CNRS
7 Avenue du Colonel Roche
31077 Toulouse, France
florent@laas.fr

Roberto Lampariello
German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Münchner Strasse 20
82234 Wessling, Germany
roberto.lampariello@dlr.de

Christian Laugier
INRIA Grenoble Rhône-Alpes
655 Avenue de l'Europe
38334 Saint Ismier, France
christian.laugier@inria.fr

Jean-Paul Laumond
LAAS-CNRS
7 Avenue du Colonel Roche
31077 Toulouse, France
jpl@laas.fr

Daniel D. Lee
University of Pennsylvania
Department of Electrical Systems Engineering
460 Levine, 200 South 33rd Street
Philadelphia, PA 19104, USA
ddlee@seas.upenn.edu

Dongjun Lee
Seoul National University
Department of Mechanical and Aerospace Engineering
301 Engineering Building, Gwanak-ro 599, Gwanak-gu
Seoul, 51-742, Korea
djlee@snu.ac.kr

Roland Lenain
IRSTEA
Department of Ecotechnology
9 Avenue Blaise Pascal - CS20085
63178 Aubiere, France
roland.lenain@irstea.fr

David Lentink
Stanford University
Department of Mechanical Engineering
416 Escondido Mall
Stanford, CA 94305, USA
dlentink@stanford.edu

John J. Leonard
Massachusetts Institute of Technology
Department of Mechanical Engineering
5-214 77 Massachusetts Avenue
Cambridge, MA 02139, USA
jleonard@mit.edu

Aleš Leonardis
University of Birmingham
Department of Computer Science
Edgbaston
Birmingham, B15 2TT, UK
a.leonardis@cs.bham.ac.uk

Stefan Leutenegger
Imperial College London
South Kensington Campus, Department of Computing
London, SW7 2AZ, UK
s.leutenegger@imperial.ac.uk

Kevin M. Lynch
Northwestern University
Department of Mechanical Engineering
2145 Sheridan Road
Evanston, IL 60208, USA
kmlynch@northwestern.edu

Anthony A. Maciejewski
Colorado State University
Department of Electrical and Computer Engineering
Fort Collins, CO 80523-1373, USA
aam@colostate.edu

Robert Mahony
Australian National University (ANU)
Research School of Engineering
115 North Road
Canberra, ACT 2601, Australia
robert.mahony@anu.edu.au

Joshua A. Marshall
Queen's University
The Robert M. Buchan Department of Mining
25 Union Street
Kingston, ON K7L 3N6, Canada
joshua.marshall@queensu.ca

Maja J. Matarić
University of Southern California
Computer Science Department
3650 McClintock Avenue
Los Angeles, CA 90089, USA
mataric@usc.edu

Yoshio Matsumoto
National Institute of Advanced Industrial Science and Technology (AIST)
Robot Innovation Research Center
1-1-1 Umezono
305-8568 Tsukuba, Japan
yoshio.matsumoto@aist.go.jp

J. Michael McCarthy
University of California at Irvine
Department of Mechanical Engineering
5200 Engineering Hall
Irvine, CA 92697-3975, USA
jmmccart@uci.edu

Claudio Melchiorri
University of Bologna
Laboratory of Automation and Robotics
Via Risorgimento 2
40136 Bologna, Italy
claudio.melchiorri@unibo.it

Arianna Menciassi
Sant'Anna School of Advanced Studies
The BioRobotics Institute
Piazza Martiri della Libertà 34
56127 Pisa, Italy
a.menciassi@sssup.it

Jean-Pierre Merlet
INRIA Sophia-Antipolis
2004 Route des Lucioles
06560 Sophia-Antipolis, France
jean-pierre.merlet@sophia.inria.fr

Giorgio Metta
Italian Institute of Technology
iCub Facility
Via Morego 30
16163 Genoa, Italy
giorgio.metta@iit.it

François Michaud
University of Sherbrooke
Department of Electrical Engineering and Computer Engineering
2500 Boul. Université
Sherbrooke, J1N4E5, Canada
francois.michaud@usherbrooke.ca

David P. Miller
University of Oklahoma
School of Aerospace and Mechanical Engineering
865 Asp Avenue
Norman, OK 73019, USA
dpmiller@ou.edu

Javier Minguez
University of Zaragoza
Department of Computer Science and Systems Engineering
C/María de Luna 1
50018 Zaragoza, Spain
jminguez@unizar.es

Pascal Morin
University Pierre and Marie Curie
Institute for Intelligent Systems and Robotics
4 Place Jussieu
75005 Paris, France
morin@isir.upmc.fr

Mario E. Munich
iRobot Corp.
1055 East Colorado Boulevard, Suite 340
Pasadena, CA 91106, USA
mariomu@ieee.org

Robin R. Murphy
Texas A&M University
Department of Computer Science and Engineering
333 H.R. Bright Building
College Station, TX 77843-3112, USA
murphy@cse.tamu.edu

Bilge Mutlu
University of Wisconsin–Madison
Department of Computer Sciences
1210 West Dayton Street
Madison, WI 53706, USA
bilge@cs.wisc.edu

Keiji Nagatani
Tohoku University
Department of Aerospace Engineering,
Graduate School of Engineering
6-6-01, Aramaki aza Aoba
980-8579 Sendai, Japan
keiji@ieee.org

Daniele Nardi
Sapienza University of Rome
Department of Computer, Control, and
Management Engineering
Via Ariosto 25
00185 Rome, Italy
nardi@dis.uniroma1.it

Eduardo Nebot
University of Sydney
Department of Aerospace, Mechanical and
Mechatronic Engineering
Sydney, NSW 2006, Australia
eduardo.nebot@sydney.edu.au

Bradley J. Nelson
ETH Zurich
Institute of Robotics and Intelligent Systems
Tannenstrasse 3
8092 Zurich, Switzerland
bnelson@ethz.ch

Duy Nguyen-Tuong
Robert Bosch GmbH
Corporate Research
Wernerstrasse 51
70469 Stuttgart, Germany
duy@robot-learning.de

Monica Nicolescu
University of Nevada
Department of Computer Science and Engineering
1664 North Virginia Street, MS 171
Reno, NV 8955, USA
monica@unr.edu

Günter Niemeyer
Disney Research
1401 Flower Street
Glendale, CA 91201-5020, USA
gunter.niemeyer@email.disney.com

Klas Nilsson
Lund Institute of Technology
Department of Computer Science
22100 Lund, Sweden
klas.nilsson@cs.lth.se

Stefano Nolfi
National Research Council (CNR)
Institute of Cognitive Sciences and Technologies
Via S. Martino della Battaglia 44
00185 Rome, Italy
stefano.nolfi@istc.cnr.it

Illah Nourbakhsh
Carnegie Mellon University
Robotics Institute
500 Forbes Avenue
Pittsburgh, PA 15213-3890, USA
illah@andrew.cmu.edu

Andreas Nüchter
University of Würzburg
Informatics VII – Robotics and Telematics
Am Hubland
97074 Würzburg, Germany
andreas@nuechti.de

Paul Y. Oh
University of Nevada
Department of Mechanical Engineering
3141 Chestnut Street
Las Vegas, PA 19104, USA
paul@coe.drexel.edu

Yoshito Okada
Tohoku University
Department of Aerospace Engineering,
Graduate School of Engineering
6-6-01, Aramaki aza Aoba
980-8579 Sendai, Japan
okada@rm.is.tohoku.ac.jp

Allison M. Okamura
Stanford University
Department of Mechanical Engineering
416 Escondido Mall
Stanford, CA 94305-2203, USA
aokamura@stanford.edu

Fiorella Operto
Scuola di Robotica
Piazza Monastero 4
16149 Genoa, Italy
operto@scuoladirobotica.it

David E. Orin
The Ohio State University
Department of Electrical and Computer
Engineering
2015 Neil Avenue
Columbus, OH 43210-1272, USA
orin.1@osu.edu

Giuseppe Oriolo
University of Rome "La Sapienza"
Department of Computer, Control, and
Management Engineering
Via Ariosto 25
00185 Rome, Italy
oriolo@diag.uniroma1.it

Christian Ott
German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Münchner Strasse 20
82234 Wessling, Germany
christian.ott@dlr.de

Ümit Özgüner
Ohio State University
Department of Electrical and Computer
Engineering
2015 Neil Avenue
Columbus, OH 43210, USA
umit@ee.eng.ohio-state.edu

Nikolaos Papanikolopoulos
University of Minnesota
Department of Computer Science and Engineering
200 Union Street SE
Minneapolis, MN 55455, USA
npapas@cs.umn.edu

Frank C. Park
Seoul National University
Mechanical and Aerospace Engineering
Kwanak-ku, Shinlim-dong, San 56-1
Seoul, 151-742, Korea
fcp@snu.ac.kr

Jaeheung Park
Seoul National University
Department of Transdisciplinary Studies
Gwanggyo-ro 145, Yeongtong-gu
Suwon, Korea
park73@snu.ac.kr

Lynne E. Parker
University of Tennessee
Department of Electrical Engineering and
Computer Science
1520 Middle Drive
Knoxville, TN 37996, USA
leparker@utk.edu

Federico Pecora
University of Örebro
School of Science and Technology
Fakultetsgatan 1
70182 Örebro, Sweden
federico.pecora@oru.se

Jan Peters
Technical University Darmstadt
Autonomous Systems Lab
Hochschulstrasse 10
64289 Darmstadt, Germany
mail@jan-peters.net

Anna Petrovskaya
Stanford University
Department of Computer Science
353 Serra Mall
Stanford, CA 94305, USA
anya@cs.stanford.edu

J. Norberto Pires
University of Coimbra
Department of Mechanical Engineering
Palácio dos Grilos, Rua da Ilha
3000-214 Coimbra, Portugal
norberto@uc.pt

Paolo Pirjanian
iRobot Corp.
8 Crosby Drive
Bedford, MA 01730, USA
paolo.pirjanian@gmail.com

Erwin Prassler
Bonn-Rhein-Sieg Univ. of Applied Sciences
Department of Computer Sciences
Grantham-Allee 20
53754 Sankt Augustin, Germany
erwin.prassler@h-brs.de

Domenico Prattichizzo
University of Siena
Department of Information Engineering
Via Roma 56
53100 Siena, Italy
prattichizzo@ing.unisi.it

Carsten Preusche
German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Münchner Strasse 20
82234 Wessling, Germany
carsten.preusche@dlr.de

William Provancher
University of Utah
Department of Mechanical Engineering
50 South Central Campus Drive
Salt Lake City, UT 84112-9208, USA
wil@mech.utah.edu

John Reid
John Deere Co.
Moline Technology Innovation Center
One John Deere Place
Moline, IL 61265, USA
reidjohnf@johndeere.com

David J. Reinkensmeyer
University of California at Irvine
Mechanical and Aerospace Engineering and
Anatomy and Neurobiology
4200 Engineering Gateway
Irvine, CA 92697-3875, USA
dreinken@uci.edu

Jonathan Roberts
Queensland University of Technology
Department of Electrical Engineering and
Computer Science
2 George Street
Brisbane, QLD 4001, Australia
jonathan.roberts@qut.edu.au

Nicholas Roy
Massachusetts Institute of Technology
Department of Aeronautics and Astronautics
77 Massachusetts Avenue 33-315
Cambridge, MA 02139, USA
nickroy@csail.mit.edu

Daniela Rus
Massachusetts Institute of Technology
CSAIL Center for Robotics
32 Vassar Street
Cambridge, MA 02139, USA
rus@csail.mit.edu

Selma Šabanović
Indiana University Bloomington
School of Informatics and Computing
919 East 10th Street
Bloomington, IN 47408, USA
selmas@indiana.edu

Kamel S. Saidi
National Institute of Standards and Technology
Building and Fire Research Laboratory
100 Bureau Drive
Gaitherbsurg, MD 20899-1070, USA
kamel.saidi@nist.gov

Claude Samson
INRIA Sophia-Antipolis
2004 Route des Lucioles
06560 Sophia-Antipolis, France
claude.samson@inria.fr

Brian Scassellati
Yale University
Computer Science, Cognitive Science, and
Mechanical Engineering
51 Prospect Street
New Haven, CT 06520-8285, USA
scaz@cs.yale.edu

Stefan Schaal
University of Southern California
Depts. of Computer Science, Neuroscience, and
Biomedical Engineering
3710 South McClintock Avenue
Los Angeles, CA 90089-2905, USA
sschaal@tuebingen.mpg.de

Steven Scheding
University of Sydney
Rio Tinto Centre for Mine Automation
Sydney, NSW 2006, Australia
steven.scheding@sydney.edu.au

Victor Scheinman
Stanford University
Department of Mechanical Engineering
440 Escondido Mall
Stanford, CA 94305-3030, USA
vds@stanford.edu

Bernt Schiele
Saarland University
Department of Computer Science
Campus E1 4
66123 Saarbrücken, Germany
schiele@mpi-inf.mpg.de

James Schmiedeler
University of Notre Dame
Department of Aerospace and Mechanical Engineering
Notre Dame, IN 46556, USA
schmiedeler.4@nd.edu

Bruno Siciliano
University of Naples Federico II
Department of Electrical Engineering and Information Technology
Via Claudio 21
80125 Naples, Italy
bruno.siciliano@unina.it

Roland Siegwart
ETH Zurich
Department of Mechanical Engineering
Leonhardstrasse 21
8092 Zurich, Switzerland
rsiegwart@ethz.ch

Reid Simmons
Carnegie Mellon University
The Robotics Institute
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
reids@cs.cmu.edu

Patrick van der Smagt
Technical University Munich
Department of Computer Science, BRML Labs
Arcisstrasse 21
80333 Munich, Germany
smagt@brml.org

Dezhen Song
Texas A&M University
Department of Computer Science
311B H.R. Bright Building
College Station, TX 77843-3112, USA
dzsong@cs.tamu.edu

Jae-Bok Song
Korea University
Department of Mechanical Engineering
Anam-ro 145, Seongbuk-gu
Seoul, 136-713, Korea
jbsong@korea.ac.kr

Cyrill Stachniss
University of Bonn
Institute for Geodesy and Geoinformation
Nussallee 15
53115 Bonn, Germany
cyrill.stachniss@igg.uni-bonn.de

Michael Stark
Max Planck Institute of Informatics
Department of Computer Vision and Multimodal Computing
Campus E1 4
66123 Saarbrücken, Germany
stark@mpi-inf.mpg.de

Amanda K. Stowers
Stanford University
Department Mechanical Engineering
416 Escondido Mall
Stanford, CA 94305-3030, USA
astowers@stanford.edu

Stefano Stramigioli
University of Twente
Faculty of Electrical Engineering, Mathematics & Computer Science, Control Laboratory
7500 AE, Enschede, The Netherlands
s.stramigioli@utwente.nl

Gaurav S. Sukhatme
University of Southern California
Department of Computer Science
3710 South McClintock Avenue
Los Angeles, CA 90089-2905, USA
gaurav@usc.edu

Satoshi Tadokoro
Tohoku University
Graduate School of Information Sciences
6-6-01 Aramaki Aza Aoba, Aoba-ku
980-8579 Sendai, Japan
tadokoro@rm.is.tohoku.ac.jp

Wataru Takano
University of Tokyo
Department of Mechano-Informatics
7-3-1 Hongo, Bunkyo-ku
113-8656 Tokyo, Japan
takano@ynl.t.u-tokyo.ac.jp

Russell H. Taylor
The Johns Hopkins University
Department of Computer Science
3400 North Charles Street
Baltimore, MD 21218, USA
rht@jhu.edu

Russ Tedrake
Massachusetts Institute of Technology
Computer Science and Artificial Intelligence Laboratory (CSAIL)
The Stata Center, Vassar Street
Cambridge, MA 02139, USA
russt@csail.mit.edu

Sebastian Thrun
Udacity Inc.
2465 Latham Street, 3rd Floor
Mountain View, CA 94040, USA
info@udacity.com

Marc Toussaint
University of Stuttgart
Machine Learning and Robotics Lab
Universitätsstrasse 38
70569 Stuttgart, Germany
marc.toussaint@ipvs.uni-stuttgart.de

James Trevelyan
The University of Western Australia
School of Mechanical and Chemical Engineering
35 Stirling Highway
Crawley, WA 6009, Australia
james.trevelyan@uwa.edu.au

Jeffrey C. Trinkle
Rensselaer Polytechnic Institute
Department of Computer Science
110 8th Street
Troy, NY 12180-3590, USA
trink@cs.rpi.edu

Masaru Uchiyama
Tohoku University
Graduate School of Engineering
6-6-01 Aobayama
980-8579 Sendai, Japan
uchiyama@space.mech.tohoku.ac.jp

H.F. Machiel Van der Loos
University of British Columbia
Department of Mechanical Engineering
2054-6250 Applied Science Lane
Vancouver, BC V6T 1Z4, Canada
vdl@mech.ubc.ca

Manuela Veloso
Carnegie Mellon University
Computer Science Department
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
mmv@cs.cmu.edu

Gianmarco Veruggio
National Research Council (CNR)
Institute of Electronics, Computer and Telecommunication Engineering
Via De Marini 6
16149 Genoa, Italy
gianmarco@veruggio.it

Luigi Villani
University of Naples Federico II
Department of Electrical Engineering and Information Technology
Via Claudio 21
80125 Naples, Italy
luigi.villani@unina.it

Kenneth J. Waldron
University of Technology Sydney
Centre of Mechatronics and Intelligent Systems
City Campus, 15 Broadway
Ultimo, NSW 2001, Australia
kenneth.waldron@uts.edu.au

Ian D. Walker
Clemson University
Department of Electrical and Computer Engineering
105 Riggs Hall
Clemson, SC 29634, USA
ianw@ces.clemson.edu

Christian Wallraven
Korea University
Department of Brain and Cognitive Engineering, Cognitive Systems Lab
Anam-Dong 5ga, Seongbuk-gu
Seoul, 136-713, Korea
wallraven@korea.ac.kr

Pierre-Brice Wieber
INRIA Grenoble Rhône-Alpes
655 Avenue de l'Europe
38334 Grenoble, France
pierre-brice.wieber@inria.fr

Brian Wilcox
California Institute of Technology
Jet Propulsion Laboratory
4800 Oak Ridge Grove Drive
Pasadena, CA 91109, USA
brian.h.wilcox@jpl.nasa.gov

Robert Wood
Harvard University
School of Engineering and Applied Sciences
149 Maxwell-Dworkin
Cambridge, MA 02138, USA
rjwood@seas.harvard.edu

Jing Xiao
University of North Carolina
Department of Computer Science
Woodward Hall
Charlotte, NC 28223, USA
xiao@uncc.edu

Katsu Yamane
Disney Research
4720 Forbes Avenue, Suite 110
Pittsburgh, PA 15213, USA
kyamane@disneyresearch.com

Mark Yim
University of Pennsylvania
Department of Mechanical Engineering and Applied Mechanics
220 South 33rd Street
Philadelphia, PA 19104, USA
yim@seas.upenn.edu

Dana R. Yoerger
Woods Hole Oceanographic Institution
Applied Ocean Physics & Engineering
266 Woods Hole Road
Woods Hole, MA 02543-1050, USA
dyoerger@whoi.edu

Kazuhito Yokoi
AIST Tsukuba Central 2
Intelligent Systems Research Institute
1-1-1 Umezono
305-8568 Tsukuba, Ibaraki, Japan
kazuhito.yokoi@aist.go.jp

Eiichi Yoshida
National Institute of Advanced Industrial Science and Technology (AIST)
CNRS-AIST Joint Robotics Laboratory, UMI3218/CRT
1-1-1 Umezono
305-8568 Tsukuba, Ibaraki, Japan
e.yoshida@aist.go.jp

Kazuya Yoshida
Tohoku University
Department of Aerospace Engineering
Aoba 01
980-8579 Sendai, Japan
yoshida@astro.mech.tohoku.ac.jp

Junku Yuh
Korea Institute of Science and Technology
National Agenda Research Division
Hwarangno 14-gil 5, Seongbuk-gu
Seoul, 136-791, Korea
yuh.junku@gmail.com

Alex Zelinsky
Department of Defence
DST Group Headquarters
72-2-03, 24 Scherger Drive
Canberra, ACT 2609, Australia
alexzelinsky@yahoo.com

Contents

List of Abbreviations LXIII

1 **Robotics and the Handbook**
Bruno Siciliano, Oussama Khatib 1
1.1 A Brief History of Robotics 1
1.2 The Robotics Community 3
1.3 This Handbook 4
Video-References 5

Part A Robotics Foundations

2 **Kinematics**
Kenneth J. Waldron, James Schmiedeler 11
2.1 Overview 12
2.2 Position and Orientation Representation 12
2.3 Joint Kinematics 21
2.4 Geometric Representation 25
2.5 Workspace 27
2.6 Forward Kinematics 28
2.7 Inverse Kinematics 29
2.8 Forward Instantaneous Kinematics 31
2.9 Inverse Instantaneous Kinematics 32
2.10 Static Wrench Transmission 33
2.11 Conclusions and Further Reading 33
References 33

3 **Dynamics**
Roy Featherstone, David E. Orin 37
3.1 Overview 38
3.2 Spatial Vector Notation 39
3.3 Canonical Equations 45
3.4 Dynamic Models of Rigid-Body Systems 47
3.5 Kinematic Trees 51
3.6 Kinematic Loops 58
3.7 Conclusions and Further Reading 61
References 63

4 **Mechanism and Actuation**
Victor Scheinman, J. Michael McCarthy, Jae-Bok Song 67
4.1 Overview 68
4.2 System Features 68
4.3 Kinematics and Kinetics 69
4.4 Serial Robots 72
4.5 Parallel Robots 73
4.6 Mechanical Structure 75
4.7 Joint Mechanisms 76

4.8 Actuators 78
4.9 Robot Performance 85
4.10 Conclusions and Further Reading 87
Video-References 87
References 87

5 **Sensing and Estimation**
Henrik I. Christensen, Gregory D. Hager 91
5.1 Introduction 91
5.2 The Perception Process 92
5.3 Sensors 94
5.4 Estimation Processes 98
5.5 Representations 109
5.6 Conclusions and Further Readings 111
References 111

6 **Model Identification**
John Hollerbach, Wisama Khalil, Maxime Gautier 113
6.1 Overview 113
6.2 Kinematic Calibration 115
6.3 Inertial Parameter Estimation 122
6.4 Identifiability and Numerical Conditioning 127
6.5 Conclusions and Further Reading 135
Video-References 136
References 137

7 **Motion Planning**
Lydia E. Kavraki, Steven M. LaValle 139
7.1 Robotics Motion Planning 139
7.2 Motion Planning Concepts 140
7.3 Sampling-Based Planning 141
7.4 Alternative Approaches 144
7.5 Differential Constraints 148
7.6 Extensions and Variations 151
7.7 Advanced Issues 154
7.8 Conclusions and Further Reading 157
Video-References 158
References 158

8 **Motion Control**
Wan Kyun Chung, Li-Chen Fu, Torsten Kröger 163
8.1 Introduction to Motion Control 164
8.2 Joint Space Versus Operational Space Control 166
8.3 Independent-Joint Control 167
8.4 PID Control 169
8.5 Tracking Control 172
8.6 Computed-Torque Control 174
8.7 Adaptive Control 177
8.8 Optimal and Robust Control 181
8.9 Trajectory Generation and Planning 183
8.10 Digital Implementation 187

8.11 Learning Control 190
Video-References 191
References 191

9 **Force Control**
Luigi Villani, Joris De Schutter 195
9.1 Background 195
9.2 Indirect Force Control 198
9.3 Interaction Tasks 205
9.4 Hybrid Force/Motion Control 211
9.5 Conclusions and Further Reading 216
Video-References 217
References 218

10 **Redundant Robots**
Stefano Chiaverini, Giuseppe Oriolo, Anthony A. Maciejewski 221
10.1 Overview 221
10.2 Task-Oriented Kinematics 224
10.3 Inverse Differential Kinematics 227
10.4 Redundancy Resolution via Optimization 232
10.5 Redundancy Resolution via Task Augmentation 233
10.6 Second-Order Redundancy Resolution 236
10.7 Cyclicity 237
10.8 Fault Tolerance 237
10.9 Conclusion and Further Reading 239
Video-References 239
References 240

11 **Robots with Flexible Elements**
Alessandro De Luca, Wayne J. Book 243
11.1 Robots with Flexible Joints 244
11.2 Robots with Flexible Links 263
Video-References 279
References 279

12 **Robotic Systems Architectures and Programming**
David Kortenkamp, Reid Simmons, Davide Brugali 283
12.1 Overview 283
12.2 History 285
12.3 Architectural Components 289
12.4 Case Study – GRACE 296
12.5 The Art of Robot Architectures 298
12.6 Implementing Robotic Systems Architectures 299
12.7 Conclusions and Further Reading 302
Video-References 302
References 302

13 **Behavior-Based Systems**
François Michaud, Monica Nicolescu 307
13.1 Robot Control Approaches 308
13.2 Basic Principles of Behavior-Based Systems 310

13.3 Basis Behaviors 313
13.4 Representation in Behavior-Based Systems 313
13.5 Learning in Behavior-Based Systems 314
13.6 Applications and Continuing Work 318
13.7 Conclusions and Further Reading 322
Video-References 322
References 323

14 **AI Reasoning Methods for Robotics**
Michael Beetz, Raja Chatila, Joachim Hertzberg, Federico Pecora 329
14.1 Why Should a Robot Use AI-Type Reasoning? 330
14.2 Knowledge Representation and Processing 330
14.3 Reasoning and Decision Making 338
14.4 Plan-Based Robot Control 346
14.5 Conclusions and Further Reading 351
Video-References 351
References 352

15 **Robot Learning**
Jan Peters, Daniel D. Lee, Jens Kober, Duy Nguyen-Tuong, J. Andrew Bagnell, Stefan Schaal 357
15.1 What Is Robot Learning 358
15.2 Model Learning 360
15.3 Reinforcement Learning 372
15.4 Conclusions 385
Video-References 386
References 386

Part B Design

16 **Design and Performance Evaluation**
Jorge Angeles, Frank C. Park 399
16.1 The Robot Design Process 400
16.2 Workspace Criteria 401
16.3 Dexterity Indices 405
16.4 Other Performance Indices 408
16.5 Other Robot Types 411
16.6 Summary 416
References 416

17 **Limbed Systems**
Shuuji Kajita, Christian Ott 419
17.1 Design of Limbed Systems 420
17.2 Conceptual Design 420
17.3 Whole Design Process Example 423
17.4 Model Induced Design 427
17.5 Various Limbed Systems 434
17.6 Performance Indices 437
Video-References 439
References 440

18 **Parallel Mechanisms**
Jean-Pierre Merlet, Clément Gosselin, Tian Huang 443
18.1 Definitions 443
18.2 Type Synthesis of Parallel Mechanisms 445
18.3 Kinematics 446
18.4 Velocity and Accuracy Analysis 447
18.5 Singularity Analysis 448
18.6 Workspace Analysis 450
18.7 Static Analysis 451
18.8 Dynamic Analysis 452
18.9 Design 452
18.10 Wire-Driven Parallel Robots 453
18.11 Application Examples 455
18.12 Conclusion and Further Reading 455
Video-References 456
References 456

19 **Robot Hands**
Claudio Melchiorri, Makoto Kaneko 463
19.1 Basic Concepts 464
19.2 Design of Robot Hands 465
19.3 Technologies for Actuation and Sensing 470
19.4 Modeling and Control of a Robot Hand 473
19.5 Applications and Trends 477
19.6 Conclusions and Further Reading 478
Video-References 478
References 479

20 **Snake-Like and Continuum Robots**
Ian D. Walker, Howie Choset, Gregory S. Chirikjian 481
20.1 Snake Robots – Short History 481
20.2 Continuum Robots – Short History 485
20.3 Snake-Like and Continuum Robot Modeling 487
20.4 Modeling of Locomotion for Snake-Like and Continuum Mechanisms 491
20.5 Conclusion and Extensions to Related Areas 492
Video-References 492
References 493

21 **Actuators for Soft Robotics**
Alin Albu-Schäffer, Antonio Bicchi 499
21.1 Background 500
21.2 Soft Robot Design 502
21.3 Modeling Actuators for Soft Robotics 508
21.4 Modeling Soft Robots 511
21.5 Stiffness Estimation 513
21.6 Cartesian Stiffness Control 515
21.7 Periodic Motion Control 518
21.8 Optimal Control of Soft Robots 521
21.9 Conclusions and Open Problems 524

Video-References 525
References 526

22 **Modular Robots**
I-Ming Chen, Mark Yim 531
22.1 Concepts and Definitions 531
22.2 Reconfigurable Modular Manipulators 533
22.3 Self-Reconfigurable Modular Robots 535
22.4 Conclusion and Further Reading 539
Video-References 540
References 540

23 **Biomimetic Robots**
Kyu-Jin Cho, Robert Wood 543
23.1 Overview 544
23.2 Components of Biomimetic Robot Design 544
23.3 Mechanisms 545
23.4 Material and Fabrication 561
23.5 Conclusion 567
Video-References 568
References 570

24 **Wheeled Robots**
Woojin Chung, Karl Iagnemma 575
24.1 Overview 575
24.2 Mobility of Wheeled Robots 576
24.3 Wheeled Robot Structures 582
24.4 Wheel–Terrain Interaction Models 586
24.5 Wheeled Robot Suspensions 589
24.6 Conclusions 592
Video-References 592
References 593

25 **Underwater Robots**
Hyun-Taek Choi, Junku Yuh 595
25.1 Background 595
25.2 Mechanical Systems 596
25.3 Power Systems 599
25.4 Underwater Actuators and Sensors 601
25.5 Computers, Communications, and Architecture 606
25.6 Underwater Manipulators 614
25.7 Conclusions and Further Reading 617
Video-References 617
References 618

26 **Flying Robots**
Stefan Leutenegger, Christoph Hürzeler, Amanda K. Stowers, Kostas Alexis, Markus W. Achtelik, David Lentink, Paul Y. Oh, Roland Siegwart 623
26.1 Background and History 624
26.2 Characteristics of Aerial Robotics 625
26.3 Basics of Aerodynamics and Flight Mechanics 629
26.4 Airplane Modeling and Design 641

26.5 Rotorcraft Modeling and Design 647
26.6 Flapping Wing Modeling and Design 653
26.7 System Integration and Realization 659
26.8 Applications of Aerial Robots 662
26.9 Conclusions and Further Reading 666
Video-References 666
References 667

27 **Micro-/Nanorobots**
Bradley J. Nelson, Lixin Dong, Fumihito Arai 671
27.1 Overview of Micro- and Nanorobotics 671
27.2 Scaling 674
27.3 Actuation at the Micro- and Nanoscales 675
27.4 Imaging at the Micro- and Nanoscales 676
27.5 Fabrication 678
27.6 Microassembly 681
27.7 Microrobotics 687
27.8 Nanorobotics 692
27.9 Conclusions 704
Video-References 704
References 705

Part C Sensing and Perception

28 **Force and Tactile Sensing**
Mark R. Cutkosky, William Provancher 717
28.1 Overview 717
28.2 Sensor Types 718
28.3 Tactile Information Processing 725
28.4 Integration Challenges 730
28.5 Conclusions and Future Developments 731
Video-References 731
References 731

29 **Inertial Sensing, GPS and Odometry**
Gregory Dudek, Michael Jenkin 737
29.1 Odometry 737
29.2 Gyroscopic Systems 739
29.3 Accelerometers 742
29.4 IMU Packages 743
29.5 Satellite-Based Positioning (GPS and GNSS) 744
29.6 GPS-IMU Integration 749
29.7 Further Reading 750
29.8 Currently Available Hardware 750
References 751

30 **Sonar Sensing**
Lindsay Kleeman, Roman Kuc 753
30.1 Sonar Principles 754
30.2 Sonar Beam Pattern 756
30.3 Speed of Sound 758

30.4 Waveforms 758
30.5 Transducer Technologies 759
30.6 Reflecting Object Models 760
30.7 Artifacts 761
30.8 TOF Ranging 762
30.9 Echo Waveform Coding 765
30.10 Echo Waveform Processing 767
30.11 CTFM Sonar 769
30.12 Multipulse Sonar 772
30.13 Sonar Rings and Arrays 773
30.14 Motion Effects 775
30.15 Biomimetic Sonars 778
30.16 Conclusions 779
Video-References 780
References 780

31 **Range Sensing**
Kurt Konolige, Andreas Nüchter 783
31.1 Range Sensing Basics 783
31.2 Sensor Technologies 785
31.3 Registration 794
31.4 Navigation and Terrain Classification and Mapping 804
31.5 Conclusions and Further Reading 807
References 807

32 **3-D Vision for Navigation and Grasping**
Danica Kragic, Kostas Daniilidis 811
32.1 Geometric Vision 812
32.2 3-D Vision for Grasping 820
32.3 Conclusion and Further Reading 822
Video-References 822
References 822

33 **Visual Object Class Recognition**
Michael Stark, Bernt Schiele, Aleš Leonardis 825
33.1 Object Classes 825
33.2 Review of the State of the Art 826
33.3 Discussion and Conclusions 837
References 838

34 **Visual Servoing**
François Chaumette, Seth Hutchinson, Peter Corke 841
34.1 The Basic Components of Visual Servoing 842
34.2 Image-Based Visual Servo 843
34.3 Pose-Based Visual Servo 851
34.4 Advanced Approaches 854
34.5 Performance Optimization and Planning 856
34.6 Estimation of 3-D Parameters 858
34.7 Determining s^* and Matching Issues 859
34.8 Target Tracking 859

34.9 Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space 860
34.10 Under Actuated Robots 861
34.11 Applications 863
34.12 Conclusions 863
Video-References 863
References 863

35 **Multisensor Data Fusion**
Hugh Durrant-Whyte, Thomas C. Henderson 867
35.1 Multisensor Data Fusion Methods 867
35.2 Multisensor Fusion Architectures 880
35.3 Applications 885
35.4 Conclusions 889
Video-References 889
References 890

Part D Manipulation and Interfaces

36 **Motion for Manipulation Tasks**
James Kuffner, Jing Xiao 897
36.1 Overview 898
36.2 Task-Level Control 900
36.3 Manipulation Planning 904
36.4 Assembly Motion 911
36.5 Unifying Feedback Control and Planning 918
36.6 Conclusions and Further Reading 920
Video-References 923
References 923

37 **Contact Modeling and Manipulation**
Imin Kao, Kevin M. Lynch, Joel W. Burdick 931
37.1 Overview 931
37.2 Kinematics of Rigid-Body Contact 932
37.3 Forces and Friction 936
37.4 Rigid-Body Mechanics with Friction 939
37.5 Pushing Manipulation 942
37.6 Contact Interfaces and Modeling 943
37.7 Friction Limit Surface 946
37.8 Contacts in Grasping and Fixture Designs 949
37.9 Conclusions and Further Reading 950
Video-References 951
References 951

38 **Grasping**
Domenico Prattichizzo, Jeffrey C. Trinkle 955
38.1 Models and Definitions 956
38.2 Controllable Twists and Wrenches 961
38.3 Compliant Grasps 965
38.4 Restraint Analysis 967

38.5 Examples 975
38.6 Conclusion and Further Reading 985
Video-References 986
References 986

39 **Cooperative Manipulation**
Fabrizio Caccavale, Masaru Uchiyama 989
39.1 Historical Overview 990
39.2 Kinematics and Statics 991
39.3 Cooperative Task Space 995
39.4 Dynamics and Load Distribution 996
39.5 Task-Space Analysis 998
39.6 Control 999
39.7 Conclusions and Further Reading 1003
Video-References 1004
References 1004

40 **Mobility and Manipulation**
Oliver Brock, Jaeheung Park, Marc Toussaint 1007
40.1 Grasping and Manipulation 1009
40.2 Control 1013
40.3 Motion Generation 1017
40.4 Learning 1021
40.5 Perception 1025
40.6 Conclusions and Further Reading 1029
Video-References 1029
References 1030

41 **Active Manipulation for Perception**
Anna Petrovskaya, Kaijen Hsiao 1037
41.1 Perception via Manipulation 1037
41.2 Object Localization 1038
41.3 Learning About an Object 1049
41.4 Recognition 1054
41.5 Conclusions 1057
Video-References 1058
References 1058

42 **Haptics**
Blake Hannaford, Allison M. Okamura 1063
42.1 Overview 1064
42.2 Haptic Device Design 1068
42.3 Haptic Rendering 1071
42.4 Control and Stability of Force Feedback Interfaces 1073
42.5 Other Types of Haptic Interfaces 1075
42.6 Conclusions and Further Reading 1079
References 1079

43 **Telerobotics**
Günter Niemeyer, Carsten Preusche, Stefano Stramigioli, Dongjun Lee 1085
43.1 Overview and Terminology 1085

43.2 Telerobotic Systems and Applications 1087
43.3 Control Architectures 1090
43.4 Bilateral Control and Force Feedback 1095
43.5 Emerging Applications of Telerobotics 1101
43.6 Conclusions and Further Reading 1104
Video-References 1104
References 1105

44 **Networked Robots**
Dezhen Song, Ken Goldberg, Nak-Young Chong 1109
44.1 Overview and Background 1109
44.2 A Brief History 1110
44.3 Communications and Networking 1112
44.4 Properties of Networked Robots 1115
44.5 Cloud Robotics 1121
44.6 Conclusion and Future Directions 1125
Video-References 1126
References 1126

Part E Moving in the Environment

45 **World Modeling**
Wolfram Burgard, Martial Hebert, Maren Bennewitz 1135
45.1 Historical Overview 1136
45.2 Models for Indoors and Structured Environments 1137
45.3 World and Terrain Models for Natural Environments 1141
45.4 Dynamic Environments 1149
45.5 Summary and Further Reading 1149
Video-References 1150
References 1150

46 **Simultaneous Localization and Mapping**
Cyrill Stachniss, John J. Leonard, Sebastian Thrun 1153
46.1 SLAM: Problem Definition 1154
46.2 The Three Main SLAM Paradigms 1157
46.3 Visual and RGB-D SLAM 1166
46.4 Conclusion and Future Challenges 1169
Video-References 1170
References 1171

47 **Motion Planning and Obstacle Avoidance**
Javier Minguez, Florant Lamiraux, Jean-Paul Laumond 1177
47.1 Nonholonomic Mobile Robots: Where Motion Planning Meets Control Theory 1178
47.2 Kinematic Constraints and Controllability 1179
47.3 Motion Planning and Small-Time Controllability 1180
47.4 Local Steering Methods and Small-Time Controllability 1181
47.5 Robots and Trailers 1184
47.6 Approximate Methods 1186
47.7 From Motion Planning to Obstacle Avoidance 1187

47.8 Definition of Obstacle Avoidance 1187
47.9 Obstacle Avoidance Techniques 1188
47.10 Robot Shape, Kinematics, and Dynamics in Obstacle Avoidance 1194
47.11 Integration Planning – Reaction 1196
47.12 Conclusions, Future Directions, and Further Reading 1198
Video-References 1199
References 1199

48 **Modeling and Control of Legged Robots**
Pierre-Brice Wieber, Russ Tedrake, Scott Kuindersma 1203
48.1 A Brief History of Legged Robots 1204
48.2 The Dynamics of Legged Locomotion 1204
48.3 Stability Analysis – Not Falling Down 1209
48.4 Generation of Dynamic Walking and Running Motions 1214
48.5 Motion and Force Control 1222
48.6 Towards More Efficient Walking 1225
48.7 Different Contact Behaviors 1227
48.8 Conclusion 1228
References 1228

49 **Modeling and Control of Wheeled Mobile Robots**
Claude Samson, Pascal Morin, Roland Lenain 1235
49.1 Background 1236
49.2 Control Models 1238
49.3 Adaptation of Control Methods for Holonomic Systems 1240
49.4 Methods Specific to Nonholonomic Systems 1241
49.5 Path Following in the Case of Nonideal Wheel-Ground Contact 1255
49.6 Complementary Issues and Bibliographical Guide 1261
Video-References 1263
References 1263

50 **Modeling and Control of Robots on Rough Terrain**
Keiji Nagatani, Genya Ishigami, Yoshito Okada 1267
50.1 Overview 1268
50.2 Modeling of Wheeled Robot in Rough Terrain 1270
50.3 Control of Wheeled Robot in Rough Terrain 1274
50.4 Modeling of Tracked Vehicle on Rough Terrain 1276
50.5 Stability Analysis of Tracked Vehicles 1278
50.6 Control of Tracked Vehicle on Rough Terrain 1279
50.7 Summary 1281
Video-References 1281
References 1282

51 **Modeling and Control of Underwater Robots**
Gianluca Antonelli, Thor I. Fossen, Dana R. Yoerger 1285
51.1 The Expanding Role of Marine Robotics in Oceanic Engineering 1285
51.2 Underwater Robotics 1287
51.3 Applications 1302
51.4 Conclusions and Further Reading 1303
Video-References 1304
References 1304

52 **Modeling and Control of Aerial Robots**
Robert Mahony, Randal W. Beard, Vijay Kumar 1307
52.1 Overview 1307
52.2 Modeling Aerial Robotic Vehicles 1309
52.3 Control 1316
52.4 Trajectory Planning 1324
52.5 Estimating the Vehicle State 1328
52.6 Conclusion 1330
Video-References 1331
References 1331

53 **Multiple Mobile Robot Systems**
Lynne E. Parker, Daniela Rus, Gaurav S. Sukhatme 1335
53.1 History 1336
53.2 Architectures for Multirobot Systems 1337
53.3 Communication 1339
53.4 Networked Mobile Robots 1340
53.5 Swarm Robots 1351
53.6 Modular Robotics 1354
53.7 Heterogeneity 1357
53.8 Task Allocation 1359
53.9 Learning 1361
53.10 Applications 1362
53.11 Conclusions and Further Reading 1366
Video-References 1366
References 1367

Part F Robots at Work

54 **Industrial Robotics**
Martin Hägele, Klas Nilsson, J. Norberto Pires, Rainer Bischoff 1385
54.1 Industrial Robotics: The Main Driver for Robotics Research and Application 1386
54.2 A Short History of Industrial Robots 1386
54.3 Industrial Robot Kinematics 1392
54.4 Typical Industrial Robot Applications 1393
54.5 Safe Human–Robot Collaboration 1405
54.6 Task Descriptions – Teaching and Programming 1409
54.7 System Integration 1414
54.8 Outlook and Long-Term Challenges 1416
Video-References 1418
References 1418

55 **Space Robotics**
Kazuya Yoshida, Brian Wilcox, Gerd Hirzinger, Roberto Lampariello 1423
55.1 Historical Developments and Advances of Orbital Robotic Systems . 1424
55.2 Historical Developments and Advances of Surface Robotic Systems. 1430
55.3 Mathematical Modeling 1437
55.4 Future Directions of Orbital and Surface Robotic Systems 1452
55.5 Conclusions and Further Reading 1457

Video-References 1457
References 1458

56 **Robotics in Agriculture and Forestry**
Marcel Bergerman, John Billingsley, John Reid, Eldert van Henten 1463
56.1 Section Scope 1464
56.2 Challenges and Opportunities 1465
56.3 Case Studies 1467
56.4 Conclusion 1487
Video-References 1488
References 1489

57 **Robotics in Construction**
Kamel S. Saidi, Thomas Bock, Christos Georgoulas 1493
57.1 Overview 1494
57.2 Offsite Applications of Robotics in Construction 1499
57.3 Onsite Applications of Single Task Construction Robots 1504
57.4 Integrated Robotized Construction Sites 1511
57.5 Currently Unsolved Technical Problems 1514
57.6 Future Directions 1516
57.7 Conclusions and Further Reading 1516
Video-References 1517
References 1517

58 **Robotics in Hazardous Applications**
James Trevelyan, William R. Hamel, Sung-Chul Kang 1521
58.1 Operation in Hazardous Environments: The Need for a Robotics Solution 1521
58.2 Applications 1523
58.3 Enabling Technologies 1537
58.4 Conclusions and Further Reading 1544
Video-References 1545
References 1546

59 **Robotics in Mining**
Joshua A. Marshall, Adrian Bonchis, Eduardo Nebot, Steven Scheding 1549
59.1 Modern Mining Practice 1550
59.2 Surface Mining 1552
59.3 Underground Mining 1562
59.4 Challenges and Industry Acceptance 1568
59.5 Challenges, Outlook, and Conclusion 1569
Video-References 1571
References 1572

60 **Disaster Robotics**
Robin R. Murphy, Satoshi Tadokoro, Alexander Kleiner 1577
60.1 Overview 1578
60.2 Disaster Characteristics and Impact on Robots 1581
60.3 Robots Actually Used at Disasters 1582
60.4 Robots at the Fukushima-Daiichi Nuclear Power Plant Accident 1588
60.5 Lessons Learned, Challenges, and Novel Approaches 1591

60.6 Evaluation 1598
60.7 Conclusions and Further Reading 1600
Video-References 1601
References 1601

61 **Robot Surveillance and Security**
Wendell H. Chun, Nikolaos Papanikolopoulos 1605
61.1 Overview 1605
61.2 Application Domains 1607
61.3 Enabling Technologies 1608
61.4 Active Research 1617
61.5 Conclusion 1622
Video-References 1623
References 1623

62 **Intelligent Vehicles**
Alberto Broggi, Alex Zelinsky, Ümit Özgüner, Christian Laugier 1627
62.1 The Motivation and Approaches to Intelligent Vehicles 1628
62.2 Enabling Technologies 1632
62.3 Road Scene Understanding 1635
62.4 Advanced Driver Assistance 1639
62.5 Driver Monitoring 1645
62.6 Towards Fully Autonomous Vehicles 1647
62.7 Future Trends and Prospects 1650
62.8 Conclusions and Further Reading 1651
Video-References 1651
References 1652

63 **Medical Robotics and Computer-Integrated Surgery**
Russell H. Taylor, Arianna Menciassi, Gabor Fichtinger, Paolo Fiorini, Paolo Dario 1657
63.1 Core Concepts 1658
63.2 Technology 1662
63.3 Systems, Research Areas, and Applications 1667
63.4 Conclusion and Future Directions 1675
Video-References 1676
References 1676

64 **Rehabilitation and Health Care Robotics**
H.F. Machiel Van der Loos, David J. Reinkensmeyer, Eugenio Guglielmelli . 1685
64.1 Overview 1686
64.2 Rehabilitation Therapy and Training Robots 1692
64.3 Aids for People with Disabilities 1703
64.4 Smart Prostheses and Orthoses 1711
64.5 Augmentation for Diagnosis and Monitoring 1713
64.6 Safety, Ethics, Access and Economics 1715
64.7 Conclusions and Further Readings 1716
Video-References 1717
References 1718

65 **Domestic Robotics**
Erwin Prassler, Mario E. Munich, Paolo Pirjanian, Kazuhiro Kosuge 1729
65.1 Mobile Domestic Robotics 1730
65.2 Enabling Technologies 1747
65.3 Smart Homes 1754
Video-References 1757
References 1757

66 **Robotics Competitions and Challenges**
Daniele Nardi, Jonathan Roberts, Manuela Veloso, Luke Fletcher 1759
66.1 Introduction 1760
66.2 Overview 1760
66.3 Competitions Inspired by Human Competitions 1762
66.4 Task-Oriented Competitions 1769
66.5 Conclusion and Further Reading 1780
Video-References 1781
References 1781

Part G Robots and Humans

67 **Humanoids**
Paul Fitzpatrick, Kensuke Harada, Charles C. Kemp, Yoshio Matsumoto, Kazuhito Yokoi, Eiichi Yoshida 1789
67.1 Why Humanoids? 1789
67.2 History 1792
67.3 What to Immitate? 1794
67.4 Locomotion 1795
67.5 Whole-Body Activities 1801
67.6 Morphological Communication 1809
67.7 Conclusions and Further Reading 1813
Video-References 1813
References 1813

68 **Human Motion Reconstruction**
Katsu Yamane, Wataru Takano 1819
68.1 Overview 1819
68.2 Models and Computations 1820
68.3 Reconstruction for Understanding 1825
68.4 Reconstruction for Robots 1829
Video-References 1830
References 1831

69 **Physical Human–Robot Interaction**
Sami Haddadin, Elizabeth Croft 1835
69.1 Classification 1836
69.2 Human Safety 1839
69.3 Human-Friendly Robot Design 1847
69.4 Control for Physical Interaction 1853
69.5 Motion Planning for Human Environments 1859
69.6 Interaction Planning 1862

69.7 Conclusions and Challenges 1867
Video-References 1868
References 1869

70 **Human–Robot Augmentation**
Massimo Bergamasco, Hugh Herr 1875
70.1 Concept and Definitions 1876
70.2 Upper Limb Wearable Systems 1877
70.3 Lower Limb Wearable Systems 1882
70.4 Whole Body Wearable Systems 1889
70.5 Control of Human–Robot Augmentation Systems 1892
70.6 Conclusions and Further Developments 1902
Video-References 1902
References 1902

71 **Cognitive Human–Robot Interaction**
Bilge Mutlu, Nicholas Roy, Selma Šabanović 1907
71.1 Human Models of Interaction 1908
71.2 Robot Models of Interaction 1914
71.3 Models of Human–Robot Interaction 1916
71.4 Conclusion and Further Reading 1927
Video-References 1927
References 1928

72 **Social Robotics**
Cynthia Breazeal, Kerstin Dautenhahn, Takayuki Kanda 1935
72.1 Overview 1936
72.2 Social Robot Embodiment 1936
72.3 Social Robots and Social-Emotional Intelligence 1938
72.4 Socio-Cognitive Skills 1941
72.5 Human Social Responses to Social Robots 1944
72.6 Social Robots and Communication Skills 1946
72.7 Long-Term Interaction with Robot Companions 1950
72.8 Tactile Interaction with Social Robots 1954
72.9 Social Robots and Teamwork 1958
72.10 Conclusion 1959
72.11 Further Reading 1960
Video-References 1960
References 1961

73 **Socially Assistive Robotics**
Maja J. Matarić, Brian Scassellati 1973
73.1 Overview 1973
73.2 The Need for Socially Assistive Robotics 1974
73.3 Advantages of Embodied Robots over Virtual Agents 1975
73.4 Motivation, Autonomy, and Companionship 1977
73.5 Influence and the Dynamics of Assistive Interaction 1978
73.6 Personalization and Adaptation to Specific Needs and Abilities 1978
73.7 Creating Long-Term Engagement and Behaviour Change 1979
73.8 SAR for Autism Spectrum Disorder (ASD) Therapy 1980
73.9 SAR Supporting Rehabilitation 1982

73.10 SAR and Eldercare 1985
73.11 SAR for Alzheimer's Dementia and Cognitive Rehabilitation 1986
73.12 Ethical and Safety Considerations 1987
References 1988

74 **Learning from Humans**
Aude G. Billard, Sylvain Calinon, Rüdiger Dillmann 1995
74.1 Learning of Robots 1995
74.2 Key Issues When Learning from Human Demonstrations 1998
74.3 Interfaces for Demonstration 2000
74.4 Algorithms to Learn from Humans 2002
74.5 Conclusions and Open Issues in Robot LfD 2008
Video-References 2009
References 2009

75 **Biologically Inspired Robotics**
Fumiya Iida, Auke Jan Ijspeert 2015
75.1 General Background 2016
75.2 Methodology 2017
75.3 Case Studies 2021
75.4 Landscape of Bio-Inspired Robotics Research and Challenges 2026
75.5 Conclusion 2028
Video-References 2028
References 2029

76 **Evolutionary Robotics**
Stefano Nolfi, Josh Bongard, Phil Husbands, Dario Floreano 2035
76.1 Method 2036
76.2 First Steps 2036
76.3 Simulation and Reality 2040
76.4 Behavior as a Complex Adaptive System 2041
76.5 Evolving Bodies 2044
76.6 Seeing the Light 2046
76.7 Computational Neuroethology 2049
76.8 Evolution and Learning 2054
76.9 Evolution of Social Behavior 2057
76.10 Evolutionary Hardware 2060
76.11 Closing Remarks 2061
Video-References 2061
References 2062

77 **Neurorobotics: From Vision to Action**
Patrick van der Smagt, Michael A. Arbib, Giorgio Metta 2069
77.1 Definitions and History 2070
77.2 The Case for Vision 2071
77.3 Vertebrate Motor Control 2075
77.4 The Role of Mirror Systems 2082
77.5 Conclusion and Further Reading 2089
References 2090

78 **Perceptual Robotics**
Heinrich Bülthoff, Christian Wallraven, Martin A. Giese 2095
78.1 Perceptual Mechanisms of Object Representations 2097
78.2 Perceptual Mechanisms of Action Representation 2103
78.3 Perceptual Validation of Robotics 2107
78.4 Conclusion and Further Reading 2108
Video-References 2109
References 2109

79 **Robotics for Education**
David P. Miller, Illah Nourbakhsh 2115
79.1 The Role of Robots in Education 2116
79.2 Educational Robot Tournaments 2117
79.3 Education Robot Platforms 2120
79.4 Education Robot Controllers and Programming Environments 2123
79.5 Robotic Technologies for Student Learning 2127
79.6 Educational Evaluation of Robot Programs 2129
79.7 Conclusions and Further Reading 2131
Video-References 2131
References 2131

80 **Roboethics: Social and Ethical Implications**
Gianmarco Veruggio, Fiorella Operto, George Bekey 2135
80.1 A Methodological Note 2137
80.2 Specificity of Robotics 2138
80.3 Cultural Differences in the Acceptance of Robots 2138
80.4 Roboethics Foreshadowed in the Literature 2139
80.5 And Expressed in Real Robotics 2139
80.6 Ethics in Science and Technology 2140
80.7 Ethical Issues in an ICT Society 2143
80.8 Human Principles and Rights 2144
80.9 Legal Issues in Robotics 2146
80.10 Roboethics Taxonomy 2147
80.11 Roboethics Enforced: From Ideals to Rules 2156
80.12 Conclusions and Further Reading 2157
Video-References 2158
References 2159

Acknowledgements 2161
About the Authors 2163
Index 2197

List of Abbreviations

Symbols

k-NN	*k*-nearest neighbor
0-D	zero-dimensional
1-D	one-dimensional
2-D	two-dimensional
2.5-D	two-and-a-half-dimensional
3-D	three-dimensional
3-D-NDT	three-dimensional!normal distributions transform
4-D	four-dimensional
6-D	six-dimensional
6R	six-revolute
7R	seven-revolute

A

A&F	agriculture and forestry
AA	agonist–antagonist
AAAI	American Association for Artificial Intelligence
AAAI	Association for the Advancement of Artificial Intelligence
AAL	ambient assisted living
ABA	articulated-body algorithm
ABF	artificial bacterial flagella
ABRT	automated!bus rapid transit
ABS	acrylonitrile–butadiene–styrene
AC	aerodynamic center
AC	alternating current
ACARP	Australian Coal Association Research Program
ACBS	automatic!constructions building system
ACC	adaptive cruise control
ACFV	autonomous!combat flying vehicle
ACM	active chord mechanism
ACM	active cord mechanism
ACT	anatomically correct testbed
ADAS	advanced driving assistance system
ADC	analog digital conveter
ADCP	acoustic Doppler current profiler
ADL	activities for daily living
ADSL	asymmetric digital subscriber line
AFC	alkaline fuel cell
AFC	armoured (or articulated) face conveyor
AFM	atomic force microscope
AFV	autonomous!flying vehicle
AGV	autonomous guided vehicle
AGV	automated!guided vehicle
AHRS	attitude and heading reference system
AHS	advanced highway system
AI	artificial intelligence
AIAA	American Institute of Aeronautics and Astronautics
AIM	assembly incidence matrix
AIP	air-independent power
AIP	anterior intraparietal sulcus
AIP	anterior interparietal area
AIS	artificial intelligence system
AIST	Institute of Advanced Industrial Science and Technology
AIST	Japan National Institute of Advanced Industrial Science and Technology
AIST	National Institute of Advanced Industrial Science and Technology (Japan)
AIT	anterior inferotemporal cortex
ALEX	active leg exoskeleton
AM	actuator for manipulation
AMASC	actuator with mechanically adjustable series compliance
AMC	Association for Computing Machinery
AMD	autonomous!mental development
AMM	audio-motor map
ANN	artificial neural network
AO	Arbeitsgemeinschaft für Ostheosynthesefragen
AOA	angle of attack
AP	antipersonnel
APF	annealed particle filter
APG	adjustable pattern generator
API	application programming interface
APOC	allowing dynamic selection and changes
AR	autoregressive
aRDnet	agile robot development network
ARM	Acorn RISC machine architecture
ARM	assistive!robot service manipulator
ARX	auto regressive estimator
ASAP	adaptive sampling and prediction
ASCII	American standard code for information interchange
ASD	autism spectrum disorder
ASIC	application-specific integrated circuit
ASIC	application-specific feature transform
ASIMO	advanced step in innovative mobility
ASK	amplitude shift keying
ASL	autonomous systems laboratory
ASM	advanced servomanipulator

ASN	active sensor network
ASR	automatic!spoken-language recognition
ASR	automatic!speech recognition
ASTRO	autonomous!space transport robotic operations
ASV	adaptive suspension vehicle
ASyMTRe	automated!synthesis of multirobot task solutions through software reconfiguration
AT	anti-tank mine
ATHLETE	all-terrain hex-legged extra-terrestrial explorer
ATLANTIS	a three layer architecture for navigating through intricate situations
ATLSS	advanced technology for large structural systems
ATR	automatic!target recognition
AuRA	autonomous robot architecture
AUV	autonomous underwater vehicle
AUV	autonomous aquatic vehicle
AUVAC	Autonomous Undersea Vehicles Application Center
AUVSI	Association for Unmanned Vehicle Systems International
AV	anti-vehicle

B

B/S	browser/server
B2B	business to business
BCI	brain-computer interface
BE	body extender
BEMT	blade element momentum theory
BEST	boosting!engineering science and technology
BET	blade element theory
BFA	bending fluidic actuator
BFP	best-first-planner
BI	brain imaging
BIP	behavior-interaction-priority
BLE	broadcast of local eligibility
BLEEX	Berkely exoskeleton
BLUE	best linear unbiased estimator
BML	behavior!mark-up language
BMS	battery management system
BN	Bayesian network
BOM	bill of material
BOw	bag-of-word
BP	behavior primitive
BP	base plate
BRICS	best practice in robotics
BRT	bus rapid transit
BWSTT	body-weight supported treadmill training

C

C	cylindrical joint
C/A	coarse-acquisition
C/S	client/server
CA	collision avoidance
CACC	cooperative adaptive cruise control
CAD	computer-aided drafting
CAD	computer-aided design
CAE	computer-aided engineering
CALM	communication access for land mobiles
CAM	computer-aided manufacturing
CAN	controller area network
CARD	computer-aided remote driving
CARE	coordination action for robotics in Europe
CASA	Civil Aviation Safety Authority
CASALA	Centre for Affective Solutions for Ambient Living Awareness
CASPER	continuous activity scheduling, planning, execution and replanning
CAT	collision avoidance technology
CAT	computer-aided tomography
CB	computional brain
CB	cluster bomb
CBRNE	chemical, biological, nuclear, radiological, or explosive
CC	compression criterion
CCD	charge-coupled device
CCD	charge-coupled detector
CCI	control command interpreter
CCP	coverage configuration protocol
CCT	conservative congruence transformation
CCW	counterclockwise
CC&D	camouflage, concealment, and deception
CD	collision detection
CD	committee draft
CD	compact disc
CDC	cardinal direction calculus
CDOM	colored dissolved organic matter
CE	computer ethic
CEA	Commissariat à l'Énergie Atomique
CEA	Atomic Energy Commission
CEBOT	cellular robotic system
CEC	Congress on Evolutionary Computation
CEPE	Computer Ethics Philosophical Enquiry
CES	Consumer Electronics Show
CF	carbon fiber
CF	contact formation
CF	climbing fiber
CFD	computational fluid dynamics
CFRP	carbon fiber reinforced prepreg
CFRP	carbon fiber reinforced plastic
CG	center of gravity

CG	computer graphics
CGI	common gateway interface
CHMM	coupled!hidden Markov model
CHMM	continuous hidden Markov model
CIC	computer integrated construction
CIE	International Commission on Illumination
CIP	Children's Innovation Project
CIRCA	cooperative intelligent real-time control architecture
CIS	computer-integrated surgery
CLARAty	coupled layered architecture for robot autonomy
CLEaR	closed-loop execution and recovery
CLIK	closed-loop inverse kinematics
CMAC	cerebellar model articulation controller
CMCs	ceramic matrix composite
CML	concurrent!mapping and localization
CMM	coordinate measurement machine
CMOMMT	cooperative multirobot observation of multiple moving target
CMOS	complementary metal-oxide-semiconductor
CMP	centroid moment pivot
CMTE	Cooperative Research Centre for Mining Technology and Equipment
CMU	Carnegie Mellon University
CNC	computer numerical control
CNN	convolutional neural network
CNP	contract net protocol
CNRS	Centre National de la Recherche Scientifique
CNT	carbon nanotube
COCO	common objects in context
COG	center of gravity
COM	center of mass
COMAN	compliant humanoid platform
COMEST	Commission mondiale d'éthique des connaissances scientifiques et des technologies
COMINT	communication intelligence
CONE	Collaborative Observatory for Nature Environments
COP	center of pressure
CoP	center of pressure
COR	center of rotation
CORBA	common object request broker architecture
CORS	continuous operating reference station
COT	cost!of transport
COTS	commercial off-the-shelf
COV	characteristic output vector
CP	complementarity problem
CP	capture point
CP	continuous path
CP	cerebral palsy
CPG	central pattern generation
CPG	central pattern generator
CPS	cyber physical system
CPSR	Computer Professional for Social Responsibility
CPU	central processing unit
CRASAR	Center for Robot-Assisted Search and Rescue
CRBA	composite-rigid-body algorithm
CRF	conditional random field
CRLB	Cramér–Rao lower bound
CSAIL	Computer Science and Artificial Intelligence Laboratory
CSIRO	Commonwealth Scientific and Industrial Research Organisation
CSMA	carrier-sense multiple-access
CSP	constraint satisfaction problem
CSSF	Canadian Scientific Submersile Facility
CT	computed tomography
CTFM	continuous-transmission frequency modulation
CU	control unit
cv-SLAM	ceiling vision SLAM
CVD	chemical vapor deposition
CVIS	cooperative vehicle infrastructure system
CVT	continuous variable transmission
CW	clockwise
CWS	contact!wrench sum

D

D	distal
D/A	digital-to-analog
DAC	digital analog converter
DARPA	Defense Advanced Research Projects Agency
DARS	distributed!autonomous robotic systems
DBN	dynamic Bayesian network
DBN	deep belief network
DC	disconnected
DC	direct current
DC	dynamic!constrained
DCS	dynamic covariance scaling
DCT	discrete!cosine transform
DD	differentially driven
DDD	dangerous, dirty, and dreary
DDF	decentralized data fusion
DDP	differential dynamic programming
DDS	data distribution service
DEA	differential elastic actuator
DEM	discrete!element method

DFA	design!for assembly
DFRA	distributed field robot architecture
DFT	discrete Fourier transform
DGPS	differential global positioning system
DH	Denavit–Hartenberg
DHMM	discrete!hidden Markov model
DHS	US Department of Homeland Security
DIRA	distributed!robot architecture
DIST	Dipartmento di Informatica Sistemica e Telematica
DL	description logic
DLR	Deutsches Zentrum für Luft- und Raumfahrt
DLR	German Aerospace Center
DMFC	direct methanol fuel cell
DMP	dynamic movement primitive
DNA	deoxyribonucleic acid
DNF	dynamic!neural field
DOD	Department of Defense
DOF	degree of freedom
DOG	difference of Gaussian
DOP	dilution of precision
DPLL	Davis–Putnam algorithm
DPM	deformable part model
DPN	dip-pen nanolithography
DPSK	differential phase shift keying
DRIE	deep reactive ion etching
DSM	dynamic!state machine
DSO	Defense Sciences Office
DSP	digital signal processor
DSRC	dedicated short-range communications
DU	dynamic!unconstrained
DVL	Doppler velocity log
DWA	dynamic window approach
DWDM	dense wave division multiplex
D&D	deactivation and decommissioning

E

e-beam	electron-beam
EAP	electroactive polymer
EBA	energy bounding algorithm
EBA	extrastriate body part area
EBID	electron-beam induced deposition
EC	externally connected
EC	exteroception
ECAI	European Conference on Artificial Intelligence
ECD	eddy current damper
ECEF	earth-centred, earth-fixed
ECER	European Conference on Educational Robotics
ECG	electrocardiogram
ECU	electronics controller unit
EDM	electrical discharge machining
EE	end-effector
EEG	electroencephalography
EGNOS	European Geostationary Navigation Overlay Service
EHC	enhanced horizon control
EHPA	exoskeleton!for human performance augmentation
EKF	extended Kalman filter
ELS	ethical, legal and societal
EM	expectation maximization
emf	electromotive force
EMG	electromyography
EMIB	emotion, motivation and intentional behavior
EMS	electrical!master–slave manipulator
EO	electrooptical
EO	elementary operator
EOA	end of arm
EOD	explosive!ordnance disposal
EP	exploratory procedure
EP	energy packet
EPFL	Ecole Polytechnique Fédérale de Lausanne
EPP	extended!physiological proprioception
EPS	expandable polystyrene
ER	electrorheological
ER	evolutionary!robotics
ERA	European robotic arm
ERP	enterprise resource planning
ERSP	evolution robotics software platform
ES	electrical!stimulation
ESA	European Space Agency
ESC	electronic speed controller
ESL	execution support language
ESM	energy!stability margin
ESM	electric support measure
ETL	Electro-Technical Laboratory
ETS-VII	Engineering Test Satellite VII
EU	European Union
EURON	European Robotics Research Network
EVA	extravehicular activity
EVRYON	evolving morphologies for human–robot symbiotic interaction

F

F5	frontal area 5
FAA	Federal Aviation Administration
FAO	Food and Agriculture Organization
FARS	Fagg–Arbib–Rizzolatti–Sakata
FARSA	framework for autonomous robotics simulation and analysis

fastSLAM	fast simultaneous localization and mapping
FB-EHPA	full-body EHPA
FCU	flight control-unit
FD	friction damper
FDA	US Food and Drug Association
FDM	fused deposition modeling
FE	finite element
FEA	finite element analysis
FEM	finite element method
FESEM	field-emission SEM
FF	fast forward
FFI	Norwegian defense research establishment
FFT	fast Fourier transform
FIFO	first-in first-out
FIRA	Federation of International Robot-soccer Association
FIRRE	family of integrated rapid response equipment
FIRST	For Inspiration and Recognition of Science and Technology
Fl-UAS	flapping wing unmanned aerial system
FLIR	forward!looking infrared
FMBT	feasible minimum buffering time
FMCW	frequency modulation continuous wave
fMRI	functional!magnetic resonance imaging
FMS	flexible!manufacturing system
FNS	functional!neural stimulation
FOA	focus of attention
FOG	fiber-optic gyro
FOPEN	foliage penetration
FOPL	first-order predicate logic
FOV	field of view
FP	fusion primitive
FPGA	field-programmable gate array
FR	false range
FRI	foot rotation indicator
FRP	fiber-reinforced plastics
FRP	fiber-reinforced prepreg
fs	force!sensor
FSA	finite-state acceptor
FSK	frequency shift keying
FSR	force sensing resistor
FSW	friction!stir welding
FTTH	fiber to the home
FW	fixed-wing

G

GA	genetic algorithm
GAPP	goal as parallel programs
GARNICS	gardening with a cognitive system
GAS	global asymptotic stability
GBAS	ground based augmentation system
GCDC	Grand Cooperative Driving Challenge
GCER	Global Conference on Educational Robotics
GCR	goal-contact relaxation
GCS	ground!control station
GDP	gross!domestic product
GenoM	generator of modules
GEO	geostationary Earth orbit
GF	grapple fixture
GFRP	glass-fiber reinforced plastic
GI	gastrointestinal
GIB	GPS intelligent buoys
GICHD	Geneva International Centre for Humanitarian Demining
GID	geometric!intersection data
GIE	generalized-inertia ellipsoid
GIS	geographic information system
GJM	generalized!Jacobian matrix
GLONASS	globalnaya navigatsionnaya sputnikovaya sistema
GLS	global navigation satellite system
GMAW	gas-shielded metal arc welding
GMM	Gaussian mixture model
GMSK	Gaussian minimum shift keying
GMTI	ground!moving target indicator
GNC	guidance, navigation, and control
GO	golgi!tendon organ
GP	Gaussian process
GPCA	generalized principal component analysis
GPRS	general!packet radio service
GPS	global positioning system
GPU	graphics processing unit
GRAB	guaranteed recursive adaptive bounding
GRACE	graduate robot attending conference
GraWoLF	gradient-based win or learn fast
GSD	geon structural description
GSN	gait sensitivity norm
GSP	Gough–Stewart platform
GUI	graphical user interface
GV	ground vehicle
GVA	gross!value added
GZMP	generalized!ZMP

H

H	helical joint
HAL	hybrid!assistive limb
HAMMER	hierarchical!attentive multiple models for execution and recognition
HASY	hand!arm system
HBBA	hybrid behavior-based architecture
HCI	human–computer interaction

HD	high definition
HD	haptic device
HD-SDI	high-definition serial digital interface
HDSL	high data rate digital subscriber line
HE	hand!exoskeleton
HF	hard finger
HF	histogram filter
HFAC	high frequency alternating current
HHMM	hierarchical!hidden Markov model
HIC	head injury criterion
HIII	Hybrid III dummy
HIP	haptic interaction point
HJB	Hamilton–Jacobi–Bellman
HJI	Hamilton–Jacobi–Isaac
HMCS	human–machine!cooperative system
HMD	head-mounted display
HMDS	hexamethyldisilazane
HMI	human–machine!interaction
HMI	human–machine!interface
HMM	hidden Markov model
HO	human operator
HOG	histogram of oriented gradient
HOG	histogram of oriented features
HPC	high-performance computing
HRI	human–robot interaction
HRI/OS	HRI operating system
HRP	humanoid robotics project
HRR	high resolution radar
HRTEM	high-resolution transmission electron microscope
HSGR	high safety goal
HST	Hubble space telescope
HSTAMIDS	handheld standoff mine detection system
HSWR	high safety wide region
HTAS	high tech automotive system
HTML	hypertext markup language
HTN	hierarchical task network
HTTP	hypertext transmission protocol
HW/SW	hardware/software

I

I/O	input/output
I3CON	industrialized, integrated, intelligent, construction
IA	interval algebra
IA	instantaneous!allocation
IAA	interaction!agent
IAB	International Association of Bioethics
IACAP	International Association for Computing and Philosophy
IAD	interaural amplitude difference
IAD	intelligent!assisting device
IARC	International Aerial Robotics Competition
IAS	intelligent!autonomous system
IBVS	image-based visual servo control
IC	integrated chip
IC	integrated circuit
ICA	independent!component analysis
ICAPS	International Conference on Automated Planning and Scheduling
ICAR	International Conference on Advanced Robotics
ICBL	International Campaign to Ban Landmines
ICC	instantaneous center of curvature
ICE	internet communications engine
ICP	iterative closest point
ICR	instantaneous center of rotation
ICRA	International Conference on Robotics and Automation
ICT	information!and communication technology
ID	inside diameter
ID	identifier
IDE	integrated!development environment
IDL	interface definition language
IE	information!ethics
IED	improvised explosive device
IEEE	Institute of Electrical and Electronics Engineers
IEKF	iterated extended Kalman filter
IETF	internet!engineering task force
IFA	Internationale Funk Ausstellung
IFOG	interferometric fiber-optic gyro
IFR	International Federation of Robotics
IFREMER	Institut français de recherche pour l'exploitation de la mer
IFRR	International Foundation of Robotics Research
IFSAR	interferometric SAR
IHIP	intermediate haptic interaction point
IIR	infinite impulse response
IIS	Internet Information Services
IIT	Istituto Italiano di Tecnologia
IJCAI	International Joint Conference on Artificial Intelligence
IK	inverse kinematics
ILLS	instrumented logical sensor system
ILO	International Labor Organization
ILQR	iterative linear quadratic regulator
IM	injury measure
IMAV	International Micro Air Vehicles
IMTS	intelligent!multimode transit system
IMU	inertial measurement unit
INS	inertia navigation system

INS	inertial navigation system
IO	input output
IO	inferior olive
IOSS	input-output-to-state stability
IP	internet protocol
IP	interphalangeal
IPA	Institute for Manufacturing Engineering and Automation
IPC	interprocess communication
IPC	international AI planning competition
IPMC	ionic polymer-metal composite
IPR	intellectual property right
IR	infrared
IRB	Institutional Review Board
IREDES	International Rock Excavation Data Exchange Standard
IRL	in real life
IRL	inverse!reinforcement learning
IRLS	iteratively reweighted least square
IRNSS	Indian regional navigational satellite system
IROS	Intelligent Robots and Systems
IS	importance sampling
ISA	industrial standard architecture
ISA	international standard atmosphere
ISAR	inverse SAR
ISDN	integrated services digital network
ISE	international submarine engineering
ISER	International Symposium on Experimental Robotics
ISM	implicit shape model
ISO	International Organization for Standardization
ISP	Internet service provider
ISR	intelligence, surveillance and reconnaissance
ISRR	International Symposium of Robotics Research
ISS	international space station
ISS	input-to-state stability
IST	Instituto Superior Técnico
IST	Information Society Technologies
IT	intrinsic tactile
IT	information!technology
IT	inferotemporal cortex
ITD	interaural time difference
IU	interaction!unit
IV	instrumental variable
IvP	interval programming
IWS	intelligent!wheelchair system
IxTeT	indexed time table

J

JAEA	Japan Atomic Energy Agency
JAMSTEC	Japan Agency for Marine-Earth Science and Technology
JAMSTEC	Japan Marine Science and Technology Center
JAUS	joint architecture for unmanned systems
JAXA	Japan Aerospace Exploration Agency
JDL	joint directors of laboratories
JEM	Japan Experiment Module
JEMRMS	Japanese experiment module remote manipulator system
JHU	Johns Hopkins University
JND	just noticeable difference
JPL	Jet Propulsion Laboratory
JPS	jigsaw positioning system
JSC	Johnson Space Center
JSIM	joint-space inertia matrix
JSP	Java server pages

K

KAIST	Korea Advanced Institute of Science and Technology
KERS	kinetic energy recovery system
KIPR	KISS Institute for Practical Robotics
KLD	Kullback–Leibler divergence
KNN	k-nearest neighbor
KR	knowledge representation
KRISO	Korea Research Institute of Ships and Ocean Engineering

L

L/D	lift-to-drag
LAAS	Laboratory for Analysis and Architecture of Systems
LADAR	laser radar
LAGR	learning!applied to ground robots
LARC	Lie algebra rank condition
LARS	Laparoscopic Assistant Robotic System
LASC	Longwall Automation Steering Committee
LBL	long-baseline system
LCAUV	long-range cruising AUV
LCC	life-cycle-costing
LCD	liquid-crystal display
LCM	light-weight communications and marshalling
LCP	linear complementarity problem
LCSP	linear constraint satisfaction program
LDA	latent Dirichlet allocation
LED	light-emitting diode
LENAR	lower!extremity nonanthropomorphic robot

LEO	low!Earth orbit
LEV	leading edge vortex
LfD	learning!from demonstration
LGN	lateral!geniculate nucleus
LHD	load!haul-dump
LIDAR	light detection and ranging
LIGA	Lithographie, Galvanoumformung, Abformung
LIP	linear inverted pendulum
LIP	lateral!intraparietal sulcus
LiPo	lithium polymer
LLC	locality constrained linear coding
LMedS	least median of squares
LMS	laser measurement system
LOG	Laplacian of Gaussian
LOPES	lower!extremity powered exoskeleton
LOS	line-of-sight
LP	linear program
LQG	linear quadratic Gaussian
LQR	linear quadratic regulator
LSS	logical sensor system
LSVM	latent support vector machine
LtA	lighter-than-air
LtA-UAS	lighter-than-air system
LTL	linear temporal logic
LVDT	linear variable differential transformer
LWR	light-weight robot

M

MACA	Afghanistan Mine Action Center
MACCEPA	mechanically adjustable compliance and controllable equilibrium position actuator
MAP	maximum a posteriori
MARS	multiappendage robotic system
MARUM	Zentrum für Marine Umweltwissenschaften
MASE	Marine Autonomous Systems Engineering
MASINT	measurement!and signatures intelligence
MAV	micro aerial vehicles
MAZE	Micro robot maze contest
MBA	motivated behavioral architecture
MBARI	Monterey Bay Aquarium Research Institute
MBE	molecular-beam epitaxy
MBS	mobile!base system
MC	Monte Carlo
MCFC	molten carbonate fuel cell
MCP	magazining, cleaning, plotting
MCP	metacarpophalangeal
MCS	mission!control system
MDARS	mobile!detection assessment and response system
MDL	minimum description length
MDP	Markov decision process
ME	mechanical!engeneering
MEG	magnetoencephalography
MEL	Mechanical Engineering Laboratory
MEMS	microelectromechanical system
MEP	motor!evoked potential
MESSIE	multi expert system for scene interpretation and evaluation
MESUR	Mars environmental survey
MF	mossy fiber
MFI	micromechanical flying insect
MFSK	multiple FSK
MHS	International Symposium on Micro Mechatronics and Human Science
MHT	multihypothesis tracking
MIA	mechanical impedance adjuster
MIME	mirror!image movement enhancer
MIMICS	multimodal immersive motion rehabilitation with interactive cognitive system
MIMO	multiple-input–multiple-output
MIP	medial intraparietal sulcus
MIPS	microprocessor without interlocked pipeline stages
MIR	mode identification and recovery
MIRO	middleware for robot
MIS	minimally invasive surgery
MIT	Massachusetts Institute of Technology
MITI	Ministry of International Trade and Industry
MKL	multiple kernel learning
ML	machine!learning
MLE	maximum likelihood estimate
MLR	mesencephalic locomotor region
MLS	multilevel surface map
MMC	metal matrix composite
MMMS	multiple master multiple-slave
MMSAE	multiple model switching adaptive estimator
MMSE	minimum mean-square error
MMSS	multiple master single-slave
MNS	mirror!neuron system
MOCVD	metallo-organic chemical vapor deposition
MOMR	multiple operator multiple robot
MOOS	mission oriented operating suite
MOOS	motion-oriented operating system
MORO	mobile robot
MOSR	multiple operator single robot
MP	moving plate
MPC	model predictive control

MPF	manifold particle filter
MPFIM	multiple!paired forward-inverse model
MPHE	multiphalanx hand exoskeleton
MPSK	Mary phase shift keying
MQAM	Mary quadrature amplitude modulation
MR	magnetorheological
MR	multiple reflection
MR	multirobot!task
MRAC	model reference adaptive control
MRDS	Microsoft robotics developers studio
MRF	Markov random field
MRHA	multiple!resource host architecture
MRI	magnetic resonance imaging
MRSR	Mars rover sample return
MRTA	multirobot!task allocation
MSAS	multifunctional satellite augmentation system
MSER	maximally stable extremal region
MSHA	US Mine Safety and Health Administration
MSK	minimum shift keying
MSL	middle-size league
MSM	master–slave!manipulator
MST	microsystem technology
MT	momentum theory
MT	multitask
MT	medial temporal area
MTBF	mean time between failures
MTI	moving target indicator
MVERT	move value estimation for robot teams
MWNT	multiwalled carbon nanotube

N

N&G	nursery and greenhouse
NAP	nonaccidental property
NASA	National Aeronautics and Space Agency
NASDA	National Space Development Agency of Japan
NASREM	NASA/NBS standard reference model
NBS	National Bureau of Standards
NC	numerical control
ND	nearness diagram navigation
NDDS	network data distribution service
NDGPS	nationwide different GPS system
NDI	nonlinear dynamic inversion
NDT	normal distributions transform
NEMO	network!mobility
NEMS	nanoelectromechanical system
NEO	neodymium
NERVE	New England Robotics Validation and Experimentation
NESM	normalized ESM
NIDRR	National Institute on Disability and Rehabilitation Research
NiMH	nickel metal hydride battery
NIMS	networked!infomechanical systems
NIOSH	United States National Institute for Occupational Safety and Health
NIRS	near infrared spectroscopy
NIST	National Institute of Standards and Technology
NLIS	national livestock identification scheme
NLP	nonlinear!programming problem
NMEA	National Marine Electronics Association
NMF	nonnegative matrix factorization
NMMI	natural machine motion initiative
NMR	nuclear!magnetic resonance
NN	neural network
NOAA	National Oceanic and Atmospheric Administration
NOAH	navigation!and obstacle avoidance help
NOC	National Oceanography Centre
NOTES	natural!orifice transluminal surgery
NPO	nonprofit organization
NPS	Naval Postgraduate School
NQE	national qualifying event
NRI	national robotics initiative
NRM	nanorobotic manipulator
NRTK	network real-time kinematic
NTPP	nontangential proper part
NTSC	National Television System Committee
NURBS	nonuniform rational B-spline
NUWC	Naval Undersea Warfare Center Division Newport
NZDF	New Zealand Defence Force

O

OAA	open!agent architecture
OASIS	onboard autonomous science investigation system
OAT	optimal arbitrary time-delay
OBU	on board unit
OC	optimal control
OCPP	optimal!coverage path planning
OCR	OC robotics
OCT	optical!coherence tomography
OCU	operator control unit
OD	outer diameter
ODE	ordinary differential equation
ODE	open dynamics engine
ODI	ordinary differential inclusion
OECD	Organization for Economic Cooperation and Development
OKR	optokinetic response
OLP	offline programming

OM	optical microscope
OM	occupancy map
ONR	US Office of Naval Research
OOF	out of field
OOTL	human!out of the loop control
OPRoS	open platform for robotic service
ORCA	open robot control architecture
ORCCAD	open robot controller computer aided design
ORI	open!roboethics initiative
ORM	obstacle restriction method
OROCOS	open robot control software
ORU	orbital replacement unit
OS	operating system
OSC	operational-space control
OSIM	operational-space inertia matrix
OSU	Ohio State University
OTH	over-the-horizon
OUR-K	ontology based unified robot knowledge
OWL	web ontology language
OxIM	Oxford intelligent machine

P

P	prismatic joint
P&O	prosthetics!and orthotic
PA	point algebra
PACT	perception!for action control theory
PAD	pleasure arousal dominance
PAFC	phosphoric acid fuel cell
PAM	pneumatic artificial muscle
PaMini	pattern-based mixed-initiative
PANi	polyaniline
PANTOMEC	pantograph mechanism driven
PAPA	privacy, accuracy, intellectual property, and access
PAS	pseudo-amplitude scan
PAT	proximity!awareness technology
PB	parametric!bias
PbD	programming!by demonstration
PBVS	pose-based visual servo control
PC	polycarbonate
PC	personal computer
PC	principal contact
PC	passivity controller
PC	proprioception
PC	Purkinje cell
PCA	principal component analysis
PCI	peripheral component interconnect
PCIe	peripheral component interconnect express
PCL	point cloud library
PCM	programmable!construction machine
PD	proportional–derivative
PDE	partial differential equation
PDGF	power!data grapple fixture
PDMS	polydimethylsiloxane
PDOP	positional dilution of precision
PDT	proximity!detection technology
PEAS	probing environment and adaptive sleeping protocol
PEFC	polymer electrolyte fuel cell
PEMFC	proton exchange membrane fuel cell
PerceptOR	perception!for off-road robotics
PET	positron emission tomography
PF	particle filter
PF	parallel!fiber
PFC	prefrontal cortex
PFH	point feature histogram
PFM	potential field method
PGM	probabilistic graphical model
PGRL	policy gradient!reinforcement learning
pHRI	physical!human–robot interaction
PI	policy iteration
PI	possible!injury
PI	propositional integral
PI	proportional–integral
PIC	programmable!intelligent computer
PID	proportional–integral–derivative
PIT	posterior!inferotemporal cortex
PKM	parallel kinematics machine
PKM	parallel kinematic machine
PL	power loading
PLC	programmable!logic controller
PLD	programmable!logic device
PLEXIL	plan execution interchange language
PLSA	probabilistic latent semantic analysis
PLZT	lead lanthanum zirconate titanate
PM	permanent magnet
PMC	polymer matrix composite
PMMA	polymethyl methacrylate
PneuNet	pneumatic network
PnP	prespective-n-point
PNT	Petri net transducer
PO	partially overlapping
PO	passivity observer
POE	local product-of-exponential
POI	point!of interest
POM	polyoxymethylene
POMDP	partially observable Markov decision process
POP	partial-order planning
PPS	precise positioning system
PPy	polypyrrole
PR	positive photoresist
PRM	probabilistic roadmap
PRM	probabilistic roadmap method
PRN	pseudo-random noise

PRoP	personal roving presence
ProVAR	professional vocational assistive robot
PRS	procedural reasoning system
PS	power source
PSD	position sensing device
PSD	position-sensitive-device
PSK	phase shift keying
PSPM	passive set-position modulation
PTAM	parallel tracking and mapping
PTU	pan–tilt unit
PUMA	programmable!universal machine for assembly
PVA	position, velocity, and attitude
PVC	polyvinyl chloride
PVD	physical vapor deposition
PVDF	polyvinylidene fluoride
PWM	pulse-width modulation
PwoF	point-contact-without-friction
PZT	lead zirconate titanate

Q

QAM	quadrature amplitude modulation
QD	quantum dot
QID	qualifier, inspection and demonstration
QOLT	quality!of life technology
QOS	quality of service
QP	quadratic programming
QPSK	quadrature phase shift keying
QRIO	quest for curiosity
QSC	quasistatic!constrained
QT	quasistatic telerobotics
QZSS	quasi-zenith satellite system

R

R	revolute joint
R.U.R.	Rossum's Universal Robots
RA	rectangle algebra
RAC	Robotics and Automation Council
RAIM	receiver autonomous integrity monitor
RALF	robotic arm large and flexible
RALPH	rapidly adapting lane position handler
RAM	random!access memory
RAMS	robot-assisted microsurgery
RAMS	random!access memory system
RANSAC	random sample consensus
RAP	reactive action package
RAS	Robotics and Automation Society
RBC	recognition!by-component
RBF	radial!basis function network
RBF	radial!basis function
RBT	robot!experiment
RC	radio control
RC	robot!controller
RCC	region connection calculus
RCC	remote center of compliance
RCM	remote!center of motion
RCP	rover chassis prototype
RCR	responsible conduct of research
RCS	real-time control system
RCS	rig control system
RDT	rapidly exploring dense tree
RECS	robotic!explosive charging system
REINFORCE	reward increment = nonnegative factor × offset reinforcement × characteristic eligibility
RERC	Rehabilitation Engineering Research Center
RF	radio frequency
RFID	radio frequency identification
RG	rate gyro
RGB-D	color camera with depth
RGB-D	red green blue distance
RGB-D	red–green–blue–depth
RHIB	rigid!hull inflatable boat
RIE	reactive-ion etching
RIG	rate-integrating gyro
RISC	reduced instruction set computer
RL	reinforcement learning
RLG	ring laser gyroscope
RLG	random loop generator
RMC	resolved momentum control
RMDP	relational Markov decision processes
RMMS	reconfigurable modular manipulator system
RMS	root mean square
RNDF	route network definition file
RNEA	recursive Newton–Euler algorithm
RNN	recurrent neural network
RNNPB	recurrent neural network with parametric bias
RNS	reaction!null-space
ROC	receiver operating curve
ROC	remote!operations centre
ROCCO	robot!construction system for computer integrated construction
ROD	robot!oriented design
ROKVISS	robotics component verification on ISS
ROKVISS	robotics!components verification on the ISS
ROM	run-of-mine
ROM	read-only memory
ROMAN	Robot and Human Interactive Communication
ROS	robot operating system
ROV	remotely operated vehicle
ROV	remotely!operated underwater vehicle

RP		rapid prototyping
RP-VITA		remote presence virtual + independent telemedicine assistant
RPC		remote procedure call
RPI		Rensselaer Polytechnic Institute
RPS		room positioning system
RRSD		Robotics and Remote Systems Division
RRT		rapidly exploring random tree
RS		Reeds and Shepp
RSJ		Robotics Society of Japan
RSS		Robotics Science and Systems
RSTA		reconnaissance, surveillance, and target acquisition
RSU		road!side unit
RT		real-time
RT		room temperature
RT		reaction!time
RTCMS	C104	Radio Technical Commission for Maritime Services Special Committee 104
RTD		resistance temperature devices
RTI		real-time innovation
RTK		real-time kinematics
rTMS		repetitive!TMS
RTS		real-time system
RTT		real-time toolkit
RV		rotary vector
RVD		rendezvous/docking
RW		rotary-wing
RWI		real-world interface
RWS		robotic workstation
R&D		research and development
R&D		research and development

S

SA	simulated annealing
SA	selective availability
SAFMC	Singapore Amazing Flying Machine Competition
SAI	simulation!and active interfaces
SAM	smoothing and mapping
SAN	semiautonomous navigation
SAR	synthetic aperture radar
SAR	socially assistive robotics
SARSA	state action-reward-state-action
SAS	synthetic aperture sonar
SAS	stability augmentation system
SAT	International Conference on Theory and Applications of Satisfiability Testing
SBAS	satellite-based augmentation system
SBL	short baseline
SBSS	space based space surveillance
SC	sparse coding
SCARA	selective compliance assembly robot arm
SCI	spinal cord!injury
sci-fi	science fiction
SCM	smart composite microstructures
SCM	soil!contact model
SD	standard deviation
SDK	standard development kit
SDK	software development kit
SDM	shape deposition manufacturing
SDR	software!for distributed robotics
SDV	spatial dynamic voting
SEA	series elastic actuator
SEE	standard!end effector
SELF	sensorized environment for life
SEM	scanning electron microscope
SET	single electron transistor
SF	soft finger
SFM	structure from motion
SFX	sensor fusion effect
SGAS	semiglobal asymptotic stability
SGD	stochastic gradient descent
SGM	semiglobal!matching
SGUUB	semiglobal uniform ultimate boundedness
SIFT	scale-invariant feature transform
SIGINT	signal!intelligence
SIR	sampling importance resampling
SISO	single input single-output
SKM	serial!kinematic machines
SLA	stereolithography
SLAM	simultaneous localization and mapping
SLICE	specification language for ICE
SLIP	spring loaded inverted pendulum
SLRV	surveyor lunar rover vehicle
SLS	selective laser sintering
SM	static margin
SMA	shape memory alloy
SMAS	solid material assembly system
SMC	sequential Monte Carlo
SME	small!and medium enterprises
SMMS	single-master multiple-slave
SMP	shape memory polymer
SMS	short message service
SMSS	single-master single-slave
SMT	satisfiabiliy modulo theory
SMU	safe!motion unit
SNAME	society of naval architects and marine engineer
SNOM	scanning near-field optical microscopy
SNR	signal-to-noise ratio
SNS	spallation neutron source
SOFC	solid oxide fuel cell
SOI	silicon-on-insulator

SOMA	stream-oriented messaging architecture
SOMR	single operator multiple robot
SOS	save our souls
SOSR	single operator single robot
SPA	sense-plan-act
SPaT	signal!phase and timing
SPAWAR	Space and Naval Warfare Systems Center
SPC	self-posture changeability
SPDM	special purpose dexterous manipulator
SPHE	single-phalanx hand exoskeleton
SPL	single!port laparoscopy
SPL	standard!platform
SPM	scanning probe microscope
SPM	spatial pyramid matching
SPMS	shearer position measurement system
SPS	standard position system
SPU	spherical, prismatic, universal
SQP	sequential!quadratic programming
SR	single-robot task
SRA	spatial!reasoning agent
SRCC	spatial remote center compliance
SRI	Stanford Research Institute
SRMS	shuttle remote manipulator system
SSA	sparse surface adjustment
SSC	smart soft composite
SSL	small-size league
SSRMS	space!station remote manipulator system
ST	single-task
STEM	science, technology, engineering and mathematics
STM	scanning tunneling microscope
STP	simple temporal problem
STriDER	self-excited tripodal dynamic experimental robot
STS	superior!temporal sulcus
SUGV	small!unmanned ground vehicle
SUN	scene understanding
SURF	robust feature
SVD	singular value decomposition
SVM	support vector machine
SVR	support vector regression
SWNT	single-walled carbon nanotube
SWRI	Southwest Research Institute

T

T-REX	teleo-reactive executive
TA	time-extended assignment
TAL	temporal action logic
TAM	taxon!affordance model
TAP	test action pair
TBG	time-base generator
TC	technical committee
TCFFHRC	Trinity College's Firefighting Robot Contest
TCP	transfer control protocol
TCP	tool center point
TCP	transmission control protocol
TCSP	temporal constraint satisfaction problem
tDCS	transcranial!direct current stimulation
TDL	task description language
TDT	tension-differential type
TECS	total energy control system
TEM	transmission electron microscope
tEODor	telerob explosive ordnance disposal and observation robot
TFP	total!factor productivity
TL	temporal logic
TMM	transfer matrix method
TMS	tether management system
TMS	transcranial!magnetic stimulation
TNT	trinitrotoluene
TOA	time of arrival
TOF	time-of-flight
ToF	time-of-flight
TORO	torque!controlled humanoid robot
TPaD	tactile pattern display
TPBVP	two-point boundary value problem
TPP	tangential proper part
TRC	Transportation Research Center
TRIC	task space retrieval using inverse optimal control
TS	technical!specification
TSEE	teleoperated!small emplacement excavator
TSP	telesensor programming
TTC	time-to-collision
TUM	Technical University of Munich
TV	television

U

U	universal joint
UAS	unmanned aircraft system
UAS	unmanned!aerial system
UAV	unmanned aerial vehicle
UAV	fusing air vehicle
UAV	fielded unmanned aerial vehicle
UB	University of Bologna
UBC	University of British Columbia
UBM	Universität der Bundeswehr Munich
UCLA	University of California, Los Angeles
UCO	uniformly completely observable
UDP	user datagram protocol
UDP	user data protocol
UGV	unmanned!ground vehicle
UHD	ultrahigh definition

UHF	ultrahigh frequency
UHV	ultrahigh-vacuum
UKF	unscented Kalman filter
ULE	upper!limb exoskeleton
UML	unified modeling language
UMV	unmanned marine vehicle
UNESCO	United Nations Educational, Scientific and Cultural Organization
UPnP	universal plug and play
URC	Ubiquitous Robotic Companion
URL	uniform resource locator
USAR	urban!search and rescue
USB	universal!serial bus
USBL	ultrashort baseline
USBL	ultrashort-baseline
USC	University of Southern California
USV	unmanned!surface vehicle
UTC	universal coordinated time
UUB	uniform ultimate boundedness
UUV	unmanned underwater vehicle
UV	ultraviolet
UVMS	underwater vehicle!manipulator system
UWB	ultrawide band
UXO	unexploded ordnance

V

V2V	vehicle-to-vehicle
VAS	visual!analog scale
VCR	video!cassette recorder
vdW	van der Waals
VE	virtual environment
VFH	vector field histogram
VHF	very high frequency
VI	value iteration
VIA	variable impedance actuator
VIP	ventral intraparietal
VM	virtual!manipulator
VME	Versa Module Europa
VO	virtual object
VO	velocity obstacle
VOC	visual object class
VOR	vestibular-ocular reflex
VR	variable reluctance
VRML	virtual reality modeling language
VS	visual servo
VS-Joint	variable stiffness joint
VSA	variable stiffness actuator
VTOL	vertical take-off and landing

W

W3C	WWW consortium
WAAS	wide-area augmentation system
WABIAN	Waseda bipedal humanoid
WABOT	Waseda robot
WAM	whole-arm manipulator
WAN	wide-area network
WASP	wireless!ad-hoc system for positioning
WAVE	wireless!access in vehicular environments
WCF	worst-case factor
WCR	worst-case range
WDVI	weighted!difference vegetation index
WG	world!graph
WGS	World Geodetic System
WHOI	Woods Hole Oceanographic Institution
WML	wireless markup language
WMR	wheeled mobile robot
WSN	wireless!sensor network
WTA	winner-take-all
WTC	World Trade Center
WWW	world wide web

X

XCOM	extrapolated center of mass
XHTML	extensible hyper text markup language
XML	extensible markup language
xUCE	urban!challenge event

Y

YARP	yet another robot platform

Z

ZMP	zero moment point
ZOH	zero order hold
ZP	zona pellucida

Multimedia Contents

1. Robotics and the Handbook

Bruno Siciliano, Oussama Khatib

Robots! Robots on Mars and in oceans, in hospitals and homes, in factories and schools; robots fighting fires, making goods and products, saving time and lives. Robots today are making a considerable impact on many aspects of modern life, from industrial manufacturing to healthcare, transportation, and exploration of the deep space and sea. Tomorrow, robots will be as pervasive and personal as today's personal computers. This chapter retraces the evolution of this fascinating field from the ancient to the modern times through a number of milestones: from the first automated mechanical artifact (1400 BC) through the establishment of the robot concept in the 1920s, the realization of the first industrial robots in the 1960s, the definition of robotics science and the birth of an active research community in the 1980s, and the expansion towards the challenges of the human world of the twenty-first century. Robotics in its long journey has inspired this handbook which is organized in three layers: the foundations of robotics science; the consolidated methodologies and technologies of robot design, sensing and perception, manipulation and interfaces, mobile and distributed robotics; the advanced applications of field and service robotics, as well as of human-centered and life-like robotics.

1.1 **A Brief History of Robotics** 1
1.2 **The Robotics Community** 3
1.3 **This Handbook** 4
Video-References 5

1.1 A Brief History of Robotics

The dream to create machines that are skilled and intelligent has been part of humanity from the beginning of time. This dream is now becoming part of our world's striking reality. Since the early civilizations, one of man's greatest ambitions has been to create artifacts in their image. The legend of the Titan Prometheus, who molded humankind from clay, or that of the giant Talus, the bronze slave forged by Hephaestus (3500 BC), testify to this quest in Greek mythology. The Egyptians' oracle statues hiding priests inside (2500 BC) were perhaps the precursor of our modern thinking machines. The clepsydra water clock introduced by the Babylonians (1400 BC) was one of the first automated mechanical artifacts. In the following centuries, human creativity has given rise to a host of devices such as the automaton theatre of Hero of Alexandria (100 AD), the hydro-powered water-raising and humanoid machines of Al-Jazari (1200), and Leonardo da Vinci's numerous ingenious designs (1500). The development of automata continued to flourish in the eighteen century both in Europe and Asia, with creations such as Jacquet-Droz's family of androids (drawer, musician and writer) and the *karakuri-ningyo* mechanical dolls (tea server and archer).

The robot *concept* was clearly established by those many creative historical realizations. Nonetheless, the emergence of the *physical* robot had to await the advent of its underlying technologies during the course of the twentieth century. In 1920, the term robot – derived from *robota* which means subordinate labour in Slav languages – was first introduced by the Czech playwright Karel Čapek in his play *Rossum's Universal Robots* (R.U.R.). In 1940, the ethics of the interaction

between robots and humans was envisioned to be governed by the well-known three fundamental laws of Isaac Asimov, the Russian science-fiction writer in his novel *Runaround*.

The middle of the twentieth century brought the first explorations of the connection between human intelligence and machines, marking the beginning of an era of fertile research in the field of artificial intelligence (AI). Around that time, the first robots were realized. They benefited from advances in the different technologies of mechanics, controls, computers and electronics. As always, new designs motivate new research and discoveries, which, in turn, lead to enhanced solutions and thus to novel concepts. This virtuous circle over time produced that knowledge and understanding which gave birth to the field of *robotics*, properly referred to as: the science and technology of robots.

The early robots built in the 1960s stemmed from the confluence of two technologies: numerical control machines for precise manufacturing, and teleoperators for remote radioactive material handling. These master-slave arms were designed to duplicate one-to-one the mechanics of the human arm, and had rudimental control and little perception about the environment. Then, during the mid-to-late twentieth century, the development of integrated circuits, digital computers and miniaturized components enabled computer-controlled robots to be designed and programmed. These robots, termed industrial robots, became essential components in the automation of flexible manufacturing systems in the late 1970s. Further to their wide application in the automotive industry, industrial robots were successfully employed in general industry, such as the metal products, the chemical, the electronics and the food industries. More recently, robots have found new applications outside the factories, in areas such as cleaning, search and rescue, underwater, space, and medical applications.

In the 1980s robotics was defined as the science which studies the *intelligent connection between perception and action*. With reference to this definition, the action of a robotic system is entrusted to a *locomotion* apparatus to move in the environment (wheels, crawlers, legs, propellers) and/or to a *manipulation* apparatus to operate on objects present in the environment (arms, end effectors, artificial hands), where suitable *actuators* animate the mechanical components of the robot. The perception is extracted from the *sensors* providing information on the state of the robot (position and speed) and its surrounding environment (force and tactile, range and vision). The intelligent connection is entrusted to a *programming, planning and control* architecture which relies on the perception and available *models* of the robot and environment and exploits learning and skill acquisition.

In the 1990s research was boosted by the need to resort to robots to address human safety in hazardous environments (*field robotics*), or to enhance the human operator ability and reduce his/her fatigue (*human augmentation*), or else by the desire to develop products with wide potential markets aimed at improving the quality of life (*service robotics*). A common denominator of such application scenarios was the need to operate in a scarcely structured environment which ultimately requires increased abilities and a higher degree of *autonomy*.

The first video attached to this chapter *Robots – A 50 Year Journey* by Oussama Khatib (2000) shows the development of robotics through the first five decades (VIDEO 805).

By the dawn of the new millennium, robotics has undergone a major transformation in scope and dimensions. This expansion has been brought about by the maturity of the field and the advances in its related technologies. From a largely dominant industrial focus, robotics has been rapidly expanding into the challenges of the human world (*human-centered and life-like robotics*). The new generation of robots is expected to safely and dependably co-habitat with humans in homes, workplaces, and communities, providing support in services, entertainment, education, healthcare, manufacturing, and assistance.

Beyond its impact on physical robots, the body of knowledge robotics has produced is revealing a much wider range of applications reaching across diverse research areas and scientific disciplines, such as: biomechanics, haptics, neurosciences, virtual simulation, animation, surgery, and sensor networks among others. In return, the challenges of the new emerging areas are proving an abundant source of stimulation and insights for the field of robotics. It is indeed at the intersection of disciplines that the most striking advances are expected to happen.

The second video attached to this chapter *Robots – The Journey Continues* by Bruno Siciliano, Oussama Khatib and Torsten Kröger (2015) shows the intensive and vibrating evolution of robotics through the last fifteen years (VIDEO 812).

1.2 The Robotics Community

The dissemination of research results and findings in archival publications and conference presentations has played an important role in the advancement of robotics in the past three decades. The extent of scientific activities in robotics has led to the establishment of professional societies and research networks devoted to the field, which have ultimately contributed to the building of an international robotics community.

In the early 1980s, robotics was entering a new era in its development following the earlier successes in industrial applications. With much energy and excitement, the pioneers of robotics launched a number of initiatives, which came to shape what became our field. The first robotics textbook by Richard Paul *Robot Manipulators: Mathematics, Programming, and Control* appeared in 1981 (MIT Press). A year later, Mike Brady, Richard Paul, and their colleagues established the first journal entirely devoted to robotics, the International Journal of Robotics Research (IJRR), which was followed by the organization in 1983 of the first International Symposium of Robotics Research (ISRR). The intent was to foster the notion of robotics as a distinct scientific field and to create a broad community of robotics. It was in the pursuit of these aims that the International Foundation of Robotics Research (IFRR) was later formally founded in 1986 with the aim of promoting the development of robotics as a science, establishing the theoretical foundations and technology basis for its ever-expanding applications, with emphasis on its potential role to benefit humans.

The year 1984 brought a host of other major developments in robotics with the establishment of the IEEE Robotics and Automation Council (RAC), led by George Saridis, the organization of the first IEEE Conference on Robotics (John Jarvis and Richard Paul) – which became the IEEE International Conference on Robotics and Automation (ICRA) the year after, the launching of the IEEE Journal of Robotics and Automation (George Bekey), and RAC Newsletter (Wesley Snyder). The transformation of the council into the IEEE Robotics and Automation Society (RAS) was formally accomplished in 1989. With the birth of RAS, the IEEE Journal of Robotics and Automation became the IEEE Transactions on Robotics and Automation – which split into the IEEE Transactions on Robotics and the IEEE Transactions on Automation Science and Engineering later in 2004; then, the RAS Newsletter became the IEEE Robotics and Automation Magazine in 1994.

Over the following years, a number of other activities were launched. New conferences such as IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 1983), International Conference on Advanced Robotics (ICAR 1983), International Symposium on Experimental Robotics (ISER 1989), Robotics: Science and Systems (RSS 2005), as well as new journals such as Robotica (1983), Journal of Robotic Systems (1984, subsequently Journal of Field Robotics 2006), Robotics and Autonomous Systems (1985), Advanced Robotics (1986), Journal of Intelligent and Robotic Systems (1988), Autonomous Robots (1994), contributed to a further substantial growth of the robotics community.

The year 1991 marked an important novelty into the ICRA flagship conference series with the creation of video proceedings, introduced earlier by Oussama Khatib and Vincent Hayward in ISER 1989. Since then video proceedings attained an important role for dissemination of research work in our robotics community. This media has become an integral part of most archival publications in recent years.

On the other hand, the introduction of graduate programs in robotics in many academic institutions around the world in the last two decades is a clear illustration of the level of maturity reached by robotics as a scientific field. Further to Paul's seminal book, other textbooks were published between the middle 1980s and middle 1990s, such as *Introduction to Robotics: Mechanics and Control* by John Craig (1985), *Robot Analysis and Control* by Harry Asada and Jean-Jacques Slotine (1986), *Robotics: Control, Sensing, Vision, and Intelligence* by King Sun Fu, Rafael González and George Lee (1987), *Robot Dynamics and Control* by Mark Spong and M. Vidyasagar (1989), *Foundation of Robotics* by Tsuneo Yoshikawa (1990), *A Mathematical Introduction to Robotic Manipulation* by Richard Murray, Zexiang Li and Shankar Sastry (1994), and *Modelling and Control of Robot Manipulators* by Lorenzo Sciavicco and Bruno Siciliano (1995).

Engagement with students is essential for the future of the robotics field. Through a joint initiative between IEEE RAS and IFRR, the School of Robotics Science was founded in 2004. Summer schools have rapidly expanded in scope and frequency, covering a wide range of robotics research areas. Such programs are providing students and young researchers with an educational opportunity of the highest scientific level, enabling them to feed and grow their potential to become the top robotics scientists in our community.

Both IEEE RAS and IFRR have constantly promoted research, education, dissemination, and industrial realizations through their own forums and through collaborations with many other organizations

in the robotics community world-wide, including European Robotics Research Network (EURON) – which has recently merged into the European Association of Robotics (euRobotics), International Federation of Robotics (IFR), and Robotics Society of Japan (RSJ).

As our field continues to expand and embrace the challenges of operation in the human world, robotics is revealing a much wider range of applications and involving diverse scientific disciplines such as biomechanics, neuroscience, haptics, animation, surgery, and sensor networks among others. Today, new communities of researchers and developers are forming, with growing connections to the core of robotics research. A strategic goal for our community is one of outreach and scientific cooperation across all regions and organizations. Our common endeavor as roboticists to reach this goal certainly holds the promise of exciting advances toward new frontiers.

1.3 This Handbook

The intensive stream of robotics research documented in the literature culminates into this unique reference, which aims at collecting in one self-contained volume the most significant achievements of our international robotics community. The *Springer Handbook of Robotics* presents a full coverage of the field from its foundations, through the research areas, up to the new emerging applications of robotics. Accordingly, the material is organized in three logical layers reflecting the historical development of the field, as illustrated in Fig. 1.1.

The foundations of robotics science, laid down in the first layer (Part A with its 14 chapters), address the theory of robot mechanics, sensing, planning, and control. The consolidated methodologies and technologies of robot design (Part B with its 12 chapters), sensing and perception (Part C with its 8 chapters), manipulation and interfaces (Part D with its 9 chapters) and mobile and distributed robotics (Part E with its 9 chapters) are presented in the second layer. The third layer is devoted to advanced applications, such as in field and service robotics (Part F with its 13 chapters) and human-centered and life-like robotics (Part G with its 14 chapters).

Part A presents the fundamental principles and methods that are used to model, design, and control a robotic system. All of the foundational topics are included in this part: kinematics, dynamics, mechanisms and actuation, sensing and estimation, model identification, motion planning, motion control, force control, redundant robots, robots with flexible elements, robotic systems architectures and programming, behaviour-based systems, AI reasoning methods for robotics, robot learning. A chapter is devoted to each of these topics. The topics are expanded and applied to specific robotic structures and systems in subsequent parts.

Part B is concerned with the design of various robotic systems. Some of the obvious mechanical structures that come to mind are arms, legs, wheels and hands; to this list can be added robots in the air

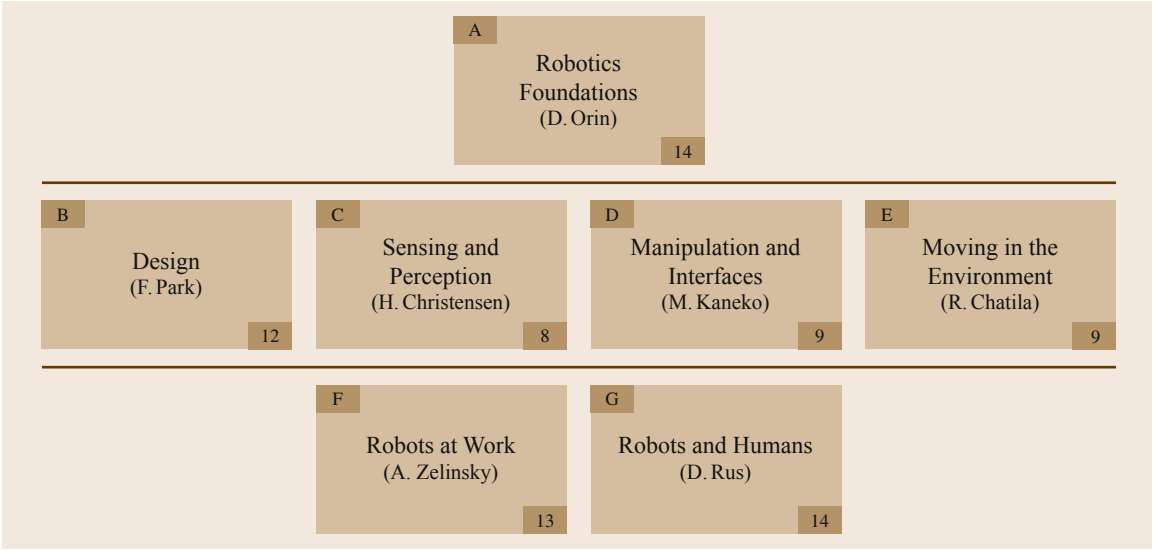


Fig. 1.1 Organization of the handbook

and subsea, as well as robot structures at the micro and nano scales. With a separate chapter devoted to design and evaluation performance, the chapters in this part successively examine limbed systems, parallel mechanisms, robot hands, snake-like and continuum robots, novel actuators for soft robotics, modular robots, biomimetic robots, wheeled robots, underwater robots, flying robots, and micro/nano robots.

Part C covers different sensory modalities and integration of sensor data across space and time to generate models of robots and the external environment. Robotics is the intelligent coupling of perception and action and as such Part C complements Part B to build systems. This part of the handbook covers sensing across contact, proprioception and exteroception. The main sensing modalities such as force and tactile sensing, inertial sensing, global positioning system (GPS) and odometry, sonar sensing, range sensing, three-dimensional (3-D) vision for navigation and grasping visual object class recognition, and visual servoing are presented. Both basic sensor models, sensor data processing and associated representations are covered. Finally, a chapter on multisensor data fusion introduces the mathematical tools needed for integration of sensor information across space and time.

Part D is concerned with interaction between robots and objects, or between humans and robots. Manipulation is supposed to handle an object through direct contact by arms or fingers or just a pusher, while interfaces are supposed to make either direct or indirect interaction between humans and robots. For enhancing dexterity in robot manipulation, motion for manipulation tasks, contact modelling and manipulation, grasping, cooperative manipulation, mobility and manipulation, active manipulation for perception are addressed in the first half of this part. For achieving a skilful manipulation or power increase in a human–robot system, haptics, telerobotics, and networked robots are discussed in the second half of Part D.

Part E covers a wide span of topics concerning the motion of robots in the environment. At first this part addresses world modelling, simultaneous localization and mapping, motion planning and obstacle avoidance. Then the modelling and control issues of legged robots, wheeled mobile robots, robots on rough terrain, underwater robots, and aerial robots are dealt with. This part completes Part A on foundations in the context of mobile robotics, and given the role of perception, is closely related to Part C on sensing. In addition, multiple mobile robot systems are discussed.

Part F covers topics related to creating field and service application-based robots that operate in all types of environments. This includes applications ranging from industrial robots to robotics competitions and challenges, through a diverse array of robotics applications: space, agriculture and forestry, construction, hazardous applications, mining, disaster scenarios, intelligent vehicles, medical and computer-integrated surgery, rehabilitation and health-care, domestic environments. This part of the handbook, which draws on Parts A, B, C, D and E, describes how robots can be put to work.

Part G covers topics related to creating robots that operate in human-centered environments. These include humanoids, human motion reconstruction, physical human–robot interaction, human robot augmentation, cognitive human–robot interaction, social robotics, socially assistive robotics, learning from humans, biologically-inspired robotics, evolutionary robotics, neurobotics, and perceptual robotics. The part concludes with the use of robotics for education, and the socio-ethical implications of robots.

Each part is accompanied by a video which illustrates the main concepts and the underlying challenges in common to the chapters in the part, while topical videos related to the specific contents of each chapter can be accessed through the web site of the multimedia contents associated with the handbook http://handbookofrobotics.org/.

The handbook has been conceived to provide a valuable resource not only for robotics experts, but also for newcomers to this expanding field, e.g., engineers, medical doctors, computer scientists, and designers. In particular, it is important to underline the tutorial value of Part A to graduate students and post-docs, the research value of Parts B to E for a wider coverage of research in robotics, and the added value of Parts F and G to engineers and scientists interested in new applications.

The contents of the various chapters have been inspired by a classic cut, i. e., avoiding the inclusion of on-going or not well-established methods. An objective perspective has been taken, while covering multiple approaches, with the goal of ensuring a high archival value to the handbook. Each chapter is preceded by a short summary, and an introductory section providing the state of the art in the area. The core sections are developed at a tutorial level. Lengthy mathematical derivations have been avoided whenever possible, while the equations, tables, and algorithms are illustrated in ready-to-use form. The final section of the chapter provides conclusions and topics for further reading. From the foundations to the social and ethical implications of robotics, the eighty chapters of the handbook provide a comprehensive collection of five decades of progress in robotics. This sum is a testimony to the level of accomplishments in our field, and a premise of further advances towards new frontiers of robotics.

Video-References

VIDEO 805 Robots – A 50 year journey
available from http://handbookofrobotics.org/view-chapter/01/videodetails/805

VIDEO 812 Robots – The journey continues
available from http://handbookofrobotics.org/view-chapter/01/videodetails/812

Multimedia Contents

Part A Robotics

Part A Robotics Foundations

Ed. by David E. Orin

2 **Kinematics**
Kenneth J. Waldron, Ultimo, Australia
James Schmiedeler, Notre Dame, USA

3 **Dynamics**
Roy Featherstone, Canberra, Australia
David E. Orin, Columbus, USA

4 **Mechanism and Actuation**
Victor Scheinman, Stanford, USA
J. Michael McCarthy, Irvine, USA
Jae-Bok Song, Seoul, Korea

5 **Sensing and Estimation**
Henrik I. Christensen, Atlanta, USA
Gregory D. Hager, Baltimore, USA

6 **Model Identification**
John Hollerbach, Salt Lake City, USA
Wisama Khalil, Nantes, France
Maxime Gautier, Nantes, France

7 **Motion Planning**
Lydia E. Kavraki, Houston, USA
Steven M. LaValle, Urbana, USA

8 **Motion Control**
Wan Kyun Chung, Pohang, Korea
Li-Chen Fu, Taipei, Taiwan
Torsten Kröger, Mountain View, USA

9 **Force Control**
Luigi Villani, Naples, Italy
Joris De Schutter, Leuven-Heverlee, Belgium

10 **Redundant Robots**
Stefano Chiaverini, Cassino, Italy
Giuseppe Oriolo, Rome, Italy
Anthony A. Maciejewski, Fort Collins, USA

11 **Robots with Flexible Elements**
Alessandro De Luca, Rome, Italy
Wayne J. Book, Atlanta, USA

12 **Robotic Systems Architectures and Programming**
David Kortenkamp, Houston, USA
Reid Simmons, Pittsburgh, USA
Davide Brugali, Dalmine, Italy

13 **Behavior-Based Systems**
François Michaud, Sherbrooke, Canada
Monica Nicolescu, Reno, USA

14 **AI Reasoning Methods for Robotics**
Michael Beetz, Bremen, Germany
Raja Chatila, Paris, France
Joachim Hertzberg, Osnabrück, Germany
Federico Pecora, Örebro, Sweden

15 **Robot Learning**
Jan Peters, Darmstadt, Germany
Daniel D. Lee, Philadelphia, USA
Jens Kober, Delft, The Netherlands
Duy Nguyen-Tuong, Stuttgart, Germany
J. Andrew Bagnell, Pittsburgh, USA
Stefan Schaal, Los Angeles, USA

The chapters contained in Part A, Robotics Foundations, present the fundamental principles and methods that are used to develop a robotic system. In order to perform the tasks that are envisioned for robots, many challenging problems have been uncovered in kinematics, dynamics, design, actuation, sensing, modeling, motion planning, control, programming, decision-making, task planning, and learning. Robots with redundant kinematic degrees of freedom or with flexible elements add to the complexity of these systems. The chapters in this part address the basic issues in each of these areas.

Some of the basic problems in robotics are outlined as follows. Robots often consist of a large number of degrees of freedom so that they can provide the rich set of three-dimensional (3-D) motions that may be required for a range of tasks. The kinematic and dynamic relationships between the joint actuators' motion and torques, and the desired motion and force for a task can be very complex. The design of the link and joint structures, as well as the actuation, to achieve the desired performance is also challenging.

The robot is a nonlinear, coupled system which is difficult to model and control because of its complex dynamics. Kinematic redundancy and flexible elements in the robots increase this complexity. The problem is exacerbated when the environment is unstructured, and often sophisticated sensing and estimation techniques are required.

In addition to control of the motion, control of the interaction forces between the robot and environment is needed when manipulating objects or interacting with humans. A fundamental robotics task is to plan collision-free motion for complex bodies from a start to a goal position among a collection of obstacles, and this can become an intractable computational problem.

In order to achieve some of the intelligence ascribed to humans, robots will need to be equipped with sophisticated action planners that employ symbolic reasoning to move in dynamic, partially known environments. In other scenarios, robots need to execute behaviors that take advantage of dynamic interactions with the environment rather than rely solely on explicit reasoning and planning. Robot software architectures also have special needs because of all of these requirements. Robot learning will be necessary to generate actions and control for ever-changing task requirements and environments, in order to achieve the level of autonomy envisioned.

While the basic issues outlined in the previous paragraphs are addressed in this part, more depth can be found in other parts of the handbook. The kinematics, dynamics, mechanical design, and control principles and methods introduced in this part can be applied to robotic structures made up of arms, hands, and legs (Part B) as well as manipulators (Part D), wheeled and other mobile robots (Part E), and field and service robots (Part F). Force control is especially important for manipulators and their interfaces (Part D). The basic sensing and estimation techniques presented here are expanded and applied to specific sensing modalities in Part C. Motion planning is an important aspect of manipulation (Part D) and mobile robots moving in the environment (Part E). Robotic systems architectures, behavior-based systems, artificial intelligence (AI) reasoning methods, and robot learning are particularly important in mobile robots moving in the environment (Part E) and robots interacting with humans (Part G).

With this overview of Part A, we now provide a brief synopsis of each chapter:

Chapter 2, Kinematics, provides a number of representations and conventions to describe the motion of the bodies in a robotic mechanism. These include rotation matrices, Euler angles, quaternions, homogeneous transformations, screw transformations, matrix exponential parameterization, and Plücker coordinates. Representations of the kinematics of all common joint types are provided, along with a modified form of the Denavit–Hartenberg convention. These representational tools are applied to compute the workspace, the forward and inverse kinematics, the forward and inverse instantaneous kinematics, Jacobian, and the static wrench transmission.

Chapter 3, Dynamics, presents the dynamic equations of motion which provide the relationships between actuation and contact forces acting on robot mechanisms, and the acceleration and motion trajectories that result. Efficient algorithms are provided for important dynamics computations which include inverse dynamics, forward dynamics, the joint-space inertia matrix, and the operational-space inertia matrix. The algorithms may be applied to fixed-base robots, mobile robots, and parallel robot mechanisms. Compact formulation of the algorithms results from using six-dimensional (6-D) spatial notation to describe rigid-body velocity, acceleration, inertia, etc.

Chapter 4, Mechanisms and Actuation, focuses on the principles that guide the design and construction of robotic systems. The kinematic equations and Jacobian are used to characterize the work envelope and mechanical advantage, and guide the selection of the robot's size and joint arrangement. The design of both serial and parallel robots is addressed. Practical consideration

is given to the design of the link and joint structures along with selection of the actuators and transmission drives to power the movement. Robot performance in terms of speed, acceleration, repeatability, and other measures is also addressed.

Chapter 5, Sensing and Estimation, provides a brief overview of common sensing methods and estimation techniques that have found broad applicability in robotics. These provide information about the state of the environment and robot system. The presentation is structured according to a perception process model that includes sensing, feature extraction, data association, parameter estimation and model integration. Several common sensing modalities are introduced and characterized. Methods for estimation in linear and nonlinear systems are discussed, including statistical estimation, the Kalman filter, and sample-based methods. The several common representations for estimation are also introduced.

Chapter 6, Model Identification, discusses methods for determining the kinematic and inertial parameters of robot manipulators. For kinematic calibration, the primary aim is to identify the geometric Denavit–Hartenberg parameters or equivalent, typically by sensing a combination of joint and endpoint positions. Inertial parameters in turn are estimated through the execution of a trajectory while additionally sensing one or more components of joint forces or torques. The chapter is organized such that both kinematic and inertial parameter identification are cast into a common framework of least-squares parameter estimation: common features relating to the identifiability of the parameters, adequacy of the measurement sets, and numerical robustness are identified and emphasized for both the kinematic and inertial parameters.

Chapter 7, Motion Planning, addresses the fundamental robotics task to plan collision-free motion for complex bodies from a start to a goal position among a collection of obstacles. The basic geometric path planning problem (Piano Mover's problem) is introduced, but the focus of the chapter is on sampling-based planning methods because of their generally wider applicability. Planning with differential constraints is considered and is important for wheeled mobile robots. Extensions and variations to the basic motion planning problem, as well as advanced issues, are discussed at the end of the chapter.

Chapter 8, Motion Control, focuses on the control of the motion of a rigid manipulator. The main challenges addressed are the complexity of the nonlinear, coupled dynamics and the model structural uncertainties. The chapter discusses topics ranging from independent-joint control and PID (proportional–integral–derivative) control to computed-torque control to manage the complex dynamics. Operational space (or task space) control is presented, and this is needed for control of tasks such as those associated with the end-effector. Adaptive and robust control are discussed to address the problems related to the uncertainties in the system. Other topics presented include trajectory generation, digital implementation, and learning control.

Chapter 9, Force Control, focuses on the control of the interaction forces between a robotic system and its environment. The chapter groups interaction control in two categories: indirect and direct force control, which achieve the control without (indirect) or with (direct) explicit closure of a force feedback loop. Impedance control and hybrid force/motion control are examples of each, respectively. The fundamental problem of interaction tasks modeling is presented to provide the basis for the force control schemes.

Chapter 10, Redundant Robots, addresses the motion generation and control of manipulators with redundant kinematic degrees of freedom. Kinematic redundancy affords a robot with an increased level of dexterity that may be used to, e.g., avoid singularities, joint limits and workspace obstacles, and also to minimize joint torques, energy, or other suitable performance criteria. The chapter discusses inverse kinematic redundancy resolution schemes which are arranged in two main categories, namely those based on the optimization of suitable performance criteria and those relying on the augmentation of the task space. The use of kinematic redundancy for fault tolerance is also analyzed.

Chapter 11, Robots with Flexible Elements, addresses the dynamic modeling and control of robots with flexibility in the joints and links. Because the control methods developed to compensate for joint versus link flexibility are structurally different, the chapter is organized such that these two types of flexibility are examined independently. The methods, however, can be extended to the case when both joint and link flexibilities are present, possibly even dynamically interacting at the same time. The chapter also examines the typical sources of flexibility in industrial robots.

Chapter 12, Robotic Systems Architectures and Programming, presents the software architectures and supporting programming tools and environments that have been developed for robotic systems. Robot architectures have special needs because of the requirements of the robot to interact asynchronously, in real time, with an uncertain, often dynamic, environment. The chapter discusses the major types of architectural com-

ponents for a layered robot control architecture – behavioral control, executives, and task planners – along with the commonly used techniques for interconnecting these components.

Chapter 13, Behavior-Based Systems, describes a control methodology aimed at situated robots operating in unconstrained, challenging, and dynamic conditions in the real world. Distributed behaviors are used as the underlying building blocks, allowing behavior-based systems to take advantage of dynamic interactions with the environment rather than rely solely on explicit reasoning and planning. The focus of the chapter is to provide the basic principles of behavior-based systems and their use in autonomous control problems and applications. The chapter presents several different classes of learning methods, but in all cases behaviors are used as the underlying building blocks for the learning process.

Chapter 14, AI Reasoning Methods for Robotics, sketches the main robotics-relevant topics of symbol-based AI reasoning. Reasoning on a mobile robot is especially challenging because of its dynamic, partially known environment. This chapter describes basic methods of knowledge representation and inference, covering both logic- and probability-based approaches. Results for several types of practical, robotics-related reasoning tasks are given, with an emphasis on temporal and spatial reasoning. Plan-based robot control is described in some detail, as it is a particularly obvious form of employing reasoning to the end of improving robot action.

Chapter 15, Robot Learning, surveys work in machine learning for techniques that are used for learning control and behavior generation in robots. In the future, robots will no longer be used to only execute the same task thousands of times, but rather they will be faced with thousands of different tasks that rarely repeat in an ever-changing environment. Robot learning will be necessary to achieve the high degree of autonomy envisioned. This chapter focuses on the core robot learning approaches capable of learning action generation and control, and in particular, techniques for model learning and reinforcement learning.

Part A presents the fundamental principles and methods that are used to model, design, and control a robotic system. All of the foundational topics are included in this part: kinematics, dynamics, mechanisms and actuation, sensing and estimation, model identification, motion planning, motion control, force control, redundant robots, robots with flexible elements, robotic systems architectures and programming, behavior-based systems, AI reasoning methods for robotics, and robot learning. A chapter is devoted to each of these topics. The topics are expanded and applied to specific robotic structures and systems in subsequent parts.

Multimedia Contents

2. Kinematics

Kenneth J. Waldron, James Schmiedeler

Kinematics pertains to the motion of bodies in a robotic mechanism without regard to the forces/torques that cause the motion. Since robotic mechanisms are by their very essence designed for motion, kinematics is the most fundamental aspect of robot design, analysis, control, and simulation. The robotics community has focused on efficiently applying different representations of position and orientation and their derivatives with respect to time to solve foundational kinematics problems.

This chapter will present the most useful representations of the position and orientation of a body in space, the kinematics of the joints most commonly found in robotic mechanisms, and a convenient convention for representing the geometry of robotic mechanisms. These representational tools will be applied to compute the *workspace*, the *forward* and *inverse kinematics*, the *forward* and *inverse instantaneous kinematics*, and the *static wrench transmission* of a robotic mechanism. For brevity, the focus will be on algorithms applicable to open-chain mechanisms.

The goal of this chapter is to provide the reader with general tools in tabulated form and a broader overview of algorithms that can be applied together to solve kinematics problems pertaining to a particular robotic mechanism.

2.1 **Overview** 12
2.2 **Position and Orientation Representation** 12
2.2.1 Position and Displacement 12
2.2.2 Orientation and Rotation 13
2.2.3 Homogeneous Transformations 16
2.2.4 Screw Transformations 17
2.2.5 Matrix Exponential Parameterization 19
2.2.6 Plücker Coordinates 20
2.3 **Joint Kinematics** 21
2.3.1 Lower Pair Joints 21
2.3.2 Higher Pair Joints 24
2.3.3 Compound Joints 24
2.3.4 6-DOF Joint 24
2.3.5 Physical Realization 25
2.3.6 Holonomic and Nonholonomic Constraints 25
2.3.7 Generalized Coordinates 25
2.4 **Geometric Representation** 25
2.5 **Workspace** 27
2.6 **Forward Kinematics** 28
2.7 **Inverse Kinematics** 29
2.7.1 Closed-Form Solutions 29
2.7.2 Numerical Methods 30
2.8 **Forward Instantaneous Kinematics** 31
2.8.1 Jacobian 31
2.9 **Inverse Instantaneous Kinematics** 32
2.9.1 Inverse Jacobian 32
2.10 **Static Wrench Transmission** 33
2.11 **Conclusions and Further Reading** 33
References 33

2.1 Overview

Unless explicitly stated otherwise, robotic mechanisms are systems of rigid bodies connected by joints. The position and orientation of a rigid body in space are collectively termed the *pose*. Therefore, robot kinematics describes the pose, velocity, acceleration, and all higher-order derivatives of the pose of the bodies that comprise a mechanism. Since kinematics does not address the forces/torques that induce motion, this chapter focuses on describing pose and velocity. These descriptions are foundational elements of dynamics (Chap. 3), motion planning (Chap. 7), and motion control (Chap. 8) algorithms.

Among the many possible topologies in which systems of bodies can be connected, two of particular importance in robotics are serial chains and fully parallel mechanisms. A serial chain is a system of rigid bodies in which each member is connected to two others, except for the first and last members that are each connected to only one other member. A fully parallel mechanism is one in which there are two members that are connected together by multiple chains of other members and joints. In practice, each of these chains is often itself a serial chain. This chapter focuses almost exclusively on algorithms applicable to serial chains. Parallel mechanisms are dealt with in more detail in Chap. 18. Another important topology is the tree structure, which is similar to a serial chain in that it has no closed loops, but differs from a serial chain in that each member might have multiple members connected to it, forming multiple *branches*. A serial chain is actually just a special case of a tree structure with no branches. Tree structures are addressed in greater depth in Chap. 3.

2.2 Position and Orientation Representation

Spatial, rigid-body kinematics can be viewed as a comparative study of different ways of representing the pose of a body. Translations and rotations, referred to in combination as rigid-body displacements, are also expressed with these representations. No one approach is optimal for all purposes, but the advantages of each can be leveraged appropriately to facilitate the solution of different problems.

The minimum number of coordinates required to locate a body in Euclidean space is six. Many representations of spatial pose employ sets with superabundant coordinates in which auxiliary relationships exist among the coordinates. The number of independent auxiliary relationships is the difference between the number of coordinates in the set and six.

This chapter and those that follow it make frequent use of *coordinate frames* or simply *frames*. A coordinate frame i consists of an origin, denoted O_i, and a triad of mutually orthogonal basis vectors, denoted $(\hat{\boldsymbol{x}}_i\, \hat{\boldsymbol{y}}_i\, \hat{\boldsymbol{z}}_i)$, that are all fixed within a particular body. The pose of a body will always be expressed relative to some other body, so it can be expressed as the pose of one coordinate frame relative to another. Similarly, rigid-body displacements can be expressed as displacements between two coordinate frames, one of which may be referred to as *moving*, while the other may be referred to as *fixed*. This indicates that the observer is located in a stationary position within the fixed frame, not that there exists any absolutely fixed frame.

2.2.1 Position and Displacement

The position of the origin of coordinate frame i relative to coordinate frame j can be denoted by the 3×1 vector

$$ {}^j\boldsymbol{p}_i = \begin{pmatrix} {}^jp_i^x \\ {}^jp_i^y \\ {}^jp_i^z \end{pmatrix} . $$

The components of this vector are the Cartesian coordinates of O_i in the j frame, which are the projections of the vector ${}^j\boldsymbol{p}_i$ onto the corresponding axes. The vector components could also be expressed as the spherical or cylindrical coordinates of O_i in the j frame. Such representations have advantages for analysis of robotic mechanisms including spherical and cylindrical joints.

A translation is a displacement in which no point in the rigid body remains in its initial position and all straight lines in the rigid body remain parallel to their initial orientations. (The points and lines are not necessarily contained within the boundaries of the finite rigid body, but rather, any point or line in space can be taken to be rigidly fixed in a body.) The translation of a body in space can be represented by the combination of its positions prior to and following the translation. Conversely, the position of a body can be represented as a translation that takes the body from a starting position (in which the coordinate frame fixed to the body coincides with the fixed coordinate frame) to the current position (in which the two frames are not coincident).

Thus, any representation of position can be used to create a representation of displacement, and vice versa.

2.2.2 Orientation and Rotation

There is significantly greater breadth in the representation of orientation than in that of position. This section does not include an exhaustive summary, but focuses on the representations most commonly applied to robotic mechanisms.

A rotation is a displacement in which at least one point in the rigid body remains in its initial position and not all lines in the body remain parallel to their initial orientations. For example, a body in a circular orbit rotates about an axis through the center of its circular path, and every point on the axis of rotation is a point in the body that remains in its initial position. As in the case of position and translation, any representation of orientation can be used to create a representation of rotation, and vice versa.

Rotation Matrices

The orientation of coordinate frame i relative to coordinate frame j can be denoted by expressing the basis vectors $(\hat{\boldsymbol{x}}_i\,\hat{\boldsymbol{y}}_i\,\hat{\boldsymbol{z}}_i)$ in terms of the basis vectors $(\hat{\boldsymbol{x}}_j\,\hat{\boldsymbol{y}}_j\,\hat{\boldsymbol{z}}_j)$. This yields $({}^j\hat{\boldsymbol{x}}_i\,{}^j\hat{\boldsymbol{y}}_i\,{}^j\hat{\boldsymbol{z}}_i)$, which when written together as a 3×3 matrix is known as the rotation matrix. The components of ${}^j\mathbf{R}_i$ are the dot products of the basis vectors of the two coordinate frames.

$$
{}^j\mathbf{R}_i = \begin{pmatrix} \hat{\boldsymbol{x}}_i\cdot\hat{\boldsymbol{x}}_j & \hat{\boldsymbol{y}}_i\cdot\hat{\boldsymbol{x}}_j & \hat{\boldsymbol{z}}_i\cdot\hat{\boldsymbol{x}}_j \\ \hat{\boldsymbol{x}}_i\cdot\hat{\boldsymbol{y}}_j & \hat{\boldsymbol{y}}_i\cdot\hat{\boldsymbol{y}}_j & \hat{\boldsymbol{z}}_i\cdot\hat{\boldsymbol{y}}_j \\ \hat{\boldsymbol{x}}_i\cdot\hat{\boldsymbol{z}}_j & \hat{\boldsymbol{y}}_i\cdot\hat{\boldsymbol{z}}_j & \hat{\boldsymbol{z}}_i\cdot\hat{\boldsymbol{z}}_j \end{pmatrix} . \tag{2.1}
$$

Because the basis vectors are unit vectors and the dot product of any two unit vectors is the cosine of the angle between them, the components are commonly referred to as direction cosines.

An elementary rotation of frame i about the $\hat{\boldsymbol{z}}_j$ axis through an angle θ is

$$
\mathbf{R}_Z(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} , \tag{2.2}
$$

while the same rotation about the $\hat{\boldsymbol{y}}_j$ axis is

$$
\mathbf{R}_Y(\theta) = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} , \tag{2.3}
$$

and about the $\hat{\boldsymbol{x}}_j$ axis is

$$
\mathbf{R}_X(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix} . \tag{2.4}
$$

Rotation matrices are combined through simple matrix multiplication such that the orientation of frame i relative to frame k can be expressed as

$$
{}^k\mathbf{R}_i = {}^k\mathbf{R}_j\,{}^j\mathbf{R}_i \, .
$$

The rotation matrix ${}^j\mathbf{R}_i$ contains nine elements, while only three parameters are required to define the orientation of a body in space. Therefore, six auxiliary relationships exist among the elements of the matrix. Because the basis vectors of coordinate frame i are mutually orthonormal, as are the basis vectors of coordinate frame j, the columns of ${}^j\mathbf{R}_i$ formed from the dot products of these vectors are also mutually orthonormal. A matrix composed of mutually orthonormal vectors is known as an orthogonal matrix and has the property that its inverse is simply its transpose. This property provides the six auxiliary relationships. Three require the column vectors to have unit length, and three require the column vectors to be mutually orthogonal. Alternatively, the orthogonality of the rotation matrix can be seen by considering the frames in reverse order. The orientation of coordinate frame j relative to coordinate frame i is the rotation matrix ${}^i\mathbf{R}_j$ whose rows are clearly the columns of the matrix ${}^j\mathbf{R}_i$.

In summary, ${}^j\mathbf{R}_i$ is the rotation matrix that transforms a vector expressed in coordinate frame i to a vector expressed in coordinate frame j. It provides a representation of the orientation of frame i relative to j and thus, can be a representation of rotation from frame i to frame j. Table 2.1 lists the equivalent rotation matrices for the other representations of orientation listed in this section. Table 2.2 contains the conversions from a known rotation matrix to these other representations.

Euler Angles

For a minimal representation, the orientation of coordinate frame i relative to coordinate frame j can be denoted as a vector of three angles $(\alpha,\beta,\gamma)^T$. These angles are known as Euler angles when each represents a rotation about an axis of a moving coordinate frame. In this way, the location of the axis of each successive rotation depends upon the preceding rotation(s), so the order of the rotations must accompany the three angles to define the orientation. For example, the symbols $(\alpha,\beta,\gamma)^T$ are used throughout this handbook to indicate Z-Y-X Euler angles. Taking the moving frame i and the fixed frame j to be initially coincident, α is the rotation about the $\hat{\boldsymbol{z}}$ axis of frame i, β is the rotation about the rotated $\hat{\boldsymbol{y}}$ axis of frame i, and finally, γ is the rotation about the twice rotated $\hat{\boldsymbol{x}}$ axis of frame i. The equivalent rotation matrix ${}^j\mathbf{R}_i$ is given in Table 2.1. Z-Y-Z and Z-X-Z Euler angles are other commonly used con-

Table 2.1 Equivalent rotation matrices for various representations of orientation, with abbreviations $c_\theta := \cos\theta$, $s_\theta := \sin\theta$, and $v_\theta := 1-\cos\theta$

Z-Y-X Euler angles $(\alpha,\beta,\gamma)^{\mathrm{T}}$	${}^{j}\mathbf{R}_i = \begin{pmatrix} c_\alpha c_\beta & c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta c_\gamma + s_\alpha s_\gamma \\ s_\alpha c_\beta & s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta c_\gamma - c_\alpha s_\gamma \\ -s_\beta & c_\beta s_\gamma & c_\beta c_\gamma \end{pmatrix}$
X-Y-Z fixed angles $(\psi,\theta,\phi)^{\mathrm{T}}$	${}^{j}\mathbf{R}_i = \begin{pmatrix} c_\phi c_\theta & c_\phi s_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ s_\phi c_\theta & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{pmatrix}$
Angle-axis $\theta\hat{\boldsymbol{w}}$	${}^{j}\mathbf{R}_i = \begin{pmatrix} w_x^2 v_\theta + c_\theta & w_x w_y v_\theta - w_z s_\theta & w_x w_z v_\theta + w_y s_\theta \\ w_x w_y v_\theta + w_z s_\theta & w_y^2 v_\theta + c_\theta & w_y w_z v_\theta - w_x s_\theta \\ w_x w_z v_\theta - w_y s_\theta & w_y w_z v_\theta + w_x s_\theta & w_z^2 v_\theta + c_\theta \end{pmatrix}$
Unit quaternions $(\epsilon_0\ \epsilon_1\ \epsilon_2\ \epsilon_3)^{\mathrm{T}}$	${}^{j}\mathbf{R}_i = \begin{pmatrix} 1-2\left(\epsilon_2^2+\epsilon_3^2\right) & 2\left(\epsilon_1\epsilon_2-\epsilon_0\epsilon_3\right) & 2\left(\epsilon_1\epsilon_3+\epsilon_0\epsilon_2\right) \\ 2\left(\epsilon_1\epsilon_2+\epsilon_0\epsilon_3\right) & 1-2\left(\epsilon_1^2+\epsilon_3^2\right) & 2\left(\epsilon_2\epsilon_3-\epsilon_0\epsilon_1\right) \\ 2\left(\epsilon_1\epsilon_3-\epsilon_0\epsilon_2\right) & 2\left(\epsilon_2\epsilon_3+\epsilon_0\epsilon_1\right) & 1-2\left(\epsilon_1^2+\epsilon_2^2\right) \end{pmatrix}$

ventions from among the 12 different possible orders of rotations.

Regardless of the order of rotations, an Euler angle representation of orientation always exhibits a singularity when the first and last rotations both occur about the same axis. This can be readily seen in the second block of Table 2.2 wherein the angles α and γ are undefined when $\beta = \pm 90°$. (For Z-Y-Z and Z-X-Z Euler angles, the singularity occurs when the second rotation is 0° or 180°.) This creates a problem in relating the angular velocity vector of a body to the time derivatives of Euler angles, which somewhat limits their usefulness in modeling robotic systems. This velocity relationship for Z-Y-X Euler angles is

$$\begin{pmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{pmatrix} = \frac{1}{\cos\beta} \begin{pmatrix} 0 & \sin\gamma & \cos\gamma \\ 0 & \cos\gamma\cos\beta & -\sin\gamma\cos\beta \\ \cos\beta & \sin\gamma\sin\beta & \cos\gamma\sin\beta \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}, \tag{2.5}$$

where $(\omega_x,\omega_y,\omega_z)^{\mathrm{T}} = {}^{i}\boldsymbol{\omega}_i$ is given in moving frame i. In some circumstances, the inverse of this relationship may be required.

$$\begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} = \begin{pmatrix} -\sin\beta & 0 & 1 \\ \cos\beta\sin\gamma & \cos\gamma & 0 \\ \cos\beta\cos\gamma & -\sin\gamma & 0 \end{pmatrix} \begin{pmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{pmatrix}. \tag{2.6}$$

Table 2.2 Conversions from a rotation matrix to various representations of orientation

Rotation matrix:

$${}^{j}\mathbf{R}_i = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

Z-Y-X Euler angles $(\alpha,\beta,\gamma)^{\mathrm{T}}$:

$$\beta = \mathrm{Atan2}\left(-r_{31}, \sqrt{r_{11}^2+r_{21}^2}\right)$$
$$\alpha = \mathrm{Atan2}\left(\frac{r_{21}}{\cos\beta}, \frac{r_{11}}{\cos\beta}\right)$$
$$\gamma = \mathrm{Atan2}\left(\frac{r_{32}}{\cos\beta}, \frac{r_{33}}{\cos\beta}\right)$$

X-Y-Z fixed angles $(\psi,\theta,\phi)^{\mathrm{T}}$:

$$\theta = \mathrm{Atan2}\left(-r_{31}, \sqrt{r_{11}^2+r_{21}^2}\right)$$
$$\psi = \mathrm{Atan2}\left(\frac{r_{21}}{\cos\theta}, \frac{r_{11}}{\cos\theta}\right)$$
$$\phi = \mathrm{Atan2}\left(\frac{r_{32}}{\cos\theta}, \frac{r_{33}}{\cos\theta}\right)$$

Angle axis $\theta\hat{\boldsymbol{w}}$:

$$\theta = \cos^{-1}\left(\frac{r_{11}+r_{22}+r_{33}-1}{2}\right)$$
$$\hat{\boldsymbol{w}} = \frac{1}{2\sin\theta}\begin{pmatrix} r_{32}-r_{23} \\ r_{13}-r_{31} \\ r_{21}-r_{12} \end{pmatrix}$$

Unit quaternions $(\epsilon_0\ \epsilon_1\ \epsilon_2\ \epsilon_3)^{\mathrm{T}}$:

$$\epsilon_0 = \frac{1}{2}\sqrt{1+r_{11}+r_{22}+r_{33}}$$
$$\epsilon_1 = \frac{r_{32}-r_{23}}{4\epsilon_0}$$
$$\epsilon_2 = \frac{r_{13}-r_{31}}{4\epsilon_0}$$
$$\epsilon_3 = \frac{r_{21}-r_{12}}{4\epsilon_0}$$

Fixed Angles

A vector of three angles can also denote the orientation of coordinate frame i relative to coordinate frame j when each angle represents a rotation about an axis of a fixed frame. Appropriately, such angles are referred to as fixed angles, and the order of the rotations must again accompany the angles to define the orientation. X-Y-Z fixed angles, denoted here as $(\psi,\theta,\phi)^{\mathrm{T}}$, are a common convention from among the, again, 12 different possible orders of rotations. Taking the moving frame i and

the fixed frame j to be initially coincident, ψ is the yaw rotation about the fixed $\hat{\boldsymbol{x}}_j$ axis, θ is the pitch rotation about the fixed $\hat{\boldsymbol{y}}_j$ axis, and ϕ is the roll rotation about the fixed $\hat{\boldsymbol{z}}_j$ axis. As can be seen by comparing the respective equivalent rotation matrices in Table 2.1 and the respective conversions in Table 2.2, a set of *X-Y-Z* fixed angles is exactly equivalent to the same set of *Z-Y-X* Euler angles ($\alpha = \phi$, $\beta = \theta$, and $\gamma = \psi$). This result holds in general such that three rotations about the three axes of a fixed frame define the same orientation as the same three rotations taken in the opposite order about the three axes of a moving frame. Likewise, all fixed-angle representations of orientations suffer from the singularity discussed for Euler angles. Also, the relationship between the time derivatives of fixed angles and the angular velocity vector is similar to the relationship for Euler angles.

Angle–Axis

A single angle θ in combination with a unit vector $\hat{\boldsymbol{w}}$ can also denote the orientation of coordinate frame i relative to coordinate frame j. In this case, frame i is rotated through the angle θ about an axis defined by the vector $\hat{\boldsymbol{w}} = (w_x\ w_y\ w_z)^{\mathrm{T}}$ relative to frame j. The vector $\hat{\boldsymbol{w}}$ is sometimes referred to as the equivalent axis of a finite rotation. The angle-axis representation, typically written as either $\theta\hat{\boldsymbol{w}}$ or $(\theta w_x\ \theta w_y\ \theta w_z)^{\mathrm{T}}$, is superabundant by one because it contains four parameters. The auxiliary relationship that resolves this is the unit magnitude of vector $\hat{\boldsymbol{w}}$. Even with this auxiliary relationship, the angle-axis representation is not unique because rotation through an angle of $-\theta$ about $-\hat{\boldsymbol{w}}$ is equivalent to a rotation through θ about $\hat{\boldsymbol{w}}$. Table 2.3 contains the conversions from the angle-axis representation to unit quaternions and vice versa. The conversions from these two representations to Euler angles or fixed angles can be easily found by using the conversions in Table 2.2 in conjunction with the equivalent rotation matrices in Table 2.1. Velocity relationships are more easily dealt with using the closely related quaternion representation.

Table 2.3 Conversions from angle-axis to unit quaternion representations of orientation and vice versa

Angle-axis $\theta\hat{\boldsymbol{w}}$ to unit quaternion $(\epsilon_0\ \epsilon_1\ \epsilon_2\ \epsilon_3)^{\mathrm{T}}$:
$\epsilon_0 = \cos\frac{\theta}{2}$
$\epsilon_1 = w_x \sin\frac{\theta}{2}$
$\epsilon_2 = w_y \sin\frac{\theta}{2}$
$\epsilon_3 = w_z \sin\frac{\theta}{2}$
Unit quaternion $(\epsilon_0\ \epsilon_1\ \epsilon_2\ \epsilon_3)^{\mathrm{T}}$ to angle-axis $\theta\hat{\boldsymbol{w}}$:
$\theta = 2\cos^{-1}\epsilon_0$
$w_x = \frac{\epsilon_1}{\sin\frac{\theta}{2}}$
$w_y = \frac{\epsilon_2}{\sin\frac{\theta}{2}}$
$w_z = \frac{\epsilon_3}{\sin\frac{\theta}{2}}$

Quaternions

The quaternion representation of orientation due to *Hamilton* [2.1], while largely superseded by the simpler vector representations of *Gibbs* [2.2] and *Graßmann* [2.3], is extremely useful for problems in robotics that result in representational singularities in the vector/matrix notation [2.4]. Quaternions do not suffer from singularities as Euler and fixed angles do.

A quaternion $\boldsymbol{\epsilon}$ is defined to have the form

$$\boldsymbol{\epsilon} = \epsilon_0 + \epsilon_1 i + \epsilon_2 j + \epsilon_3 k\ ,$$

where the components ϵ_0, ϵ_1, ϵ_2, and ϵ_3 are scalars, sometimes referred to as Euler parameters, and i, j, and k are operators. The operators are defined to satisfy the following combinatory rules

$$\begin{aligned} &ii = jj = kk = -1\ , \\ &ij = k\ , jk = i\ , ki = j\ , \\ &ji = -k\ , kj = -i\ , ik = -j\ . \end{aligned}$$

Two quaternions are added by adding the respective components separately, so the operators act as separators. The null element for addition is the quaternion $\mathbf{0} = 0 + 0i + 0j + 0k$, and quaternion sums are associative, commutative, and distributive. The null element for multiplication is $\mathbf{I} = 1 + 0i + 0j + 0k$, as can be seen using $\mathbf{I}\boldsymbol{\epsilon} = \boldsymbol{\epsilon}$ for any quaternion $\boldsymbol{\epsilon}$. Quaternion products are associative and distributive, but not commutative, and following the conventions of the operators and addition, have the form

$$\begin{aligned} \boldsymbol{ab} = {} & a_0b_0 - a_1b_1 - a_2b_2 - a_3b_3 \\ & + (a_0b_1 + a_1b_0 + a_2b_3 - a_3b_2)i \\ & + (a_0b_2 + a_2b_0 + a_3b_1 - a_1b_3)j \\ & + (a_0b_3 + a_3b_0 + a_1b_2 - a_2b_1)k\ . \end{aligned} \tag{2.7}$$

It is convenient to define the conjugate of a quaternion

$$\tilde{\boldsymbol{\epsilon}} = \epsilon_0 - \epsilon_1 i - \epsilon_2 j - \epsilon_3 k\ ,$$

so that

$$\boldsymbol{\epsilon}\tilde{\boldsymbol{\epsilon}} = \tilde{\boldsymbol{\epsilon}}\boldsymbol{\epsilon} = \epsilon_0^2 + \epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2\ .$$

A unit quaternion can then be defined such that $\boldsymbol{\epsilon}\tilde{\boldsymbol{\epsilon}} = 1$. Often, ϵ_0 is referred to as the scalar part of the quaternion, and $(\epsilon_1\ \epsilon_2\ \epsilon_3)^{\mathrm{T}}$ is referred to as the vector part.

Unit quaternions are used to describe orientation, and the unit magnitude provides the auxiliary relationship to resolve the use of superabundant (four) coordinates. A vector is defined in quaternion notation as a quaternion with $\epsilon_0 = 0$. Thus, a vector $\boldsymbol{p} = (p_x\ p_y\ p_z)^{\mathrm{T}}$ can be expressed as a quaternion $\boldsymbol{p} = p_x i + p_y j + p_z k$. For any unit quaternion $\boldsymbol{\epsilon}$, the operation $\boldsymbol{\epsilon p \tilde{\epsilon}}$ performs a rotation of the vector $\boldsymbol{p}$ about the direction $(\epsilon_1\ \epsilon_2\ \epsilon_3)^{\mathrm{T}}$. This is clearly seen by expanding the operation $\boldsymbol{\epsilon p \tilde{\epsilon}}$ and comparing the results with the equivalent rotation matrix listed in Table 2.1. Also, as shown in Table 2.3, unit quaternions are closely related to the angle-axis representation. ϵ_0 corresponds (but is not equal) to the angle of rotation, while ϵ_1, ϵ_2, and ϵ_3 define the axis of rotation.

For velocity analysis, the time derivative of the quaternion can be related to the angular velocity vector as

$$\begin{pmatrix} \dot{\epsilon}_0 \\ \dot{\epsilon}_1 \\ \dot{\epsilon}_2 \\ \dot{\epsilon}_3 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} -\epsilon_1 & -\epsilon_2 & -\epsilon_3 \\ \epsilon_0 & \epsilon_3 & -\epsilon_2 \\ -\epsilon_3 & \epsilon_0 & \epsilon_1 \\ \epsilon_2 & -\epsilon_1 & \epsilon_0 \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}, \tag{2.8}$$

where $(\omega_x, \omega_y, \omega_z)^{\mathrm{T}} = {}^j\boldsymbol{\omega}_i$ is given in fixed frame j. Defining $\boldsymbol{\epsilon}_{1:3} = (\epsilon_1, \epsilon_2, \epsilon_3)^{\mathrm{T}}$, it is straightforward to verify that

$$\dot{\epsilon}_0 = -\frac{1}{2}\, {}^j\boldsymbol{\omega}_i^{\mathrm{T}}\, \boldsymbol{\epsilon}_{1:3}$$
$$\dot{\boldsymbol{\epsilon}}_{1:3} = \frac{1}{2}\left(\epsilon_0\, {}^j\boldsymbol{\omega}_i - \boldsymbol{\epsilon}_{1:3} \times {}^j\boldsymbol{\omega}_i\right) .$$

In the case that the angular velocity is given in the moving coordinates, these rate equations take a similar form.

$$\dot{\epsilon}_0 = -\frac{1}{2}\, {}^i\boldsymbol{\omega}_i^{\mathrm{T}}\, \boldsymbol{\epsilon}_{1:3}$$
$$\dot{\boldsymbol{\epsilon}}_{1:3} = \frac{1}{2}\left(\epsilon_0\, {}^i\boldsymbol{\omega}_i + \boldsymbol{\epsilon}_{1:3} \times {}^i\boldsymbol{\omega}_i\right)$$

The corresponding rate matrix for this case is the same as (2.8) with the exception that the off-diagonal elements in the bottom 3×3 submatrix have opposite sign.

While a unit quaternion represents only the orientation of a body, quaternions may be dualized [2.5–7] to create an algebra that provides a description of the position and orientation of a body in space. Other combined representations are discussed in the following sections.

2.2.3 Homogeneous Transformations

The preceding sections have addressed representations of position and orientation separately. With homogeneous transformations, position vectors and rotation matrices are combined together in a compact notation. Any vector ${}^i\boldsymbol{r}$ expressed relative to the i coordinate frame can be expressed relative to the j coordinate frame if the position and orientation of the i frame are known relative to the j frame. Using the notation of Sect. 2.2.1, the position of the origin of coordinate frame i relative to coordinate frame j can be denoted by the vector ${}^j\boldsymbol{p}_i = ({}^jp_i^x\ {}^jp_i^y\ {}^jp_i^z)^{\mathrm{T}}$. Using the notation of Sect. 2.2.2, the orientation of frame i relative to frame j can be denoted by the rotation matrix ${}^j\mathbf{R}_i$. Thus,

$${}^j\boldsymbol{r} = {}^j\mathbf{R}_i\, {}^i\boldsymbol{r} + {}^j\boldsymbol{p}_i\, . \tag{2.9}$$

This equation can be written

$$\begin{pmatrix} {}^j\boldsymbol{r} \\ 1 \end{pmatrix} = \begin{pmatrix} {}^j\mathbf{R}_i & {}^j\boldsymbol{p}_i \\ \mathbf{0}^{\mathrm{T}} & 1 \end{pmatrix} \begin{pmatrix} {}^i\boldsymbol{r} \\ 1 \end{pmatrix}, \tag{2.10}$$

where

$${}^j\mathbf{T}_i = \begin{pmatrix} {}^j\mathbf{R}_i & {}^j\boldsymbol{p}_i \\ \mathbf{0}^{\mathrm{T}} & 1 \end{pmatrix} \tag{2.11}$$

is the 4×4 homogeneous transformation matrix and $({}^j\boldsymbol{r}\ 1)^{\mathrm{T}}$ and $({}^i\boldsymbol{r}\ 1)^{\mathrm{T}}$ are the homogeneous representations of the position vectors ${}^j\boldsymbol{r}$ and ${}^i\boldsymbol{r}$. The matrix ${}^j\mathbf{T}_i$ transforms vectors from coordinate frame i to coordinate frame j. Its inverse ${}^j\mathbf{T}_i^{-1}$ transforms vectors from coordinate frame j to coordinate frame i.

$${}^j\mathbf{T}_i^{-1} = {}^i\mathbf{T}_j = \begin{pmatrix} {}^j\mathbf{R}_i^{\mathrm{T}} & -{}^j\mathbf{R}_i^{\mathrm{T}}\, {}^j\boldsymbol{p}_i \\ \mathbf{0}^{\mathrm{T}} & 1 \end{pmatrix}. \tag{2.12}$$

Composition of 4×4 homogeneous transformation matrices is accomplished through simple matrix multiplication, just as in the case of 3×3 rotation matrices. Therefore, ${}^k\mathbf{T}_i = {}^k\mathbf{T}_j\, {}^j\mathbf{T}_i$. Since matrix multiplications do not commute, the order or sequence is important.

The homogeneous transformation of a simple rotation about an axis is sometimes denoted ***Rot*** such that a rotation of θ about an axis $\hat{\boldsymbol{z}}$ is

$$\mathbf{Rot}(\hat{\boldsymbol{z}}, \theta) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.13}$$

Similarly, the homogeneous transformation of a simple translation along an axis is sometimes denoted ***Trans*** such that a translation of d along an axis $\hat{\boldsymbol{x}}$ is

$$\mathbf{Trans}(\hat{\boldsymbol{x}}, d) = \begin{pmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.14}$$

Homogeneous transformations are particularly attractive when compact notation is desired and/or when ease of programming is the most important consideration. This is not, however, a computationally efficient representation since it introduces a large number of additional multiplications by ones and zeros. Although homogeneous transformation matrices technically contain 16 elements, four are defined to be zero or one, and the remaining elements are composed of a rotation matrix and a position vector. Therefore, the only truly superabundant coordinates come from the rotation matrix component, so the relevant auxiliary relationships are those associated with the rotation matrix.

2.2.4 Screw Transformations

The transformation in (2.9) can be viewed as composed of a rotation between coordinate frames i and j and a separate displacement between those frames. To get from frame i to frame j, one could perform the rotation first, followed by the displacement, or vice versa. Alternatively, the spatial displacement between the frames can be expressed, except in the case of a pure translation, as a rotation about a unique line combined with a translation parallel to that line.

Chasles' Theorem

Chasles' theorem, in the form stated by *Chirikjian* and *Kyatkin* [2.8], has two parts. The first states that:

> *Any displacement of a body in space can be accomplished by means of a translation of a designated point from its initial to its final position, followed by a rotation of the whole body about that point to bring it into its final orientation.*

The second part states that:

> *Any displacement of a body in space can be accomplished by means of a rotation of the body about a unique line in space accompanied by a translation of the body parallel to that line.*

Such a line is called a screw axis, and it is this second result that is usually thought of as Chasles' theorem.

The first part of the theorem is almost axiomatic. A designated point in a body anywhere in Euclidean space can be displaced from a given initial position to a given final position. By further requiring that all points in the body traverse the same displacement, the body translates so that the designated point moves from its initial position to its final position. The body can then be rotated about that point into any given final orientation.

The second part of the theorem depends on this representation of a spatial displacement and requires a more complex argument. A preliminary theorem due to Euler allows greater specificity about the rotation of the body: *Any displacement of a body in which one point remains fixed is equivalent to a rotation of that body about a unique axis passing through that point.* Geometrically, embedding three points in the moving body and letting one be the fixed point about which rotation occurs, each of the other two will have initial and final positions. The right bisector planes of the lines joining the initial and final positions in each case necessarily contain the fixed point. Any line in the bisector plane can be the axis of a rotation that carries the corresponding point from its initial to its final position. Therefore, the unique line common to the two bisector planes is such that rotation about it will carry any point in the body from its initial to its final position. The rigidity condition requires that all planes in the body that contain that line rotate through the same angle.

For any rotation of a rigid body described by a rotation matrix ${}^{j}\mathbf{R}_{i}$, Euler's theorem states that there is a unique eigenvector $\hat{\boldsymbol{w}}$ such that

$$ {}^{j}\mathbf{R}_{i}\hat{\boldsymbol{w}} = \hat{\boldsymbol{w}} \,, \tag{2.15} $$

where $\hat{\boldsymbol{w}}$ is a unit vector parallel to the axis of rotation. This expression requires a unit eigenvalue of ${}^{j}\mathbf{R}_{i}$ corresponding to the eigenvector $\hat{\boldsymbol{w}}$. The remaining two eigenvalues are $\cos\theta \pm i\sin\theta$, where i is the complex operator and θ is the angle of rotation of the body about the axis.

Combining the first part of Chasles' theorem with Euler's theorem, a general spatial displacement can be expressed as a translation taking a point from its initial to its final position, followed by a unique rotation about a unique axis through that point that carries the body from its initial to its final orientation. Resolving the translation into components in the direction of and orthogonal to the rotation axis, every point in the body has the same displacement component in the direction of the axis because rotation about it does not affect that component. Projected onto a plane normal to the rotation axis, the kinematic geometry of the displacement is identical to that of planar motion. Just as there is a unique point in the plane about which a body could be rotated between two given positions, there is a unique point in the projection plane. If the rotation axis is moved to pass through that point, the spatial displacement can be accomplished by a rotation about that axis combined with a translation along it, as stated by the theorem.

The line about which rotation takes place is called the screw axis of the displacement. The ratio of the linear displacement d to the rotation θ is referred to as the

pitch h of the screw axis [2.4]. Thus,

$$d = h\theta \,. \tag{2.16}$$

The screw axis of a pure translation is not unique. Any line parallel to the translation can be regarded as the screw axis, and since the rotation θ is zero, the axis of a translation is said to have infinite pitch.

A screw axis is most conveniently represented in any coordinate frame by means of a unit vector $\hat{\boldsymbol{w}}$ parallel to it and the position vector $\boldsymbol{\rho}$ of any point lying on it. Additional specification of the pitch h and the angle of rotation θ completely defines the location of a second coordinate frame relative to the first frame. Thus, a total of eight coordinates define a screw transformation, which is superabundant by two. The unit magnitude of vector $\hat{\boldsymbol{w}}$ provides one auxiliary relationship, but in general, there is no second auxiliary relationship because the same screw axis is defined by all points lying on it, which is to say that the vector $\boldsymbol{\rho}$ contains one free coordinate.

Algebraically, a screw displacement is represented by

$$^j\boldsymbol{r} = {}^j\mathbf{R}_i(^i\boldsymbol{r} - \boldsymbol{\rho}) + d\hat{\boldsymbol{w}} + \boldsymbol{\rho} \,. \tag{2.17}$$

Comparing this expression to (2.9) yields

$$^j\boldsymbol{p}_i = d\hat{\boldsymbol{w}} + \left(\mathbf{1}_{3\times3} - {}^j\mathbf{R}_i\right)\boldsymbol{\rho} \,. \tag{2.18}$$

$\mathbf{1}_{3\times3}$ denotes the 3×3 identity matrix. An expression for d is easily obtained by taking the inner product of both sides of the equation with $\hat{\boldsymbol{w}}$.

$$d = \hat{\boldsymbol{w}}^{\mathrm{T}}{}^j\boldsymbol{p}_i \,. \tag{2.19}$$

The matrix $\mathbf{1}_{3\times3} - {}^j\mathbf{R}_i$ is singular, so (2.18) cannot be solved to give a unique value of $\boldsymbol{\rho}$, but since $\boldsymbol{\rho}$ can represent any point on the screw axis, this would not be appropriate. One component of $\boldsymbol{\rho}$ can be arbitrarily chosen, and any two of the component equations can then be solved to find the two other components of $\boldsymbol{\rho}$. All other points on the screw axis are then given by $\boldsymbol{\rho} + k\hat{\boldsymbol{w}}$, where k can take any value.

Table 2.4 contains the conversions between screw transformations and homogeneous transformations. Note that the equivalent rotation matrix for a screw transformation has the same form as the equivalent rotation matrix for an angle-axis representation of orientation in Table 2.1. Also, the auxiliary relationship that the vector $\boldsymbol{\rho}$ be orthogonal to the screw axis ($\hat{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{\rho} = 0$) is used in Table 2.4 to provide a unique conversion to the screw transformation. The inverse result, that of finding the rotation matrix ${}^j\mathbf{R}_i$ and the translation ${}^j\boldsymbol{p}_i$ corresponding to a given screw displacement, is found from Rodrigues' equation.

Rodrigues' Equation

Given a screw axis, the angular displacement of a body about it, and the translation of the body along it, the displacement of an arbitrary point in that body can be found. Viewing a matrix transformation as describing the displacement of the body, this is equivalent to finding the matrix transformation equivalent to a given screw displacement.

Referring to Fig. 2.1, the position vectors of a point before and after a screw displacement can be geometrically related as

$$\begin{aligned} ^j\boldsymbol{r} = {}^i\boldsymbol{r} + d\hat{\boldsymbol{w}} + \sin\theta\,\hat{\boldsymbol{w}} \times (^i\boldsymbol{r} - \boldsymbol{\rho}) \\ - (1 - \cos\theta)(^i\boldsymbol{r} - \boldsymbol{\rho}) - (^i\boldsymbol{r} - \boldsymbol{\rho}) \cdot \hat{\boldsymbol{w}}\,\hat{\boldsymbol{w}} \,, \end{aligned} \tag{2.20}$$

where ${}^i\boldsymbol{r}$ and ${}^j\boldsymbol{r}$ denote the initial and final positions of the point, $\hat{\boldsymbol{w}}$ and $\boldsymbol{\rho}$ specify the screw axis, and θ and d give the displacement about it. This result is usually referred to as Rodrigues' equation [2.9], which can be written as a matrix transformation [2.10],

$$^j\boldsymbol{r} = {}^j\mathbf{R}_i{}^i\boldsymbol{r} + {}^j\boldsymbol{p}_i \,, \tag{2.21}$$

since, when expanded, it gives three linear equations for the components of ${}^j\boldsymbol{r}$ in terms of those of ${}^i\boldsymbol{r}$.

$$^j\mathbf{R}_i = \begin{pmatrix} w_x^2 v_\theta + c_\theta & w_x w_y v_\theta - w_z s_\theta & w_x w_z v_\theta + w_y s_\theta \\ w_x w_y v_\theta + w_z s_\theta & w_y^2 v_\theta + c_\theta & w_y w_z v_\theta - w_x s_\theta \\ w_x w_z v_\theta - w_y s_\theta & w_y w_z v_\theta + w_x s_\theta & w_z^2 v_\theta + c_\theta \end{pmatrix}$$

$$^j\boldsymbol{p}_i = \left(\mathbf{1}_{3\times3} - {}^j\mathbf{R}_i\right)\boldsymbol{\rho} + h\theta\hat{\mathbf{w}} \,,$$

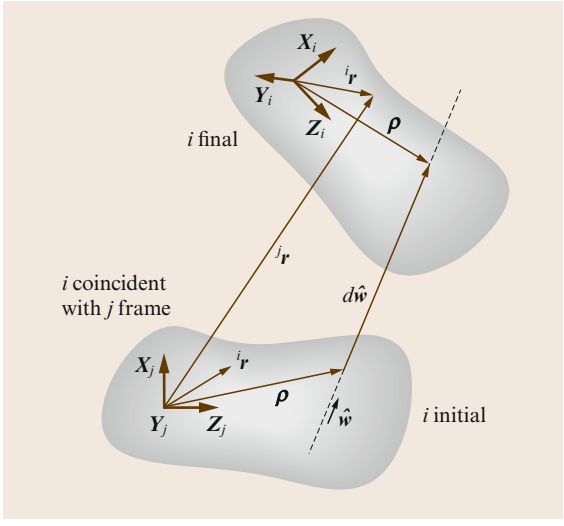


Fig. 2.1 Initial and final positions of an arbitrary point in a body undergoing a screw displacement; ${}^i\boldsymbol{r}$ is the position of the point relative to the moving frame, which is coincident with the fixed frame j in its initial position; ${}^j\boldsymbol{r}$ is the position of the point relative to the fixed frame after the screw displacement of the moving body

Table 2.4 Conversions from a screw transformation to a homogeneous transformation and vice versa, with abbreviations $c_\theta := \cos\theta$, $s_\theta := \sin\theta$, and $v_\theta := 1-\cos\theta$

Screw transformation to homogeneous transformation	${}^j\mathbf{R}_i = \begin{pmatrix} w_x^2 v_\theta + c_\theta & w_x w_y v_\theta - w_z s_\theta & w_x w_z v_\theta + w_y s_\theta \\ w_x w_y v_\theta + w_z s_\theta & w_y^2 v_\theta + c_\theta & w_y w_z v_\theta - w_x s_\theta \\ w_x w_z v_\theta - w_y s_\theta & w_y w_z v_\theta + w_x s_\theta & w_z^2 v_\theta + c_\theta \end{pmatrix}$ ${}^j\mathbf{p}_i = (\mathbf{1}_{3\times3} - {}^j\mathbf{R}_i)\boldsymbol{\rho} + h\theta\,\hat{\boldsymbol{w}}$
Homogeneous transformation to screw transformation	$\boldsymbol{l} = \begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix}^{\mathrm{T}}$ $\theta = \operatorname{sign}\left(\boldsymbol{l}^{\mathrm{T}} {}^j\boldsymbol{p}_i\right) \left\lvert \cos^{-1}\left(\frac{r_{11}+r_{22}+r_{33}-1}{2}\right)\right\rvert$ $h = \frac{\boldsymbol{l}^{\mathrm{T}} {}^j\boldsymbol{p}_i}{2\theta\sin\theta}$ $\boldsymbol{\rho} = \frac{(\mathbf{1}_{3\times3} - {}^j\mathbf{R}_i^{\mathrm{T}}) {}^j\boldsymbol{p}_i}{2(1-\cos\theta)}$ $\hat{\boldsymbol{w}} = \frac{\boldsymbol{l}}{2\sin\theta}$

where the abbreviations are $c_\theta := \cos\theta$, $s_\theta := \sin\theta$, and $v_\theta = 1-\cos\theta$. The rotation matrix ${}^j\mathbf{R}_i$ expressed in this form is also called the screw matrix, and these equations give the elements of ${}^j\mathbf{R}_i$ and ${}^j\boldsymbol{p}_i$ in terms of the screw parameters.

An exception arises in the case of a pure translation, for which $\theta = 0$ and Rodrigues' equation becomes

$$ {}^j\boldsymbol{r} = {}^i\boldsymbol{r} + d\hat{\boldsymbol{w}} \,. \tag{2.22}$$

Substituting for this case, ${}^j\mathbf{R}_i = \mathbf{1}_{3\times3}$ and ${}^j\boldsymbol{p}_i = d\hat{\boldsymbol{w}}$.

Additional information on screw theory can be found in [2.11–15].

2.2.5 Matrix Exponential Parameterization

The position and orientation of a body can also be expressed in a unified fashion with an exponential mapping. This approach is introduced first with its application to pure rotation and expanded to rigid-body motion. More details on the approach can be found in [2.16] and [2.17].

Exponential Coordinates for Rotation

The set of all orthogonal matrices with determinant 1, which is the set of all rotation matrices $\mathbf{R}$, is a *group* under the operation of matrix multiplication denoted as $SO(3) \subset \mathbb{R}^{3\times3}$ [2.18]. This stands for *special orthogonal* wherein *special* alludes to the det $\mathbf{R}$ being $+1$ instead of ±1. The set of rotation matrices satisfies the four axioms of a group:

- *Closure*: $\mathbf{R}_1\mathbf{R}_2 \in SO(3) \quad \forall\mathbf{R}_1, \mathbf{R}_2 \in SO(3)$
- *Identity*: $\mathbf{1}_{3\times3}\mathbf{R} = \mathbf{R}\mathbf{1}_{3\times3} = \mathbf{R} \quad \forall\mathbf{R} \in SO(3)$
- *Inverse*: $\mathbf{R}^{\mathrm{T}} \in SO(3)$ is the unique inverse of $\mathbf{R}$ $\forall\mathbf{R} \in SO(3)$
- *Associativity*: $(\mathbf{R}_1\mathbf{R}_2)\mathbf{R}_3 = \mathbf{R}_1(\mathbf{R}_2\mathbf{R}_3) \quad \forall\mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3 \in SO(3)$.

In the angle-axis representation presented in Sect. 2.2.2, orientation is expressed as an angle θ of rotation about an axis defined by the unit vector $\hat{\boldsymbol{w}}$. The equivalent rotation matrix found in Table 2.1 can be expressed as the exponential map

$$\begin{aligned} \mathbf{R} &= \mathrm{e}^{\mathbf{S}(\hat{\boldsymbol{w}})\theta} \\ &= \mathbf{1}_{3\times3} + \theta\mathbf{S}(\hat{\boldsymbol{w}}) + \frac{\theta^2}{2!}\mathbf{S}(\hat{\boldsymbol{w}})^2 \\ &\quad + \frac{\theta^3}{3!}\mathbf{S}(\hat{\boldsymbol{w}})^3 + \ldots \,, \end{aligned} \tag{2.23}$$

where $\mathbf{S}(\hat{\boldsymbol{w}})$ is the unit skew-symmetric matrix

$$\mathbf{S}(\hat{\boldsymbol{w}}) = \begin{pmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{pmatrix} . \tag{2.24}$$

Thus, the exponential map transforms a skew-symmetric matrix $\mathbf{S}(\hat{\boldsymbol{w}})$ that corresponds to an axis of rotation $\hat{\boldsymbol{w}}$ into an orthogonal matrix $\mathbf{R}$ that corresponds to a rotation about the axis $\hat{\boldsymbol{w}}$ through an angle of θ. It can be shown that the closed-form expression for $\mathrm{e}^{\mathbf{S}(\hat{\boldsymbol{w}})\theta}$, which can be efficiently computed, is

$$\mathrm{e}^{\mathbf{S}(\hat{\boldsymbol{w}})\theta} = \mathbf{1}_{3\times3} + \mathbf{S}(\hat{\boldsymbol{w}})\sin\theta + \mathbf{S}(\hat{\boldsymbol{w}})^2(1-\cos\theta) \,. \tag{2.25}$$

The components of $(\theta w_x\ \theta w_y\ \theta w_z)^{\mathrm{T}}$, which are related to the elements of the rotation matrix $\mathbf{R}$ in Table 2.2, are referred to as the *exponential coordinates* for $\mathbf{R}$.

Exponential Coordinates for Rigid-Body Motion

As indicated in Sect. 2.2.3, the position and orientation of a body can be expressed by the combination of a position vector $\boldsymbol{p} \in \mathbb{R}^3$ and a rotation matrix $\mathbf{R} \in SO(3)$. The product space of $\mathbb{R}^3$ with $SO(3)$ is the group known as $SE(3)$, which stands for *special Euclidean*.

$$SE(3) = \{(\boldsymbol{p}, \mathbf{R}) : \boldsymbol{p} \in \mathbb{R}^3, \mathbf{R} \in SO(3)\} = \mathbb{R}^3 \times SO(3) \,.$$

The set of homogeneous transformations satisfies the four axioms of a group:

- *Closure*: $\mathbf{T}_1\mathbf{T}_2 \in SE(3) \quad \forall \mathbf{T}_1, \mathbf{T}_2 \in SE(3)$
- *Identity*: $\mathbf{1}_{4\times4}\mathbf{T} = \mathbf{T}\mathbf{1}_{4\times4} = \mathbf{T} \quad \forall \mathbf{T} \in SE(3)$
- *Inverse*: the unique inverse of $\mathbf{T} \quad \forall \mathbf{T} \in SE(3)$ is given in (2.12)
- *Associativity*: $(\mathbf{T}_1\mathbf{T}_2)\mathbf{T}_3 = \mathbf{T}_1(\mathbf{T}_2\mathbf{T}_3) \quad \forall \mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3 \in SE(3)$.

In the screw transformation representation in Sect. 2.2.4, position and orientation are expressed by the angle θ of rotation about a screw axis defined by the unit vector $\hat{\boldsymbol{w}}$, the point $\boldsymbol{\rho}$ on the axis such that $\hat{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{\rho} = 0$, and the pitch h of the screw axis. The equivalent homogeneous transformation found in Table 2.4 can be expressed as the exponential map

$$\mathbf{T} = \mathrm{e}^{\hat{\boldsymbol{\xi}}\theta} = \mathbf{1}_{4\times4} + \hat{\boldsymbol{\xi}}\theta + \frac{(\hat{\boldsymbol{\xi}}\theta)^2}{2!} + \frac{(\hat{\boldsymbol{\xi}}\theta)^3}{3!} + \dots \,, \tag{2.26}$$

where

$$\hat{\boldsymbol{\xi}} = \begin{pmatrix} S(\hat{\boldsymbol{w}}) & \boldsymbol{v} \\ \mathbf{0}^{\mathrm{T}} & 0 \end{pmatrix} \tag{2.27}$$

is the generalization of the unit skew-symmetric matrix $\mathbf{S}(\hat{\boldsymbol{w}})$ known as a *twist*. The twist coordinates of $\hat{\boldsymbol{\xi}}$ are given by $\boldsymbol{\xi} := (\hat{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{v}^{\mathrm{T}})^{\mathrm{T}}$. It can be shown that the closed-form expression for $\mathrm{e}^{\hat{\boldsymbol{\xi}}\theta}$ is

$$\mathrm{e}^{\hat{\boldsymbol{\xi}}\theta} = \begin{pmatrix} \mathrm{e}^{\mathbf{S}(\hat{\boldsymbol{w}})\theta} & (\mathbf{1}_{3\times3} - \mathrm{e}^{\mathbf{S}(\hat{\boldsymbol{w}})\theta})(\hat{\boldsymbol{w}} \times \boldsymbol{v}) + \hat{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{v}\theta\hat{\boldsymbol{w}} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{pmatrix} . \tag{2.28}$$

Comparison of this result with the conversion between homogeneous and screw transformations in Table 2.4 yields

$$\boldsymbol{v} = \boldsymbol{\rho} \times \hat{\boldsymbol{w}} \tag{2.29}$$

and

$$h = \hat{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{v} \,. \tag{2.30}$$

Thus, the exponential map for a twist transforms the initial pose of a body into its final pose. It gives the relative rigid-body motion. The vector $\boldsymbol{\xi}\theta$ contains the *exponential coordinates* for the rigid-body transformation.

As for screw transformations, the case of pure translation is unique. In this case, $\hat{\boldsymbol{w}} = \mathbf{0}$, so

$$\mathrm{e}^{\hat{\boldsymbol{\xi}}\theta} = \begin{pmatrix} \mathbf{1}_{3\times3} & \theta\boldsymbol{v} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{pmatrix} . \tag{2.31}$$

2.2.6 Plücker Coordinates

A minimum of four coordinates are needed to define a line in space. The Plücker coordinates of a line form a six-dimensional (6-D) vector, so they are superabundant by two. They can be viewed as a pair of three-dimensional (3-D) vectors; one is parallel to the line, and the other is the *moment* of that vector about the origin. Thus, if $\boldsymbol{u}$ is any vector parallel to the line and $\boldsymbol{\rho}$ is the position of any point on the line relative to the origin, the Plücker coordinates (L, M, N, P, Q, and R) are given by

$$(L, M, N) = \boldsymbol{u}^{\mathrm{T}}; \; (P, Q, R) = (\boldsymbol{\rho} \times \boldsymbol{u})^{\mathrm{T}} \,. \tag{2.32}$$

For simply defining a line, the magnitude of $\boldsymbol{u}$ is not unique, nor is the component of $\boldsymbol{\rho}$ parallel to $\boldsymbol{u}$. Two auxiliary relationships are imposed to reduce the set to just four independent coordinates. One is that the scalar product of the two three-dimensional vectors is identically zero.

$$LP + MQ + NR \equiv 0 \,. \tag{2.33}$$

The other is the invariance of the line designated when the coordinates are all multiplied by the same scaling factor.

$$(L, M, N, P, Q, R) \equiv (kL, kM, kN, kP, kQ, kR) \,. \tag{2.34}$$

This relationship may take the form of constraining $\boldsymbol{u}$ to have unit magnitude so that L, M, and N are the direction cosines.

In this handbook, it is often useful to express velocities in Plücker coordinates, wherein unlike the definition of lines, the magnitudes of the two three-dimensional vectors are not arbitrary. This leads to the motor notation of *von Mises* [2.9, 19] and *Everett* [2.20]. For instantaneously coincident coordinate frames, one fixed and the other embedded in the moving body, $\boldsymbol{\omega}$ is the angular velocity of the body and $\boldsymbol{v}_O$ is the velocity of the origin O of the body-fixed frame when both are expressed relative to the fixed frame. This provides a Plücker coordinate system for the spatial velocity $\mathbf{v}$ of the body. The Plücker coordinates of $\mathbf{v}$

are simply the Cartesian coordinates of $\boldsymbol{\omega}$ and $\boldsymbol{v}_O$,

$$\mathbf{v} = \begin{pmatrix} \boldsymbol{\omega} \\ \boldsymbol{v}_O \end{pmatrix} . \tag{2.35}$$

The transformation from Plücker coordinate system i to Plücker coordinate system j for spatial velocities is achieved with the spatial transform ${}^{j}\mathbf{X}_i$. If $\mathbf{v}_i$ and $\mathbf{v}_j$ denote the spatial velocities of a body relative to the i and j frames, respectively, and ${}^{j}\boldsymbol{p}_i$ and ${}^{j}\mathbf{R}_i$ denote the position and orientation of frame i relative to frame j,

$$\mathbf{v}_j = {}^{j}\mathbf{X}_i \mathbf{v}_i \, , \tag{2.36}$$

where

$${}^{j}\mathbf{X}_i = \begin{pmatrix} {}^{j}\mathbf{R}_i & \mathbf{0}_{3\times 3} \\ \mathbf{S}({}^{j}\boldsymbol{p}_i)\,{}^{j}\mathbf{R}_i & {}^{j}\mathbf{R}_i \end{pmatrix} , \tag{2.37}$$

such that

$${}^{j}\mathbf{X}_i{}^{-1} = {}^{i}\mathbf{X}_j = \begin{pmatrix} {}^{i}\mathbf{R}_j & \mathbf{0}_{3\times 3} \\ -{}^{i}\mathbf{R}_j\mathbf{S}({}^{j}\boldsymbol{p}_i) & {}^{i}\mathbf{R}_j \end{pmatrix} , \tag{2.38}$$

and

$${}^{k}\mathbf{X}_i = {}^{k}\mathbf{X}_j {}^{j}\mathbf{X}_i \, , \tag{2.39}$$

and $\mathbf{S}({}^{j}\boldsymbol{p}_i)$ is the skew-symmetric matrix

$$\begin{pmatrix} 0 & -{}^{j}p_i^z & {}^{j}p_i^y \\ {}^{j}p_i^z & 0 & -{}^{j}p_i^x \\ -{}^{j}p_i^y & {}^{j}p_i^x & 0 \end{pmatrix} . \tag{2.40}$$

Spatial vector notation, which includes the spatial velocities and transforms briefly mentioned here, is treated in greater depth in Sect. 3.2. Specifically, Table 3.1 gives a computationally efficient algorithm for applying a spatial transform.

2.3 Joint Kinematics

Unless explicitly stated otherwise, the kinematic description of robotic mechanisms typically employs a number of idealizations. The links that compose the robotic mechanism are assumed to be perfectly rigid bodies having surfaces that are geometrically perfect in both position and shape. Accordingly, these rigid bodies are connected together at joints where their idealized surfaces are in ideal contact without any clearance between them. The respective geometries of these surfaces in contact determine the freedom of motion between the two links, or the *joint kinematics*.

A kinematic joint is a connection between two bodies that constrains their relative motion. Two bodies that are in contact with one another create a simple kinematic joint. The surfaces of the two bodies that are in contact are able to move over one another, thereby permitting relative motion of the two bodies. Simple kinematic joints are classified as lower pair joints if contact occurs over surfaces [2.21] and as higher pair joints if contact occurs only at points or along lines.

A joint model describes the motion of a frame fixed in one body of a joint relative to a frame fixed in the other body. The motion is expressed as a function of the joint's motion variables, and other elements of a joint model include the rotation matrix, position vector, free modes, and constrained modes. The free modes of a joint define the directions in which motion is allowed. They are represented by the $6 \times n_i$ matrix $\boldsymbol{\Phi}_i$ whose columns are the Plücker coordinates of the allowable motion. This matrix relates the spatial velocity vector across the joint $\boldsymbol{v}_{\mathrm{rel,i}}$ to the joint velocity vector $\dot{\boldsymbol{q}}_i$,

$$\boldsymbol{v}_{\mathrm{rel,i}} = \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i \, . \tag{2.41}$$

In contrast, the constrained modes of a joint define the directions in which motion is not allowed. They are represented by the $6 \times (6 - n_i)$ matrix $\boldsymbol{\Phi}_i^{\mathrm{c}}$ that is complementary to $\boldsymbol{\Phi}_i$. Tables 2.5 and 2.6 contain the formulas of the joint models for all of the joints described in this section. They are used extensively for the dynamic analysis presented in Chap. 3. Additional information on joints can be found in Chap. 4.

2.3.1 Lower Pair Joints

Lower pair joints are mechanically attractive since wear is spread over the whole surface and lubricant is trapped in the small clearance space (in nonidealized systems) between the surfaces, resulting in relatively good lubrication. As can be proved [2.23] from the requirement for surface contact, there are only six possible forms of lower pair joints: revolute, prismatic, helical, cylindrical, spherical, and planar joints.

Revolute

The most general form of a revolute joint, often abbreviated as *R* and sometimes referred to colloquially as a hinge or pin joint, is a lower pair composed of two

Table 2.5 Joint model formulas for one-degree-of-freedom lower pair joints, with abbreviations $c_{\theta_i} := \cos\theta_i$ and $s_{\theta_i} := \sin\theta_i$ (adapted in part from Table 4.1 in [2.22])

Joint type	Joint rotation matrix ${}^{j}\mathbf{R}_i$	Position vector ${}^{j}\boldsymbol{p}_i$	Free modes $\boldsymbol{\Phi}_i$	Constrained modes $\boldsymbol{\Phi}_i^{\mathrm{c}}$	Pose state vars.	$\dot{\boldsymbol{q}}_i$
Revolute R	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	θ_i	$\dot{\theta}_i$
Prismatic P	$\mathbf{1}_{3\times 3}$	$\begin{pmatrix} 0 \\ 0 \\ d_i \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$	d_i	$\dot{d}_i$
Helical H (pitch h)	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ h\theta_i \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ h \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -h \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	θ_i	$\dot{\theta}_i$

congruent surfaces of revolution. The surfaces are the same except one of them is an external surface, convex in any plane normal to the axis of revolution, and one is an internal surface, concave in any plane normal to the axis. The surfaces may not be solely in the form of right circular cylinders, since surfaces of that form do not provide any constraint on axial sliding. A revolute joint permits only rotation of one of the bodies joined relative to the other. The position of one body relative to the other may be expressed as the angle between two lines normal to the joint axis, one fixed in each body. Thus, the joint has one *degree of freedom* (DOF). When the $\hat{\boldsymbol{z}}$ axis of coordinate frame i is aligned with a revolute joint axis, the formulas in Table 2.5 define the revolute joint model.

Prismatic

The most general form of a prismatic joint, often abbreviated as *P* and sometimes referred to colloquially as a sliding joint, is a lower pair formed from two congruent general cylindrical surfaces. These may not be right circular cylindrical surfaces. A general cylindrical surface is obtained by extruding any curve in a constant direction. Again, one surface is internal and the other is an external surface. A prismatic joint permits only sliding of one of the members joined relative to the other along the direction of extrusion. The position of one body relative to the other is determined by the distance between two points on a line parallel to the direction of sliding, with one point fixed in each body. Thus, this joint also has one degree of freedom. When the $\hat{\boldsymbol{z}}$ axis of coordinate frame i is aligned with a prismatic joint axis, the formulas in Table 2.5 define the prismatic joint model.

Helical

The most general form of a helical joint, often abbreviated as *H* and sometimes referred to colloquially as a screw joint, is a lower pair formed from two helicoidal surfaces formed by extruding any curve along a helical path. The simple example is a bolt and nut wherein the basic generating curve is a pair of straight lines. The angle θ of rotation about the axis of the helical joint is directly related to the distance d of displacement of one body relative to the other along that axis by the expression $d = h\theta$, where the constant h is called the pitch of the helical joint. When the $\hat{\boldsymbol{z}}$ axis of coordinate frame i is aligned with a helical joint axis, the formulas in Table 2.5 define the helical joint model.

Cylindrical

A cylindrical joint, often abbreviated as C, is a lower pair formed by contact of two congruent right circular cylinders, one an internal surface and the other an external surface. It permits both rotation about the cylinder axis and sliding parallel to it. Therefore, it is a joint with two degrees of freedom. Lower pair joints with more than one degree of freedom are easily replaced by kinematically equivalent compound joints (Sect. 2.3.3) that are serial chains of one-degree-of-freedom lower pairs. In the present case, the cylindrical joint can be replaced by a revolute in series with a pris-

Table 2.6 Joint model formulas for higher-degree-of-freedom lower pair joints, universal joint, rolling contact joint, and 6-DOF joint, with abbreviations $c_{\theta_i} := \cos\theta_i$ and $s_{\theta_i} := \sin\theta_i$ (adapted in part from Table 4.1 in [2.22]) *The Euler angles α_i, β_i, and γ_i could be used in place of the unit quaternion $\boldsymbol{\epsilon}_i$ to represent orientation

Joint type	Joint rotation matrix ${}^jR_i$	Position vector ${}^jp_i$	Free modes Φ_i	Constrained modes Φ_i^c	Pose Variables	Velocity Variables $\dot{q}_i$
Cylindrical C	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ d_i \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	θ_i d_i	$\begin{pmatrix} \dot{\theta}_i \\ \dot{d}_i \end{pmatrix}$
Spherical* S	$\begin{pmatrix} \text{Table 2.1} \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\boldsymbol{\epsilon}_i$	$\boldsymbol{\omega}_{i\,\mathrm{rel}}$
Planar	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} c_{\theta_i} d_{xi} - s_{\theta_i} d_{yi} \\ s_{\theta_i} d_{xi} + c_{\theta_i} d_{yi} \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	θ_i d_{xi} d_{yi}	$\begin{pmatrix} \dot{\theta}_i \\ \dot{d}_{xi} \\ \dot{d}_{yi} \end{pmatrix}$
Flat planar rolling contact (fixed radius r)	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} r\theta_i c_{\theta_i} - r s_{\theta_i} \\ -r\theta_i s_{\theta_i} - r c_{\theta_i} \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \\ r \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -r & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	θ_i	$\dot{\theta}_i$
Universal U	$\begin{pmatrix} c_{\alpha_i} c_{\beta_i} & -s_{\alpha_i} & c_{\alpha_i} s_{\beta_i} \\ s_{\alpha_i} c_{\beta_i} & c_{\alpha_i} & s_{\alpha_i} s_{\beta_i} \\ -s_{\beta_i} & 0 & c_{\beta_i} \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} -s_{\beta_i} & 0 \\ 0 & 1 \\ c_{\beta_i} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} c_{\beta_i} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ s_{\beta_i} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	α_i β_i	$\begin{pmatrix} \dot{\alpha}_i \\ \dot{\beta}_i \end{pmatrix}$
6-DOF*	$\begin{pmatrix} \text{see} \\ \text{Table 2.1} \end{pmatrix}$	${}^0p_i$	$\mathbf{1}_{6\times 6}$		$\boldsymbol{\varepsilon}_i$ ${}^0p_i$	$\begin{pmatrix} \boldsymbol{\omega}_i \\ v_i \end{pmatrix}$

matic joint whose direction of sliding is parallel to the revolute axis. While simpler to implement using the geometric representation discussed in Sect. 2.4, this approach has disadvantages for dynamic simulation. Modeling a single cylindrical joint as a combination of a prismatic and a revolute joint requires the addition of a virtual link between the two with zero mass and zero length. The massless link can create computational problems. When the $\hat{z}$ axis of coordinate frame i is aligned with a cylindrical joint axis, the formulas in Table 2.6 define the cylindrical joint model.

Spherical

A spherical joint, often abbreviated as S, is a lower pair formed by contact of two congruent spherical surfaces. Once again, one is an internal surface, and the other is an external surface. A spherical joint permits rotation about any line through the center of the sphere. Thus, it permits independent rotation about axes in up to three different directions and has three degrees of freedom. A spherical joint is easily replaced by a kinematically equivalent compound joint consisting of three revolutes that have axes that all intersect in a single point – the center of the spherical joint. The revolute joint axes

do not need to be successively orthogonal, but often they are implemented that way. The arrangement is, in general, kinematically equivalent to a spherical joint, but it does exhibit a singularity when the revolute joint axes become coplanar. This is as compared to the native spherical joint that never has such a singularity. Likewise, if a spherical joint is modeled in simulation as three revolutes, computational difficulties again can arise from the necessary inclusion of massless virtual links having zero length. The joint model formulas of a spherical joint are given in Table 2.6.

Planar

A planar joint is formed by planar contacting surfaces. Like the spherical joint, it is a lower pair joint with three degrees of freedom. A kinematically equivalent compound joint consisting of a serial chain of three revolutes with parallel axes can replace a planar joint. As was the case with the spherical joint, the compound joint exhibits a singularity when the revolute axes become coplanar. When the $\hat{z}$ axis of coordinate frame i is aligned with the normal to the plane of contact, the formulas in Table 2.6 define the planar joint model.

2.3.2 Higher Pair Joints

Some higher pair joints also have attractive properties, particularly rolling pairs in which one body rolls without slipping over the surface of the other. This is mechanically attractive since the absence of sliding means the absence of abrasive wear. However, since ideal contact occurs at a point, or along a line, application of a load across the joint may lead to very high local stresses resulting in other forms of material failure and, hence, wear. Higher pair joints can be used to create kinematic joints with special geometric properties, as in the case of a gear pair or a cam and follower pair.

Rolling Contact

Rolling contact actually encompasses several different geometries. Rolling contact in planar motion permits one degree of freedom of relative motion as in the case of a roller bearing, for example. Planar rolling contact can take place along a line, thereby spreading the load and wear somewhat. Three-dimensional rolling contact allows rotation about any axis through the point of contact that is, in principle, unique. Hence, a three-dimensional rolling contact pair permits relative motion with three degrees of freedom. When the $\hat{z}$ axis of coordinate frame i is aligned with the axis of rotation and passes through the center of the roller of fixed radius r, the formulas in Table 2.6 define the planar rolling contact joint model for a roller on a flat surface.

Regardless of whether the joint is planar or three-dimensional, the *no-slip* condition associated with a rolling contact joint requires that the instantaneous relative velocity between the points on the two bodies in contact be zero. If P is the point of rolling contact between bodies i and j,

$$\boldsymbol{v}_{\mathrm{P}_i/\mathrm{P}_j} = 0\,. \tag{2.42}$$

Likewise, relative acceleration is in the direction of the common normal to the two surfaces at the point of contact. Because the constraint associated with the joint is expressed in terms of velocity and cannot be expressed in terms of position alone, it is nonholonomic, as discussed in Sect. 2.3.6. A more detailed discussion of the kinematic constraints for rolling contact is found in Chap. 24.

2.3.3 Compound Joints

Compound kinematic joints are connections between two bodies formed by chains of other members and simple kinematic joints. A compound joint may constrain the relative motion of the two bodies joined in the same way as a simple joint. In such a case, the two joints are said to be kinematically equivalent.

Universal

A universal joint, often abbreviated as *U* and referred to as a Cardan or Hooke joint, is a compound joint with two degrees of freedom. It consists of a serial chain of two revolutes whose axes intersect orthogonally. The joint model for a universal joint, in which, from Euler angle notation, α_i is the first rotation about the Z-axis and then β_i is the rotation about the Y-axis, is given in Table 2.6. This is a joint for which the matrices $\boldsymbol{\Phi}_i$ and $\boldsymbol{\Phi}_i^{\mathrm{c}}$ are not constant, so in general, $\dot{\boldsymbol{\Phi}}_i \neq 0$ and $\dot{\boldsymbol{\Phi}}_i^{\mathrm{c}} \neq 0$. As seen in Table 2.6, the orientation of the first joint axis (expressed in the outboard coordinate frame) varies with β_i.

2.3.4 6-DOF Joint

The motion of two bodies not jointed together can be modeled as a six-degree-of-freedom *joint* that introduces no constraints. This is particularly useful for mobile robots, such as aircraft, that make at most intermittent contact with the ground, and thus, a body in free motion relative to the fixed frame is termed a *floating base*. Such a free motion joint model enables the position and orientation of a floating base in space to be expressed with six joint variables. The 6-DOF joint model is included in Table 2.6.

2.3.5 Physical Realization

In an actual robotic mechanism, the joints may have physical limits beyond which motion is prohibited. The workspace (Sect. 2.5) of a robotic manipulator is determined by considering the combined limits and freedom of motion of all the joints within the mechanism. Revolute joints are easily actuated by rotating motors and are, therefore, extremely common in robotic systems. They may also be present as passive, unactuated joints. Also common, although less so than revolutes, prismatic joints are relatively easily actuated by means of linear actuators such as hydraulic or pneumatic cylinders, ball screws, or screw jacks. They always have motion limits since unidirectional sliding can, in principle, produce infinite displacements. Helical joints are most often found in robotic mechanisms as constituents of linear actuators such as screw jacks and ball screws and are seldom used as primary kinematic joints. Joints with more than one degree of freedom are generally used passively in robotic mechanisms because each degree of freedom of an active joint must be separately actuated. Passive spherical joints are quite often found in robotic mechanisms, while passive planar joints are only occasionally found. The effect of an actuated spherical joint is achieved by employing the kinematically equivalent combination of three revolutes and actuating each. Universal joints are used in robotic mechanisms in both active and passive forms.

Serial chains are commonly denoted by the abbreviations for the joints they contain in the order in which they appear in the chain. For example, an RPR chain contains three links, the first jointed to the base with a revolute and to the second with a prismatic, while the second and third are jointed together with another revolute. If all of the joints are identical, the notation consists of the number of joints preceding the joint abbreviation, such as 6R for a six-axis serial-chain manipulator containing only revolute joints.

Joints are realized with hardware that is more complex than the idealizations presented in Sects. 2.3.1 and 2.3.2. For example, a revolute joint may be achieved with a ball bearing composed of a set of bearing balls trapped between two journals. The balls ideally roll without slipping on the journals, thereby taking advantage of the special properties of rolling contact joints. A prismatic joint may be realized by means of a roller-rail assembly.

2.3.6 Holonomic and Nonholonomic Constraints

With the exception of rolling contact, all of the constraints associated with the joints discussed in the preceding sections can be expressed mathematically by equations containing only the joint position variables. These are called holonomic constraints. The number of equations, and hence the number of constraints, is $6-n$, where n is the number of degrees of freedom of the joint. The constraints are intrinsically part of the axial joint model.

A nonholonomic constraint is one that cannot be expressed in terms of the position variables alone, but includes the time derivative of one or more of those variables. These constraint equations cannot be integrated to obtain relationships solely between the joint variables. The most common example in robotic systems arises from the use of a wheel or roller that rolls without slipping on another member. Nonholonomic constraints, particularly as they apply to wheeled robots, are discussed in more detail in Chap. 24.

2.3.7 Generalized Coordinates

In a robotic mechanism consisting of N bodies, $6N$ coordinates are required to specify the position and orientation of all the bodies relative to a coordinate frame. Since some of those bodies are jointed together, a number of constraint equations will establish relationships among some of these coordinates. In this case, the $6N$ coordinates can be expressed as functions of a smaller set of coordinates $\boldsymbol{q}$ that are all independent. The coordinates in this set are known as generalized coordinates, and motions associated with these coordinates are consistent with all of the constraints. The joint variables $\boldsymbol{q}$ of a robotic mechanism form a set of generalized coordinates [2.24, 25].

2.4 Geometric Representation

The geometry of a robotic mechanism is conveniently defined by attaching coordinate frames to each link. While these frames could be located arbitrarily, it is advantageous both for consistency and computational efficiency to adhere to a convention for locating the frames on the links. *Denavit* and *Hartenberg* [2.26] introduced the foundational convention that has been adapted in a number of different ways, one of which is the convention introduced by *Khalil* and *Dombre* [2.27] used throughout this handbook. In all of its forms, the convention requires only four rather than six parameters to locate one coordinate frame relative to another. The

four parameters consist of two link parameters, the link length a_i and the link twist α_i, and two joint parameters, the joint offset d_i and the joint angle θ_i. This parsimony is achieved through judicious placement of the coordinate frame origins and axes such that the $\hat{\boldsymbol{x}}$ axis of one frame both intersects and is perpendicular to the $\hat{\boldsymbol{z}}$ axis of the following coordinate frame. The convention is applicable to robotic mechanisms consisting of revolute and prismatic joints, so when multiple-degree-of-freedom joints are present, they are modeled as combinations of revolute and prismatic joints, as discussed in Sect. 2.3.

There are essentially four different forms of the convention for locating coordinate frames in a robotic mechanism. Each exhibits its own advantages by managing trade-offs of intuitive presentation. In the original *Denavit* and *Hartenberg* [2.26] convention, joint i is located between links i and $i+1$, so it is on the outboard side of link i. Also, the joint offset d_i and joint angle θ_i are measured along and about the $i-1$ joint axis, so the subscripts of the joint parameters do not match that of the joint axis. *Waldron* [2.28] and *Paul* [2.29] modified the labeling of axes in the original convention such that joint i is located between links $i-1$ and i in order to make it consistent with the base member of a serial chain being member 0. This places joint i at the inboard side of link i and is the convention used in all of the other modified versions. Furthermore, Waldron and Paul addressed the mismatch between subscripts of the joint parameters and joint axes by placing the $\hat{\boldsymbol{z}}_i$ axis along the $i+1$ joint axis. This, of course, relocates the subscript mismatch to the correspondence between the joint axis and the $\hat{\boldsymbol{z}}$ axis of the coordinate frame. *Craig* [2.30] eliminated all of the subscript mismatches by placing the $\hat{\boldsymbol{z}}_i$ axis along joint i, but at the expense of the homogeneous transformation ${}^{i-1}\mathbf{T}_i$ being formed with a mixture of joint parameters with subscript i and link parameters with subscript $i-1$. *Khalil* and *Dombre* [2.27] introduced another variation similar to Craig's except that it defines the link parameters a_i and α_i along and about the $\hat{\boldsymbol{x}}_{i-1}$ axis. In this case, the homogeneous transformation ${}^{i-1}\mathbf{T}_i$ is formed only by parameters with subscript i, and the subscript mismatch is such that a_i and α_i indicate the length and twist of link $i-1$ rather than link i. Thus, in summary, the advantages of the convention used throughout this handbook compared to the alternative conventions are that the $\hat{\boldsymbol{z}}$ axes of the coordinate frames share the common subscript of the joint axes and the four parameters that define the spatial transform from coordinate frame i to coordinate frame $i-1$ all share the common subscript i.

In this handbook, the convention for serial chain mechanisms is shown in Fig. 2.2 and summarized as follows. The numbering of bodies and joints follows the convention:

- The N moving bodies of the robotic mechanism are numbered from 1 to N. The number of the base is 0.
- The N joints of the robotic mechanism are numbered from 1 to N, with joint i located between members $i-1$ and i.

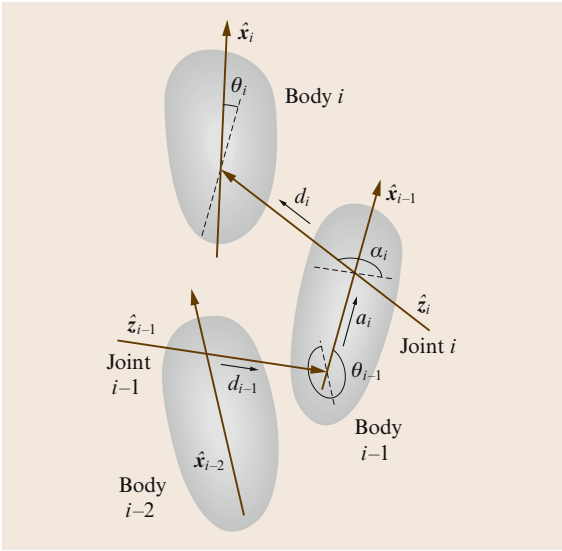


Fig. 2.2 Schematic of the numbering of bodies and joints in a robotic mechanism, the convention for attaching coordinate frames to the bodies, and the definitions of the four parameters, a_i, α_i, d_i, and θ_i, that locate one frame relative to another

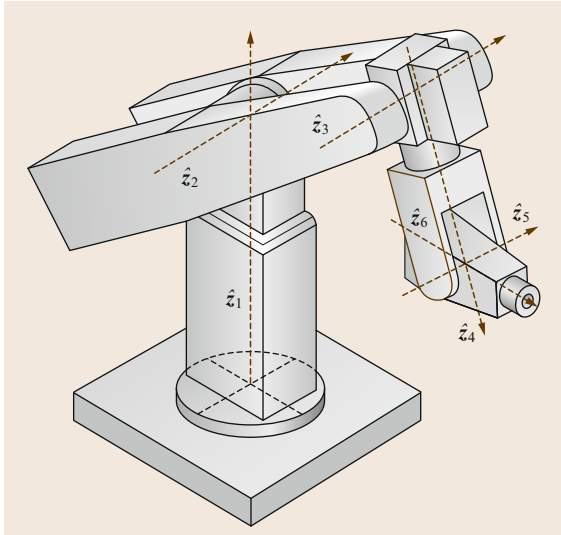


Fig. 2.3 Example six-degree-of-freedom serial chain manipulator composed of an articulated arm with no joint offsets and a spherical wrist

Table 2.7 Geometric parameters of the example serial chain manipulator in Fig. 2.3

i	α_i	a_i	d_i	θ_i
1	0	0	0	θ_1
2	$-\frac{\pi}{2}$	0	0	θ_2
3	0	a_3	0	θ_3
4	$-\frac{\pi}{2}$	0	d_4	θ_4
5	$\frac{\pi}{2}$	0	0	θ_5
6	$-\frac{\pi}{2}$	0	0	θ_6

With this numbering scheme, the attachment of coordinate frames follows the convention:

- The $\hat{\boldsymbol{z}}_i$ axis is located along the axis of joint i,
- The $\hat{\boldsymbol{x}}_{i-1}$ axis is located along the common normal between the $\hat{\boldsymbol{z}}_{i-1}$ and $\hat{\boldsymbol{z}}_i$ axes.

Using the attached frames, the four parameters that locate one frame relative to another are defined as:

- a_i is the distance from $\hat{\boldsymbol{z}}_{i-1}$ to $\hat{\boldsymbol{z}}_i$ along $\hat{\boldsymbol{x}}_{i-1}$,
- α_i is the angle from $\hat{\boldsymbol{z}}_{i-1}$ to $\hat{\boldsymbol{z}}_i$ about $\hat{\boldsymbol{x}}_{i-1}$,
- d_i is the distance from $\hat{\boldsymbol{x}}_{i-1}$ to $\hat{\boldsymbol{x}}_i$ along $\hat{\boldsymbol{z}}_i$,
- θ_i is the angle from $\hat{\boldsymbol{x}}_{i-1}$ to $\hat{\boldsymbol{x}}_i$ about $\hat{\boldsymbol{z}}_i$.

The geometric parameters for the example manipulator shown in Fig. 2.3 are listed in Table 2.7. All of the joints of this manipulator are revolutes, and joint 1 has a vertical orientation. Joint 2 is perpendicular to joint 1 and intersects it. Joint 3 is parallel to joint 2, and the length of link 2 is a_3. Joint 4 is perpendicular to joint 3 and intersects it. Joint 5 likewise intersects joint 4 perpendicularly at an offset of d_4 from joint 3. Finally, joint 6 intersects joint 5 perpendicularly.

With this convention, coordinate frame i can be located relative to coordinate frame $i-1$ by executing a rotation through an angle α_i about the $\hat{\boldsymbol{x}}_{i-1}$ axis, a translation of distance a_i along $\hat{\boldsymbol{x}}_{i-1}$, a rotation through an angle θ_i about the $\hat{\boldsymbol{z}}_i$ axis, and a translation of distance d_i along $\hat{\boldsymbol{z}}_i$. Through concatenation of these individual transformations,

$$\mathbf{Rot}(\hat{\boldsymbol{x}}_{i-1}, \alpha_i)\mathbf{Trans}(\hat{\boldsymbol{x}}_{i-1}, a_i)\mathbf{Rot}(\hat{\boldsymbol{z}}_i, \theta_i)\,\mathbf{Trans}(\hat{\boldsymbol{z}}_i, d_i)\,, \tag{2.43}$$

the equivalent homogeneous transformation is,

$$^{i-1}\mathbf{T}_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_i \\ \sin\theta_i\cos\alpha_i & \cos\theta_i\cos\alpha_i & -\sin\alpha_i & -\sin\alpha_i d_i \\ \sin\theta_i\sin\alpha_i & \cos\theta_i\sin\alpha_i & \cos\alpha_i & \cos\alpha_i d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.44}$$

The identification of geometric parameters is addressed in Chap. 8.

2.5 Workspace

Most generally, the workspace of a robotic manipulator is the total volume swept out by the end-effector as the manipulator executes all possible motions. The workspace is determined by the geometry of the manipulator and the limits of the joint motions. It is more specific to define the reachable workspace as the total locus of points at which the end-effector can be placed and the dextrous workspace [2.31] as the subset of those points at which the end-effector can be placed while having an arbitrary orientation. Dexterous workspaces exist only for certain idealized geometries, so real industrial manipulators with joint motion limits almost never possess dexterous workspaces.

Many serial-chain robotic manipulators are designed such that their joints can be partitioned into a regional structure and an orientation structure. The joints in the regional structure accomplish the positioning of the end-effector in space, and the joints in the orientation structure accomplish the orientation of the end-effector. Typically, the inboard joints of a serial chain manipulator comprise the regional structure, while the outboard joints comprise the orientation structure. Also, since prismatic joints provide no capability for rotation, they are generally not employed within the orientation structure.

The regional workspace volume can be calculated from the known geometry of the serial-chain manipulator and motion limits of the joints. With three inboard joints comprising the regional structure, the area of workspace for the outer two (joints 2 and 3) is computed first, and then the volume is calculated by integrating over the joint variable of the remaining inboard joint (joint 1). In the case of a prismatic joint, this simply involves multiplying the area by the total length of travel of the prismatic joint. In the more common case of a revolute joint, it involves rotating the area about the joint axis through the full range of motion of the revolute [2.32]. By the theorem of Pappus, the associated volume V is

$$V = A\bar{r}\gamma\,, \tag{2.45}$$

where A is the area, $\bar{r}$ is the distance from the area's centroid to the axis, and γ is the angle through which the area is rotated. The boundaries of the area are determined by tracing the motion of a reference point in the end-effector, typically the center of rotation of the wrist that serves as the orientation structure. Starting with each of the two joints at motion limits and with joint 2 locked, joint 3 is moved until its second motion limit is reached. Joint 3 is then locked, and joint 2 is freed to move to its second motion limit. Joint 2 is again locked, while joint 3 is freed to move back to its original motion limit. Finally, joint 3 is locked, and joint 2 freed to move likewise to its original motion limit. In this way, the trace of the reference point is a closed curve whose area and centroid can be calculated mathematically.

More details on manipulator workspace can be found in Chaps. 4 and 16.

2.6 Forward Kinematics

The forward kinematics problem for a serial-chain manipulator is to find the position and orientation of the end-effector relative to the base given the positions of all of the joints and the values of all of the geometric link parameters. Often, a frame fixed in the end-effector is referred to as the *tool frame*, and while fixed in the final link N, it in general has a constant offset in both position and orientation from frame N. Likewise, a *station frame* is often located in the base to establish the location of the task to be performed. This frame generally has a constant offset in its pose relative to frame 0, which is also fixed in the base.

A more general expression of the forward kinematics problem is to find the relative position and orientation of any two designated members given the geometric structure of the robotic mechanism and the values of a number of joint positions equal to the number of degrees of freedom of the mechanism. The forward kinematics problem is critical for developing manipulator coordination algorithms because joint positions are typically measured by sensors mounted on the joints and it is necessary to calculate the positions of the joint axes relative to the fixed frame.

In practice, the forward kinematics problem is solved by calculating the transformation between a coordinate frame fixed in the end-effector and another coordinate frame fixed in the base, i. e., between the tool and station frames. This is straightforward for a serial chain since the transformation describing the position of the end-effector relative to the base is obtained by simply concatenating transformations between frames fixed in adjacent links of the chain. The convention for the geometric representation of a manipulator presented in Sect. 2.4 reduces this to finding an equivalent 4×4 homogeneous transformation matrix that relates the spatial displacement of the end-effector coordinate frame to the base frame.

For the example serial-chain manipulator shown in Fig. 2.3 and neglecting the addition of tool and station

Table 2.8 Forward kinematics of the example serial chain manipulator in Fig. 2.3, with abbreviations $c_{\theta_i} := \cos\theta_i$ and $s_{\theta_i} := \sin\theta_i$

$$
{}^0\mathbf{T}_6 = \begin{pmatrix} r_{11} & r_{12} & r_{13} & {}^0p_6^x \\ r_{21} & r_{22} & r_{23} & {}^0p_6^y \\ r_{31} & r_{32} & r_{33} & {}^0p_6^z \\ 0 & 0 & 0 & 1 \end{pmatrix},
$$

$$
\begin{aligned}
r_{11} &= c_{\theta_1}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})(s_{\theta_4}s_{\theta_6} - c_{\theta_4}c_{\theta_5}c_{\theta_6}) \\
&\quad - c_{\theta_1}s_{\theta_5}c_{\theta_6}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\
&\quad + s_{\theta_1}(s_{\theta_4}c_{\theta_5}c_{\theta_6} + c_{\theta_4}s_{\theta_6})\,, \\
r_{21} &= s_{\theta_1}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})(s_{\theta_4}s_{\theta_6} - c_{\theta_4}c_{\theta_5}c_{\theta_6}) \\
&\quad - s_{\theta_1}s_{\theta_5}c_{\theta_6}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\
&\quad - c_{\theta_1}(s_{\theta_4}c_{\theta_5}c_{\theta_6} + c_{\theta_4}s_{\theta_6})\,, \\
r_{31} &= (c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3})(s_{\theta_4}s_{\theta_6} - c_{\theta_4}c_{\theta_5}c_{\theta_6}) \\
&\quad + s_{\theta_5}c_{\theta_6}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})\,, \\
r_{12} &= c_{\theta_1}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})(c_{\theta_4}c_{\theta_5}s_{\theta_6} + s_{\theta_4}c_{\theta_6}) \\
&\quad + c_{\theta_1}s_{\theta_5}s_{\theta_6}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\
&\quad + s_{\theta_1}(c_{\theta_4}c_{\theta_6} - s_{\theta_4}c_{\theta_5}s_{\theta_6})\,, \\
r_{22} &= s_{\theta_1}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})(c_{\theta_4}c_{\theta_5}s_{\theta_6} + s_{\theta_4}c_{\theta_6}) \\
&\quad + s_{\theta_1}s_{\theta_5}s_{\theta_6}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\
&\quad - c_{\theta_1}(c_{\theta_4}c_{\theta_6} - s_{\theta_4}c_{\theta_5}s_{\theta_6})\,, \\
r_{32} &= (c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3})(c_{\theta_4}c_{\theta_5}s_{\theta_6} + s_{\theta_4}c_{\theta_6}) \\
&\quad - s_{\theta_5}s_{\theta_6}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})\,, \\
r_{13} &= c_{\theta_1}c_{\theta_4}s_{\theta_5}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3}) \\
&\quad - c_{\theta_1}c_{\theta_5}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\
&\quad - s_{\theta_1}s_{\theta_4}s_{\theta_5}\,, \\
r_{23} &= s_{\theta_1}c_{\theta_4}s_{\theta_5}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3}) \\
&\quad - s_{\theta_1}c_{\theta_5}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) + c_{\theta_1}s_{\theta_4}s_{\theta_5}\,, \\
r_{33} &= c_{\theta_4}s_{\theta_5}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\
&\quad + c_{\theta_5}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})\,, \\
{}^0p_6^x &= a_3c_{\theta_1}c_{\theta_2} - d_4c_{\theta_1}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3})\,, \\
{}^0p_6^y &= a_3s_{\theta_1}c_{\theta_2} - d_4s_{\theta_1}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3})\,, \\
{}^0p_6^z &= -a_3s_{\theta_2} + d_4(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})\,.
\end{aligned}
$$

frames, the transformation is

$$ {}^0\mathbf{T}_6 = {}^0\mathbf{T}_1\,{}^1\mathbf{T}_2\,{}^2\mathbf{T}_3\,{}^3\mathbf{T}_4\,{}^4\mathbf{T}_5\,{}^5\mathbf{T}_6 \,. \tag{2.46} $$

Table 2.8 contains the elements of ${}^0\mathbf{T}_6$ that are calculated using Table 2.7 and (2.44).

Once again, homogeneous transformations provide a compact notation, but are computationally inefficient for solving the forward kinematics problem. A reduction in computation can be achieved by separating the position and orientation portions of the transformation to eliminate all multiplications by the 0 and 1 elements of the matrices. In Chap. 3, calculations are made using the spatial vector notation briefly introduced here in Sect. 2.2.6 and explained in detail in Sect. 3.2. This approach does not employ homogeneous transformations, but rather separates out the rotation matrices and positions to achieve computation efficiency. Table 3.1 provides the detailed formulas, with the product of spatial transforms particularly relevant to the forward kinematics problem.

Kinematic trees are the general structure of robotic mechanisms that do not contain closed loops, and the forward kinematics of tree structures are addressed in Chap. 3. The forward kinematics problem for closed chains is much more complicated because of the additional constraints present. Solution methods for closed chains are included in Chap. 18.

2.7 Inverse Kinematics

The inverse kinematics problem for a serial-chain manipulator is to find the values of the joint positions given the position and orientation of the end-effector relative to the base and the values of all of the geometric link parameters. Once again, this is a simplified statement applying only to serial chains. A more general statement is: given the relative positions and orientations of two members of a mechanism, find the values of all of the joint positions. This amounts to finding all of the joint positions given the homogeneous transformation between the two members of interest.

In the common case of a six-degree-of-freedom serial chain manipulator, the known transformation is ${}^0\mathbf{T}_6$. Reviewing the formulation of this transformation in Sect. 2.6, it is clear that the inverse kinematics problem for serial-chain manipulators requires the solution of sets of nonlinear equations. In the case of a six-degree-of-freedom manipulator, three of these equations relate to the position vector within the homogeneous transformation, and the other three relate to the rotation matrix. In the latter case, these three equations cannot come from the same row or column because of the dependency within the rotation matrix. With these nonlinear equations, it is possible that no solutions exist or multiple solutions exist [2.33]. For a solution to exist, the desired position and orientation of the end-effector must lie in the workspace of the manipulator. In cases where solutions do exist, they often cannot be presented in closed form, so numerical methods are required.

2.7.1 Closed-Form Solutions

Closed-form solutions are desirable because they are faster than numerical solutions and readily identify all possible solutions. The disadvantage of closed-form solutions is that they are not general, but robot dependent. The most effective methods for finding closed-form solutions are ad hoc techniques that take advantage of particular geometric features of specific mechanisms. In general, closed-form solutions can only be obtained for six-degree-of-freedom systems with special kinematic structure characterized by a large number of the geometric parameters defined in Sect. 2.4 being zero-valued. Most industrial manipulators have such structure because it permits more efficient coordination software. Sufficient conditions for a six-degree-of-freedom manipulator to have closed-form inverse kinematics solutions are [2.34–36]:

1. Three consecutive revolute joint axes intersect at a common point, as in a spherical wrist.
2. Three consecutive revolute joint axes are parallel.

Closed-form solution approaches are generally divided into algebraic and geometric methods.

Algebraic Methods

Algebraic methods involve identifying the significant equations containing the joint variables and manipulating them into a soluble form. A common strategy is reduction to a transcendental equation in a single variable such as,

$$ C_1 \cos\theta_i + C_2 \sin\theta_i + C_3 = 0 \,, \tag{2.47} $$

where C_1, C_2, and C_3 are constants. The solution to such an equation is

$$ \theta_i = 2\tan^{-1}\left(\frac{C_2 \pm \sqrt{C_2^2 - C_3^2 + C_1^2}}{C_1 - C_3}\right) . \tag{2.48} $$

Special cases in which one or more of the constants are zero are also common.

Reduction to a pair of equations having the form,

$$C_1 \cos\theta_i + C_2 \sin\theta_i + C_3 = 0\,, \tag{2.49}$$

$$C_1 \sin\theta_i - C_2 \cos\theta_i + C_4 = 0\,, \tag{2.50}$$

is another particularly useful strategy because only one solution results,

$$\theta_i = \mathrm{Atan2}(-C_1C_4 - C_2C_3, C_2C_4 - C_1C_3)\,. \tag{2.51}$$

Geometric Methods

Geometric methods involve identifying points on the manipulator relative to which position and/or orientation can be expressed as a function of a reduced set of the joint variables. This often amounts to decomposing the spatial problem into separate planar problems. The resulting equations are solved using algebraic manipulation. The two sufficient conditions for existence of a closed-form solution for a six-degree-of-freedom manipulator that are listed above enable the decomposition of the problem into inverse position kinematics and inverse orientation kinematics. This is the decomposition into regional and orientation structures discussed in Sect. 2.5, and the solution is found by rewriting (2.46),

$${}^0\mathbf{T}_6\,{}^6\mathbf{T}_5\,{}^5\mathbf{T}_4\,{}^4\mathbf{T}_3 = {}^0\mathbf{T}_1\,{}^1\mathbf{T}_2\,{}^2\mathbf{T}_3\,. \tag{2.52}$$

The example manipulator in Fig. 2.3 has this structure, and its regional structure is commonly known as an articulated or anthropomorphic arm or an elbow manipulator. The solution to the inverse position kinematics problem for such a structure is summarized in Table 2.9. Because there are two solutions for θ_1 and likewise two solutions for both θ_2 and θ_3 corresponding to each θ_1 solution, there are a total of four solutions to the inverse position kinematics problem of the articulated arm manipulator. The orientation structure is simply a spherical wrist, and the corresponding solution to the inverse orientation kinematics problem is summarized in Table 2.10. Two solutions for θ_5 are given in Table 2.10, but only one solution for both θ_4 and θ_6 corresponds to each. Thus, the inverse orientation kinematics problem of a spherical wrist has two solutions. Combining the regional and orientation structures, the total number of inverse kinematics solutions for the manipulator in Fig. 2.3 is eight.

2.7.2 Numerical Methods

Unlike the algebraic and geometric methods used to find closed-form solutions, numerical methods are not robot dependent, so they can be applied to any kinematic structure. The disadvantages of numerical methods are that they can be slower and in some cases, they do not allow computation of all possible solutions. For a six-degree-of-freedom serial-chain manipulator with only revolute and prismatic joints, the translation and rotation equations can always be reduced to a polynomial in a single variable of degree not greater than 16 [2.37]. Thus, such a manipulator can have as many as 16 real solutions to the inverse kinematics problem [2.38]. Since closed-form solution of a polynomial equation is only possible if the polynomial is of degree four or less, it follows that many manipulator geometries are not soluble in closed form. In general, a greater number of nonzero geometric parameters corresponds to a polynomial of higher degree in the reduction. For such manipulator structures, the most common numerical methods can be divided into categories of symbolic elimination methods, continuation methods, and iterative methods.

Symbolic Elimination Methods

Symbolic elimination methods involve analytical manipulations to eliminate variables from the system of nonlinear equations to reduce it to a smaller set of equations. *Raghavan* and *Roth* [2.39] used dialytic elimination to reduce the inverse kinematics problem of a general 6R serial-chain manipulator to a polyno-

Table 2.9 Inverse position kinematics of the articulated arm within the example serial chain manipulator in Fig. 2.3

$\theta_1 = \mathrm{Atan2}\left({}^0p_6^y, {}^0p_6^x\right)$ or $\mathrm{Atan2}\left(-{}^0p_6^y, -{}^0p_6^x\right)$
$\theta_3 = -\mathrm{Atan2}\left(D, \pm\sqrt{1-D^2}\right)$,   where $D := \frac{({}^0p_6^x)^2+({}^0p_6^y)^2+({}^0p_6^z)^2-a_3^2-d_4^2}{2a_3d_4}$,
$\theta_2 = \mathrm{Atan2}\left({}^0p_6^z, \sqrt{({}^0p_6^x)^2+({}^0p_6^y)^2}\right)$ $-\mathrm{Atan2}\left(d_4\cos\theta_3, a_3 - d_4\sin\theta_3\right)$

Table 2.10 Inverse orientation kinematics of the spherical wrist within the example serial chain manipulator in Fig. 2.3, with abbreviations $c_{\theta_i} := \cos\theta_i$ and $s_{\theta_i} := \sin\theta_i$

$\theta_5 = \mathrm{Atan2}\left(\pm\sqrt{1-\left(r_{13}s_{\theta_1} - r_{23}c_{\theta_1}\right)^2}\ , r_{13}s_{\theta_1} - r_{23}c_{\theta_1}\right)$
$\theta_4 = \mathrm{Atan2}\left(\mp\left(r_{13}c_{\theta_1} + r_{23}s_{\theta_1}\right)s_{(\theta_2+\theta_3)} \mp r_{33}c_{(\theta_2+\theta_3)}\right.$, $\left.\pm\left(r_{13}c_{\theta_1} + r_{23}s_{\theta_1}\right)c_{(\theta_2+\theta_3)} \mp r_{23}s_{(\theta_2+\theta_3)}\right)$
$\theta_6 = \mathrm{Atan2}\left(\pm\left(r_{12}s_{\theta_1} + r_{22}c_{\theta_1}\right), \pm\left(r_{11}s_{\theta_1} - r_{21}c_{\theta_1}\right)\right)$,   where the $\pm$ choice for θ_5 dictates all of the subsequent $\pm$ and $\mp$ for θ_4 and θ_6 .

mial of degree 16 and to find all possible solutions. The roots provide solutions for one of the joint variables, while the other variables are computed by solving linear systems. *Manocha* and *Canny* [2.40] improved the numerical properties of this technique by reformulating the problem as a generalized eigenvalue problem. An alternative approach to elimination makes use of Gröbner bases [2.41, 42].

Continuation Methods

Continuation methods involve tracking a solution path from a start system with known solutions to a target system whose solutions are sought as the start system is transformed into the target system. These techniques have been applied to inverse kinematics problems [2.43], and special properties of polynomial systems can be exploited to find all possible solutions [2.44].

Iterative Methods

A number of different iterative methods can be employed to solve the inverse kinematics problem. Most of them converge to a single solution based on an initial guess, so the quality of that guess greatly impacts the solution time. Newton–Raphson methods provide a fundamental approach that uses a first-order approximation of the original equations. *Pieper* [2.34] was among the first to apply the method to inverse kinematics, and others have followed [2.45, 46]. Optimization approaches formulate the problem as a nonlinear optimization problem and employ search techniques to move from an initial guess to a solution [2.47, 48]. Resolved motion rate control converts the problem to a differential equation [2.49], and a modified predictor–corrector algorithm can be used to perform the joint velocity integration [2.50]. Control-theory-based methods cast the differential equation into a control problem [2.51]. Interval analysis [2.52] is perhaps one of the most promising iterative methods because it offers rapid convergence to a solution and can be used to find all possible solutions. For complex mechanisms, the damped least-squares approach [2.53] is particularly attractive, and more detail is provided in Chap. 10.

2.8 Forward Instantaneous Kinematics

The forward instantaneous kinematics problem for a serial-chain manipulator is: given the positions of all members of the chain and the rates of motion about all the joints, find the total velocity of the end-effector. Here the rate of motion about the joint is the angular velocity of rotation about a revolute joint or the translational velocity of sliding along a prismatic joint. The total velocity of a member is the velocity of the origin of the coordinate frame fixed to it combined with its angular velocity. That is, the total velocity has six independent components and therefore, completely represents the velocity field of the member. It is important to note that this problem definition includes an assumption that the pose of the mechanism is completely known. In most situations, this means that either the forward or inverse position kinematics problem must be solved before the forward instantaneous kinematics problem can be addressed. The same is true of the inverse instantaneous kinematics problem discussed in the following section. The forward instantaneous kinematics problem is important when doing acceleration analysis for the purpose of studying dynamics. The total velocities of the members are needed for the computation of Coriolis and centripetal acceleration components.

2.8.1 Jacobian

Differentiation with respect to time of the forward position kinematics equations yields a set of equations of the form

$$^{k}\boldsymbol{v}_{N} = \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{2.53}$$

where $^{k}\boldsymbol{v}_{N}$ is the spatial velocity of the end-effector expressed in any frame k, $\dot{\boldsymbol{q}}$ is an n-dimensional vector composed of the joint rates, and $\mathbf{J}(\boldsymbol{q})$ is a $6\times n$ matrix whose elements are, in general, nonlinear functions of $\boldsymbol{q}$. $\mathbf{J}(\boldsymbol{q})$ is called the Jacobian matrix of this algebraic system and is expressed relative to the same coordinate frame as the spatial velocity $^{k}\boldsymbol{v}_{N}$ [2.54]. Alternately, (2.53) can be expressed as

$$^{k}\boldsymbol{v}_{N} = [\mathbf{J}_1 \;\; \mathbf{J}_2 \;\; \cdots \;\; \mathbf{J}_N]\;\dot{\boldsymbol{q}}\,, \tag{2.54}$$

where N is the number of joints (each with possibly more than 1 degree-of-freedom) and $\mathbf{J}_i$ provides the column(s) of $\mathbf{J}(\boldsymbol{q})$ which correspond(s) to $\dot{\boldsymbol{q}}_i$. If the joint positions are known, (2.53) yields six linear algebraic equations in the joint rates. If the joint rates are given,

a solution of (2.53) is a solution of the forward instantaneous kinematics problem. Note that $\mathbf{J}(\boldsymbol{q})$ can be regarded as a known matrix for this purpose provided all the joint positions are known.

Algorithm 2.1 Jacobian Computation Algorithm for a Serial-Chain Mechanism

inputs: ${}^{k}\mathbf{X}_N$, ${}^{N}\mathbf{X}_{N-1}, \ldots, {}^{i}\mathbf{X}_{i-1}, \ldots, {}^{2}\mathbf{X}_1$
output: $\mathbf{J}$

$\mathbf{X} = {}^{k}\mathbf{X}_N$
for $i = N$ **to** 1 **do**
  $\mathbf{J}_i = \mathbf{X}\,\boldsymbol{\Phi}_i$
  if $i > 1$ **then**
    $\mathbf{X} = \mathbf{X}\,{}^{i}\mathbf{X}_{i-1}$
  end if
end for

Using the spatial vector notation briefly introduced in Sect. 2.2.6 and explained in detail in Sect. 3.2, the Jacobian can be easily computed from the free modes $\boldsymbol{\Phi}_i$ of the joints and the associated spatial transforms ${}^{j}\mathbf{X}_i$.

$$\mathbf{J}_i = {}^{k}\mathbf{X}_i\,\boldsymbol{\Phi}_i\,. \tag{2.55}$$

To understand why this holds, note that $\boldsymbol{\Phi}_i$ describes the spatial velocities created by joint i in local coordinates and ${}^{k}\mathbf{X}_i$ transforms spatial velocities expressed in frame i to frame k. The quantities $\mathbf{J}_i$ can be efficiently computed from ${}^{k}\mathbf{X}_N$ and the link-to-link transforms ${}^{i}\mathbf{X}_{i-1}$. Algorithm 2.1 contains an algorithm for computing the columns of the Jacobian in this manner. Table 3.1 provides efficient methods to carry out the multiplications required in this algorithm which exploit the structure of spatial transformation matrices. Note that the quantity ${}^{k}\mathbf{X}_N$ can be computed with forward kinematics and each ${}^{i}\mathbf{X}_{i-1}$ can be computed with simple joint kinematics. Thus, use of the algorithm in Alg. 2.1 simplifies the problem of forward instantaneous kinematics to one of forward standard kinematics. Additional information about the Jacobian can be found in Chap. 10.

2.9 Inverse Instantaneous Kinematics

The important problem from the point of view of robotic coordination is the inverse instantaneous kinematics problem. More information on robot coordination can be found in Chaps. 7 and 8. The inverse instantaneous kinematics problem for a serial chain manipulator is: given the positions of all members of the chain and the total velocity of the end-effector, find the rates of motion of all joints. When controlling a movement of an industrial robot that operates in the point-to-point mode, it is not only necessary to compute the final joint positions needed to assume the desired final hand position. It is also necessary to generate a smooth trajectory for motion between the initial and final positions. There are, of course, an infinite number of possible trajectories for this purpose. However, the most straightforward and successful approach employs algorithms based on the solution of the inverse instantaneous kinematics problem. This technique originated in the work of *Whitney* [2.55] and of *Pieper* [2.34].

2.9.1 Inverse Jacobian

In order to solve the linear system of equations in the joint rates obtained by decomposing (2.53) into its component equations when $\mathbf{v}_N$ is known, it is necessary to invert the Jacobian matrix. The equation becomes

$$\dot{\boldsymbol{q}} = \mathbf{J}^{-1}(\boldsymbol{q})\mathbf{v}_N\,. \tag{2.56}$$

Since $\mathbf{J}$ is a 6×6 matrix, numerical inversion is not very attractive. It is quite possible for $\mathbf{J}$ to become singular ($|\mathbf{J}| = 0$), in which case the inverse does not exist. More information on singularities can be found in Chaps. 4 and 18. Even when the Jacobian matrix does not become singular, it may become ill-conditioned, leading to degraded performance in significant portions of the manipulator's workspace. Most industrial robot geometries are simple enough that the Jacobian matrix can be inverted analytically, leading to a set of explicit equations for the joint rates [2.56–58]. This greatly reduces the number of operations needed as compared to numerical inversion. For more complex manipulator geometries, though, numerical inversion is the only solution option. The Jacobian of a redundant manipulator is not square, so it cannot be inverted. Chapter 10 discusses how various pseudoinverses can be used in such cases.

2.10 Static Wrench Transmission

A general force system can be shown to be equivalent to a single force together with a moment acting about its line of action. This is called a wrench. There is a deep isometry between the geometries of systems of wrench axes and that of systems of instantaneous screw axes [2.59]. Static wrench analysis of a manipulator establishes the relationship between wrenches applied to the end-effector and forces/torques applied to the joints. This is essential for controlling a manipulator's interactions with its environment. Examples include tasks involving fixed or quasi-fixed workpieces such as inserting a component in place with a specified force and tightening a nut to a prescribed torque. More information can be found in Chaps. 9 and 37. Through the principle of virtual work, the relationship between wrenches applied to the end-effector and forces/torques applied to the joints can be shown to be

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\mathbf{f}\,, \tag{2.57}$$

where $\boldsymbol{\tau}$ is the n-dimensional vector of applied joint forces/torques for an n-degree-of-freedom manipulator and $\boldsymbol{f}$ is the spatial force vector

$$\mathbf{f} = \begin{pmatrix} \boldsymbol{n} \\ \boldsymbol{f} \end{pmatrix}\,, \tag{2.58}$$

in which $\boldsymbol{n}$ and $\boldsymbol{f}$ are the vectors of torques and forces, respectively, applied to the end-effector, both expressed in the coordinate frame relative to which the Jacobian is also expressed. Thus, in the same way the Jacobian maps the joint rates to the spatial velocity of the end-effector, its transpose maps the wrenches applied to the end-effector to the equivalent joint forces/torques. As in the velocity case, when the Jacobian is not square, the inverse relationship is not uniquely defined.

2.11 Conclusions and Further Reading

This chapter presents an overview of how the fundamentals of kinematics can be applied to robotic mechanisms. The topics include various representations of the position and orientation of a rigid body in space, the freedom of motion and accompanying mathematical models of joints, a geometric representation that describes the bodies and joints of a robotic mechanism, the workspace of a manipulator, the problems of forward and inverse kinematics, the problems of forward and inverse instantaneous kinematics including the definition of the Jacobian, and finally the transmission of static wrenches. This chapter is certainly not a comprehensive account of robot kinematics. Fortunately, a number of excellent texts provide a broad introduction to robotics with significant focus on kinematics [2.17, 27, 29, 30, 51, 60–64].

From a historical perspective, robotics fundamentally changed the nature of the field of mechanism kinematics. Before the first work on the generation of coordination equations for robots [2.34, 55], the focus of the field was almost entirely on single-degree-of-freedom mechanisms. This is why robotics, following on from the advent of digital computing, led to a renaissance of work in mechanism kinematics. More details can be found in Chap. 4. The evolution of the field has continued as it has broadened from the study of simple serial chains for industrial robots, the focus of the analysis in this chapter, to parallel machines (Chap. 18), human-like grippers (Chap. 19), robotic vehicles (Chaps. 17 and 24–26), and even small-scale robots (Chap. 27).

References

2.1 W. R. Hamilton: On quaternions, or on a new system of imaginaries in algebra, Philos. Mag. **18** (2000)

2.2 E.B. Wilson: *Vector Analysis* (Dover, New York 1960), based upon the lectures of J.W. Gibbs (reprint of the 2nd edn. published by Charles Scribner's Sons, 1909)

2.3 H. Graßmann: *Die Wissenschaft der extensiven Größe oder die Ausdehnungslehre* (Wigand, Leipzig 1844)

2.4 J.M. McCarthy: *Introduction to Theoretical Kinematics* (MIT Press, Cambridge 1990)

2.5 W.K. Clifford: Preliminary sketch of bi-quarternions, Proc. Lond. Math. Soc. **4**, 381–395 (1873)

2.6 A.P. Kotelnikov: *Screw calculus and some applications to geometry and mechanics* (Annal. Imp. Univ., Kazan 1895)

2.7 E. Study: *Geometrie der Dynamen* (Teubner, Leipzig 1903)

2.8 G.S. Chirikjian, A.B. Kyatkin: *Engineering Applications of Noncommutative Harmonic Analysis* (CRC, Boca Raton 2001)

2.9 R. von Mises: Anwendungen der Motorrechnung, Z. Angew. Math. Mech. **4**(3), 193–213 (1924)

2.10 J.E. Baker, I.A. Parkin: *Fundamentals of Screw Motion: Seminal Papers by Michel Chasles and Olinde Rodrigues*, School of Information Technologies (University of Sydney, Sydney 2003), translated from O. Rodrigues: Des lois géométriques qui régissent les déplacements d'un système dans l'espace, J. Math. Pures Applicqu. Liouville 5, 380–440 (1840)

2.11 R.S. Ball: *A Treatise on the Theory of Screws* (Cambridge Univ. Press, Cambridge 1998)

2.12 J.K. Davidson, K.H. Hunt: *Robots and Screw Theory: Applications of Kinematics and Statics to Robotics* (Oxford Univ. Press, Oxford 2004)

2.13 K.H. Hunt: *Kinematic Geometry of Mechanisms* (Clarendon, Oxford 1978)

2.14 J.R. Phillips: *Freedom in Machinery. Vol 1. Introducing Screw Theory* (Cambridge Univ. Press, Cambridge 1984)

2.15 J.R. Phillips: *Freedom in Machinery. Vol 2. Screw Theory Exemplified* (Cambridge Univ. Press, Cambridge 1990)

2.16 G.S. Chirikjian: Rigid-body kinematics. In: *Robotics and Automation Handbook*, ed. by T. Kurfess (CRC, Boca Raton 2005), Chap. 2

2.17 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)

2.18 A. Karger, J. Novak: *Space Kinematics and Lie Groups* (Routledge, New York 1985)

2.19 R. von Mises: Motorrechnung, ein neues Hilfsmittel in der Mechanik, Z. Angew. Math. Mech. **2**(2), 155–181 (1924)

2.20 J.D. Everett: On a new method in statics and kinematics, Mess. Math. **45**, 36–37 (1875)

2.21 F. Reuleaux: *Kinematics of Machinery* (Dover, New York 1963), reprint of *Theoretische Kinematik*, 1875, in German

2.22 R. Featherstone: *Rigid Body Dynamics Algorithms* (Kluwer, Boston 2007)

2.23 K.J. Waldron: A method of studying joint geometry, Mechan. Mach. Theory **7**, 347–353 (1972)

2.24 T.R. Kane, D.A. Levinson: *Dynamics, Theory and Applications* (McGraw-Hill, New York 1985)

2.25 J.L. Lagrange: *Oeuvres de Lagrange* (Gauthier-Villars, Paris 1867)

2.26 J. Denavit, R.S. Hartenberg: A kinematic notation for lower-pair mechanisms based on matrices, J. Appl. Mech. **22**, 215–221 (1955)

2.27 W. Khalil, E. Dombre: *Modeling, Identification and Control of Robots* (Taylor Francis, New York 2002)

2.28 K.J. Waldron: A study of overconstrained linkage geometry by solution of closure equations, Part I: A method of study, Mech. Mach. Theory **8**(1), 95–104 (1973)

2.29 R. Paul: *Robot Manipulators: Mathematics, Programming and Control* (MIT Press, Cambridge 1982)

2.30 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Addison-Wesley, Reading 1986)

2.31 K.J. Waldron, A. Kumar: The dextrous workspace, ASME Mech. Conf., Los Angeles (1980), ASME paper No. 80-DETC-108

2.32 R. Vijaykumar, K.J. Waldron, M.J. Tsai: Geometric optimization of manipulator structures for working volume and dexterity, Int. J. Robotics Res. **5**(2), 91–103 (1986)

2.33 J. Duffy: *Analysis of Mechanisms and Robot Manipulators* (Wiley, New York 1980)

2.34 D. Pieper: The Kinematics of Manipulators Under Computer Control, Ph.D. Thesis (Stanford University, Stanford 1968)

2.35 C.S.G. Lee: Robot arm kinematics, dynamics, and control, Computer **15**(12), 62–80 (1982)

2.36 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT Press, Cambridge 2001)

2.37 H.Y. Lee, C.G. Liang: A new vector theory for the analysis of spatial mechanisms, Mech. Mach. Theory **23**(3), 209–217 (1988)

2.38 R. Manseur, K.L. Doty: A robot manipulator with 16 real inverse kinematic solutions, Int. J. Robotics Res. **8**(5), 75–79 (1989)

2.39 M. Raghavan, B. Roth: Kinematic analysis of the 6R manipulator of general geometry, 5th Int. Symp. Robotics Res. (1990)

2.40 D. Manocha, J. Canny: *Real Time Inverse Kinematics for General 6R Manipulators*, Tech. Rep. (University of California, Berkeley 1992)

2.41 B. Buchberger: Applications of Gröbner bases in non-linear computational geometry, Lect. Notes Comput. Sci. **296**, 52–80 (1989)

2.42 P. Kovacs: Minimum degree solutions for the inverse kinematics problem by application of the Buchberger algorithm. In: *Advances in Robot Kinematics*, ed. by S. Stifter, J. Lenarcic (Springer, New York 1991) pp. 326–334

2.43 L.W. Tsai, A.P. Morgan: Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods, ASME J. Mech. Transm. Autom. Des. **107**, 189–195 (1985)

2.44 C.W. Wampler, A.P. Morgan, A.J. Sommese: Numerical continuation methods for solving polynomial systems arising in kinematics, ASME J. Mech. Des. **112**, 59–68 (1990)

2.45 R. Manseur, K.L. Doty: Fast inverse kinematics of 5-revolute-axis robot manipulators, Mechan. Mach. Theory **27**(5), 587–597 (1992)

2.46 S.C.A. Thomopoulos, R.Y.J. Tam: An iterative solution to the inverse kinematics of robotic manipulators, Mechan. Mach. Theory **26**(4), 359–373 (1991)

2.47 J.J. Uicker Jr., J. Denavit, R.S. Hartenberg: An interactive method for the displacement analysis of spatial mechanisms, J. Appl. Mech. **31**, 309–314 (1964)

2.48 J. Zhao, N. Badler: Inverse kinematics positioning using nonlinear programming for highly articulated figures, Trans. Comput. Graph. **13**(4), 313–336 (1994)

2.49 D.E. Whitney: Resolved motion rate control of manipulators and human prostheses, IEEE Trans. Man Mach. Syst. **10**, 47–63 (1969)

2.50 H. Cheng, K. Gupta: A study of robot inverse kinematics based upon the solution of differential equations, J. Robotic Syst. **8**(2), 115–175 (1991)

2.51 L. Sciavicco, B. Siciliano: *Modeling and Control of Robot Manipulators* (Springer, London 2000)

2.52 R.S. Rao, A. Asaithambi, S.K. Agrawal: Inverse kinematic solution of robot manipulators using interval analysis, ASME J. Mech. Des. **120**(1), 147–150 (1998)
2.53 C.W. Wampler: Manipulator inverse kinematic solutions based on vector formulations and damped least squares methods, IEEE Trans. Syst. Man Cybern. **16**, 93–101 (1986)
2.54 D.E. Orin, W.W. Schrader: Efficient computation of the jacobian for robot manipulators, Int. J. Robotics Res. **3**(4), 66–75 (1984)
2.55 D.E. Whitney: The mathematics of coordinated control of prosthetic arms and manipulators, J. Dynamic Sys. Meas. Control **122**, 303–309 (1972)
2.56 R.P. Paul, B.E. Shimano, G. Mayer: Kinematic control equations for simple manipulators, IEEE Trans. Syst. Man Cybern. **11**(6), 339–455 (1981)
2.57 R.P. Paul, C.N. Stephenson: Kinematics of robot wrists, Int. J. Robotics Res. **20**(1), 31–38 (1983)
2.58 R.P. Paul, H. Zhang: Computationally efficient kinematics for manipulators with spherical wrists based on the homogeneous transformation representation, Int. J. Robotics Res. **5**(2), 32–44 (1986)
2.59 K.J. Waldron, K.H. Hunt: Series-parallel dualities in actively coordinated mechanisms, Int. J. Robotics Res. **10**, 473–480 (1991)
2.60 H. Asada, J.J.E. Slotine: *Robot Analysis and Control* (Wiley, New York 1986)
2.61 F.L. Lewis, C.T. Abdallah, D.M. Dawson: *Control of Robot Manipulators* (Macmillan, New York 1993)
2.62 R.J. Schilling: *Fundamentals of Robotics: Analysis and Control* (Prentice Hall, Englewood Cliffs 1990)
2.63 M.W. Spong, M. Vidyasagar: *Robot Dynamics and Control* (Wiley, New York 1989)
2.64 T. Yoshikawa: *Foundations of Robotics* (MIT Press, Cambridge 1990)

Multimedia Contents

3. Dynamics

Roy Featherstone, David E. Orin

The dynamic equations of motion provide the relationships between actuation and contact forces acting on robot mechanisms, and the acceleration and motion trajectories that result. Dynamics is important for mechanical design, control, and simulation. A number of algorithms are important in these applications, and include computation of the following: *inverse dynamics*, *forward dynamics*, the *joint-space inertia matrix*, and the *operational-space inertia matrix*. This chapter provides efficient algorithms to perform each of these calculations on a rigid-body model of a robot mechanism. The algorithms are presented in their most general form and are applicable to robot mechanisms with general connectivity, geometry, and joint types. Such mechanisms include fixed-base robots, mobile robots, and parallel robot mechanisms.

In addition to the need for computational efficiency, algorithms should be formulated with a compact set of equations for ease of development and implementation. The use of spatial notation has been very effective in this regard, and is used in presenting the dynamics algorithms. Spatial vector algebra is a concise vector notation for describing rigid-body velocity, acceleration, inertia, etc., using six-dimensional (6-D) vectors and tensors.

The goal of this chapter is to introduce the reader to the subject of robot dynamics and to provide the reader with a rich set of algorithms, in a compact form, that they may apply to their particular robot mechanism. These algorithms are presented in tables for ready access.

3.1 **Overview** 38
3.1.1 Spatial Vector Notation 38
3.1.2 Canonical Equations 38
3.1.3 Dynamic Models of Rigid-Body Systems 38
3.1.4 Kinematic Trees 39
3.1.5 Kinematic Loops 39

3.2 **Spatial Vector Notation** 39
3.2.1 Motion and Force 40
3.2.2 Basis Vectors 40
3.2.3 Spatial Velocity and Force 40
3.2.4 Addition and Scalar Multiplication 41
3.2.5 Scalar Product 41
3.2.6 Coordinate Transforms 41
3.2.7 Vector Products 41
3.2.8 Differentiation 42
3.2.9 Acceleration 42
3.2.10 Spatial Momentum 42
3.2.11 Spatial Inertia 42
3.2.12 Equation of Motion 43
3.2.13 Computer Implementation 44
3.2.14 Summary 45

3.3 **Canonical Equations** 45
3.3.1 Joint-Space Formulation 45
3.3.2 Lagrange Formulation 46
3.3.3 Operational-Space Formulation 46
3.3.4 Impact Model 47

3.4 **Dynamic Models of Rigid-Body Systems** 47
3.4.1 Connectivity 47
3.4.2 Link Geometry 49
3.4.3 Link Inertias 49
3.4.4 Joint Models 49
3.4.5 Example System 50

3.5 **Kinematic Trees** 51
3.5.1 The Recursive Newton–Euler Algorithm 51
3.5.2 The Articulated-Body Algorithm 54
3.5.3 The Composite-Rigid-Body Algorithm 56
3.5.4 Operational-Space Inertia Matrix 57

3.6 **Kinematic Loops** 58
3.6.1 Formulation of Closed-Loop Algorithm 58
3.6.2 Closed-Loop Algorithm 60

3.7 **Conclusions and Further Reading** 61
3.7.1 Multibody Dynamics 62
3.7.2 Alternative Representations 62
3.7.3 Alternative Formulations 62
3.7.4 Efficiency 62
3.7.5 Accuracy 62
3.7.6 Software Packages 63
3.7.7 Symbolic Simplification 63
3.7.8 Algorithms for Parallel Computers .. 63
3.7.9 Topologically-Varying Systems 63

References 63

3.1 Overview

Robot dynamics provides the relationships between actuation and contact forces, and the acceleration and motion trajectories that result. The dynamic equations of motion provide the basis for a number of computational algorithms that are useful in mechanical design, control, and simulation. A growing area of their use is in computer animation of mobile systems, especially using human and humanoid models. In this Chapter, the fundamental dynamic relationships for robot mechanisms are presented, along with efficient algorithms for the most common computations. Spatial vector notation, a concise representation which makes use of 6-D vectors and tensors, is used in the algorithms.

This chapter presents efficient low-order algorithms for four major computations:

1. Inverse dynamics, in which the required joint actuator torques/forces are computed from a specification of the robot's trajectory (position, velocity, and acceleration),
2. Forward dynamics in which the applied joint actuator torques/forces are specified and the joint accelerations are to be determined,
3. The joint-space inertia matrix, which maps the joint accelerations to the joint torques/forces, and
4. The operational-space inertia matrix, which maps task accelerations to task forces in operational or Cartesian space.

Inverse dynamics is used in feedforward control and trajectory planning. Forward dynamics is required for simulation. The joint-space inertia (mass) matrix is used in analysis, in feedback control to linearize the dynamics, and is an integral part of many forward dynamics formulations. The operational-space inertia matrix is used in control at the task or end-effector level.

3.1.1 Spatial Vector Notation

Section 3.2 presents the spatial vector notation, which is used to express the algorithms in this chapter in a clear and concise manner. It was originally developed by *Featherstone* [3.1] to provide a concise vector notation for describing rigid-body velocity, acceleration, inertia, etc., using 6-D vectors and tensors. Section 3.2 explains the meanings of spatial vectors and operators, and provides a detailed tabulation of the correspondence between spatial and standard three-dimensional (3-D) quantities and operators, so that the algorithms in the later sections can be understood. Formulae for efficient computer implementation of spatial arithmetic are also provided. Effort is taken in the discussion of spatial vectors to distinguish between the coordinate vectors and the quantities they represent. This illuminates some of the important characteristics of spatial vectors.

3.1.2 Canonical Equations

The dynamic equations of motion are provided in Sect. 3.3 in two fundamental forms: the joint-space formulation and the operational-space formulation. The terms in the joint-space formulation have traditionally been derived using a Lagrangian approach in which they are developed independently of any reference coordinate frame. The Lagrange formulation provides a description of the relationship between the joint actuator forces and the motion of the mechanism, and fundamentally operates on the kinetic and potential energy in the system. The resulting joint-space formulation has a number of notable properties that have proven useful for developing control algorithms. The equations to relate the terms in the joint-space and operational-space formulations, along with an impact model, are also provided in this section.

3.1.3 Dynamic Models of Rigid-Body Systems

The algorithms in this chapter are model-based and require a data structure describing a robot mechanism as one of their input arguments. Section 3.4 gives a description of the components of this model: a connectivity graph, link geometry parameters, link inertia parameters, and a set of joint models. The description of the connectivity is general so that it covers both kinematic trees and closed-loop mechanisms. Kinematic trees and the spanning tree for a closed-loop mechanism share a common notation. In order to describe

the link and joint geometry, two coordinate frames are associated with each joint, one each attached to the *predecessor* and *successor* links. These frames are defined to be compatible with the modified Denavit–Hartenberg convention of *Craig* [3.2] for serial mechanisms containing single-degree-of-freedom (DOF) joints, but are nevertheless applicable to general rigid-body systems containing general multi-DOF joints. The relationship between connected links is described using the general joint model of *Roberson* and *Schwertassek* [3.3]. A humanoid robot is given as an example to illustrate the link and joint numbering scheme, as well as the assignment of coordinate frames to describe the links and joints. The example includes a floating base, and revolute, universal, and spherical joints.

3.1.4 Kinematic Trees

The algorithms presented in Sect. 3.5 calculate the inverse dynamics, forward dynamics, joint-space inertia matrix, and operational-space inertia matrix for any robot mechanism that is a kinematic tree. An $O(n)$ algorithm for inverse dynamics is provided, where n is the number of degrees of freedom in the mechanism. It uses a Newton–Euler formulation of the problem, and is based on the very efficient recursive Newton–Euler algorithm (RNEA) of *Luh* et al. [3.4]. Two algorithms are provided for forward dynamics. The first is the $O(n)$ articulated-body algorithm (ABA) which was developed by *Featherstone* [3.1]. The second is the $O(n^2)$ composite-rigid-body algorithm (CRBA), developed by *Walker* and *Orin* [3.5], to compute the joint-space inertia matrix (JSIM). This matrix, together with a vector computed using the RNEA, provide the coefficients of the equation of motion, which can then be solved directly for the accelerations [3.5]. The operational-space inertia matrix (OSIM) is a kind of articulated-body inertia, and two algorithms are given to calculate it. The first uses the basic definition of the OSIM, and the second is a straightforward $O(n)$ algorithm which is based on efficient solution of the forward dynamics problem. The inputs, outputs, model data, and pseudocode for each algorithm are summarized in tables for ready access.

3.1.5 Kinematic Loops

The above algorithms apply only to mechanisms having the connectivity of kinematic trees, including unbranched kinematic chains. A final algorithm is provided in Sect. 3.6 for the forward dynamics of closed-loop systems, including parallel robot mechanisms. The algorithm makes use of the dynamic equations of motion for a spanning tree of the closed-loop system, and supplements these with loop-closure constraint equations. Three different methods are outlined to solve the resulting linear system of equations. Method 2 is particularly useful if $n \gg n^c$, where n^c is the number of constraints due to the loop-closing joints. This method offers the opportunity to use $O(n)$ algorithms on the spanning tree [3.6]. The section ends with an efficient algorithm to compute the loop-closure constraints by transforming them to a single coordinate system. Since the loop-closure constraint equations are applied at the acceleration level, standard *Baumgarte* stabilization [3.7] is used to prevent the accumulation of position and velocity errors in the loop-closure constraints.

The final section in this chapter provides a conclusion and suggestions for further reading. The area of robot dynamics has been, and continues to be, a very rich area of investigation. This section outlines the major contributions that have been made in the area and the work most often cited. Unfortunately, space does not permit us to provide a comprehensive review of the extensive literature in the area.

3.2 Spatial Vector Notation

There is no single standard notation for robot dynamics. The notations currently in use include 3-D vectors, 4×4 matrices, and several types of 6-D vector: screws, motors, Lie algebra elements, and spatial vectors. Six-dimensional vector notations are generally the best, being more compact than 3-D vectors, and more powerful than 4×4 matrices. We therefore use 6-D vectors throughout this chapter. In particular, we shall use the spatial vector algebra described in [3.8]. This section provides a brief summary of spatial vectors. Descriptions of 4×4 matrix notations can be found in [3.2, 9], and descriptions of other 6-D vector notations can be found in [3.10–12].

In this handbook, vectors are usually denoted by bold italic letters (e.g., $\boldsymbol{f}$, $\boldsymbol{v}$). However, to avoid a few name clashes, we shall use upright bold letters to denote spatial vectors (e.g., $\mathbf{f}$, $\mathbf{v}$). Note that this applies only to vectors, not tensors. Also, *in this section only*, we will underline coordinate vectors to distinguish them from the vectors that they represent (e.g., $\underline{\boldsymbol{v}}$ and $\underline{\mathbf{v}}$, representing $\boldsymbol{v}$ and $\mathbf{v}$).

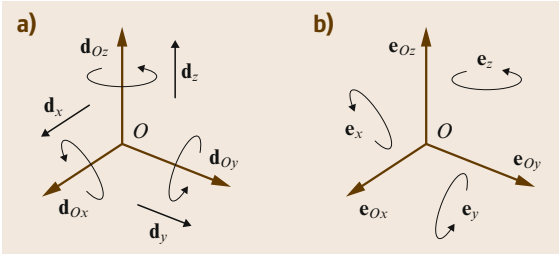


Fig.3.1a,b Plücker basis vectors for motions **(a)** and forces **(b)**

3.2.1 Motion and Force

For mathematical reasons, it is useful to distinguish between those vectors that describe the motions of rigid bodies, and those that describe the forces acting upon them. We therefore place motion vectors in a vector space called M^6, and force vectors in a space called F^6. (The superscripts indicate the dimension.) Motion vectors describe quantities like velocity, acceleration, infinitesimal displacement, and directions of motion freedom; force vectors describe force, momentum, contact normals, and so on.

3.2.2 Basis Vectors

Suppose that $\boldsymbol{v}$ is a 3-D vector, and that $\underline{\boldsymbol{v}} = (v_x, v_y, v_z)^{\mathsf{T}}$ is the Cartesian coordinate vector that represents $\boldsymbol{v}$ in the orthonormal basis $\{\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}\}$. The relationship between $\boldsymbol{v}$ and $\underline{\boldsymbol{v}}$ is then given by the formula

$$\boldsymbol{v} = \hat{\boldsymbol{x}} v_x + \hat{\boldsymbol{y}} v_y + \hat{\boldsymbol{z}} v_z \,.$$

This same idea applies also to spatial vectors, except that we use Plücker coordinates instead of Cartesian coordinates, and a Plücker basis instead of an orthonormal basis.

Plücker coordinates were introduced in Sect. 2.2.6, but the basis vectors are shown in Fig. 3.1. There are 12 basis vectors in total: six for motion vectors and six for forces. Given a Cartesian coordinate frame, O_{xyz}, the Plücker basis vectors are defined as follows: three unit rotations about the directed lines Ox, Oy, and Oz, denoted by $\mathbf{d}_{Ox}$, $\mathbf{d}_{Oy}$, and $\mathbf{d}_{Oz}$, three unit translations in the directions x, y, and z, denoted by $\mathbf{d}_x$, $\mathbf{d}_y$, and $\mathbf{d}_z$, three unit couples about the x, y, and z directions, denoted by $\mathbf{e}_x$, $\mathbf{e}_y$, and $\mathbf{e}_z$, and three unit forces along the lines Ox, Oy, and Oz, denoted by $\mathbf{e}_{Ox}$, $\mathbf{e}_{Oy}$, and $\mathbf{e}_{Oz}$.

3.2.3 Spatial Velocity and Force

Given any point O, the velocity of a rigid body can be described by a pair of 3-D vectors, $\boldsymbol{\omega}$ and $\boldsymbol{v}_O$, which specify the body's angular velocity and the linear velocity of the body-fixed point currently at O. Note that $\boldsymbol{v}_O$ is not the velocity of O itself, but the velocity of the body-fixed point that happens to coincide with O at the current instant.

The velocity of this same rigid body can also be described by a single spatial motion vector, $\mathbf{v} \in \mathsf{M}^6$. To obtain $\mathbf{v}$ from $\boldsymbol{\omega}$ and $\boldsymbol{v}_O$, we first introduce a Cartesian frame, O_{xyz}, with its origin at O. This frame defines a Cartesian coordinate system for $\boldsymbol{\omega}$ and $\boldsymbol{v}_O$, and also a Plücker coordinate system for $\mathbf{v}$. Given these coordinate systems, it can be shown that

$$\begin{aligned}\mathbf{v} = {} & \mathbf{d}_{Ox}\omega_x + \mathbf{d}_{Oy}\omega_y + \mathbf{d}_{Oz}\omega_z + \mathbf{d}_x v_{Ox} \\ & + \mathbf{d}_y v_{Oy} + \mathbf{d}_z v_{Oz} \,, \end{aligned} \tag{3.1}$$

where $\omega_x, \ldots, v_{Oz}$ are the Cartesian coordinates of $\boldsymbol{\omega}$ and $\boldsymbol{v}_O$ in O_{xyz}. Thus, the Plücker coordinates of $\mathbf{v}$ are the Cartesian coordinates of $\boldsymbol{\omega}$ and $\boldsymbol{v}_O$. The coordinate vector representing $\mathbf{v}$ in O_{xyz} can be written

$$\underline{\mathbf{v}}_O = \begin{pmatrix} \omega_x \\ \vdots \\ v_{Oz} \end{pmatrix} = \begin{pmatrix} \underline{\boldsymbol{\omega}} \\ \underline{\boldsymbol{v}}_O \end{pmatrix} . \tag{3.2}$$

The notation on the far right of this equation is simply a convenient abbreviation of the list of Plücker coordinates.

The definition of spatial force is very similar. Given any point O, any system of forces acting on a single rigid body is equivalent to a single force $\boldsymbol{f}$ acting on a line passing through O, together with a pure couple, $\boldsymbol{n}_O$, which is the moment of the force system about O. Thus, the two vectors $\boldsymbol{f}$ and $\boldsymbol{n}_O$ describe the force acting on a rigid body in much the same way that $\boldsymbol{\omega}$ and $\boldsymbol{v}_O$ describe its velocity. This same force can also be described by a single spatial force vector, $\mathbf{f} \in \mathsf{F}^6$. Introducing the frame O_{xyz}, as before, it can be shown that

$$\begin{aligned}\mathbf{f} = {} & \mathbf{e}_x n_{Ox} + \mathbf{e}_y n_{Oy} + \mathbf{e}_z n_{Oz} + \mathbf{e}_{Ox} f_x \\ & + \mathbf{e}_{Oy} f_y + \mathbf{e}_{Oz} f_z \,, \end{aligned} \tag{3.3}$$

where $n_{Ox}, \ldots, f_z$ are the Cartesian coordinates of $\boldsymbol{n}_O$ and $\boldsymbol{f}$ in O_{xyz}. The coordinate vector representing $\mathbf{f}$ in O_{xyz} can then be written

$$\underline{\mathbf{f}}_O = \begin{pmatrix} n_{Ox} \\ \vdots \\ f_z \end{pmatrix} = \begin{pmatrix} \underline{\boldsymbol{n}}_O \\ \underline{\boldsymbol{f}} \end{pmatrix} . \tag{3.4}$$

Again, these are the Plücker coordinates of $\mathbf{f}$ in O_{xyz}, and the notation on the far right is simply a convenient abbreviation of the list of Plücker coordinates.

3.2.4 Addition and Scalar Multiplication

Spatial vectors behave in the obvious way under addition and scalar multiplication. For example, if $\mathbf{f}_1$ and $\mathbf{f}_2$ both act on the same rigid body, then their resultant is $\mathbf{f}_1 + \mathbf{f}_2$; if two different bodies have velocities of $\mathbf{v}_1$ and $\mathbf{v}_2$, then the velocity of the second body relative to the first is $\mathbf{v}_2 - \mathbf{v}_1$; and if $\mathbf{f}$ denotes a force of 1 N acting along a particular line in space, then $\alpha\,\mathbf{f}$ denotes a force of α N acting along the same line.

3.2.5 Scalar Product

A scalar product is defined between any two spatial vectors, provided that one of them is a motion and the other a force. Given any $\mathbf{m} \in \mathsf{M}^6$ and $\mathbf{f} \in \mathsf{F}^6$, the scalar product can be written either $\mathbf{f}\cdot\mathbf{m}$ or $\mathbf{m}\cdot\mathbf{f}$, and expresses the work done by a force $\mathbf{f}$ acting on a body with motion $\mathbf{m}$. Expressions like $\mathbf{f}\cdot\mathbf{f}$ and $\mathbf{m}\cdot\mathbf{m}$ are not defined. If $\underline{\mathbf{m}}$ and $\underline{\mathbf{f}}$ are coordinate vectors representing $\mathbf{m}$ and $\mathbf{f}$ in the same coordinate system, then

$$\mathbf{m}\cdot\mathbf{f} = \underline{\mathbf{m}}^{\mathrm{T}}\underline{\mathbf{f}}\,. \tag{3.5}$$

3.2.6 Coordinate Transforms

Motion and force vectors obey different transformation rules. Let A and B be two coordinate frames, each defining a coordinate system of the same name; and let $\underline{\mathbf{m}}_A$, $\underline{\mathbf{m}}_B$, $\underline{\mathbf{f}}_A$, and $\underline{\mathbf{f}}_B$ be coordinate vectors representing the spatial vectors $\mathbf{m} \in \mathsf{M}^6$ and $\mathbf{f} \in \mathsf{F}^6$ in A and B coordinates, respectively. The transformation rules are then

$$\underline{\mathbf{m}}_B = {}^B\boldsymbol{X}_A\,\underline{\mathbf{m}}_A \tag{3.6}$$

and

$$\underline{\mathbf{f}}_B = {}^B\boldsymbol{X}_A^F\,\underline{\mathbf{f}}_A\,, \tag{3.7}$$

where ${}^B\boldsymbol{X}_A$ and ${}^B\boldsymbol{X}_A^F$ are the coordinate transformation matrices from A to B for motion and force vectors, respectively. These matrices are related by the identity

$${}^B\boldsymbol{X}_A^F \equiv ({}^B\boldsymbol{X}_A)^{-T} \equiv ({}^A\boldsymbol{X}_B)^{\mathrm{T}}\,. \tag{3.8}$$

Suppose that the position and orientation of frame A relative to frame B is described by a position vector ${}^B\underline{\boldsymbol{p}}_A$ and a 3×3 rotation matrix ${}^B\boldsymbol{R}_A$, as described in Sect. 2.2. The formula for ${}^B\boldsymbol{X}_A$ is then

$$\begin{aligned} {}^B\boldsymbol{X}_A &= \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \boldsymbol{S}({}^B\underline{\boldsymbol{p}}_A) & \mathbf{1} \end{pmatrix} \begin{pmatrix} {}^B\boldsymbol{R}_A & \mathbf{0} \\ \mathbf{0} & {}^B\boldsymbol{R}_A \end{pmatrix} \\ &= \begin{pmatrix} {}^B\boldsymbol{R}_A & \mathbf{0} \\ \boldsymbol{S}({}^B\underline{\boldsymbol{p}}_A){}^B\boldsymbol{R}_A & {}^B\boldsymbol{R}_A \end{pmatrix}, \end{aligned} \tag{3.9}$$

and its inverse is

$${}^A\boldsymbol{X}_B = \begin{pmatrix} {}^A\boldsymbol{R}_B & \mathbf{0} \\ \mathbf{0} & {}^A\boldsymbol{R}_B \end{pmatrix} \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ -\boldsymbol{S}({}^B\underline{\boldsymbol{p}}_A) & \mathbf{1} \end{pmatrix}. \tag{3.10}$$

The quantity $\boldsymbol{S}(\underline{\boldsymbol{p}})$ is the skew-symmetric matrix that satisfies $\boldsymbol{S}(\underline{\boldsymbol{p}})\underline{\boldsymbol{v}} = \underline{\boldsymbol{p}} \times \underline{\boldsymbol{v}}$ for any 3-D vector $\underline{\boldsymbol{v}}$. It is defined by the equation

$$\boldsymbol{S}(\underline{\boldsymbol{p}}) = \begin{pmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{pmatrix}. \tag{3.11}$$

3.2.7 Vector Products

There are two vector (cross) products defined on spatial vectors. The first takes two motion-vector arguments, and produces a motion-vector result. It is defined by the formula

$$\begin{aligned} \underline{\mathbf{m}}_1 \times \underline{\mathbf{m}}_2 &= \begin{pmatrix} \underline{\boldsymbol{m}}_1 \\ \underline{\boldsymbol{m}}_{1O} \end{pmatrix} \times \begin{pmatrix} \underline{\boldsymbol{m}}_2 \\ \underline{\boldsymbol{m}}_{2O} \end{pmatrix} \\ &= \begin{pmatrix} \underline{\boldsymbol{m}}_1 \times \underline{\boldsymbol{m}}_2 \\ \underline{\boldsymbol{m}}_1 \times \underline{\boldsymbol{m}}_{2O} + \underline{\boldsymbol{m}}_{1O} \times \underline{\boldsymbol{m}}_2 \end{pmatrix}. \end{aligned} \tag{3.12}$$

The second takes a motion vector as left-hand argument and a force vector as right-hand argument, and produces a force-vector result. It is defined by the formula

$$\begin{aligned} \underline{\mathbf{m}} \times \underline{\mathbf{f}} &= \begin{pmatrix} \underline{\boldsymbol{m}} \\ \underline{\boldsymbol{m}}_O \end{pmatrix} \times \begin{pmatrix} \underline{\boldsymbol{f}}_O \\ \underline{\boldsymbol{f}} \end{pmatrix} \\ &= \begin{pmatrix} \underline{\boldsymbol{m}} \times \underline{\boldsymbol{f}}_O + \underline{\boldsymbol{m}}_O \times \underline{\boldsymbol{f}} \\ \underline{\boldsymbol{m}} \times \underline{\boldsymbol{f}} \end{pmatrix}. \end{aligned} \tag{3.13}$$

These products arise in differentiation formulae.

It is possible to define a spatial cross-product operator, in analogy with (3.11), as follows

$$\boldsymbol{S}(\underline{\mathbf{m}}) = \begin{pmatrix} \boldsymbol{S}(\underline{\boldsymbol{m}}) & \mathbf{0} \\ \boldsymbol{S}(\underline{\boldsymbol{m}}_O) & \boldsymbol{S}(\underline{\boldsymbol{m}}) \end{pmatrix}, \tag{3.14}$$

in which case

$$\underline{\mathbf{m}}_1 \times \underline{\mathbf{m}}_2 = \boldsymbol{S}(\underline{\mathbf{m}}_1)\,\underline{\mathbf{m}}_2\,, \tag{3.15}$$

but

$$\underline{\mathbf{m}} \times \underline{\mathbf{f}} = -\boldsymbol{S}(\underline{\mathbf{m}})^{\mathrm{T}}\underline{\mathbf{f}}\,. \tag{3.16}$$

Observe that $\boldsymbol{S}(\underline{\mathbf{m}})$ maps motion vectors to motion vectors, but $\boldsymbol{S}(\underline{\boldsymbol{m}})^{\mathrm{T}}$ maps force vectors to force vectors.

3.2.8 Differentiation

The derivative of a spatial vector is defined by

$$\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{s}(x) = \lim_{\delta x \to 0} \frac{\mathbf{s}(x+\delta x) - \mathbf{s}(x)}{\delta x}\,, \tag{3.17}$$

where $\mathbf{s}$ here stands for any spatial vector. The derivative is a spatial vector of the same kind (motion or force) as that being differentiated.

The formula for differentiating a spatial coordinate vector in a moving coordinate system is

$$\left(\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{s}\right)_A = \frac{\mathrm{d}}{\mathrm{d}t}\underline{\mathbf{s}}_A + \underline{\mathbf{v}}_A \times \underline{\mathbf{s}}_A\,, \tag{3.18}$$

where $\mathbf{s}$ is any spatial vector, $\mathrm{d}\mathbf{s}/\mathrm{d}t$ is the time derivative of $\mathbf{s}$, A is the moving coordinate system, $(\mathrm{d}\mathbf{s}/\mathrm{d}t)_A$ is the coordinate vector that represents $\mathrm{d}\mathbf{s}/\mathrm{d}t$ in A coordinates, $\underline{\mathbf{s}}_A$ is the coordinate vector that represents $\mathbf{s}$ in A coordinates, $\mathrm{d}\underline{\mathbf{s}}_A/\mathrm{d}t$ is the time derivative of $\underline{\mathbf{s}}_A$ (which is the componentwise derivative, since $\underline{\mathbf{s}}_A$ is a coordinate vector), and $\underline{\mathbf{v}}_A$ is the velocity of the A coordinate frame, expressed in A coordinates.

The time derivative of a spatial vector that changes only because it is moving is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{s} = \mathbf{v} \times \mathbf{s}\,, \tag{3.19}$$

where $\mathbf{v}$ is the velocity of $\mathbf{s}$. This formula is useful for differentiating quantities that do not change in their own right, but are attached to moving rigid bodies (e.g., joint axis vectors).

3.2.9 Acceleration

Spatial acceleration is defined as the rate of change of spatial velocity. Unfortunately, this means that spatial acceleration differs from the classical textbook definition of rigid-body acceleration, which we shall call *classical acceleration*. Essentially, the difference can be summarized as follows

$$\underline{\mathbf{a}} = \begin{pmatrix} \underline{\dot{\boldsymbol{\omega}}} \\ \underline{\dot{\boldsymbol{v}}}_O \end{pmatrix} \quad \text{and} \quad \underline{\mathbf{a}}' = \begin{pmatrix} \underline{\dot{\boldsymbol{\omega}}} \\ \underline{\dot{\boldsymbol{v}}}'_O \end{pmatrix}, \tag{3.20}$$

where $\underline{\mathbf{a}}$ is the spatial acceleration, $\underline{\mathbf{a}}'$ is the classical acceleration, $\underline{\dot{\boldsymbol{v}}}_O$ is the derivative of $\underline{\boldsymbol{v}}_O$ taking O to be fixed in space, and $\underline{\dot{\boldsymbol{v}}}'_O$ is the derivative of $\underline{\boldsymbol{v}}_O$ taking O to be fixed in the body. The two accelerations are related by

$$\underline{\mathbf{a}}' = \underline{\mathbf{a}} + \begin{pmatrix} \mathbf{0} \\ \underline{\boldsymbol{\omega}} \times \underline{\boldsymbol{v}}_O \end{pmatrix}. \tag{3.21}$$

If $\boldsymbol{r}$ is a position vector giving the position of the body-fixed point at O relative to any fixed point, then

$$\begin{aligned} \boldsymbol{v}_O &= \dot{\boldsymbol{r}}\,, \\ \dot{\boldsymbol{v}}'_O &= \ddot{\boldsymbol{r}}\,, \\ \dot{\boldsymbol{v}}_O &= \ddot{\boldsymbol{r}} - \boldsymbol{\omega} \times \boldsymbol{v}_O\,. \end{aligned} \tag{3.22}$$

The practical difference is that spatial accelerations are easier to use. For example, if the bodies B_1 and B_2 have velocities of $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively, and $\mathbf{v}_{\mathrm{rel}}$ is the relative velocity of B_2 with respect to B_1, then

$$\mathbf{v}_2 = \mathbf{v}_1 + \mathbf{v}_{\mathrm{rel}}\,.$$

The relationship between their spatial accelerations is obtained simply by differentiating the velocity formula

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{v}_2 = \mathbf{v}_1 + \mathbf{v}_{\mathrm{rel}}) \quad \Rightarrow \quad \mathbf{a}_2 = \mathbf{a}_1 + \mathbf{a}_{\mathrm{rel}}\,.$$

Observe that spatial accelerations are composed by addition, exactly like velocities. There are no Coriolis or centrifugal terms to worry about. This is a significant improvement on the formulae for composing classical accelerations, such as those in [3.2, 13, 14].

3.2.10 Spatial Momentum

Suppose that a rigid body has a mass of m, a center of mass at C, and a rotational inertia of $\bar{\boldsymbol{I}}^{\mathrm{cm}}$ about C (Fig. 3.2). If this body is moving with a spatial velocity of $\underline{\mathbf{v}}_C = (\underline{\boldsymbol{\omega}}^{\mathrm{T}}\,\underline{\boldsymbol{v}}_C^{\mathrm{T}})^{\mathrm{T}}$, then its linear momentum is $\boldsymbol{h} = m\boldsymbol{v}_C$, and its intrinsic angular momentum is $\boldsymbol{h}_C = \bar{\boldsymbol{I}}^{\mathrm{cm}}\boldsymbol{\omega}$. Its moment of momentum about a general point, O, is $\boldsymbol{h}_O = \boldsymbol{h}_C + \boldsymbol{c} \times \boldsymbol{h}$, where $\boldsymbol{c} = \overrightarrow{OC}$. We can assemble these vectors into a spatial momentum vector as follows

$$\underline{\mathbf{h}}_C = \begin{pmatrix} \underline{\boldsymbol{h}}_C \\ \underline{\boldsymbol{h}} \end{pmatrix} = \begin{pmatrix} \bar{\boldsymbol{I}}^{\mathrm{cm}}\underline{\boldsymbol{\omega}} \\ m\underline{\boldsymbol{v}}_C \end{pmatrix} \tag{3.23}$$

and

$$\underline{\mathbf{h}}_O = \begin{pmatrix} \underline{\boldsymbol{h}}_O \\ \underline{\boldsymbol{h}} \end{pmatrix} = \begin{pmatrix} \mathbf{1} & S(\underline{\boldsymbol{c}}) \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \underline{\mathbf{h}}_C\,. \tag{3.24}$$

Spatial momentum is a force vector, and transforms accordingly.

3.2.11 Spatial Inertia

The spatial momentum of a rigid body is the product of its spatial inertia and velocity

$$\mathbf{h} = \boldsymbol{I}\mathbf{v}\,, \tag{3.25}$$

where $\boldsymbol{I}$ is the spatial inertia. Expressed in Plücker coordinates at C, we have

$$\mathbf{h}_C = \boldsymbol{I}_C \mathbf{v}_C \,, \tag{3.26}$$

which implies

$$\boldsymbol{I}_C = \begin{pmatrix} \bar{\boldsymbol{I}}^{\mathrm{cm}} & \mathbf{0} \\ \mathbf{0} & m\mathbf{1} \end{pmatrix} . \tag{3.27}$$

This is the general formula for the spatial inertia of a rigid body expressed at its center of mass. To express it at another point, O, we proceed as follows. From (3.24), (3.26), and (3.27)

$$\begin{aligned} \underline{\mathbf{h}}_O &= \begin{pmatrix} \mathbf{1} & \boldsymbol{S}(\underline{\boldsymbol{c}}) \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \begin{pmatrix} \bar{\boldsymbol{I}}^{\mathrm{cm}} & \mathbf{0} \\ \mathbf{0} & m\mathbf{1} \end{pmatrix} \underline{\mathbf{v}}_C \\ &= \begin{pmatrix} \mathbf{1} & \boldsymbol{S}(\underline{\boldsymbol{c}}) \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \begin{pmatrix} \bar{\boldsymbol{I}}^{\mathrm{cm}} & \mathbf{0} \\ \mathbf{0} & m\mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \boldsymbol{S}(\underline{\boldsymbol{c}})^{\mathrm{T}} & \mathbf{1} \end{pmatrix} \underline{\mathbf{v}}_O \\ &= \begin{pmatrix} \bar{\boldsymbol{I}}^{\mathrm{cm}} + m\boldsymbol{S}(\underline{\boldsymbol{c}})\boldsymbol{S}(\underline{\boldsymbol{c}})^{\mathrm{T}} & m\boldsymbol{S}(\underline{\boldsymbol{c}}) \\ m\boldsymbol{S}(\underline{\boldsymbol{c}})^{\mathrm{T}} & m\mathbf{1} \end{pmatrix} \underline{\mathbf{v}}_O \,; \end{aligned}$$

but we also have that $\underline{\mathbf{h}}_O = \boldsymbol{I}_O \underline{\mathbf{v}}_O$, so

$$\boldsymbol{I}_O = \begin{pmatrix} \bar{\boldsymbol{I}}^{\mathrm{cm}} + m\boldsymbol{S}(\underline{\boldsymbol{c}})\boldsymbol{S}(\underline{\boldsymbol{c}})^{\mathrm{T}} & m\boldsymbol{S}(\underline{\boldsymbol{c}}) \\ m\boldsymbol{S}(\underline{\boldsymbol{c}})^{\mathrm{T}} & m\mathbf{1} \end{pmatrix} . \tag{3.28}$$

This equation can also be written

$$\boldsymbol{I}_O = \begin{pmatrix} \bar{\boldsymbol{I}}_O & m\boldsymbol{S}(\underline{\boldsymbol{c}}) \\ m\boldsymbol{S}(\underline{\boldsymbol{c}})^{\mathrm{T}} & m\mathbf{1} \end{pmatrix} , \tag{3.29}$$

where

$$\bar{\boldsymbol{I}}_O = \bar{\boldsymbol{I}}^{\mathrm{cm}} + m\boldsymbol{S}(\underline{\boldsymbol{c}})\boldsymbol{S}(\underline{\boldsymbol{c}})^{\mathrm{T}} \tag{3.30}$$

is the rotational inertia of the rigid body about O.

Spatial inertia matrices are symmetric and positive-definite. In the general case, 21 numbers are required to specify a spatial inertia (e.g., for an articulated-body or operational-space inertia); but a rigid-body inertia needs only 10 parameters: the mass, the coordinates of the center of mass, and the six independent elements of either $\bar{\boldsymbol{I}}^{\mathrm{cm}}$ or $\bar{\boldsymbol{I}}_O$.

The transformation rule for spatial inertias is

$$\boldsymbol{I}_B = {}^B\boldsymbol{X}_A^F \, \boldsymbol{I}_A \, {}^A\boldsymbol{X}_B \,, \tag{3.31}$$

where A and B are any two coordinate systems. In practice, we often need to calculate $\boldsymbol{I}_A$ from $\boldsymbol{I}_B$, given only ${}^B\boldsymbol{X}_A$. The formula for this transformation is

$$\boldsymbol{I}_A = ({}^B\boldsymbol{X}_A)^{\mathrm{T}} \boldsymbol{I}_B \, {}^B\boldsymbol{X}_A \,. \tag{3.32}$$

If two bodies, having inertias $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$, are rigidly connected to form a single composite body, then the inertia of the composite, $\boldsymbol{I}_{\mathrm{tot}}$, is the sum of the inertias of its parts

$$\boldsymbol{I}_{\mathrm{tot}} = \boldsymbol{I}_1 + \boldsymbol{I}_2 \,. \tag{3.33}$$

This single equation takes the place of three equations in the traditional 3-D vector approach: one to compute the composite mass, one to compute the composite center of mass, and one to compute the composite rotational inertia. If a rigid body with inertia $\boldsymbol{I}$ is moving with a velocity of $\mathbf{v}$, then its kinetic energy is

$$T = \frac{1}{2} \mathbf{v} \cdot \boldsymbol{I}\mathbf{v} \,. \tag{3.34}$$

If a rigid body, B, is part of a larger system, then it is possible to define an apparent-inertia matrix for B, which describes the relationship between a force acting on B and its resulting acceleration, taking into account the effects of the other bodies in the system. Such quantities are called articulated-body inertias. If B happens to be the end-effector of a robot, then its apparent inertia is called an operational-space inertia.

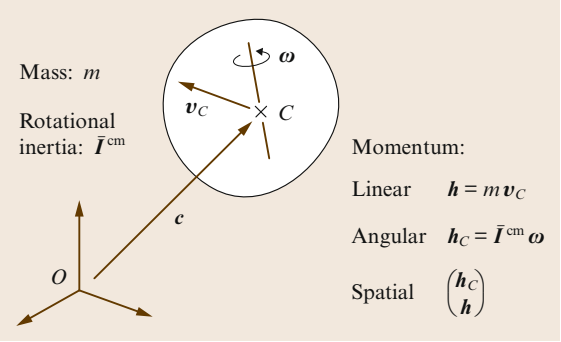


Fig. 3.2 Spatial momentum

3.2.12 Equation of Motion

The spatial equation of motion states that the net force acting on a rigid body equals its rate of change of momentum

$$\mathbf{f} = \frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{I}\mathbf{v}) = \boldsymbol{I}\mathbf{a} + \dot{\boldsymbol{I}}\mathbf{v} \,.$$

It can be shown that the expression $\dot{\boldsymbol{I}}\mathbf{v}$ evaluates to $(\mathbf{v} \times \boldsymbol{I}\mathbf{v})$ [3.8, 15], so the equation of motion can be written

$$\mathbf{f} = \boldsymbol{I}\mathbf{a} + \mathbf{v} \times \boldsymbol{I}\mathbf{v} \,. \tag{3.35}$$

This single equation incorporates both Newton's and Euler's equations of motion for a rigid body. To

Table 3.1 Summary of spatial vector notation

Spatial quantities:	
$\mathbf{v}$	Velocity of a rigid body
$\mathbf{a}$	Spatial acceleration of a rigid body ($\mathbf{a} = \dot{\mathbf{v}}$)
$\mathbf{a}'$	Classical description of rigid-body acceleration expressed as a 6-D vector
$\mathbf{f}$	Force acting on a rigid body
$\boldsymbol{I}$	Inertia of a rigid body
$\boldsymbol{X}$	Plücker coordinate transform for motion vectors
$\boldsymbol{X}^F$	Plücker coordinate transform for force vectors ($\boldsymbol{X}^F = \boldsymbol{X}^{-T}$)
${}^B\boldsymbol{X}_A$	Plücker transform from A coordinates to B coordinates
$\mathbf{m}$	A generic motion vector (any element of M^6)
3-D quantities:	
O	Coordinate system origin
$\boldsymbol{r}$	Position of the body-fixed point at O relative to any fixed point in space
$\boldsymbol{\omega}$	Angular velocity of a rigid body
$\boldsymbol{v}_O$	Linear velocity of the body-fixed point at O ($\boldsymbol{v}_O = \dot{\boldsymbol{r}}$)
$\dot{\boldsymbol{\omega}}$	Angular acceleration of a rigid body
$\dot{\boldsymbol{v}}_O$	The derivative of $\boldsymbol{v}_O$ taking O to be fixed in space
$\dot{\boldsymbol{v}}'_O$	The derivative of $\boldsymbol{v}_O$ taking O to be fixed in the body; the classical acceleration of the body-fixed point at O ($\dot{\boldsymbol{v}}'_O = \ddot{\boldsymbol{r}}$)
$\boldsymbol{f}$	Linear force acting on a rigid body, or the resultant of a system of linear forces
$\boldsymbol{n}_O$	Moment about O of a linear force or system of linear forces
m	Mass of a rigid body
$\boldsymbol{c}$	Position of a rigid body's center of mass, measured relative to O
$\boldsymbol{h}$	First moment of mass of a rigid body, $\boldsymbol{h} = m\boldsymbol{c}$; can also denote linear momentum
$\bar{\boldsymbol{I}}^{\mathrm{cm}}$	Moment of inertia about a body's centre of mass
$\bar{\boldsymbol{I}}$	Moment of inertia about O
${}^B\boldsymbol{R}_A$	Orthonormal rotation matrix transforming from A coordinates to B coordinates
${}^A\boldsymbol{p}_B$	Location of the origin of B coordinates relative to the origin of A coordinates, expressed in A coordinates

verify this, we can recover them as follows. Expressing (3.35) at the body's center of mass, and using (3.16), (3.14), and (3.22), we have

$$\begin{aligned}\begin{pmatrix}\underline{\boldsymbol{n}}_C\\ \underline{\boldsymbol{f}}\end{pmatrix} &= \begin{pmatrix}\bar{\boldsymbol{I}}^{\mathrm{cm}} & \mathbf{0}\\ \mathbf{0} & m\mathbf{1}\end{pmatrix}\begin{pmatrix}\dot{\underline{\boldsymbol{\omega}}}\\ \dot{\underline{\boldsymbol{v}}}_C\end{pmatrix} - \begin{pmatrix}S(\underline{\boldsymbol{\omega}})^{\mathrm{T}} & S(\underline{\boldsymbol{v}}_C)^{\mathrm{T}}\\ \mathbf{0} & S(\underline{\boldsymbol{\omega}})^{\mathrm{T}}\end{pmatrix}\begin{pmatrix}\bar{\boldsymbol{I}}^{\mathrm{cm}}\underline{\boldsymbol{\omega}}\\ m\underline{\boldsymbol{v}}_C\end{pmatrix}\\ &= \begin{pmatrix}\bar{\boldsymbol{I}}^{\mathrm{cm}} & \mathbf{0}\\ \mathbf{0} & m\mathbf{1}\end{pmatrix}\begin{pmatrix}\dot{\underline{\boldsymbol{\omega}}}\\ \ddot{\underline{\boldsymbol{c}}} - \underline{\boldsymbol{\omega}}\times\underline{\boldsymbol{v}}_C\end{pmatrix} + \begin{pmatrix}\underline{\boldsymbol{\omega}}\times\bar{\boldsymbol{I}}^{\mathrm{cm}}\underline{\boldsymbol{\omega}}\\ m\underline{\boldsymbol{\omega}}\times\underline{\boldsymbol{v}}_C\end{pmatrix}\\ &= \begin{pmatrix}\bar{\boldsymbol{I}}^{\mathrm{cm}}\dot{\underline{\boldsymbol{\omega}}} + \underline{\boldsymbol{\omega}}\times\bar{\boldsymbol{I}}^{\mathrm{cm}}\underline{\boldsymbol{\omega}}\\ m\ddot{\underline{\boldsymbol{c}}}\end{pmatrix}.\end{aligned} \tag{3.36}$$

Table 3.1 (continued)

Equations
$\mathbf{v} = \begin{pmatrix}\boldsymbol{\omega}\\ \boldsymbol{v}_O\end{pmatrix} \quad \mathbf{a} = \begin{pmatrix}\dot{\boldsymbol{\omega}}\\ \dot{\boldsymbol{v}}_O\end{pmatrix} = \begin{pmatrix}\dot{\boldsymbol{\omega}}\\ \ddot{\boldsymbol{r}} - \boldsymbol{\omega}\times\dot{\boldsymbol{r}}\end{pmatrix}$
$\mathbf{f} = \begin{pmatrix}\boldsymbol{n}_O\\ \boldsymbol{f}\end{pmatrix} \quad \mathbf{a}' = \begin{pmatrix}\dot{\boldsymbol{\omega}}\\ \dot{\boldsymbol{v}}'_O\end{pmatrix} = \begin{pmatrix}\dot{\boldsymbol{\omega}}\\ \ddot{\boldsymbol{r}}\end{pmatrix} = \mathbf{a} + \begin{pmatrix}\mathbf{0}\\ \boldsymbol{\omega}\times\boldsymbol{v}_O\end{pmatrix}$
$\boldsymbol{I} = \begin{pmatrix}\bar{\boldsymbol{I}} & S(\boldsymbol{h})\\ S(\boldsymbol{h})^{\mathrm{T}} & m\mathbf{1}\end{pmatrix} = \begin{pmatrix}\bar{\boldsymbol{I}}^{\mathrm{cm}} + mS(\boldsymbol{c})S(\boldsymbol{c})^{\mathrm{T}} & mS(\boldsymbol{c})\\ mS(\boldsymbol{c})^{\mathrm{T}} & m\mathbf{1}\end{pmatrix}$
${}^B\boldsymbol{X}_A = \begin{pmatrix}{}^B\boldsymbol{R}_A & \mathbf{0}\\ {}^B\boldsymbol{R}_A S({}^A\boldsymbol{p}_B)^{\mathrm{T}} & {}^B\boldsymbol{R}_A\end{pmatrix} = \begin{pmatrix}{}^B\boldsymbol{R}_A & \mathbf{0}\\ S({}^B\boldsymbol{p}_A)\,{}^B\boldsymbol{R}_A & {}^B\boldsymbol{R}_A\end{pmatrix}$
$\mathbf{v}\cdot\mathbf{f} = \mathbf{f}\cdot\mathbf{v} = \mathbf{v}^{\mathrm{T}}\mathbf{f} = \boldsymbol{\omega}\cdot\boldsymbol{n}_O + \boldsymbol{v}_O\cdot\boldsymbol{f}$
$\mathbf{v}\times\mathbf{m} = \begin{pmatrix}\boldsymbol{\omega}\times\boldsymbol{m}\\ \boldsymbol{v}_O\times\boldsymbol{m} + \boldsymbol{\omega}\times\boldsymbol{m}_O\end{pmatrix} = \begin{pmatrix}S(\boldsymbol{\omega}) & \mathbf{0}\\ S(\boldsymbol{v}_O) & S(\boldsymbol{\omega})\end{pmatrix}\begin{pmatrix}\boldsymbol{m}\\ \boldsymbol{m}_O\end{pmatrix}$
$\mathbf{v}\times\mathbf{f} = \begin{pmatrix}\boldsymbol{\omega}\times\boldsymbol{n}_O + \boldsymbol{v}_O\times\boldsymbol{f}\\ \boldsymbol{\omega}\times\boldsymbol{f}\end{pmatrix} = \begin{pmatrix}S(\boldsymbol{\omega}) & S(\boldsymbol{v}_O)\\ \mathbf{0} & S(\boldsymbol{\omega})\end{pmatrix}\begin{pmatrix}\boldsymbol{n}_O\\ \boldsymbol{f}\end{pmatrix}$

Compact computer representations

Mathematical object	Size	Computer representation	Size
$\begin{pmatrix}\boldsymbol{\omega}\\ \boldsymbol{v}_O\end{pmatrix}$	6×1	$(\boldsymbol{\omega}\,;\,\boldsymbol{v}_O)$	3+3
$\begin{pmatrix}\boldsymbol{n}_O\\ \boldsymbol{f}\end{pmatrix}$	6×1	$(\boldsymbol{n}_O\,;\,\boldsymbol{f})$	3+3
$\begin{pmatrix}\bar{\boldsymbol{I}} & S(\boldsymbol{h})\\ S(\boldsymbol{h})^{\mathrm{T}} & m\mathbf{1}\end{pmatrix}$	6×6	$(m\,;\,\boldsymbol{h}\,;\,\bar{\boldsymbol{I}})$	1+3+9
$\begin{pmatrix}\boldsymbol{R} & \mathbf{0}\\ \boldsymbol{R}S(\boldsymbol{p})^{\mathrm{T}} & \boldsymbol{R}\end{pmatrix}$	6×6	$(\boldsymbol{R}\,;\,\boldsymbol{p})$	9+3

Efficient spatial arithmetic formulae

Expression	Computed value
$\boldsymbol{X}\,\mathbf{v}$	$(\boldsymbol{R}\,\boldsymbol{\omega}\,;\,\boldsymbol{R}(\boldsymbol{v}_O - \boldsymbol{p}\times\boldsymbol{\omega}))$
$\boldsymbol{X}^F\,\mathbf{f}$	$(\boldsymbol{R}(\boldsymbol{n}_O - \boldsymbol{p}\times\boldsymbol{f})\,;\,\boldsymbol{R}\boldsymbol{f})$
$\boldsymbol{X}^{-1}$	$(\boldsymbol{R}^{\mathrm{T}}\,;\,-\boldsymbol{R}\boldsymbol{p})$
$\boldsymbol{X}^{-1}\,\mathbf{v}$	$(\boldsymbol{R}^{\mathrm{T}}\boldsymbol{\omega}\,;\,\boldsymbol{R}^{\mathrm{T}}\boldsymbol{v}_O + \boldsymbol{p}\times\boldsymbol{R}^{\mathrm{T}}\boldsymbol{\omega})$
$(\boldsymbol{X}^F)^{-1}\,\mathbf{f}$	$(\boldsymbol{R}^{\mathrm{T}}\boldsymbol{n}_O + \boldsymbol{p}\times\boldsymbol{R}^{\mathrm{T}}\boldsymbol{f}\,;\,\boldsymbol{R}^{\mathrm{T}}\boldsymbol{f})$
$\boldsymbol{X}_1\boldsymbol{X}_2$	$(\boldsymbol{R}_1\boldsymbol{R}_2\,;\,\boldsymbol{p}_2 + \boldsymbol{R}_2^{\mathrm{T}}\boldsymbol{p}_1)$
$\boldsymbol{I}_1 + \boldsymbol{I}_2$	$(m_1 + m_2\,;\,\boldsymbol{h}_1 + \boldsymbol{h}_2\,;\,\bar{\boldsymbol{I}}_1 + \bar{\boldsymbol{I}}_2)$
$\boldsymbol{I}\,\mathbf{v}$	$(\bar{\boldsymbol{I}}\boldsymbol{\omega} + \boldsymbol{h}\times\boldsymbol{v}_O\,;\,m\,\boldsymbol{v}_O - \boldsymbol{h}\times\boldsymbol{\omega})$
$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{I}\boldsymbol{X}$	$(m\,;\,\boldsymbol{R}^{\mathrm{T}}\boldsymbol{h} + m\boldsymbol{p}\,;\,\boldsymbol{R}^{\mathrm{T}}\bar{\boldsymbol{I}}\boldsymbol{R} - S(\boldsymbol{p})S(\boldsymbol{R}^{\mathrm{T}}\boldsymbol{h}) - S(\boldsymbol{R}^{\mathrm{T}}\boldsymbol{h} + m\boldsymbol{p})S(\boldsymbol{p}))$

For meaning of $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{I}\boldsymbol{X}$ see (3.32)

3.2.13 Computer Implementation

The easiest way to implement spatial vector arithmetic on a computer is to start with an existing matrix arithmetic tool, like MATLAB and write (or download from the Web) routines to do the following:

1. Calculate $\boldsymbol{S}(\mathbf{m})$ from $\mathbf{m}$ according to (3.14).
2. Compose $\boldsymbol{X}$ from $\boldsymbol{R}$ and $\boldsymbol{p}$ according to (3.9).
3. Compose $\boldsymbol{I}$ from m, $\boldsymbol{c}$, and $\bar{\boldsymbol{I}}^{\mathrm{cm}}$ according to (3.28).

All other spatial arithmetic operations can be performed using standard matrix arithmetic routines. However, some additional routines could usefully be added to this list, such as:

- Routines to calculate $\boldsymbol{R}$ from various other representations of rotation.
- Routines to convert between spatial and 4×4 matrix quantities.

This is the recommended approach whenever human productivity is more important than computational efficiency. A software package along these lines can be found at [3.16].

If greater efficiency is required, then a more elaborate spatial arithmetic library must be used, in which

1. A dedicated data structure is defined for each kind of spatial quantity, and
2. A suite of calculation routines are provided, each implementing a spatial arithmetic operation by means of an efficient formula.

Some examples of suitable data structures and efficient formulae are shown in Table 3.1. Observe that the suggested data structures for rigid-body inertias and Plücker transforms contain only a third as many numbers as the 6×6 matrices they represent. The efficient arithmetic formulae listed in this table offer cost savings ranging from a factor of 1.5 to a factor of 6 relative to the use of general 6×6 and 6×1 matrix arithmetic. Even more efficient formulae can be found in [3.17].

3.2.14 Summary

Spatial vectors are 6-D vectors that combine the linear and angular aspects of rigid-body motion, resulting in a compact notation that is very suitable for describing dynamics algorithms. To avoid a few name clashes with 3-D vectors, we have used bold upright letters to denote spatial vectors, while tensors are still denoted by italics. In the sections that follow, upright letters will be used to denote both spatial vectors and vectors that are concatenations of other vectors, like $\dot{\mathbf{q}}$.

Table 3.1 presents a summary of the spatial quantities and operators introduced in this section, together with the formulae that define them in terms of 3-D quantities and operators. It also presents data structures and formulae for efficient computer implementation of spatial arithmetic. This table should be read in conjunction with Tables 2.5 and 2.6, which show how to compute the orientation, position, and spatial velocities for a variety of joint types. Note that ${}^{j}\boldsymbol{R}_{i}$ and ${}^{j}\boldsymbol{p}_{i}$ in these tables correspond to ${}^{B}\boldsymbol{R}_{A}^{T}$ and ${}^{A}\boldsymbol{p}_{B}$, respectively, when read in conjunction with Table 3.1.

3.3 Canonical Equations

The equations of motion of a robot mechanism are usually presented in one of two canonical forms: the joint-space formulation,

$$\boldsymbol{H}(\mathbf{q})\ddot{\mathbf{q}}+\boldsymbol{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+\boldsymbol{\tau}_{\mathrm{g}}(\mathbf{q})=\boldsymbol{\tau}\ , \tag{3.37}$$

or the operational-space formulation,

$$\boldsymbol{\Lambda}(\boldsymbol{x})\dot{\mathbf{v}}+\boldsymbol{\mu}(\boldsymbol{x},\mathbf{v})+\boldsymbol{\rho}(\boldsymbol{x})=\mathbf{f}\,. \tag{3.38}$$

These equations show the functional dependencies explicitly: $\boldsymbol{H}$ is a function of $\mathbf{q}$, $\boldsymbol{\Lambda}$ is a function of $\boldsymbol{x}$, and so on. Once these dependencies are understood, they are usually omitted. In (3.38), $\boldsymbol{x}$ is a vector of operational-space coordinates, while $\mathbf{v}$ and $\mathbf{f}$ are spatial vectors denoting the velocity of the end-effector and the external force acting on it. If the robot is redundant, then the coefficients of this equation must be defined as functions of $\mathbf{q}$ and $\dot{\mathbf{q}}$ rather than $\boldsymbol{x}$ and $\mathbf{v}$.

These two equations are further explained below, along with a description of the Lagrange formulation of (3.37), and the impulsive equations of motion for impact.

3.3.1 Joint-Space Formulation

The symbols $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$, and $\boldsymbol{\tau}$ denote n-dimensional vectors of joint position, velocity, acceleration and force variables, respectively, where n is the number of degrees of motion freedom of the robot mechanism. $\boldsymbol{H}$ is an $n\times n$ symmetric, positive-definite matrix, and is called the generalized, or joint-space, inertia matrix (JSIM). $\boldsymbol{C}$ is an $n\times n$ matrix such that $\boldsymbol{C}\,\dot{\mathbf{q}}$ is the vector of Coriolis and centrifugal terms (collectively known as *velocity product* terms); and $\boldsymbol{\tau}_{\mathrm{g}}$ is the vector of gravity terms. More terms can be added to this equation, as required, to account for other dynamical effects (e.g., viscous friction). The effects of a force $\mathbf{f}$ exerted on the mechanism at the end-effector can be accounted for by adding the term $\boldsymbol{J}^{\mathrm{T}}\mathbf{f}$ to the right side of (3.37), where $\boldsymbol{J}$ is the Jacobian of the end-effector (Sect. 2.8.1).

$\mathbf{q}$ specifies the coordinates of a point in the mechanism's configuration space. If the mechanism is a kinematic tree (Sect. 3.4), then $\mathbf{q}$ contains every joint variable in the mechanism, otherwise it contains only an independent subset. The elements of $\mathbf{q}$ are generalized coordinates. Likewise, the elements of $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$, and $\boldsymbol{\tau}$ are generalized velocities, accelerations, and forces.

3.3.2 Lagrange Formulation

Various methods exist for deriving the terms in (3.37). The two that are most commonly used in robotics are the Newton–Euler formulation and the Lagrange formulation. The former works directly with Newton's and Euler's equations for a rigid body, which are contained within the spatial equation of motion, (3.35). This formulation is especially amenable to the development of efficient recursive algorithms for dynamics computations, such as those described in Sects. 3.5 and 3.6.

The Lagrange formulation proceeds via the Lagrangian of the robot mechanism,

$$L = T - U \,, \tag{3.39}$$

where T and U are the total kinetic and potential energy, respectively, of the mechanism. The kinetic energy is given by

$$T = \frac{1}{2}\dot{\mathbf{q}}^{\mathrm{T}}\boldsymbol{H}\dot{\mathbf{q}} \,. \tag{3.40}$$

The dynamic equations of motion can then be developed using Lagrange's equation for each generalized coordinate

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i \,. \tag{3.41}$$

The resulting equation can be written in scalar form

$$\sum_{j=1}^{n} H_{ij}\ddot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} C_{ijk}\dot{q}_j\dot{q}_k + \tau_{gi} = \tau_i \,, \tag{3.42}$$

which shows the structure of the velocity-product terms. C_{ijk} are known as Christoffel symbols of the first type, and are given by

$$C_{ijk} = \frac{1}{2}\left(\frac{\partial H_{ij}}{\partial q_k} + \frac{\partial H_{ik}}{\partial q_j} - \frac{\partial H_{jk}}{\partial q_i}\right) \,. \tag{3.43}$$

They are functions of only the position variables, q_i. The elements of $\boldsymbol{C}$ in (3.37) can be defined as

$$C_{ij} = \sum_{k=1}^{n} C_{ijk}\dot{q}_k \,. \tag{3.44}$$

However, $\boldsymbol{C}$ is not unique, and other definitions are possible.

With the choice of $\boldsymbol{C}$ given in (3.44), it is possible to show that the matrix $\boldsymbol{N}$, given by

$$\boldsymbol{N}(\mathbf{q},\dot{\mathbf{q}}) = \dot{\boldsymbol{H}}(\mathbf{q}) - 2\boldsymbol{C}(\mathbf{q},\dot{\mathbf{q}}) \,, \tag{3.45}$$

is skew-symmetric [3.18]. Thus, for any $n \times 1$ vector $\boldsymbol{\alpha}$,

$$\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{N}(\mathbf{q},\dot{\mathbf{q}})\,\boldsymbol{\alpha} = 0 \,. \tag{3.46}$$

This property is quite useful in control, especially when considering $\boldsymbol{\alpha} = \dot{\mathbf{q}}$, which gives

$$\dot{\mathbf{q}}^{\mathrm{T}}\boldsymbol{N}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}} = 0 \,. \tag{3.47}$$

By applying the principle of conservation of energy, it can be shown that (3.47) holds for any choice of the matrix $\boldsymbol{C}$ [3.18, 19].

3.3.3 Operational-Space Formulation

In (3.38), $\boldsymbol{x}$ is a 6-D vector of operational-space coordinates giving the position and orientation of the robot's end-effector; $\mathbf{v}$ is the velocity of the end-effector; and $\mathbf{f}$ is the force exerted on the end-effector. $\boldsymbol{x}$ is typically a list of Cartesian coordinates, and Euler angles or quaternion components, and is related to $\mathbf{v}$ via a differential equation of the form

$$\dot{\boldsymbol{x}} = \boldsymbol{E}(\boldsymbol{x})\mathbf{v} \,. \tag{3.48}$$

$\boldsymbol{\Lambda}$ is the operational-space inertia matrix, which is the apparent inertia of the end-effector taking into account the effect of the rest of the robot's mechanism (i. e., it is an articulated-body inertia). $\boldsymbol{\mu}$ and $\boldsymbol{\rho}$ are vectors of velocity-product and gravity terms, respectively.

Operational space (also known as task space) is the space in which high-level motion and force commands are issued and executed. The operational-space formulation is therefore particularly useful in the context of motion and force control systems (Sects. 8.2 and 9.2). Equation (3.38) can be generalized to operational spaces with dimensions other than six, and to operational spaces that incorporate the motions of more than one end-effector [3.20].

The terms in (3.37) and (3.38) are related by the following formulae

$$\mathbf{v} = \boldsymbol{J}\dot{\mathbf{q}} \,, \tag{3.49}$$

$$\dot{\mathbf{v}} = \boldsymbol{J}\ddot{\mathbf{q}} + \dot{\boldsymbol{J}}\dot{\mathbf{q}} \,, \tag{3.50}$$

$$\boldsymbol{\tau} = \boldsymbol{J}^{\mathrm{T}}\mathbf{f} \,, \tag{3.51}$$

$$\boldsymbol{\Lambda} = (\boldsymbol{J}\boldsymbol{H}^{-1}\boldsymbol{J}^{\mathrm{T}})^{-1} \,, \tag{3.52}$$

$$\boldsymbol{\mu} = \boldsymbol{\Lambda}(\boldsymbol{J}\boldsymbol{H}^{-1}\boldsymbol{C}\dot{\mathbf{q}} - \dot{\boldsymbol{J}}\dot{\mathbf{q}}) \,, \tag{3.53}$$

and

$$\boldsymbol{\rho} = \boldsymbol{\Lambda}\boldsymbol{J}\boldsymbol{H}^{-1}\boldsymbol{\tau}_{\mathrm{g}} \,. \tag{3.54}$$

These equations assume that $m \leq n$ (m is the dimension of operational-space coordinates), and that the Jacobian $\boldsymbol{J}$ has full rank. More details can be found in [3.21].

3.3.4 Impact Model

If a robot strikes a rigid body in its environment, then an impulsive force arises at the moment of impact and causes a step change in the robot's velocity. Let us assume that the impact occurs between the end effector and a rigid body in the environment, and that a spatial impulse of $\mathbf{f}'$ is exerted on the end-effector. This impulse causes a step change of $\Delta\mathbf{v}$ in the end-effector's velocity; and the two are related by the operational-space equation of impulsive motion [3.22],

$$\boldsymbol{\Lambda}\Delta\mathbf{v} = \mathbf{f}' \,. \tag{3.55}$$

In joint space, the equation of impulsive motion for a robot mechanism is

$$\boldsymbol{H}\Delta\dot{\mathbf{q}} = \boldsymbol{\tau}' \,, \tag{3.56}$$

where $\boldsymbol{\tau}'$ and $\Delta\dot{\mathbf{q}}$ denote the joint-space impulse and velocity change, respectively. In the case of a collision involving the robot's end-effector, we have

$$\boldsymbol{\tau}' = \boldsymbol{J}^{\mathrm{T}}\mathbf{f}' \tag{3.57}$$

and

$$\Delta\mathbf{v} = \boldsymbol{J}\Delta\dot{\mathbf{q}} \,, \tag{3.58}$$

which follow from (3.51) and (3.49). Equations (3.55)–(3.57) imply that

$$\Delta\dot{\mathbf{q}} = \bar{\boldsymbol{J}}\Delta\mathbf{v} \,, \tag{3.59}$$

where $\bar{\boldsymbol{J}}$ is the inertia-weighted pseudoinverse of $\boldsymbol{J}$ and is given by

$$\bar{\boldsymbol{J}} = \boldsymbol{H}^{-1}\boldsymbol{J}^{\mathrm{T}}\boldsymbol{\Lambda} \,. \tag{3.60}$$

$\bar{\boldsymbol{J}}$ is also known as the dynamically consistent generalized inverse of the Jacobian matrix [3.21]. Note that the expression $\boldsymbol{\Lambda}\boldsymbol{J}\boldsymbol{H}^{-1}$, which appears in (3.53) and (3.54), is equal to $\bar{\boldsymbol{J}}^{\mathrm{T}}$ since $\boldsymbol{H}$ and $\boldsymbol{\Lambda}$ are both symmetric. Although we have introduced $\bar{\boldsymbol{J}}$ in the context of impulsive dynamics, it is more typically used in normal (i. e., non-impulsive) dynamics equations.

3.4 Dynamic Models of Rigid-Body Systems

A basic rigid-body model of a robot mechanism has four components: a connectivity graph, link and joint geometry parameters, link inertia parameters, and a set of joint models. To this model, one can add various force-producing elements, such as springs, dampers, joint friction, actuators, and drives. The actuators and drives, in particular, may have quite elaborate dynamic models of their own. It is also possible to add extra motion freedoms to model elasticity in the joint bearings or links (Chap. 11). This section describes a basic model. More on this topic can be found in books such as [3.3, 8, 23].

3.4.1 Connectivity

A connectivity graph is an undirected graph in which each node represents a rigid body and each arc represents a joint. The graph must be connected; and exactly one node represents a fixed base or reference frame. If the graph represents a mobile robot (i. e., a robot that is not connected to a fixed base), then it is necessary to introduce a fictitious 6-DOF joint between the fixed base and any one body in the mobile robot. The chosen body is then known as a *floating base*. If a single graph is to represent a collection of mobile robots, then each robot has its own floating base, and each floating base has its own 6-DOF joint. Note that a 6-DOF joint imposes no constraints on the two bodies it connects, so the introduction of a 6-DOF joint alters the connectivity of the graph without altering the physical properties of the system it represents.

In graph-theory terminology, a loop is an arc that connects a node to itself, and a cycle is a closed path that does not traverse any arc more than once. In the connectivity graph of a robot mechanism, loops are not allowed, and cycles are called kinematic loops. A mechanism that contains kinematic loops is called a closed-loop mechanism; and a mechanism that does not is called an open-loop mechanism or a kinematic tree. Every closed-loop mechanism has a spanning tree, which defines an open-loop mechanism, and every joint that is not in the spanning tree is called a loop-closing joint. The joints in the tree are called tree joints.

The fixed base serves as the root node of a kinematic tree, and the root node of any spanning tree on a closed-loop mechanism. A kinematic tree is said to be branched if at least one node has at least two children, and unbranched otherwise. An unbranched kinematic

tree is also called a kinematic chain, and a branched tree can be called a branched kinematic chain. A typical industrial robot arm, without a gripper, is a kinematic chain, while a typical humanoid robot is a kinematic tree with a floating base.

In a system containing N_B moving bodies and N_J joints, where N_J includes the 6-DOF joints mentioned above, the bodies and joints are numbered as follows. First, the fixed base is numbered body 0. The other bodies are then numbered from 1 to N_B in any order such that each body has a higher number than its parent. If the system contains kinematic loops then one must first choose a spanning tree, and commit to that choice, since the identity of a body's parent is determined by the spanning tree. This style of numbering is called a *regular numbering scheme*.

Having numbered the bodies, we number the tree joints from 1 to N_B such that joint i connects body i to its parent. The loop-closing joints, if any, are then numbered from N_B+1 to N_J in any order. Each loop-closing joint k closes one independent kinematic loop, and we number the loops from 1 to N_L (where $N_L = N_J - N_B$ is the number of independent loops) such that loop l is the one closed by joint $k = N_B + l$. Kinematic loop l is the unique cycle in the graph that traverses joint k, but does not traverse any other loop-closing joint.

For an unbranched kinematic tree, these rules produce a unique numbering in which the bodies are numbered consecutively from base to tip, and the joints are numbered such that joint i connects bodies i and $i-1$. In all other cases, regular numberings are not unique.

Although the connectivity graph is undirected, it is necessary to assign a direction to each joint for the purpose of defining joint velocity and force. This is necessary for both tree joints and loop-closing joints. Specifically, a joint is said to connect from one body to another. We may call them the predecessor $p(i)$ and successor $s(i)$ for joint i, respectively. Joint velocity is then defined as the velocity of the successor relative to the predecessor; and joint force is defined as a force acting on the successor. It is standard practice (but not a necessity) for all tree joints to connect from the parent to the child.

The connectivity of a kinematic tree, or the spanning tree on a closed-loop mechanism, is described by an N_B-element array of parent body numbers, where the i-th element $p(i)$ is the parent of body i. Note that the parent $p(i)$ for body i is also the predecessor $p(i)$ for joint i, and thus the common notation. Many algorithms rely on the property $p(i) < i$ to perform their calculations in the correct order. The set of all body numbers for the children of body i, $c(i)$, is also useful in many recursive algorithms.

The connectivity data for kinematic loops may be described in a variety of ways. A representation that facilitates use in recursive algorithms includes the following conventions. Loop-closing joint k joins bodies $p(k)$ (the predecessor) and $s(k)$ (the successor). The set $LR(i)$ for body i gives the numbers of the loops for which body i is the root. Using the property $p(i) < i$ for bodies in the spanning tree, the root of a loop is chosen as the body with the lowest number. In addition, the set $LB(i)$ for body i gives the numbers of the loops to which body i belongs but is not the root.

An example of a closed-loop system is given in Fig. 3.3. The system consists of a humanoid mechanism with topologically varying contacts, with the environment and within the mechanism, which form closed loops. The system has $N_B = 16$ moving bodies and $N_J = 19$ joints with $N_L = N_J - N_B = 3$ loops. The main body (1) is considered to be a *floating base* for this mobile robot system. It is connected to the fixed base (0) through a fictitious 6-DOF joint (1). To complete the example, the loop-closing joint and the body numbers $p(k)$ and $s(k)$ as well as the root body for each loop are given in Table 3.2. The body-based sets $c(i)$ and

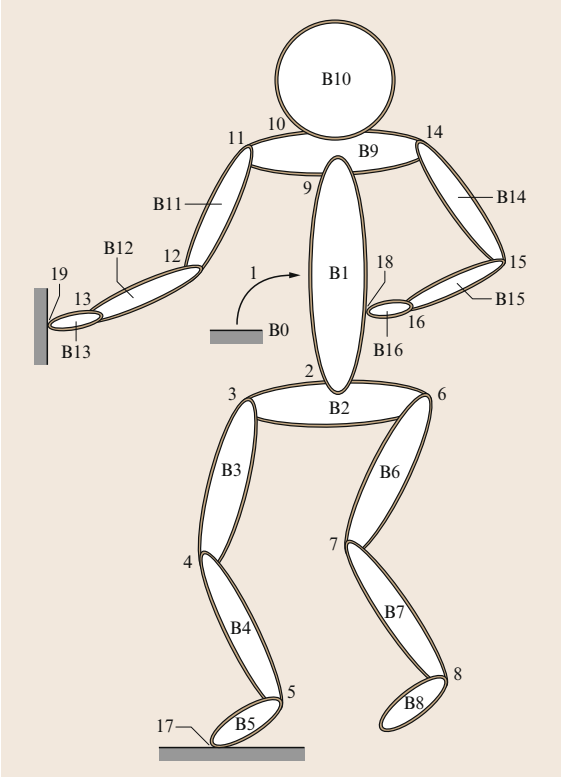


Fig. 3.3 Humanoid robot example. Note: to distinguish between body numbers and joint numbers in this figure, body numbers are preceded by a B for clarity

$LB(i)$ are given in Table 3.3. Note that $LR(0) = \{1, 3\}$ and $LR(1) = \{2\}$ and all other LR sets are null for this example.

3.4.2 Link Geometry

When two bodies are connected by a joint, a complete description of the connection consists of a description of the joint itself, and the locations of two coordinate frames, one in each body, which specify where in each body the joint is located. If there are N_J joints in the system, then there are a total of $2N_J$ joint-attachment frames. One half of these frames are identified with the numbers 1 to N_J, and the remainder with the labels $J1$ to JN_J. Each joint i connects from frame Ji to frame i.

For joints 1 to N_B (i. e., the tree joints), frame i is rigidly attached to body i. For joints $N_B + 1$ to N_J, frame k for loop-closing joint k will be rigidly attached to body $s(k)$. The second coordinate frame Ji is attached to the predecessor $p(i)$ for each joint i, whether it is a tree joint or a loop-closing joint. Coordinate frame Ji provides a base frame for joint i in that the joint rotation and/or translation is defined relative to this frame.

Figure 3.4 shows the coordinate frames and transforms associated with each joint in the system. The overall transform from frame $p(i)$ coordinates to frame i coordinates for a tree joint is given by

$$ {}^{i}\boldsymbol{X}_{p(i)} = {}^{i}\boldsymbol{X}_{Ji}\,{}^{Ji}\boldsymbol{X}_{p(i)} = \boldsymbol{X}_J(i)\boldsymbol{X}_L(i) \,. \tag{3.61} $$

The transform $\boldsymbol{X}_L(i)$ is a fixed link transform which sets the base frame Ji of joint i relative to $p(i)$. It may be used to transform spatial motion vectors from $p(i)$ to Ji coordinates. The transform $\boldsymbol{X}_J(i)$ is a variable joint transform which completes the transformation across joint i from Ji to i coordinates.

Table 3.2 Loop-closing joints and roots of loops for the humanoid example

Loop l	Loop-closing joint k	$p(k)$	$s(k)$	Root
1	17	0	5	0
2	18	16	1	1
3	19	0	13	0

Table 3.3 Body-based sets for the humanoid example

Body i	$c(i)$	$LB(i)$	Body i	$c(i)$	$LB(i)$
0	1		9	10, 11, 14	2, 3
1	2, 9	1, 3	10		
2	3, 6	1	11	12	3
3	4	1	12	13	3
4	5	1	13		3
5		1	14	15	2
6	7		15	16	2
7	8		16		2
8					

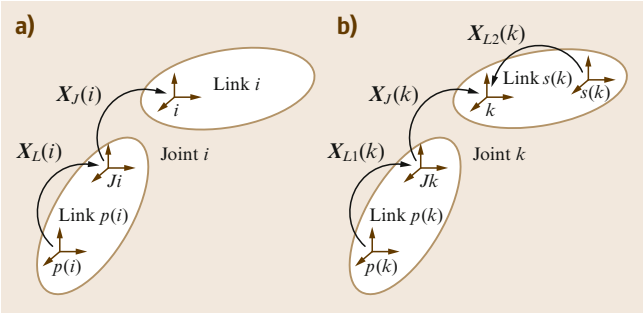


Fig.3.4a,b Coordinate frames and transforms associated with **(a)** a tree joint and **(b)** a loop-closing joint

Similarly, the overall transform from frame $p(k)$ coordinates to frame k coordinates for a loop-closing joint is given by

$$ {}^{k}\boldsymbol{X}_{p(k)} = {}^{k}\boldsymbol{X}_{Jk}\,{}^{Jk}\boldsymbol{X}_{p(k)} = \boldsymbol{X}_J(k)\,\boldsymbol{X}_{L1}(k) \,. \tag{3.62} $$

An additional transform $\boldsymbol{X}_{L2}(k)$ is defined from frame $s(k)$ coordinates to frame k coordinates and is given by

$$ \boldsymbol{X}_{L2}(k) = {}^{k}\boldsymbol{X}_{s(k)} \,. \tag{3.63} $$

Link and joint geometry data can be specified in a variety of different ways. The most common method is to use *Denavit–Hartenberg* parameters [3.24]. However, standard Denavit–Hartenberg parameters are not completely general, and are insufficient for describing the geometry for a branched kinematic tree, or for a mechanism containing certain kinds of multi-DOF joints. A modified form of Denavit–Hartenberg parameters [3.2] is used for single-DOF joints in this Handbook (Sect. 2.4). The parameters have been extended for branched kinematic trees [3.23] and closed-loop mechanisms.

3.4.3 Link Inertias

The link inertia data consists of the masses, positions of centers of mass, and rotational inertias of each link in the mechanism. The inertia parameters for link i are expressed in coordinate frame i, and are therefore constants.

3.4.4 Joint Models

The relationship between connected links is described using the general joint model of *Roberson* and *Schwertassek* [3.3]. For a kinematic tree or spanning tree on

a closed-loop mechanism, an $n_i \times 1$ vector, $\dot{\boldsymbol{q}}_i$, relates the velocity of link i to the velocity of its parent, link $p(i)$, where n_i is the number of degrees of freedom at the joint connecting the two links. For a loop-closing joint in a closed-loop mechanism, the relationship is between the velocity of link $s(i)$ (the successor) and the velocity of link $p(i)$ (the predecessor). In either case, the relationship is between the velocity of coordinate frames i and Ji.

Let $\mathbf{v}_{\text{rel}}$ and $\mathbf{a}_{\text{rel}}$ denote the velocity and acceleration across joint i, that is, the velocity and acceleration of link $s(i)$ relative to $p(i)$. The free modes of the joint are represented by the $6 \times n_i$ matrix $\boldsymbol{\Phi}_i$, such that $\mathbf{v}_{\text{rel}}$ and $\mathbf{a}_{\text{rel}}$ are given as follows

$$\mathbf{v}_{\text{rel}} = \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i \tag{3.64}$$

and

$$\mathbf{a}_{\text{rel}} = \boldsymbol{\Phi}_i \ddot{\boldsymbol{q}}_i + \dot{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i \,, \tag{3.65}$$

where $\boldsymbol{\Phi}_i$ and $\dot{\boldsymbol{\Phi}}_i$ depend on the type of joint [3.3]. The matrix $\boldsymbol{\Phi}_i$ has full column rank, so we can define a complementary matrix, $\boldsymbol{\Phi}_i^{\text{c}}$, such that the 6×6 matrix $\left(\boldsymbol{\Phi}_i \boldsymbol{\Phi}_i^{\text{c}}\right)$ is invertible. We can regard the columns of this matrix as forming a basis on M^6 such that the first n_i basis vectors define the directions in which motion is allowed, and the remaining $6 - n_i = n_i^{\text{c}}$ vectors define directions in which motion is not allowed. Thus, $\boldsymbol{\Phi}_i^{\text{c}}$ represents the constrained modes of joint i.

The force transmitted across joint i from its predecessor to its successor, $\mathbf{f}_i$, is given as follows

$$\mathbf{f}_i = \left(\boldsymbol{\Psi}_i \boldsymbol{\Psi}_i^{\text{c}}\right) \begin{pmatrix} \boldsymbol{\tau}_i \\ \boldsymbol{\lambda}_i \end{pmatrix} , \tag{3.66}$$

where $\boldsymbol{\tau}_i$ is the $n_i \times 1$ vector of applied forces along the free modes, $\boldsymbol{\lambda}_i$ is the $(6 - n_i) \times 1$ vector of constraint forces, and $\boldsymbol{\Psi}_i$ and $\boldsymbol{\Psi}_i^{\text{c}}$ are computed as follows

$$\left(\boldsymbol{\Psi}_i \boldsymbol{\Psi}_i^{\text{c}}\right) = \left(\boldsymbol{\Phi}_i \boldsymbol{\Phi}_i^{\text{c}}\right)^{-T} \,. \tag{3.67}$$

For most common joint types, it is possible to choose $\boldsymbol{\Phi}_i$ and $\boldsymbol{\Phi}_i^{\text{c}}$ such that the matrix $(\boldsymbol{\Phi}_i \boldsymbol{\Phi}_i^{\text{c}})$ is numerically orthonormal, so that $(\boldsymbol{\Psi}_i \boldsymbol{\Psi}_i^{\text{c}})$ is numerically equal to $(\boldsymbol{\Phi}_i \boldsymbol{\Phi}_i^{\text{c}})$. Note that (3.67) implies the following relationships: $(\boldsymbol{\Psi}_i)^{\text{T}} \boldsymbol{\Phi}_i = \mathbf{1}_{n_i \times n_i}$, $(\boldsymbol{\Psi}_i)^{\text{T}} \boldsymbol{\Phi}_i^{\text{c}} = \mathbf{0}_{n_i \times (6-n_i)}$, $(\boldsymbol{\Psi}_i^{\text{c}})^{\text{T}} \boldsymbol{\Phi}_i = \mathbf{0}_{(6-n_i) \times n_i}$, and $(\boldsymbol{\Psi}_i^{\text{c}})^{\text{T}} \boldsymbol{\Phi}_i^{\text{c}} = \mathbf{1}_{(6-n_i) \times (6-n_i)}$. When applied to (3.66), the following useful relationship results

$$\boldsymbol{\tau}_i = \boldsymbol{\Phi}_i^{\text{T}} \mathbf{f}_i \,. \tag{3.68}$$

The value of $\dot{\boldsymbol{\Phi}}_i$ in (3.65) depends on the type of joint. The general formula is

$$\dot{\boldsymbol{\Phi}}_i = \mathring{\boldsymbol{\Phi}}_i + \mathbf{v}_i \times \boldsymbol{\Phi}_i \,, \tag{3.69}$$

where $\mathbf{v}_i$ is the velocity of link i, and $\mathring{\boldsymbol{\Phi}}_i$ is the apparent derivative of $\boldsymbol{\Phi}_i$, as seen by an observer moving with link i, and is given by

$$\mathring{\boldsymbol{\Phi}}_i = \frac{\partial \boldsymbol{\Phi}_i}{\partial \boldsymbol{q}_i} \dot{\boldsymbol{q}}_i \,. \tag{3.70}$$

For most common joint types, $\mathring{\boldsymbol{\Phi}}_i = \mathbf{0}$.

Single-DOF joints ($n_i = 1$) are especially straightforward to work with when using the Denavit–Hartenberg convention. Motion is chosen along (prismatic) or about (revolute) the $\hat{\mathbf{z}}_i$ coordinate axis. In this case, $\boldsymbol{\Phi}_i = (0\,0\,0\,0\,0\,1)^{\text{T}}$ for a prismatic joint and $\boldsymbol{\Phi}_i = (0\,0\,1\,0\,0\,0)^{\text{T}}$ for a revolute joint. Also, $\mathring{\boldsymbol{\Phi}}_i = \mathbf{0}$.

The fictitious 6-DOF joint for a floating base for a mobile robot is also handled relatively easily. For this case, $\boldsymbol{\Phi}_i = \mathbf{1}$ (6×6 identity matrix) and $\mathring{\boldsymbol{\Phi}}_i = \mathbf{0}$.

The revolute joint and floating-base joint, as well as the universal joint ($n_i = 2$) and spherical joint ($n_i = 3$) are illustrated in the example in the next section. For additional details on joint kinematics, see Sect. 2.3.

3.4.5 Example System

In order to illustrate the conventions used for the link and joint models, coordinate frames are attached to the first five links (bodies) and fixed base of the humanoid robot as shown in Fig. 3.5. Note that frame Ji is attached to link $p(i) = i - 1$ for each of the five joints. For this example, the origin of frame $J1$ is set coincident with the origin of frame 0, and the origins of frames $J2$, $J3$, $J4$, and $J5$ are coincident with the origins of frames 2, 3, 4, and 5, respectively.

Note that $J1$ could be set at any position/orientation on the fixed base (B0) to permit the most convenient

Table 3.4 Number of degrees of freedom (n_i), fixed rotation $\left({}^{Ji}\boldsymbol{R}_{p(i)}\right)$ and position $\left({}^{p(i)}\boldsymbol{p}_{Ji}\right)$ from frame $p(i)$ to base frame Ji for joint i of the example system. Note that $2l_i$ is the nominal length of link i along its long axis

Joint	n_i	${}^{Ji}\boldsymbol{R}_{p(i)}$	${}^{p(i)}\boldsymbol{p}_{Ji}$
1	6	$\mathbf{1}_{3\times3}$	$\mathbf{0}_{3\times1}$
2	1	$\mathbf{1}_{3\times3}$	$\begin{pmatrix} 0 \\ 0 \\ -l_1 \end{pmatrix}$
3	3	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ -l_2 \\ 0 \end{pmatrix}$
4	1	$\mathbf{1}_{3\times3}$	$\begin{pmatrix} 0 \\ 2l_3 \\ 0 \end{pmatrix}$
5	2	$\begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 2l_4 \\ 0 \end{pmatrix}$

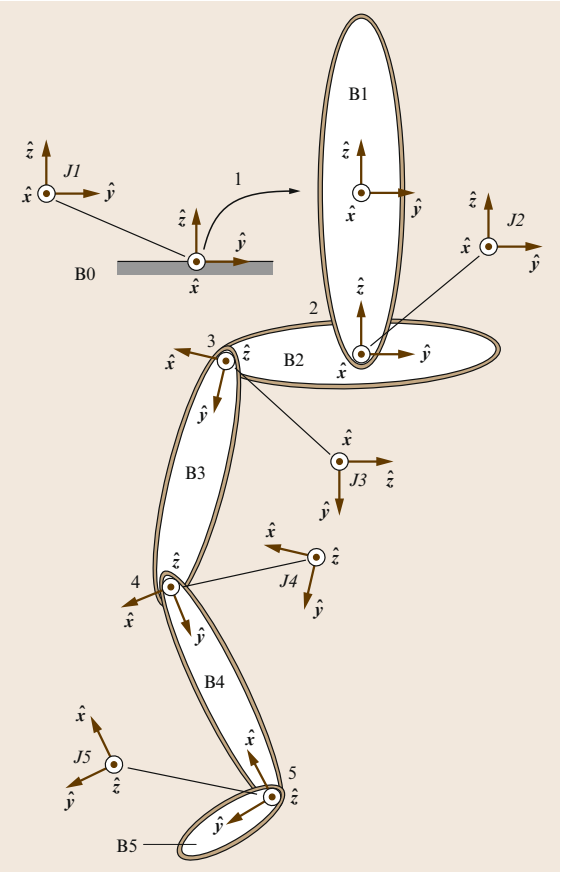


Fig. 3.5 Coordinate frames for the first five links and joints of the humanoid robot example

representation for the motion of the floating base (B1) relative to the fixed base. Also, the origin of $J2$ could be set anywhere along $\hat{z}_2$.

The number of degrees of freedom, and fixed rotation and position of the base frames Ji for each joint of the example system, are given in Table 3.4. The rotation ${}^{Ji}\boldsymbol{R}_{p(i)}$ transforms 3-D vectors in $p(i)$ coordinates to Ji coordinates. The position ${}^{p(i)}\boldsymbol{p}_{Ji}$ is the vector giving the position of the origin O_{Ji} relative to $O_{p(i)}$, expressed in $p(i)$ coordinates. The spatial transform $\boldsymbol{X}_L(i) = {}^{Ji}\boldsymbol{X}_{p(i)}$ may be composed from these 3-D quantities through the equation for ${}^{B}\boldsymbol{X}_A$ in Table 3.1. The humanoid has a floating base, the torso, a revolute joint between the torso and pelvis (about $\hat{z}_2$), a spherical joint at the hip, a revolute joint at the knee, and a universal joint at the ankle. As shown in Fig. 3.5, the leg is slightly bent and the foot is turned out to the side ($\approx 90^\circ$ rotation about $\hat{\boldsymbol{y}}_3$ at the hip).

The free modes, velocity variables, and position variables for all of the joint types in the humanoid are given in Tables 2.5 and 2.6. The expressions for ${}^{j}\boldsymbol{R}_i$ and ${}^{j}\boldsymbol{p}_i$ in these tables give ${}^{i}\boldsymbol{R}_{Ji}^{\mathrm{T}}$ and ${}^{Ji}\boldsymbol{p}_i$, respectively, through which the joint transform $\boldsymbol{X}_J(i) = {}^{i}\boldsymbol{X}_{Ji}$ may be composed. Revolute joints follow the Denavit–Hartenberg convention with rotation about the $\hat{\boldsymbol{z}}_i$ axis. The ankle has a pitch rotation of α_5 about the $\hat{\boldsymbol{z}}_{J5}$ axis followed by a roll rotation of β_5 about the $\hat{\boldsymbol{y}}_5$ axis (see the Z–Y–X Euler angle definitions in Table 2.1). The hip is modeled as a ball-and-socket, spherical joint. To avoid the singularities that are associated with Euler angles, the quaternion $\boldsymbol{\epsilon}_i$ may be used to represent the orientation at the hip. The relationship between the quaternion rate $\dot{\boldsymbol{\epsilon}}_i$ and the relative rotation rate $\boldsymbol{\omega}_{i\,\mathrm{rel}}$ is given in (2.8) of the Handbook.

The floating base uses the position of the torso ${}^{0}\boldsymbol{p}_1$ and quaternion $\boldsymbol{\epsilon}_1$ for its position and orientation state variables, respectively. The position of the torso may be computed by integrating the velocity for the link, as expressed in fixed base coordinates: ${}^{0}\boldsymbol{v}_1 = {}^{0}\boldsymbol{R}_1\boldsymbol{v}_1$, where $\boldsymbol{v}_1$ is the velocity of the torso in moving coordinates.

Note that $\mathring{\boldsymbol{\Phi}}_i = 0$ for all joints except the universal joint. Since the components of $\hat{\boldsymbol{z}}_{J5}$ in link 5 coordinates vary with β_5, $\hat{\boldsymbol{z}}_{J5} \neq 0$. See Sect. 2.3 of the Handbook for further details of the joint kinematics.

3.5 Kinematic Trees

The dynamics of a kinematic tree is simpler, and easier to calculate, than the dynamics of a closed-loop mechanism. Indeed, many algorithms for closed-loop mechanisms work by first calculating the dynamics of a spanning tree, and then subjecting it to the loop-closure constraints.

This section describes the following dynamics algorithms for kinematic trees: the recursive Newton–Euler algorithm (RNEA) for inverse dynamics, the articulated-body algorithm (ABA) for forward dynamics, the composite-rigid-body algorithm (CRBA) for calculating the joint-space inertia matrix (JSIM), and two algorithms to calculate the operational-space inertia matrix (OSIM). Implementations of the first three can be found in [3.16].

3.5.1 The Recursive Newton–Euler Algorithm

This is an $O(n)$ algorithm for calculating the inverse dynamics of a fixed-base kinematic tree, and is based on the very efficient RNEA of *Luh* et al. [3.4]. A floating-

base version can be found in [3.8, 15]. Given the joint position and velocity variables, this algorithm calculates the applied joint torque/force variables required to produce a given set of joint accelerations.

The link velocities and accelerations are first computed through an outward recursion from the fixed base to the leaf links of the tree. The required forces on each link are computed using the Newton–Euler equations (3.35) during this recursion. A second, inward recursion uses the force balance equations at each link to compute the spatial force across each joint and the value of each joint torque/force variable. The key step for computational efficiency is to refer most quantities to local link coordinates. Also, the effects of gravity on each link are efficiently included in the equations by accelerating the base of the mechanism upward.

The calculation proceeds in four steps, as follows, with two steps in each of the two recursions.

Step 1

Calculate the velocity and acceleration of each link in turn, starting with the known velocity and acceleration of the fixed base, and working towards the tips (i. e., the leaf nodes in the connectivity graph).

The velocity of each link in a kinematic tree is given by the recursive formula

$$\mathbf{v}_i = \mathbf{v}_{p(i)} + \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i \,, \quad (\mathbf{v}_0 = \mathbf{0}) \,, \tag{3.71}$$

where $\mathbf{v}_i$ is the velocity of link i, $\boldsymbol{\Phi}_i$ is the motion matrix of joint i, and $\dot{\boldsymbol{q}}_i$ is the vector of joint velocity variables for joint i.

The equivalent formula for accelerations is obtained by differentiating (3.71), giving

$$\mathbf{a}_i = \mathbf{a}_{p(i)} + \boldsymbol{\Phi}_i \ddot{\boldsymbol{q}}_i + \dot{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i \,, \quad (\mathbf{a}_0 = \mathbf{0}) \,, \tag{3.72}$$

where $\mathbf{a}_i$ is the acceleration of link i, and $\ddot{\boldsymbol{q}}_i$ is the vector of joint acceleration variables.

The effect of a uniform gravitational field on the mechanism can be simulated by initializing $\mathbf{a}_0$ to $-\mathbf{a}_g$ instead of zero, where $\mathbf{a}_g$ is the gravitational acceleration vector. In this case, $\mathbf{a}_i$ is not the true acceleration of link i, but the sum of its true acceleration and $-\mathbf{a}_g$.

Step 2

Calculate the equation of motion for each link. This step computes the forces required to cause the accelerations calculated in step 1. The equation of motion for link i is

$$\mathbf{f}_i^{\mathrm{a}} = \boldsymbol{I}_i \mathbf{a}_i + \mathbf{v}_i \times \boldsymbol{I}_i \mathbf{v}_i \,, \tag{3.73}$$

where $\boldsymbol{I}_i$ is the spatial inertia of link i, and $\mathbf{f}_i^{\mathrm{a}}$ is the net force acting on link i.

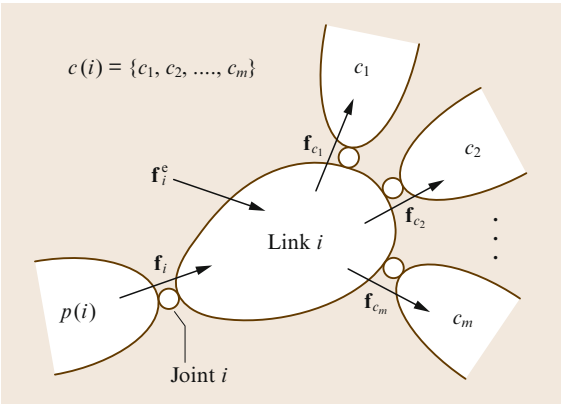


Fig. 3.6 Forces acting on link i

Step 3

Calculate the spatial force across each joint. Referring to Fig. 3.6, the net force acting on link i is

$$\mathbf{f}_i^{\mathrm{a}} = \mathbf{f}_i^{\mathrm{e}} + \mathbf{f}_i - \sum_{j \in c(i)} \mathbf{f}_j \,,$$

where $\mathbf{f}_i$ is the force transmitted across joint i, $\mathbf{f}_i^{\mathrm{e}}$ is the sum of all relevant external forces acting on link i, and $c(i)$ is the set of children of link i. Rearranging this equation gives the following recursive formula for calculating the joint forces

$$\mathbf{f}_i = \mathbf{f}_i^{\mathrm{a}} - \mathbf{f}_i^{\mathrm{e}} + \sum_{j \in c(i)} \mathbf{f}_j \,, \tag{3.74}$$

where i iterates from N_B to 1.

$\mathbf{f}_i^{\mathrm{e}}$ may include contributions from springs, dampers, force fields, contact with the environment, and so on, but its value is assumed to be known, or at least to be calculable from known quantities.

If gravity has not been simulated by a fictitious base acceleration, then the gravitational force acting on link i must be included in $\mathbf{f}_i^{\mathrm{e}}$.

Step 4

Calculate the joint force variables, $\boldsymbol{\tau}_i$. By definition, they are given by the equation

$$\boldsymbol{\tau}_i = \boldsymbol{\Phi}_i^{\mathrm{T}} \mathbf{f}_i \,. \tag{3.75}$$

Coordinate-Free Algorithm

Equations (3.71)–(3.75) imply the algorithm shown in Algorithm 3.1, which is the coordinate-free version of the RNEA. This is the simplest form of the algorithm, and it is suitable for mathematical analysis and related purposes. However, it is not suitable for numerical

computation because a numerical version must use coordinate vectors.

Algorithm 3.1 Coordinate-free recursive Newton–Euler algorithm (RNEA) for inverse dynamics

$\mathbf{v}_0 = \mathbf{0}$
$\mathbf{a}_0 = -\mathbf{a}_g$
for $i = 1$ **to** N_B **do**
  $\mathbf{v}_i = \mathbf{v}_{p(i)} + \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i$
  $\mathbf{a}_i = \mathbf{a}_{p(i)} + \boldsymbol{\Phi}_i \ddot{\boldsymbol{q}}_i + \dot{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i$
  $\mathbf{f}_i = \boldsymbol{I}_i \mathbf{a}_i + \mathbf{v}_i \times \boldsymbol{I}_i \mathbf{v}_i - \mathbf{f}_i^e$
end for
for $i = N_B$ **to** 1 **do**
  $\boldsymbol{\tau}_i = \boldsymbol{\Phi}_i^T \mathbf{f}_i$
  if $p(i) \neq 0$ **then**
    $\mathbf{f}_{p(i)} = \mathbf{f}_{p(i)} + \mathbf{f}_i$
  end if
end for

Link-Coordinates Algorithm

In general, we say that an algorithm is implemented in link coordinates if a coordinate system is defined for each link, and the calculations pertaining to link i are performed in the coordinate system associated with link i. The alternative is to implement the algorithm in absolute coordinates, in which case all calculations are performed in a single coordinate system, typically that of the base link. In practice, the RNEA is more computationally efficient when implemented in link coordinates, and the same is true of most other dynamics algorithms.

To convert the RNEA to link coordinates, we first examine the equations to see which ones involve quantities from more than one link. Equations (3.73) and (3.75) each involve quantities pertaining to link i only, and therefore need no modification. Such equations are said to be local to link i. The remaining equations involve quantities from more than one link, and therefore require the insertion of coordinate transformation matrices. The modified versions of (3.71), (3.72), and (3.74) are

$$\mathbf{v}_i = {}^i\boldsymbol{X}_{p(i)}\, \mathbf{v}_{p(i)} + \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i \,, \tag{3.76}$$

$$\mathbf{a}_i = {}^i\boldsymbol{X}_{p(i)}\, \mathbf{a}_{p(i)} + \boldsymbol{\Phi}_i \ddot{\boldsymbol{q}}_i + \dot{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i \,, \tag{3.77}$$

and

$$\mathbf{f}_i = \mathbf{f}_i^a - {}^i\boldsymbol{X}_0^F\, {}^0\mathbf{f}_i^e + \sum_{j \in c(i)} {}^i\boldsymbol{X}_j^F\, \mathbf{f}_j \,. \tag{3.78}$$

Equation (3.78) assumes that external forces are expressed in absolute (i. e., link 0) coordinates.

Algorithm 3.2 Recursive Newton–Euler algorithm using spatial vectors

inputs: $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$, *model* , ${}^0\mathbf{f}_i^e$
output: $\boldsymbol{\tau}$
model data : N_B, jtype(i), $p(i)$, $\boldsymbol{X}_L(i)$, $\boldsymbol{I}_i$

$\mathbf{v}_0 = \mathbf{0}$
$\mathbf{a}_0 = -\mathbf{a}_g$
for $i = 1$ **to** N_B **do**
  $\boldsymbol{X}_J(i) = \text{xjcalc}(\text{jtype}(i), \boldsymbol{q}_i)$
  ${}^i\boldsymbol{X}_{p(i)} = \boldsymbol{X}_J(i)\, \boldsymbol{X}_L(i)$
  if $p(i) \neq 0$ **then**
    ${}^i\boldsymbol{X}_0 = {}^i\boldsymbol{X}_{p(i)}\, {}^{p(i)}\boldsymbol{X}_0$
  end if
  $\boldsymbol{\Phi}_i = \text{pcalc}(\text{jtype}(i), \boldsymbol{q}_i)$
  $\mathring{\boldsymbol{\Phi}}_i = \text{pdcalc}(\text{jtype}(i), \boldsymbol{q}_i, \dot{\boldsymbol{q}}_i)$
  $\mathbf{v}_i = {}^i\boldsymbol{X}_{p(i)}\, \mathbf{v}_{p(i)} + \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i$
  $\boldsymbol{\zeta}_i = \mathring{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i + \mathbf{v}_i \times \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i$
  $\mathbf{a}_i = {}^i\boldsymbol{X}_{p(i)}\, \mathbf{a}_{p(i)} + \boldsymbol{\Phi}_i \ddot{\boldsymbol{q}}_i + \boldsymbol{\zeta}_i$
  $\mathbf{f}_i = \boldsymbol{I}_i \mathbf{a}_i + \mathbf{v}_i \times \boldsymbol{I}_i \mathbf{v}_i - {}^i\boldsymbol{X}_0^{-T}\, {}^0\mathbf{f}_i^e$
end for
for $i = N_B$ **to** 1 **do**
  $\boldsymbol{\tau}_i = \boldsymbol{\Phi}_i^T \mathbf{f}_i$
  if $p(i) \neq 0$ **then**
    $\mathbf{f}_{p(i)} = \mathbf{f}_{p(i)} + {}^i\boldsymbol{X}_{p(i)}^T\, \mathbf{f}_i$
  end if
end for

The complete algorithm is shown in Algorithm 3.2. The function `jtype` returns the type code for joint i; the function `xjcalc` calculates the joint transformation matrix for the specified type of joint; and the functions `pcalc` and `pdcalc` calculate $\boldsymbol{\Phi}_i$ and $\mathring{\boldsymbol{\Phi}}_i$. The formulae used by these functions for a variety of joint types can be found in Tables 2.5 and 2.6, bearing in mind that the rotation matrices that must be used are the transposes of those listed in these tables. In the general case, both `pcalc` and `pdcalc` are needed. However, for most common joint types, $\boldsymbol{\Phi}_i$ is a known constant in link coordinates, and $\mathring{\boldsymbol{\Phi}}_i$ is therefore zero. If it is known in advance that all joints will have this property, then the algorithm can be simplified accordingly. The quantities $\boldsymbol{I}_i$ and $\boldsymbol{X}_L(i)$ are known constants in link coordinates, and are part of the data structure describing the robot mechanism.

The last assignment in the first loop initializes each $\mathbf{f}_i$ to the expression $\mathbf{f}_i^a - {}^i\boldsymbol{X}_0^F\, {}^0\mathbf{f}_i^e$ (using the identity ${}^i\boldsymbol{X}_0^F = {}^i\boldsymbol{X}_0^{-T}$). The summation on the right-hand side of (3.78) is then performed in the second loop. This algorithm includes code to calculate ${}^i\boldsymbol{X}_0$, which is used to transform the external forces to link coordinates. If there are no external forces, then this code can be omitted. If there is only a single external force (e.g., a force

at the end-effector of a robot arm) then this code can be replaced with code that transforms the external force vector successively from one link coordinate system to the next, using ${}^{i}\boldsymbol{X}_{p(i)}$.

Note: although the phrase *link coordinates* suggests that we are using moving coordinate frames, the algorithm is in fact implemented in stationary coordinates that happen to coincide with the moving coordinates at the current instant.

3-D Vector RNEA

The original version of the RNEA was developed and expressed using 3-D vectors (e.g. [3.2, 4]). Algorithm 3.3 shows a special case of this algorithm, in which the joints are assumed to be revolute, and the joint axes are assumed to coincide with the z axes of the link coordinate systems. (Without these assumptions, the equations would be a lot longer.) It also assumes that the external forces are zero.

Algorithm 3.3 Recursive Newton–Euler algorithm in 3-D vectors, for revolute joints only

inputs: $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$, *model*
output: $\boldsymbol{\tau}$
model data : N_B, $p(i)$, $\boldsymbol{R}_L(i)$, ${}^{p(i)}\boldsymbol{p}_i$, m_i, $\boldsymbol{c}_i$, $\bar{\boldsymbol{I}}_i^{\mathrm{cm}}$

$\boldsymbol{\omega}_0 = \mathbf{0}$
$\dot{\boldsymbol{\omega}}_0 = \mathbf{0}$
$\dot{\boldsymbol{v}}'_0 = -\dot{\boldsymbol{v}}'_{\mathrm{g}}$
for $i = 1$ **to** N_B **do**
  ${}^{i}\boldsymbol{R}_{p(i)} = \mathrm{rotz}(\boldsymbol{q}_i)\,\boldsymbol{R}_L(i)$
  $\boldsymbol{\omega}_i = {}^{i}\boldsymbol{R}_{p(i)}\,\boldsymbol{\omega}_{p(i)} + \hat{\boldsymbol{z}}_i\dot{q}_i$
  $\dot{\boldsymbol{\omega}}_i = {}^{i}\boldsymbol{R}_{p(i)}\,\dot{\boldsymbol{\omega}}_{p(i)} + \left({}^{i}\boldsymbol{R}_{p(i)}\,\boldsymbol{\omega}_{p(i)}\right) \times \hat{\boldsymbol{z}}_i\dot{q}_i + \hat{\boldsymbol{z}}_i\ddot{q}_i$
  $\dot{\boldsymbol{v}}'_i = {}^{i}\boldsymbol{R}_{p(i)}\left(\dot{\boldsymbol{v}}'_{p(i)} + \dot{\boldsymbol{\omega}}_{p(i)} \times {}^{p(i)}\boldsymbol{p}_i + \boldsymbol{\omega}_{p(i)} \times \boldsymbol{\omega}_{p(i)} \times {}^{p(i)}\boldsymbol{p}_i\right)$
  $\boldsymbol{f}_i = m_i(\dot{\boldsymbol{v}}'_i + \dot{\boldsymbol{\omega}}_i \times \boldsymbol{c}_i + \boldsymbol{\omega}_i \times \boldsymbol{\omega}_i \times \boldsymbol{c}_i)$
  $\boldsymbol{n}_i = \bar{\boldsymbol{I}}_i^{\mathrm{cm}}\dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \bar{\boldsymbol{I}}_i^{\mathrm{cm}}\boldsymbol{\omega}_i + \boldsymbol{c}_i \times \boldsymbol{f}_i$
end for
for $i = N_B$ **to** 1 **do**
  $\tau_i = \hat{\boldsymbol{z}}_i^{\mathrm{T}}\boldsymbol{n}_i$
  if $p(i) \neq 0$ **then**
    $\boldsymbol{f}_{p(i)} = \boldsymbol{f}_{p(i)} + {}^{i}\boldsymbol{R}_{p(i)}^{\mathrm{T}}\boldsymbol{f}_i$
    $\boldsymbol{n}_{p(i)} = \boldsymbol{n}_{p(i)} + {}^{i}\boldsymbol{R}_{p(i)}^{\mathrm{T}}\boldsymbol{n}_i + {}^{p(i)}\boldsymbol{p}_i \times {}^{i}\boldsymbol{R}_{p(i)}^{\mathrm{T}}\boldsymbol{f}_i$
  end if
end for

In this algorithm, $\dot{\boldsymbol{v}}'_{\mathrm{g}}$ is the linear acceleration due to gravity, expressed in base (link 0) coordinates; `rotz` computes the transpose of the matrix shown in (2.2); $\boldsymbol{R}_L(i)$ is the rotational component of $\boldsymbol{X}_L(i)$; ${}^{i}\boldsymbol{R}_{p(i)}$ is the rotational component of ${}^{i}\boldsymbol{X}_{p(i)}$; `pcalc` and `pdcalc` are not used because $\boldsymbol{\Phi}_i$ is the known constant $(\hat{\boldsymbol{z}}^{\mathrm{T}}\ \mathbf{0}^{\mathrm{T}})^{\mathrm{T}}$; $\dot{\boldsymbol{v}}'_i$ is the linear acceleration of the origin of link i coordinates (O_i), and is the linear component of the classical acceleration of link i; ${}^{p(i)}\boldsymbol{p}_i$ is the position of O_i relative to $O_{p(i)}$ expressed in $p(i)$ coordinates; and m_i, $\boldsymbol{c}_i$, and $\bar{\boldsymbol{I}}_i^{\mathrm{cm}}$ are the inertia parameters of link i. (See Table 3.1 for the equations relating these 3-D quantities to the corresponding spatial quantities.)

At first sight, the 3-D vector algorithm looks significantly different from the spatial vector algorithm. Nevertheless, it can be obtained directly from the spatial vector algorithm simply by expanding the spatial vectors to their 3-D components, restricting the joint type to revolute, converting spatial accelerations to classical accelerations (i. e., replacing each instance of $\dot{\boldsymbol{v}}_i$ with $\dot{\boldsymbol{v}}'_i - \boldsymbol{\omega}_i \times \boldsymbol{v}_i$ as per (3.22)), and applying some 3-D vector identities to bring the equations into the form shown in the table. The conversion from spatial to classical acceleration has one interesting side-effect: $\boldsymbol{v}_i$ cancels out of the equation of motion, and therefore does not need to be calculated. As a result, the 3-D version of the algorithm has a slight speed advantage over the spatial version.

3.5.2 The Articulated-Body Algorithm

The ABA is an $O(N_B)$ algorithm for calculating the forward dynamics of a kinematic tree. However, under normal circumstances, $O(N_B) = O(n)$, so we shall refer to it as an $O(n)$ algorithm. The ABA was developed by *Featherstone* [3.1] and is an example of a constraint-propagation algorithm. Given the joint position, velocity, and applied torque/force variables, this algorithm calculates the joint accelerations. With the joint accelerations determined, numerical integration may be used to provide a simulation of the mechanism.

The key concept in the ABA is illustrated in Fig. 3.7. The subtree rooted at link i interacts with the rest of the kinematic tree only through a force $\mathbf{f}_i$ that is transmitted across joint i. Suppose we break the tree at this point, and consider only the motion of the subtree subject to an unknown force, $\mathbf{f}_i$, acting on link i. It is possible to show that the acceleration of link i is related to the applied force according to the equation

$$\mathbf{f}_i = \boldsymbol{I}_i^A\mathbf{a}_i + \mathbf{p}_i^A\ , \tag{3.79}$$

where $\boldsymbol{I}_i^A$ is called the articulated-body inertia of link i in the subtree (which we can now call an articulated body), and $\mathbf{p}_i^A$ is the associated bias force, which is the force required to produce zero acceleration in link i. Note that $\mathbf{p}_i^A$ depends on the velocities of the individual bodies in the articulated body. Equation (3.79) takes into account the complete dynamics of the subtree. Thus, if we happened to know the correct value of $\mathbf{f}_i$, then (3.79)

would immediately give us the correct acceleration of link i.

The reason we are interested in the quantities $\boldsymbol{I}_i^A$ and $\mathbf{p}_i^A$ is that they allow us to calculate $\ddot{\boldsymbol{q}}_i$ from $\mathbf{a}_{p(i)}$, which in turn allows us to calculate $\mathbf{a}_i$, which then allows us to calculate more joint accelerations, and so on. Combining (3.79) with (3.75) and (3.72) gives

$$\boldsymbol{\tau}_i = \boldsymbol{\Phi}_i^{\mathrm{T}} \mathbf{f}_i = \boldsymbol{\Phi}_i^{\mathrm{T}} \left(\boldsymbol{I}_i^A (\mathbf{a}_{p(i)} + \boldsymbol{\Phi}_i \ddot{\boldsymbol{q}}_i + \dot{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i) + \mathbf{p}_i^A \right) ,$$

which can be solved for $\ddot{\boldsymbol{q}}_i$ to give

$$\ddot{\boldsymbol{q}}_i = \boldsymbol{D}_i \left(\boldsymbol{u}_i - \boldsymbol{U}_i^{\mathrm{T}} \mathbf{a}_{p(i)} \right) , \tag{3.80}$$

where

$$\begin{aligned} \boldsymbol{U}_i &= \boldsymbol{I}_i^A \boldsymbol{\Phi}_i , \\ \boldsymbol{D}_i &= \left(\boldsymbol{\Phi}_i^{\mathrm{T}} \boldsymbol{U}_i \right)^{-1} = \left(\boldsymbol{\Phi}_i^{\mathrm{T}} \boldsymbol{I}_i^A \boldsymbol{\Phi}_i \right)^{-1} , \\ \boldsymbol{u}_i &= \boldsymbol{\tau}_i - \boldsymbol{U}_i^{\mathrm{T}} \boldsymbol{\zeta}_i - \boldsymbol{\Phi}_i^{\mathrm{T}} \mathbf{p}_i^A \end{aligned}$$

and

$$\boldsymbol{\zeta}_i = \dot{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i = \mathring{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i + \mathbf{v}_i \times \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i .$$

$\mathbf{a}_i$ can then be calculated via (3.72).

It turns out that the articulated-body inertias and bias forces can be calculated efficiently via the recursive formulae

$$\boldsymbol{I}_i^A = \boldsymbol{I}_i + \sum_{j \in c(i)} \left(\boldsymbol{I}_j^A - \boldsymbol{U}_j \boldsymbol{D}_j \boldsymbol{U}_j^{\mathrm{T}} \right) \tag{3.81}$$

and

$$\mathbf{p}_i^A = \mathbf{p}_i + \sum_{j \in c(i)} \left(\mathbf{p}_j^A + \boldsymbol{I}_j^A \boldsymbol{\zeta}_j + \boldsymbol{U}_j \boldsymbol{D}_j \boldsymbol{u}_j \right) , \tag{3.82}$$

where

$$\mathbf{p}_i = \mathbf{v}_i \times \boldsymbol{I}_i \mathbf{v}_i - \mathbf{f}_i^{\mathrm{e}} .$$

These formulae are obtained by examining the relationship between $\mathbf{f}_i$ and $\mathbf{a}_i$ in Fig. 3.7 and assuming that $\boldsymbol{I}_j^A$ and $\mathbf{p}_j^A$ are already known for every $j \in c(i)$. For more details see [3.1, 8, 15, 25].

The complete algorithm is shown in Algorithm 3.4.

Algorithm 3.4 Articulated-body algorithm for forward dynamics

inputs: $\mathbf{q}$, $\dot{\mathbf{q}}$, $\boldsymbol{\tau}$, *model*, ${}^0\mathbf{f}_i^{\mathrm{e}}$
output: $\ddot{\mathbf{q}}$
model data: N_B, jtype(i), $p(i)$, $\boldsymbol{X}_L(i)$, $\boldsymbol{I}_i$

$\mathbf{v}_0 = \mathbf{0}$
$\mathbf{a}_0 = -\mathbf{a}_{\mathrm{g}}$
for $i = 1$ **to** N_B **do**
  $\boldsymbol{X}_J(i) = \mathrm{xjcalc}(\mathrm{jtype}(i), \boldsymbol{q}_i)$
  ${}^i\boldsymbol{X}_{p(i)} = \boldsymbol{X}_J(i)\, \boldsymbol{X}_L(i)$
  if $p(i) \neq 0$ **then**
    ${}^i\boldsymbol{X}_0 = {}^i\boldsymbol{X}_{p(i)}\, {}^{p(i)}\boldsymbol{X}_0$
  end if
  $\boldsymbol{\Phi}_i = \mathrm{pcalc}(\mathrm{jtype}(i), \boldsymbol{q}_i)$
  $\mathring{\boldsymbol{\Phi}}_i = \mathrm{pdcalc}(\mathrm{jtype}(i), \boldsymbol{q}_i, \dot{\boldsymbol{q}}_i)$
  $\mathbf{v}_i = {}^i\boldsymbol{X}_{p(i)}\, \mathbf{v}_{p(i)} + \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i$
  $\boldsymbol{\zeta}_i = \mathring{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i + \mathbf{v}_i \times \boldsymbol{\Phi}_i \dot{\boldsymbol{q}}_i$
  $\boldsymbol{I}_i^A = \boldsymbol{I}_i$
  $\mathbf{p}_i^A = \mathbf{v}_i \times \boldsymbol{I}_i \mathbf{v}_i - {}^i\boldsymbol{X}_0^{-T}\, {}^0\mathbf{f}_i^{\mathrm{e}}$
end for
for $i = N_B$ **to** 1 **do**
  $\boldsymbol{U}_i = \boldsymbol{I}_i^A \boldsymbol{\Phi}_i$
  $\boldsymbol{D}_i = \left(\boldsymbol{\Phi}_i^{\mathrm{T}} \boldsymbol{U}_i \right)^{-1}$
  $\boldsymbol{u}_i = \boldsymbol{\tau}_i - \boldsymbol{U}_i^{\mathrm{T}} \boldsymbol{\zeta}_i - \boldsymbol{\Phi}_i^{\mathrm{T}} \mathbf{p}_i^A$
  if $p(i) \neq 0$ **then**
    $\boldsymbol{I}_{p(i)}^A = \boldsymbol{I}_{p(i)}^A + {}^i\boldsymbol{X}_{p(i)}^{\mathrm{T}} \left(\boldsymbol{I}_i^A - \boldsymbol{U}_i \boldsymbol{D}_i \boldsymbol{U}_i^{\mathrm{T}} \right) {}^i\boldsymbol{X}_{p(i)}$
    $\mathbf{p}_{p(i)}^A = \mathbf{p}_{p(i)}^A + {}^i\boldsymbol{X}_{p(i)}^{\mathrm{T}} \left(\mathbf{p}_i^A + \boldsymbol{I}_i^A \boldsymbol{\zeta}_i + \boldsymbol{U}_i \boldsymbol{D}_i \boldsymbol{u}_i \right)$
  end if
end for
for $i = 1$ **to** N_B **do**
  $\mathbf{a}_i = {}^i\boldsymbol{X}_{p(i)}\, \mathbf{a}_{p(i)}$
  $\ddot{\boldsymbol{q}}_i = \boldsymbol{D}_i \left(\boldsymbol{u}_i - \boldsymbol{U}_i^{\mathrm{T}} \mathbf{a}_i \right)$
  $\mathbf{a}_i = \mathbf{a}_i + \boldsymbol{\Phi}_i \ddot{\boldsymbol{q}}_i + \boldsymbol{\zeta}_i$
end for

It is expressed in link coordinates, as per the RNEA in Algorithm 3.2. It makes a total of three passes through the kinematic tree. The first pass iterates from the base out to the tips; it calculates the link velocities using (3.76), the velocity-product term $\boldsymbol{\zeta}_i = \dot{\boldsymbol{\Phi}}_i \dot{\boldsymbol{q}}_i$, and it initializes the variables $\boldsymbol{I}_i^A$ and $\mathbf{p}_i^A$ to the values $\boldsymbol{I}_i$ and $\mathbf{p}_i (= \mathbf{v}_i \times \boldsymbol{I}_i \mathbf{v}_i - {}^i\boldsymbol{X}_0^F\, {}^0\mathbf{f}_i^{\mathrm{e}})$, respectively. The second pass iterates from the tips back to the base; it calculates the articulated-body inertia and bias force for each link using (3.81) and (3.82). The third pass iterates from the base to the tips; it calculates the link and joint accelerations using (3.80) and (3.77).

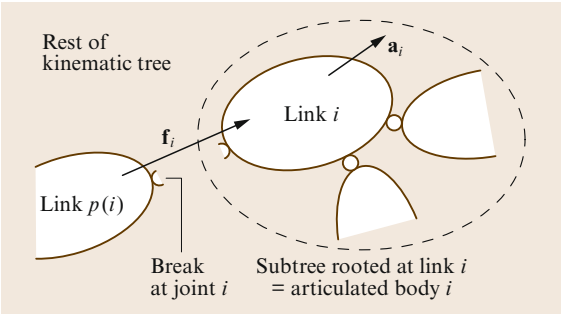


Fig. 3.7 Definition of articulated body i

3.5.3 The Composite-Rigid-Body Algorithm

The CRBA is an algorithm for calculating the joint-space inertia matrix (JSIM) of a kinematic tree. The most common use for the CRBA is as part of a forward dynamics algorithm. It first appeared as method 3 in [3.5].

Forward dynamics, in joint space, is the task of calculating $\ddot{\mathbf{q}}$ from $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\boldsymbol{\tau}$. Starting from (3.37), the most obvious way to proceed is to calculate $\boldsymbol{H}$ and $\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}$, and then solve the linear equation

$$\boldsymbol{H}\ddot{\mathbf{q}} = \boldsymbol{\tau} - (\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}) \tag{3.83}$$

for $\ddot{\mathbf{q}}$. If the mechanism is a kinematic tree, then $\boldsymbol{H}$ and $\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}$ can be computed in $O(n^2)$ and $O(n)$ operations, respectively, and (3.83) can be solved in $O(n^3)$ operations. Algorithms that take this approach are therefore known collectively as $O(n^3)$ algorithms. However, this figure of $O(n^3)$ should be regarded as the worst-case complexity, since the actual complexity depends on the amount of branching in the tree [3.26]. Furthermore, even in the worst case, the n^3 term has a small coefficient, and does not dominate until approximately $n = 60$.

$\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}$ can be calculated using an inverse dynamics algorithm. If $ID(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ is the result of an inverse dynamics calculation with arguments $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, then

$$ID(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) = \boldsymbol{\tau} = \boldsymbol{H}\ddot{\mathbf{q}} + \boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}} ,$$

so

$$\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}} = ID(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{0}) . \tag{3.84}$$

Thus, the value of $\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}$ for a kinematic tree can be calculated efficiently using the RNEA with $\ddot{\mathbf{q}} = \mathbf{0}$.

The key concept in the CRBA is to note that the JSIM only depends on the joint positions, and not their rates. The CRBA makes the simplifying assumption that the rate at each joint is zero. By also assuming that gravity is zero, $\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}$ is eliminated from (3.83). Furthermore, for a revolute joint, a unit of joint acceleration applied at the j-th joint produces the j-th column of the JSIM. This partitions the mechanism into two composite rigid bodies connected by the j-th joint, and simplifies the dynamics considerably. This concept has been generalized so that the CRBA may be applied to any joint type within a kinematic tree structure.

It can be shown that the general form of the JSIM for a kinematic tree is

$$\boldsymbol{H}_{ij} = \begin{cases} \boldsymbol{\Phi}_i^{\mathrm{T}} \boldsymbol{I}_i^{\mathrm{C}} \boldsymbol{\Phi}_j & \text{if } i \in c^*(j) \\ \boldsymbol{\Phi}_i^{\mathrm{T}} \boldsymbol{I}_j^{\mathrm{C}} \boldsymbol{\Phi}_j & \text{if } j \in c^*(i) \\ \mathbf{0} & \text{otherwise,} \end{cases} \tag{3.85}$$

where $c^*(i)$ is the set of links in the subtree rooted at link i, including link i itself, and

$$\boldsymbol{I}_i^{\mathrm{C}} = \sum_{j \in c^*(i)} \boldsymbol{I}_j . \tag{3.86}$$

See [3.8, 15]. In fact, $\boldsymbol{I}_i^{\mathrm{C}}$ is the inertia of the composite rigid body formed by the rigid assembly of all the links in $c^*(i)$, and this is where the algorithm gets its name.

Equations (3.85) and (3.86) are the basis of the algorithm shown in Algorithm 3.5, which is the CRBA in link coordinates.

Algorithm 3.5 Composite-rigid-body algorithm for calculating the JSIM

inputs: *model* , *RNEA partial results*
output: $\boldsymbol{H}$
model data : $N_B, p(i), \boldsymbol{I}_i$
RNEA data : $\boldsymbol{\Phi}_i, {}^i\boldsymbol{X}_{p(i)}$

$\boldsymbol{H} = \mathbf{0}$
for $i = 1$ **to** N_B **do**
  $\boldsymbol{I}_i^{\mathrm{C}} = \boldsymbol{I}_i$
end for
for $i = N_B$ **to** 1 **do**
  $\boldsymbol{F} = \boldsymbol{I}_i^{\mathrm{C}} \boldsymbol{\Phi}_i$
  $\boldsymbol{H}_{ii} = \boldsymbol{\Phi}_i^{\mathrm{T}} \boldsymbol{F}$
  if $p(i) \neq 0$ **then**
    $\boldsymbol{I}_{p(i)}^{\mathrm{C}} = \boldsymbol{I}_{p(i)}^{\mathrm{C}} + {}^i\boldsymbol{X}_{p(i)}^{\mathrm{T}} \boldsymbol{I}_i^{\mathrm{C}} {}^i\boldsymbol{X}_{p(i)}$
  end if
  $j = i$
  while $p(j) \neq 0$ **do**
    $\boldsymbol{F} = {}^j\boldsymbol{X}_{p(j)}^{\mathrm{T}} \boldsymbol{F}$
    $j = p(j)$
    $\boldsymbol{H}_{ij} = \boldsymbol{F}^{\mathrm{T}} \boldsymbol{\Phi}_j$
    $\boldsymbol{H}_{ji} = \boldsymbol{H}_{ij}^{\mathrm{T}}$
  end while
end for

This algorithm assumes that the matrices ${}^i\boldsymbol{X}_{p(i)}$ and $\boldsymbol{\Phi}_i$ have already been calculated, e.g., during the calculation of $\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}$. If this is not the case, then the relevant lines from Algorithm 3.2 can be inserted into the first loop. The matrix $\boldsymbol{F}$ is a local variable. The first step, $\boldsymbol{H} = \mathbf{0}$, can be omitted if there are no branches in the tree.

Having calculated $\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}$ and $\boldsymbol{H}$, the final step is to solve (3.83) for $\ddot{\mathbf{q}}$. This can be done using a standard Cholesky or $\boldsymbol{LDL}^{\mathrm{T}}$ factorization. Note that $\boldsymbol{H}$ can be highly ill-conditioned [3.27], reflecting an underlying ill-conditioning of the kinematic tree itself, so it is recommended to use double-precision arithmetic for every step in the forward dynamics calculation. (This advice applies also to the ABA.)

Exploiting Sparsity

Equation (3.85) implies that some elements of $\boldsymbol{H}$ will automatically be zero if there are branches in the kinematic tree. An example of this effect is shown in Fig. 3.8. Observe that nearly half of the elements are zero. It is possible to exploit this sparsity using the factorization algorithms described in [3.26]. Depending on the amount of branching in the tree, the sparse algorithms can run many times faster than the standard algorithms.

3.5.4 Operational-Space Inertia Matrix

Two different algorithms are presented to calculate the OSIM. The first is an $O(n^3)$ algorithm that uses the basic definition of the OSIM along with efficient factorization of the JSIM. The second is an $O(n)$ algorithm that is based on efficient solution of the forward dynamics problem.

Algorithm Using Basic Definition

If a robot has a relatively small number of freedoms (e.g., six), then the most efficient method for calculating the OSIM is via (3.52). The procedure is as follows:

1. Calculate $\boldsymbol{H}$ via the CRBA.
2. Factorize $\boldsymbol{H}$ into $\boldsymbol{H} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$ (Cholesky factorization).
3. Use back-substitution to calculate $\boldsymbol{Y} = \boldsymbol{L}^{-1}\boldsymbol{J}^{\mathrm{T}}$.
4. $\boldsymbol{\Lambda}^{-1} = \boldsymbol{Y}^{\mathrm{T}}\boldsymbol{Y}$.
5. Factorize $\boldsymbol{\Lambda}^{-1}$ (optional).

The final step is only possible if the end-effector has a full six DOFs, and is only necessary if the application requires $\boldsymbol{\Lambda}$ rather than $\boldsymbol{\Lambda}^{-1}$. In the second step, an $\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}$ factorization can be used instead of Cholesky, or one can use one of the efficient factorizations described in [3.26] for branched kinematic trees.

The other terms in (3.38) can be calculated via (3.53) and (3.54). In particular, (3.38) can be rewritten in the form

$$\dot{\mathbf{v}} + \boldsymbol{\Lambda}^{-1}(\boldsymbol{x})[\boldsymbol{\mu}(\boldsymbol{x}, \mathbf{v}) + \boldsymbol{\rho}(\boldsymbol{x})] = \boldsymbol{\Lambda}^{-1}(\boldsymbol{x})\,\mathbf{f}\,, \qquad (3.87)$$

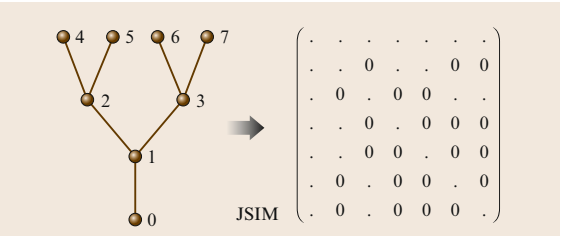


Fig. 3.8 Branch-induced sparsity: branches in the kinematic tree cause certain elements in the JSIM to be zero

and the quantity $\boldsymbol{\Lambda}^{-1}(\boldsymbol{\mu} + \boldsymbol{\rho})$ can be calculated from the formula

$$\boldsymbol{\Lambda}^{-1}(\boldsymbol{\mu} + \boldsymbol{\rho}) = \boldsymbol{J}\boldsymbol{H}^{-1}(\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}) - \dot{\boldsymbol{J}}\dot{\mathbf{q}}\,. \qquad (3.88)$$

The term $\dot{\boldsymbol{J}}\dot{\mathbf{q}}$ is the velocity-product acceleration of the end-effector (3.50). It is calculated as a by-product of calculating $\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}$ via the RNEA (3.84). Specifically, $\dot{\boldsymbol{J}}\dot{\mathbf{q}} = \mathbf{a}_{\mathrm{ee}} - \mathbf{a}_0$, where $\mathbf{a}_{\mathrm{ee}}$ is the calculated acceleration of the end-effector (expressed in the same coordinates as $\dot{\mathbf{v}}$) and $\mathbf{a}_0$ is the acceleration of the base ($-\mathbf{a}_{\mathrm{g}}$).

$O(n)$ Algorithm

For a sufficiently large value of n, it becomes more efficient to use an $O(n)$ algorithm. Several such algorithms can be found in [3.28–30]. In this section, a more straightforward algorithm is given, which is based on an $O(n)$ calculation of the joint-space forward dynamics problem, e.g., via the ABA. It is a variation of the unit force method [3.29] and computes the inverse of the OSIM.

Starting with (3.87), observe that $\boldsymbol{\Lambda}^{-1}$ is a function of position only, and certain terms in the dynamic equations can be neglected without affecting its value. Specifically, if the joint rates, $\dot{\mathbf{q}}$, joint forces, $\boldsymbol{\tau}$, and gravitational forces are all set to zero, the value of $\boldsymbol{\Lambda}$ will remain unchanged. Under these conditions,

$$\dot{\mathbf{v}} = \boldsymbol{\Lambda}^{-1}\mathbf{f}\,. \qquad (3.89)$$

Let us define $\hat{\mathbf{e}}_i$ to be a 6-D coordinate vector with a 1 in the i-th coordinate and zeros elsewhere. If we set $\mathbf{f} = \hat{\mathbf{e}}_i$ in (3.89), then $\dot{\mathbf{v}}$ will equal column i of $\boldsymbol{\Lambda}^{-1}$. Let us also define the function $FD(i, j, \mathbf{q}, \dot{\mathbf{q}}, \mathbf{a}_0, \boldsymbol{\tau}, \mathbf{f})$, which performs a forward-dynamics calculation and returns the true acceleration of link i (i. e., $\mathbf{a}_i - \mathbf{a}_0$), expressed in the same coordinates as $\mathbf{f}$ (typically the base coordinates). The arguments $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\boldsymbol{\tau}$ set the values of the joint position, velocity, and force variables, while j and $\mathbf{f}$ specify that an external force of $\mathbf{f}$ is to be applied to link j. The argument $\mathbf{a}_0$ specifies a fictitious base acceleration to include gravitational effects, and is set to either $\mathbf{0}$ or $-\mathbf{a}_{\mathrm{g}}$.

With these definitions, we have

$$(\boldsymbol{\Lambda}^{-1})^i = FD(ee, ee, \mathbf{q}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \hat{\mathbf{e}}_i) \qquad (3.90)$$

and

$$\boldsymbol{\Lambda}^{-1}(\boldsymbol{\mu} + \boldsymbol{\rho}) = -FD(ee, ee, \mathbf{q}, \dot{\mathbf{q}}, -\mathbf{a}_{\mathrm{g}}, \boldsymbol{\tau}, \mathbf{0})\,, \qquad (3.91)$$

where $(\boldsymbol{\Lambda}^{-1})^i$ is column i of $\boldsymbol{\Lambda}^{-1}$, and ee is the body number of the end-effector. It therefore follows that the coefficients of (3.87) can be calculated using the algorithm shown in Algorithm 3.6. This algorithm is $O(n)$.

Algorithm 3.6 Algorithm to compute the inverse of the operational-space inertia matrix and other terms

for $j = 1$ **to** 6 **do**
  $\dot{\mathbf{v}}^j = FD(ee, ee, \mathbf{q}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \hat{\mathbf{e}}_j)$
end for
$\boldsymbol{\Lambda}^{-1} = \left[\dot{\mathbf{v}}^1\ \dot{\mathbf{v}}^2\ \cdots\ \dot{\mathbf{v}}^6\right]$
$\boldsymbol{\Lambda}^{-1}(\boldsymbol{\mu} + \boldsymbol{\rho}) = -FD(ee, ee, \mathbf{q}, \dot{\mathbf{q}}, -\mathbf{a}_g, \boldsymbol{\tau}, \mathbf{0})$

The efficiency of the algorithm may be increased significantly when computing $\boldsymbol{\Lambda}^{-1}$ by noting that: (1) $\mathbf{v}_i$, $\boldsymbol{\zeta}_i$, and $\boldsymbol{\tau}_i$ in the ABA calculation (Algorithm 3.4) may be set to zero, and (2) $\boldsymbol{I}_i^A$ and the quantities that depend upon it ($\boldsymbol{U}_i$ and $\boldsymbol{D}_i$) need only be computed once, since they do not vary with the applied force. Also, note that the algorithm may be applied to multiple end-effectors by modifying *FD* to accept a list of end-effector body numbers in its first argument, and return a composite vector containing the accelerations of all the specified bodies. Algorithm 3.6 is then enclosed in a `for` loop that controls the second argument to *FD*, and iterates over all of the end-effector body numbers [3.20].

However, for the case of branched mechanisms with multiple end-effectors, several published algorithms achieve even better efficiency, and should be used instead [3.20, 29, 31–33]. The best achievable complexity is $O(n + md + m^2)$, where m is the number of end-effectors and d is the depth of the system's connectivity tree [3.33]; but the fastest algorithm for a typical humanoid exploits branch-induced sparsity [3.32, 33].

3.6 Kinematic Loops

All of the algorithms in the last section were for kinematic trees. In this section, a final algorithm is provided for the forward dynamics of closed-loop systems. The algorithm supplements the dynamic equations of motion for a spanning tree of the closed-loop system with the loop-closure constraint equations. Three different methods are given to solve the resulting linear system of equations. An efficient algorithm is given to compute the loop-closure constraints.

Systems with closed kinematic loops exhibit more complicated dynamics than kinematic trees. For example:

1. The degree of motion freedom of a kinematic tree is fixed, but that of a closed-loop system can vary.
2. The degree of instantaneous motion freedom is always the same as the degree of finite motion freedom in a kinematic tree, but they can be different in a closed-loop system.
3. Every force in a kinematic tree can be determined, but some forces in a closed-loop system can be indeterminate. This occurs whenever a closed-loop system is overconstrained.

Two examples of these phenomena are shown in Fig. 3.9. The mechanism in Fig. 3.9a has no finite motion freedom, but it has two degrees of infinitesimal motion freedom. The mechanism in Fig. 3.9b has one degree of freedom when $\theta \neq 0$, but if $\theta = 0$ then the two arms, A and B, are able to move independently, and the mechanism has two degrees of freedom. Moreover, at the boundary between these two motion regimes, the mechanism has three degrees of infinitesimal motion freedom. Both these mechanisms are planar, and are therefore overconstrained. As a result, the out-of-plane components of the joint constraint forces are indeterminate. This kind of indeterminacy has no effect on the motions of these mechanisms, but it does complicate the calculation of their dynamics.

3.6.1 Formulation of Closed-Loop Algorithm

A closed-loop system can be modeled as a spanning tree subject to a set of loop-closure constraint forces. If

$$\boldsymbol{H}\ddot{\mathbf{q}} + \boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g = \boldsymbol{\tau}$$

is the equation of motion of the spanning tree on its own, then the equation of motion for the closed-loop system is

$$\boldsymbol{H}\ddot{\mathbf{q}} + \boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g = \boldsymbol{\tau} + \boldsymbol{\tau}^a + \boldsymbol{\tau}^c\,, \tag{3.92}$$

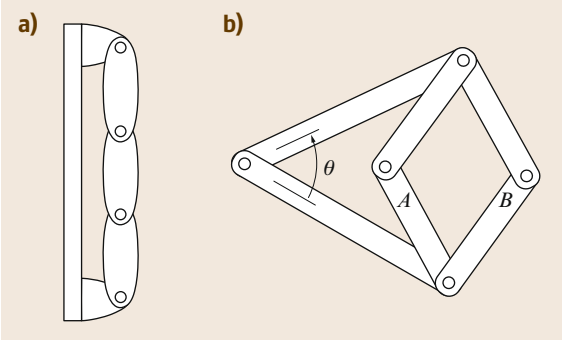


Fig.3.9a,b Pathological closed-loop systems. **(a)** Example of varying motion freedom, **(b)** Example with differing finite and infinitesimal motion freedoms

where $\boldsymbol{\tau}^{\mathrm{a}}$ and $\boldsymbol{\tau}^{\mathrm{c}}$ are vectors of loop-closure active and constraint forces, respectively, expressed in the generalized force coordinates of the spanning tree. $\boldsymbol{\tau}^{\mathrm{a}}$ is a known quantity, and $\boldsymbol{\tau}^{\mathrm{c}}$ is unknown. $\boldsymbol{\tau}^{\mathrm{a}}$ comes from the force elements acting at the loop-closing joints (springs, dampers and actuators). If there are no such force elements, then $\boldsymbol{\tau}^{\mathrm{a}} = \mathbf{0}$.

The loop-closure constraints restrict the motion of the spanning tree. At the acceleration level, these constraints can be expressed in the form of a linear equation,

$$\boldsymbol{L}\ddot{\mathbf{q}} = \mathbf{l}\,, \tag{3.93}$$

where $\boldsymbol{L}$ is an $n^{\mathrm{c}} \times n$ matrix. n^{c} is the number of constraints due to the loop-closing joints, and is given by the formula

$$n^{\mathrm{c}} = \sum_{k=N_B+1}^{N_J} n_k^{\mathrm{c}}\,, \tag{3.94}$$

where n_k^{c} is the number of constraints imposed by joint k. If $\mathrm{rank}(\boldsymbol{L}) < n^{\mathrm{c}}$ then the loop-closure constraints are linearly dependent, and the closed-loop mechanism is overconstrained. The mobility of a closed-loop system (i. e., its degree of motion freedom) is given by the formula

$$\text{mobility} = n - \mathrm{rank}(\boldsymbol{L})\,. \tag{3.95}$$

Given a constraint equation in the form of (3.93), it follows that the constraint forces can be expressed in the form

$$\boldsymbol{\tau}^{\mathrm{c}} = \boldsymbol{L}^{\mathrm{T}}\boldsymbol{\lambda}\,, \tag{3.96}$$

where $\boldsymbol{\lambda} = \left(\boldsymbol{\lambda}_{N_B+1}^{\mathrm{T}} \cdots \boldsymbol{\lambda}_{N_J}^{\mathrm{T}}\right)^{\mathrm{T}}$ is an $n^{\mathrm{c}} \times 1$ vector of unknown constraint-force variables (or Lagrange multipliers). If the mechanism is overconstrained, then $\boldsymbol{L}^{\mathrm{T}}$ will have a null space, and the component of $\boldsymbol{\lambda}$ lying in this null space will be indeterminate.

It is often possible to identify redundant constraints in advance. For example, if a kinematic loop is known to be planar, then the out-of-plane loop-closure constraints are redundant. In these circumstances, it is advantageous to remove the corresponding rows of $\boldsymbol{L}$ and elements of $\mathbf{l}$ and $\boldsymbol{\lambda}$. The removed elements of $\boldsymbol{\lambda}$ can be assigned a value of zero.

Combining (3.92), (3.93), and (3.96) produces the following equation of motion for a closed-loop system

$$\begin{pmatrix} \boldsymbol{H} & \boldsymbol{L}^{\mathrm{T}} \\ \boldsymbol{L} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \ddot{\mathbf{q}} \\ -\boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\tau} + \boldsymbol{\tau}^{\mathrm{a}} - (\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}) \\ \mathbf{l} \end{pmatrix}. \tag{3.97}$$

The system matrix is symmetric, but indefinite. If $\boldsymbol{L}$ has full rank, then the system matrix will be nonsingular, otherwise it will be singular, and one or more elements of $\boldsymbol{\lambda}$ will be indeterminate.

Equation (3.97) can be solved in any of the following ways:

1. Solve it directly for $\ddot{\mathbf{q}}$ and $\boldsymbol{\lambda}$.
2. Solve for $\boldsymbol{\lambda}$ first, and then use the result to solve for $\ddot{\mathbf{q}}$.
3. Solve (3.93) for $\ddot{\mathbf{q}}$, substitute the result into (3.92), eliminate the unknown constraint forces, and solve for the remaining unknowns.

Method 1 is the simplest, but generally also the least efficient. This method is appropriate when the system matrix is nonsingular. As the size of the system matrix is $(n + n^{\mathrm{c}}) \times (n + n^{\mathrm{c}})$, this method is $O((n + n^{\mathrm{c}})^3)$.

Method 2 is particularly useful if $n \gg n^{\mathrm{c}}$, and offers the opportunity to use $O(n)$ algorithms on the spanning tree [3.6]. From (3.97),

$$\boldsymbol{L}\boldsymbol{H}^{-1}\boldsymbol{L}^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{l} - \boldsymbol{L}\boldsymbol{H}^{-1}\left[\boldsymbol{\tau} + \boldsymbol{\tau}^{\mathrm{a}} - (\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}})\right]\,. \tag{3.98}$$

This equation can be formulated in $O(n\,(n^{\mathrm{c}})^2)$ operations via $O(n)$ algorithms, and solved in $O((n^{\mathrm{c}})^3)$. Once $\boldsymbol{\lambda}$ is known, $\boldsymbol{\tau}^{\mathrm{c}}$ can be calculated via (3.96) in $O(n\,n^{\mathrm{c}})$ operations, and (3.92) solved by an $O(n)$ algorithm; so the total complexity is $O(n\,(n^{\mathrm{c}})^2 + (n^{\mathrm{c}})^3)$. If $\boldsymbol{L}$ is rank deficient, then $\boldsymbol{L}\boldsymbol{H}^{-1}\boldsymbol{L}^{\mathrm{T}}$ will be singular; but it is still a positive-semidefinite matrix, and presents a slightly easier factorization problem than a singular instance of the indefinite system matrix in (3.97).

Method 3 is useful if $n - n^{\mathrm{c}}$ is small, or if $\boldsymbol{L}$ is expected to be rank deficient. Equation (3.93) is solved using a special version of Gaussian elimination (or similar procedure), which is equipped with a numerical rank test, and which is designed to solve underdetermined systems. The solution is an equation of the form

$$\ddot{\mathbf{q}} = \boldsymbol{K}\boldsymbol{y} + \ddot{\mathbf{q}}_0\,,$$

where $\ddot{\mathbf{q}}_0$ is any particular solution to (3.93), $\boldsymbol{K}$ is an $n \times (n - \mathrm{rank}(\boldsymbol{L}))$ matrix with the property $\boldsymbol{L}\boldsymbol{K} = \mathbf{0}$, and $\boldsymbol{y}$ is a vector of $n - \mathrm{rank}(\boldsymbol{L})$ unknowns. (Typically, $\boldsymbol{y}$ is a linearly independent subset of the elements of $\ddot{\mathbf{q}}$.) Substituting this expression for $\ddot{\mathbf{q}}$ into (3.92), and premultiplying both sides by $\boldsymbol{K}^{\mathrm{T}}$ to eliminate $\boldsymbol{\tau}^{\mathrm{c}}$, produces

$$\boldsymbol{K}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{K}\boldsymbol{y} = \boldsymbol{K}^{\mathrm{T}}\left(\boldsymbol{\tau} + \boldsymbol{\tau}^{\mathrm{a}} - (\boldsymbol{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_{\mathrm{g}}) - \boldsymbol{H}\ddot{\mathbf{q}}_0\right)\,. \tag{3.99}$$

This method also has cubic complexity, but it can be the most efficient if $n - n^{\mathrm{c}}$ is small. It is also reported to be more stable than method 1 [3.34].

3.6.2 Closed-Loop Algorithm

Algorithms for calculating $\boldsymbol{H}$ and $\boldsymbol{C}\dot{\mathbf{q}}+\boldsymbol{\tau}_{\mathrm{g}}$ can be found in Sect. 3.5.3 and 3.5.1, respectively, which just leaves $\boldsymbol{L}$, $\mathbf{l}$ and $\boldsymbol{\tau}^{\mathrm{a}}$. To keep things simple, we will assume that all loop-closing joints are zero-DOF joints.

There is no loss of generality with this assumption: one simply breaks open the loops by cutting links instead of joints (Fig. 3.10). However, there may be some loss of efficiency. With this assumption, we only need to calculate $\boldsymbol{L}$ and $\mathbf{l}$, since $\boldsymbol{\tau}^{\mathrm{a}}=\mathbf{0}$.

Loop Constraints

In the general case, the velocity constraint equation for loop k is

$$\left(\boldsymbol{\Psi}_k^{\mathrm{c}}\right)^{\mathrm{T}}\left(\mathbf{v}_{s(k)}-\mathbf{v}_{p(k)}\right)=\mathbf{0}\,, \tag{3.100}$$

and the acceleration constraint is

$$\left(\boldsymbol{\Psi}_k^{\mathrm{c}}\right)^{\mathrm{T}}\left(\mathbf{a}_{s(k)}-\mathbf{a}_{p(k)}\right)+\left(\dot{\boldsymbol{\Psi}}_k^{\mathrm{c}}\right)^{\mathrm{T}}\left(\mathbf{v}_{s(k)}-\mathbf{v}_{p(k)}\right)=\mathbf{0}\,. \tag{3.101}$$

However, if every loop-closing joint has zero DOF, then these equations simplify to

$$\mathbf{v}_{s(k)}-\mathbf{v}_{p(k)}=\mathbf{0} \tag{3.102}$$

and

$$\mathbf{a}_{s(k)}-\mathbf{a}_{p(k)}=\mathbf{0}\,. \tag{3.103}$$

Let us define a loop Jacobian, $\boldsymbol{J}_k$, with the property that

$$\mathbf{v}_{s(k)}-\mathbf{v}_{p(k)}=\boldsymbol{J}_k\dot{\mathbf{q}}\,. \tag{3.104}$$

$\boldsymbol{J}_k$ is a $6\times n$ matrix defined by the formula

$$\boldsymbol{J}_k=\left(e_{1k}\boldsymbol{\Phi}_1\;\cdots\;e_{N_Bk}\boldsymbol{\Phi}_{N_B}\right)\,, \tag{3.105}$$

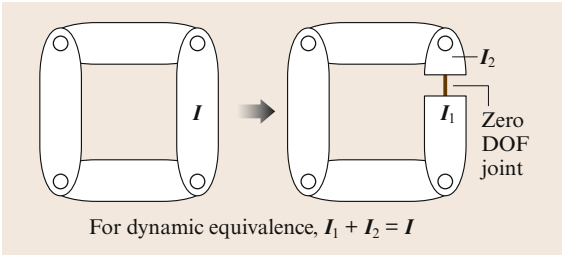


Fig. 3.10 Inserting a zero-DOF joint in preparation for cutting the loop open at that joint

where

$$e_{ik}=\begin{cases}+1 & \text{if } s(k)\in c^*(i) \text{ and } p(k)\notin c^*(i)\,,\\ -1 & \text{if } p(k)\in c^*(i) \text{ and } s(k)\notin c^*(i)\,,\\ 0 & \text{otherwise.}\end{cases}$$

In other words, $e_{ik}=+1$ if joint i lies on the path to $s(k)$ but not the path to $p(k)$; $e_{ik}=-1$ if joint i lies on the path to $p(k)$ but not the path to $s(k)$; and $e_{ik}=0$ if joint i lies on both paths or on neither.

The loop acceleration constraint can now be written

$$\begin{aligned}\mathbf{0}&=\mathbf{a}_{s(k)}-\mathbf{a}_{p(k)}\\&=\boldsymbol{J}_k\ddot{\mathbf{q}}+\dot{\boldsymbol{J}}_k\dot{\mathbf{q}}\\&=\boldsymbol{J}_k\ddot{\mathbf{q}}+\mathbf{a}^{\mathrm{vp}}_{s(k)}-\mathbf{a}^{\mathrm{vp}}_{p(k)}\,,\end{aligned} \tag{3.106}$$

where $\mathbf{a}_i^{\mathrm{vp}}$ is the velocity-product acceleration of link i, which is the acceleration it would have if $\ddot{\mathbf{q}}$ were zero. The velocity-product acceleration of every link is calculated during the calculation of the vector $\boldsymbol{C}\dot{\mathbf{q}}+\boldsymbol{\tau}_{\mathrm{g}}$ (3.84). If the RNEA is used to calculate $\boldsymbol{C}\dot{\mathbf{q}}+\boldsymbol{\tau}_{\mathrm{g}}$, then $\mathbf{a}_i^{\mathrm{vp}}$ will be the value of $\mathbf{a}_i$ calculated by the RNEA with its acceleration argument set to zero.

The matrices $\boldsymbol{L}$ and $\mathbf{l}$ can now be expressed as follows

$$\boldsymbol{L}=\begin{pmatrix}\boldsymbol{L}_{N_B+1}\\ \vdots\\ \boldsymbol{L}_{N_J}\end{pmatrix}\quad\text{and}\quad\mathbf{l}=\begin{pmatrix}\boldsymbol{l}_{N_B+1}\\ \vdots\\ \boldsymbol{l}_{N_J}\end{pmatrix}, \tag{3.107}$$

where

$$\boldsymbol{L}_k=\boldsymbol{J}_k \tag{3.108}$$

and

$$\boldsymbol{l}_k=\mathbf{a}^{\mathrm{vp}}_{p(k)}-\mathbf{a}^{\mathrm{vp}}_{s(k)}\,. \tag{3.109}$$

Constraint Stabilization

In practice, it is necessary to stabilize loop-closure constraints, or they will simply fly apart during simulation because of numerical integration errors. The standard technique is due to *Baumgarte* [3.3, 7, 35], and consists of replacing each constraint equation of the form

$$a_{\mathrm{e}}=0\,,$$

with one of the form

$$a_{\mathrm{e}}+K_{\mathrm{v}}v_{\mathrm{e}}+K_{\mathrm{p}}p_{\mathrm{e}}=0\,,$$

where a_{e}, v_{e}, and p_{e} are the acceleration, velocity, and position errors, respectively, and K_{v} and K_{p} are positive constants. Typically, one chooses a time constant,

t_c, according to how quickly one wants the position and velocity errors to decay. K_p and K_v are then given by the formulae $K_v = 2/t_c$ and $K_p = 1/t_c^2$. However, there is no good rule for choosing t_c. If t_c is too long, then loop-constraint errors accumulate faster than they decay; if t_c is too short, then the equations of motion become excessively stiff, causing a loss of numerical integration accuracy. A reasonable value for a large, slow industrial robot is $t_c = 0.1$, while a smaller, faster robot might need $t_c = 0.01$.

To incorporate stabilization terms into the loop constraint equation, we replace (3.109) with

$$\boldsymbol{l}_k = \mathbf{a}^{\mathrm{vp}}_{p(k)} - \mathbf{a}^{\mathrm{vp}}_{s(k)} - K_v(\mathbf{v}_{s(k)} - \mathbf{v}_{p(k)}) - K_p \mathbf{p}_{ek} \ , \quad (3.110)$$

where $\mathbf{p}_{ek}$ is a vector representing the position error in loop k. In absolute coordinates (i. e., link 0 coordinates), $\mathbf{p}_{ek}$ is given by

$$\mathbf{p}_{ek} = \mathrm{x_to_vec}({}^0\boldsymbol{X}_{p(k)}\, \boldsymbol{X}_{L1}^{-1}(k)\, \boldsymbol{X}_{L2}(k)\, {}^{s(k)}\boldsymbol{X}_0) \ , \quad (3.111)$$

where the $\boldsymbol{X}_{L1}(k)$ and $\boldsymbol{X}_{L2}(k)$ transforms are defined in (3.62) and (3.63), and shown in Fig. 3.4 for joint k, and $\mathrm{x_to_vec}\left({}^B\boldsymbol{X}_A\right)$ computes a vector approximating the displacement from frame A to frame B, assuming this displacement to be infinitesimal. x_to_vec can be defined as

$$\mathrm{x_to_vec}(\boldsymbol{X}) = \frac{1}{2}\begin{pmatrix} X_{23} - X_{32} \\ X_{31} - X_{13} \\ X_{12} - X_{21} \\ X_{53} - X_{62} \\ X_{61} - X_{43} \\ X_{42} - X_{51} \end{pmatrix} . \quad (3.112)$$

Algorithm

Algorithm 3.7 shows an algorithm for calculating $\boldsymbol{L}$ and $\mathbf{l}$ for the special case when all the loop-closing joints have 0-DOF. It combines simplicity with good performance by transforming every quantity that is needed to formulate the loop-closure constraints into a single coordinate system, in this case absolute (link 0) coordinates, so that no further transforms are needed.

The first loop calculates the transforms from absolute to link coordinates, and uses them to transform $\boldsymbol{\Phi}_i$ to absolute coordinates. Only the $\boldsymbol{\Phi}_i$ that are needed in the loop-closure constraints are transformed.

The second loop calculates the nonzero elements of $\boldsymbol{L}$ (which can be sparse), according to (3.105). The inner `while` loop terminates on the root of the loop, which is the highest-numbered common ancestor of links $p(k)$ and $s(k)$. It could be the fixed base if they have no other common ancestor. The second loop ends with the calculation of $\mathbf{l}$, in absolute coordinates, according to (3.110).

Algorithm 3.7 Algorithm to calculate loop-closure constraints

inputs: *model* , *RNEA partial results*
outputs: $\boldsymbol{L}$, $\mathbf{l}$
model data : N_B, $p(i)$, N_J, $p(k)$, $s(k)$, $LB(i)$, $\boldsymbol{X}_{L1}(k)$, $\boldsymbol{X}_{L2}(k)$, K_p, K_v
RNEA data : $\boldsymbol{\Phi}_i$, ${}^i\boldsymbol{X}_{p(i)}$, $\mathbf{v}_{p(k)}$, $\mathbf{v}_{s(k)}$, $\mathbf{a}^{\mathrm{vp}}_{p(k)}$, $\mathbf{a}^{\mathrm{vp}}_{s(k)}$

for $i = 1$ **to** N_B **do**
  if $p(i) \neq 0$ **then**
    ${}^i\boldsymbol{X}_0 = {}^i\boldsymbol{X}_{p(i)}\, {}^{p(i)}\boldsymbol{X}_0$
  end if
  if $LB(i) \neq$ null **then**
    ${}^0\boldsymbol{\Phi}_i = {}^i\boldsymbol{X}_0^{-1}\, \boldsymbol{\Phi}_i$
  end if
end for
$\boldsymbol{L} = \mathbf{0}$
for $k = N_B + 1$ **to** N_J **do**
  $i = p(k)$
  $j = s(k)$
  while $i \neq j$ **do**
    if $i > j$ **then**
      $\boldsymbol{L}_{k,i} = -{}^0\boldsymbol{\Phi}_i$
      $i = p(i)$
    else
      $\boldsymbol{L}_{k,j} = {}^0\boldsymbol{\Phi}_j$
      $j = p(j)$
    end if
  end while
  $\mathbf{a}_e = {}^{s(k)}\boldsymbol{X}_0^{-1}\, \mathbf{a}^{\mathrm{vp}}_{s(k)} - {}^{p(k)}\boldsymbol{X}_0^{-1}\, \mathbf{a}^{\mathrm{vp}}_{p(k)}$
  $\mathbf{v}_e = {}^{s(k)}\boldsymbol{X}_0^{-1}\, \mathbf{v}_{s(k)} - {}^{p(k)}\boldsymbol{X}_0^{-1}\, \mathbf{v}_{p(k)}$
  $\mathbf{p}_e = \mathrm{x_to_vec}\left({}^{p(k)}\boldsymbol{X}_0^{-1}\, \boldsymbol{X}_{L1}^{-1}(k)\, \boldsymbol{X}_{L2}(k)\, {}^{s(k)}\boldsymbol{X}_0\right)$
  $\boldsymbol{l}_k = -\mathbf{a}_e - K_v\, \mathbf{v}_e - K_p\, \mathbf{p}_e$
end for

3.7 Conclusions and Further Reading

This chapter has presented the fundamentals of rigid-body dynamics as they apply to robot mechanisms. It has covered the following topics: the spatial vector algebra, which provides a concise notation for describing and implementing dynamics equations and algorithms; the canonical equations of motion that are most fre-

quently used in robotics; how to construct a dynamic model of a robot; and several efficient model-based algorithms for calculating inverse dynamics, forward dynamics, and the joint-space and operational-space inertia matrices.

There are many topics in dynamics that have not been mentioned in this chapter, but can be found in later chapters of this handbook. The dynamics of robots with elastic links and joints is covered in Chap. 11; the problem of identifying the parameters of a dynamic model is covered in Chap. 6; the dynamics of physical contact between a robot and the objects in its environment is described in Chap. 37; and the dynamics of robots with floating bases is described in Chap. 55.

We conclude this chapter by noting that a brief history of robot dynamics can be found in [3.36], and that a more extensive treatment of robot dynamics can be found in books such as [3.8, 10, 15, 29, 37–40]. Finally, some suggestions for further reading are listed below.

3.7.1 Multibody Dynamics

Robot dynamics can be regarded as a subset (or a specific application) of the broader discipline of multibody dynamics. Books on multibody dynamics include [3.3, 14, 35, 41–46]. Of course, multibody dynamics is, in turn, a subset of classical mechanics; and the mathematical foundations of the subject can be found in any good book on classical mechanics, such as [3.13].

3.7.2 Alternative Representations

This chapter has used spatial vectors to express the equations of motion. There are various alternatives to the use of spatial vectors: other kinds of 6-D vector, 3-D vectors, 4×4 matrices, and the spatial operator algebra. All 6-D vector formalisms are similar, but are not exactly the same. The main alternatives to spatial vectors are: screws [3.10–12], motors [3.47], Lie algebras [3.12, 48], and ad hoc notations. (An ad hoc notation is one in which 3-D vectors are grouped into pairs for the purpose of reducing the volume of algebra.) Three-dimensional vectors are the formalism used in most classical mechanics and multibody texts, and are also a precursor to 6-D vector and 4×4 matrix formalisms. 4×4 matrices are popular in robotics because they are very useful for kinematics. However, they are not so useful for dynamics. 4×4 matrix formulations of dynamics can be found in [3.37, 49, 50]. The spatial operator algebra was developed at the Jet Propulsion Laboratory (JPL) by Rodriguez, Jain, and others. It uses $6N$-dimensional vectors and $6N \times 6N$ matrices, the latter regarded as linear operators. Examples of this notation can be found in [3.38, 51–53].

3.7.3 Alternative Formulations

This chapter used a vectorial formulation of the equations of motion that is usually called the Newtonian or Newton–Euler formulation. The main alternative is the Lagrangian formulation, in which the equations of motion are obtained via Lagrange's equation. Examples of the Lagrangian formulation can be found in [3.9, 10, 18, 54, 55]. *Kane*'s method has also been applied in robotics [3.56, 57].

3.7.4 Efficiency

Because of the need for real-time implementation, especially in control, the robotics community has focused on the problem of computational efficiency. For *inverse dynamics*, the $O(n)$ recursive Newton–Euler algorithm (RNEA) of *Luh* et al. [3.4] remains the most important algorithm. Further improvements to the algorithm are given in [3.58, 59]. For *forward dynamics*, the two algorithms presented in this chapter remain the most important for computational considerations: the $O(n)$ articulated-body algorithm (ABA) developed by *Featherstone* [3.1] and the $O(n^3)$ algorithm based on the composite-rigid-body algorithm (CRBA) of *Walker* and *Orin* [3.5]. Improvements were made in the ABA over the years [3.15, 17, 25] so that it was more efficient than the CRBA-based algorithm for decreasingly smaller values of n. However, more recent application of the CRBA to branched kinematic trees [3.26] and robotic systems with motion-controlled appendages [3.60] continue to show the viability of the CRBA approach.

For the *joint-space inertia matrix*, the CRBA [3.5] is the most important algorithm. A number of improvements and modifications have been made over the years to increase its computational efficiency [3.15, 61–63]. For the *operational-space inertia matrix*, efficient $O(n)$ algorithms have been developed [3.28–30] and applied to increasingly complex systems [3.20, 31, 33]. The exploitation of branch-induced sparsity also results in an efficient algorithm [3.32].

3.7.5 Accuracy

Concerns can arise over the numerical accuracy of a dynamics algorithm, the accuracy of a simulation (i. e., numerical integration accuracy), or the accuracy of a dynamic model. The numerical accuracy of dynamics algorithms has received relatively little attention compared with efficiency. The RNEA, CRBA, and ABA have all been tested for accuracy on a large variety of rigid-body systems, but the same cannot be said of most other algorithms. Rigid-body systems are often ill-conditioned, in the sense that a small change

in the applied force (or a model parameter) can produce a large change in the resulting acceleration. This phenomenon was studied by *Featherstone* [3.27], who discovered that the ill-conditioning gets worse with increasing body count, and that it can grow in proportion to $O(n^4)$ in the worst case. Other publications on this topic include [3.8, 34, 64, 65].

3.7.6 Software Packages

A number of software packages have been developed to provide dynamic simulation capabilities for multibody systems, and in particular, robotic systems. Several have been written in MATLAB for ease of integration with other analysis, control, and simulation programs. Many packages are open source, and some are offered at a relatively low cost to the user. They differ in their capabilities in a variety of ways including: speed, topologies and joint models supported, accuracy, underlying dynamic formulation and associated order of complexity, user interface, graphics support, numerical integration routines, integration with other code, application support, and cost. Among those commonly cited are: Adams [3.66], Autolev [3.67], Bullet [3.68], DART [3.69], DynaMechs [3.70], Gazebo [3.71], Open Dynamics Engine [3.72], Robotics Studio [3.73], Robotics Toolbox [3.74], Robotran [3.75, 76], SD/FAST [3.77], Simbody [3.78], SimMechanics [3.79], SYMORO [3.80, 81] and Webots [3.82].

3.7.7 Symbolic Simplification

The technique of symbolic simplification takes a general-purpose dynamics algorithm, and applies it symbolically to a specific dynamic model. The result is a list of assignment statements detailing what the algorithm would have done if it had been executed for real. This list is then inspected and pruned of all unnecessary calculations, and the remainder is output to a text file in the form of computer source code. This code can run as much as ten times faster than the original general-purpose algorithm, but it is specific to one dynamic model. Both Autolev [3.67] and SD/FAST [3.77] use this technique. Other publications on symbolic simplification for dynamics include [3.76, 80, 81, 83–88].

3.7.8 Algorithms for Parallel Computers

In order to speed up the common dynamics computations, a number of algorithms have been developed for parallel and pipelined computers. For *inverse dynamics*, early work focused on speeding up the $O(n)$ RNEA on up to n processors [3.89, 90] while subsequent work resulted in $O(\log_2 n)$ algorithms [3.91, 92]. For the $O(n^2)$ CRBA to compute the *joint-space inertia matrix*, early work resulted in $O(\log_2 n)$ algorithms for n processors to compute the composite-rigid-body inertias and diagonal elements of the matrix [3.93, 94]. Subsequent work resulted in $O(\log_2 n)$ algorithms for $O(n^2)$ processors to compute the entire matrix [3.95, 96]. For *forward dynamics*, speedup was obtained for a multiple manipulator system on a parallel/pipelined supercomputer [3.97]. The first $O(\log_2 n)$ algorithm for n processors was developed for an unbranched serial chain [3.98]. More recent work has focused on $O(\log_2 n)$ algorithms for more complex structures [3.65, 99, 100].

3.7.9 Topologically-Varying Systems

There are many robot mechanisms whose topology varies over time because of a change of contact conditions, especially with the environment. In legged vehicles, use of a compliant ground-contact model to compute the contact forces reduced the closed-loop structure to a tree structure [3.101]. However, for cases in which the contacts are very stiff, numerical integration problems may result. In more recent work [3.40, 102] in which hard contact constraints are assumed, an efficient method was used to reduce the large number of coordinate variables from that which may be necessary in general-purpose motion analysis systems [3.43]. Also, they were able to automatically identify the variables as the structure varied and developed a method for computing the velocity boundary conditions after configuration changes [3.40, 102].

References

3.1 R. Featherstone: The calculation of robot dynamics using articulated-body inertias, Int. J. Robotics Res. **2**(1), 13–30 (1983)

3.2 J.J. Craig: *Introduction to Robotics: Mechanics and Control*, 3rd edn. (Prentice Hall, Upper Saddle River 2005)

3.3 R.E. Roberson, R. Schwertassek: *Dynamics of Multibody Systems* (Springer, Berlin, Heidelberg 1988)

3.4 J.Y.S. Luh, M.W. Walker, R.P.C. Paul: On-line computational scheme for mechanical manipulators, Trans. ASME J. Dyn. Syst. Meas. Control **102**(2), 69–76 (1980)

3.5 M.W. Walker, D.E. Orin: Efficient dynamic computer simulation of robotic mechanisms, Trans. ASME J. Dyn. Syst. Meas. Control **104**, 205–211 (1982)

3.6 D. Baraff: Linear-time dynamics using lagrange multipliers, Proc. 23rd Annu. Conf. Comp. Graph. Interact. Tech., New Orleans (1996) pp. 137–146

3.7 J. Baumgarte: Stabilization of constraints and integrals of motion in dynamical systems, Comput. Methods Appl. Mech. Eng. **1**, 1–16 (1972)

3.8 R. Featherstone: *Rigid Body Dynamics Algorithms* (Springer, New York 2008)

3.9 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)

3.10 J. Angeles: *Fundamentals of Robotic Mechanical Systems*, 2nd edn. (Springer, New York 2003)

3.11 R.S. Ball: *A Treatise on the Theory of Screws* (Cambridge Univ. Press, London 1900), Republished (1998)

3.12 J.M. Selig: *Geometrical Methods in Robotics* (Springer, New York 1996)

3.13 D.T. Greenwood: *Principles of Dynamics* (Prentice-Hall, Englewood Cliffs 1988)

3.14 F.C. Moon: *Applied Dynamics* (Wiley, New York 1998)

3.15 R. Featherstone: *Robot Dynamics Algorithms* (Kluwer, Boston 1987)

3.16 R. Featherstone: *Spatial v2*, http://royfeatherstone.org/spatial/v2 (2012)

3.17 S. McMillan, D.E. Orin: Efficient computation of articulated-body inertias using successive axial screws, IEEE Trans. Robotics Autom. **11**, 606–611 (1995)

3.18 L. Sciavicco, B. Siciliano: *Modeling and Control of Robot Manipulators*, 2nd edn. (Springer, London 2000)

3.19 J. Slotine, W. Li: On the adaptive control of robot manipulators, Int. J. Robotics Res. **6**(3), 49–59 (1987)

3.20 K.S. Chang, O. Khatib: Operational space dynamics: Efficient algorithms for modeling and control of branching mechanisms, Proc. IEEE Int. Conf. Robotics Autom., San Francisco (2000) pp. 850–856

3.21 O. Khatib: A unified approach to motion and force control of robot manipulators: The operational space formulation, IEEE J. Robotics Autom. **3**(1), 43–53 (1987)

3.22 Y.F. Zheng, H. Hemami: Mathematical modeling of a robot collision with its environment, J. Robotics Syst. **2**(3), 289–307 (1985)

3.23 W. Khalil, E. Dombre: *Modeling, Identification and Control of Robots* (Kogan Page Sci., London 2002)

3.24 J. Denavit, R.S. Hartenberg: A kinematic notation for lower-pair mechanisms based on matrices, J. Appl. Mech. **22**, 215–221 (1955)

3.25 H. Brandl, R. Johanni, M. Otter: A very efficient algorithm for the simulation of robots and similar multibody systems without inversion of the mass matrix, Proc. IFAC/IFIP/IMACS Int. Symp. Theory Robots, Vienna (1986)

3.26 R. Featherstone: Efficient factorization of the joint space inertia matrix for branched kinematic trees, Int. J. Robotics Res. **24**(6), 487–500 (2005)

3.27 R. Featherstone: An empirical study of the joint space inertia matrix, Int. J. Robotics Res. **23**(9), 859–871 (2004)

3.28 K. Kreutz-Delgado, A. Jain, G. Rodriguez: Recursive formulation of operational space control, Proc. IEEE Int. Conf. Robotics Autom., Sacramento (1991) pp. 1750–1753

3.29 K.W. Lilly: *Efficient Dynamic Simulation of Robotic Mechanisms* (Kluwer, Boston 1993)

3.30 K.W. Lilly, D.E. Orin: Efficient O(N) recursive computation of the operational space inertia matrix, IEEE Trans. Syst. Man Cybern. **23**(5), 1384–1391 (1993)

3.31 G. Rodriguez, A. Jain, K. Kreutz-Delgado: Spatial operator algebra for multibody system dynamics, J. Astronaut. Sci. **40**(1), 27–50 (1992)

3.32 R. Featherstone: Exploiting sparsity in operational-space dynamics, Int. J. Robotics Res. **29**(10), 1353–1368 (2010)

3.33 P. Wensing, R. Featherstone, D.E. Orin: A reduced-order recursive algorithm for the computation of the operational-space inertia matrix, Proc. IEEE Int. Conf. Robotics Autom., St. Paul (2012) pp. 4911–4917

3.34 R.E. Ellis, S.L. Ricker: Two numerical issues in simulating constrained robot dynamics, IEEE Trans. Syst. Man Cybern. **24**(1), 19–27 (1994)

3.35 J. Wittenburg: *Dynamics of Systems of Rigid Bodies* (Teubner, Stuttgart 1977)

3.36 R. Featherstone, D.E. Orin: Robot dynamics: Equations and algorithms, Proc. IEEE Int. Conf. Robotics Autom., San Francisco (2000) pp. 826–834

3.37 C.A. Balafoutis, R.V. Patel: *Dynamic Analysis of Robot Manipulators: A Cartesian Tensor Approach* (Kluwer, Boston 1991)

3.38 A. Jain: *Robot and Multibody Dynamics: Analysis and Algorithms* (Springer, New York 2011)

3.39 L.W. Tsai: *Robot Analysis and Design: The Mechanics of Serial and Parallel Manipulators* (Wiley, New York 1999)

3.40 K. Yamane: *Simulating and Generating Motions of Human Figures* (Springer, Berlin, Heidelberg 2004)

3.41 F.M.L. Amirouche: *Fundamentals of Multibody Dynamics: Theory and Applications* (Birkhäuser, Boston 2006)

3.42 M.G. Coutinho: *Dynamic Simulations of Multibody Systems* (Springer, New York 2001)

3.43 E.J. Haug: *Computer Aided Kinematics and Dynamics of Mechanical Systems* (Allyn and Bacon, Boston 1989)

3.44 R.L. Huston: *Multibody Dynamics* (Butterworths, Boston 1990)

3.45 A.A. Shabana: *Computational Dynamics*, 2nd edn. (Wiley, New York 2001)

3.46 V. Stejskal, M. Valášek: *Kinematics and Dynamics of Machinery* (Marcel Dekker, New York 1996)

3.47 L. Brand: *Vector and Tensor Analysis*, 4th edn. (Wiley/Chapman Hall, New York/London 1953)

3.48 F.C. Park, J.E. Bobrow, S.R. Ploen: A lie group formulation of robot dynamics, Int. J. Robotics Res. **14**(6), 609–618 (1995)

3.49 M.E. Kahn, B. Roth: The near minimum-time control of open-loop articulated kinematic chains, J. Dyn. Syst. Meas. Control **93**, 164–172 (1971)

3.50 J.J. Uicker: Dynamic force analysis of spatial linkages, Trans. ASME J. Appl. Mech. **34**, 418–424 (1967)
3.51 A. Jain: Unified formulation of dynamics for serial rigid multibody systems, J. Guid. Control Dyn. **14**(3), 531–542 (1991)
3.52 G. Rodriguez: Kalman filtering, smoothing, and recursive robot arm forward and inverse dynamics, IEEE J. Robotics Autom. **RA-3**(6), 624–639 (1987)
3.53 G. Rodriguez, A. Jain, K. Kreutz-Delgado: A spatial operator algebra for manipulator modelling and control, Int. J. Robotics Res. **10**(4), 371–381 (1991)
3.54 J.M. Hollerbach: A recursive lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity, IEEE Trans. Syst. Man Cybern. **SMC-10**(11), 730–736 (1980)
3.55 M.W. Spong, S. Hutchinson, M. Vidyasagar: *Robot Modeling and Control* (Wiley, Hoboken 2006)
3.56 K.W. Buffinton: Kane's Method in Robotics. In: *Robotics and Automation Handbook*, ed. by T.R. Kurfess (CRC, Boca Raton 2005), 6-1–6-31
3.57 T.R. Kane, D.A. Levinson: The use of kane's dynamical equations in robotics, Int. J. Robotics Res. **2**(3), 3–21 (1983)
3.58 C.A. Balafoutis, R.V. Patel, P. Misra: Efficient modeling and computation of manipulator dynamics using orthogonal cartesian tensors, IEEE J. Robotics Autom. **4**, 665–676 (1988)
3.59 X. He, A.A. Goldenberg: An algorithm for efficient computation of dynamics of robotic manipulators, Proc. 4th Int. Conf. Adv. Robotics, Columbus (1989) pp. 175–188
3.60 W. Hu, D.W. Marhefka, D.E. Orin: Hybrid kinematic and dynamic simulation of running machines, IEEE Trans. Robotics **21**(3), 490–497 (2005)
3.61 C.A. Balafoutis, R.V. Patel: Efficient computation of manipulator inertia matrices and the direct dynamics problem, IEEE Trans. Syst. Man Cybern. **19**, 1313–1321 (1989)
3.62 K.W. Lilly, D.E. Orin: Alternate formulations for the manipulator inertia matrix, Int. J. Robotics Res. **10**, 64–74 (1991)
3.63 S. McMillan, D.E. Orin: Forward dynamics of multilegged vehicles using the composite rigid body method, Proc. IEEE Int. Conf. Robotics Autom. (1998) pp. 464–470
3.64 U.M. Ascher, D.K. Pai, B.P. Cloutier: Forward dynamics: Elimination methods, and formulation stiffness in robot simulation, Int. J. Robotics Res. **16**(6), 749–758 (1997)
3.65 R. Featherstone: A divide-and-conquer articulated-body algorithm for parallel $O(\log(n))$ calculation of rigid-body dynamics. Part 2: Trees, loops and accuracy, Int. J. Robotics Res. **18**(9), 876–892 (1999)
3.66 MSC Software Corporation: Adams, http://www.mscsoftware.com/
3.67 T. Kane, D. Levinson: *Autolev user's manual* (OnLine Dynamics Inc., Sunnyvale 2005)
3.68 Real-Time Physics Simulation: Bullet, http://bulletphysics.org/wordpress (2015)
3.69 Georgia Tech Graphics Lab and Humanoid Robotics Lab: DART, http://dartsim.github.io (2011)
3.70 S. McMillan, D.E. Orin, R.B. McGhee: DynaMechs: An object oriented software package for efficient dynamic simulation of underwater robotic vehicles. In: *Underwater Robotic Vehicles: Design and Control*, ed. by J. Yuh (TSI Press, Albuquerque 1995) pp. 73–98
3.71 Open Source Robotics Foundation: Gazebo, http://gazebosim.org (2002)
3.72 R. Smith: Open Dynamics Engine User Guide, http://opende.sourceforge.net (2006)
3.73 Microsoft Corporation: Robotics Developer Studio, http://www.microsoft.com/robotics (2010)
3.74 P.I. Corke: A robotics toolbox for MATLAB, IEEE Robotics Autom. Mag. **3**(1), 24–32 (1996)
3.75 Robotran: http://www.robotran.be (Center for Research in Mechatronics, Université catholique de Louvain 2015)
3.76 J.C. Samin, P. Fisette: *Symbolic Modeling of Multibody Systems* (Kluwer, Dordrecht 2003)
3.77 M.G. Hollars, D.E. Rosenthal, M.A. Sherman: *SD/FAST User's Manual* (Symbolic Dynamics Inc., Mountain View 1994)
3.78 M. Sherman, P. Eastman: Simbody, https://simtk.org/home/simbody (2015)
3.79 G.D. Wood, D.C. Kennedy: *Simulating Mechanical Systems in Simulink with SimMechanics* (MathWorks Inc., Natick 2003)
3.80 W. Khalil, D. Creusot: SYMORO+: A system for the symbolic modeling of robots, Robotica **15**, 153–161 (1997)
3.81 W. Khalil, A. Vijayalingam, B. Khomutenko, I. Mukhanov, P. Lemoine, G. Ecorchard: OpenSYMORO: An open-source software package for symbolic modelling of robots, Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (2014) pp. 126–1211
3.82 Cyberbotics Ltd.: *Webots User Guide*, http://www.cyberbotics.com (2015)
3.83 I.C. Brown, P.J. Larcombe: A survey of customised computer algebra programs for multibody dynamic modelling. In: *Symbolic Methods in Control System Analysis and Design*, ed. by N. Munro (Inst. Electr. Eng., London 1999) pp. 53–77
3.84 J.J. Murray, C.P. Neuman: ARM: An algebraic robot dynamic modeling program, Proc. IEEE Int. Conf. Robotics Autom., Atlanta (1984) pp. 103–114
3.85 J.J. Murray, C.P. Neuman: Organizing customized robot dynamic algorithms for efficient numerical evaluation, IEEE Trans. Syst. Man Cybern. **18**(1), 115–125 (1988)
3.86 F.C. Park, J. Choi, S.R. Ploen: Symbolic formulation of closed chain dynamics in independent coordinates, Mech. Mach. Theory **34**, 731–751 (1999)
3.87 M. Vukobratovic, N. Kircanski: *Real-Time Dynamics of Manipulation Robots* (Springer, New York 1985)
3.88 J. Wittenburg, U. Wolz: Mesa Verde: A symbolic program for nonlinear articulated-rigid-body dynamics, ASME Des. Eng. Div. Conf., Cincinnati (1985) pp. 1–8, ASME Paper No. 85-DET-151
3.89 J.Y.S. Luh, C.S. Lin: Scheduling of parallel computation for a computer-controlled mechanical manipulator, IEEE Trans. Syst. Man Cybern. **12**(2), 214–234 (1982)

3.90 D.E. Orin: Pipelined approach to inverse plant plus jacobian control of robot manipulators, Proc. IEEE Int. Conf. Robotics Autom., Atlanta (1984) pp. 169–175

3.91 R.H. Lathrop: Parallelism in manipulator dynamics, Int. J. Robotics Res. **4**(2), 80–102 (1985)

3.92 C.S.G. Lee, P.R. Chang: Efficient parallel algorithm for robot inverse dynamics computation, IEEE Trans. Syst. Man Cybern. **16**(4), 532–542 (1986)

3.93 M. Amin-Javaheri, D.E. Orin: Systolic architectures for the manipulator inertia matrix, IEEE Trans. Syst. Man Cybern. **18**(6), 939–951 (1988)

3.94 C.S.G. Lee, P.R. Chang: Efficient parallel algorithms for robot forward dynamics computation, IEEE Trans. Syst. Man Cybern. **18**(2), 238–251 (1988)

3.95 M. Amin-Javaheri, D.E. Orin: Parallel algorithms for computation of the manipulator inertia matrix, Int. J. Robotics Res. **10**(2), 162–170 (1991)

3.96 A. Fijany, A.K. Bejczy: A class of parallel algorithms for computation of the manipulator inertia matrix, IEEE Trans. Robotics Autom. **5**(5), 600–615 (1989)

3.97 S. McMillan, P. Sadayappan, D.E. Orin: Parallel dynamic simulation of multiple manipulator systems: Temporal versus spatial methods, IEEE Trans. Syst. Man Cybern. **24**(7), 982–990 (1994)

3.98 A. Fijany, I. Sharf, G.M.T. D'Eleuterio: Parallel $O(\log N)$ algorithms for computation of manipulator forward dynamics, IEEE Trans. Robotics Autom. **11**(3), 389–400 (1995)

3.99 R. Featherstone: A divide-and-conquer articulated-body algorithm for parallel $O(\log(n))$ calculation of rigid-body dynamics. Part 1: Basic algorithm, Int. J. Robotics Res. **18**(9), 867–875 (1999)

3.100 R. Featherstone, A. Fijany: A technique for analyzing constrained rigid-body systems and its application to the constraint force algorithm, IEEE Trans. Robotics Autom. **15**(6), 1140–1144 (1999)

3.101 P.S. Freeman, D.E. Orin: Efficient dynamic simulation of a quadruped using a decoupled tree-structured approach, Int. J. Robotics Res. **10**, 619–627 (1991)

3.102 Y. Nakamura, K. Yamane: Dynamics computation of structure-varying kinematic chains and its application to human figures, IEEE Trans. Robotics Autom. **16**(2), 124–134 (2000)

Multimedia Contents

4. Mechanism and Actuation

Victor Scheinman, J. Michael McCarthy, Jae-Bok Song

This chapter focuses on the principles that guide the design and construction of robotic systems. The kinematics equations and Jacobian of the robot characterize its range of motion and mechanical advantage, and guide the selection of its size and joint arrangement. The tasks a robot is to perform and the associated precision of its movement determine detailed features such as mechanical structure, transmission, and actuator selection. Here we discuss in detail both the mathematical tools and practical considerations that guide the design of mechanisms and actuation for a robot system.

The following sections (Sect. 4.1) discuss characteristics of the mechanisms and actuation that affect the performance of a robot. Sections 4.2–4.6 discuss the basic features of a robot manipulator and their relationship to the mathematical model that is used to characterize its performance. Sections 4.7 and 4.8 focus on the details of the structure and actuation of the robot and how they combine to yield various types of robots. The final Sect. 4.9 relates these design features to various performance metrics.

4.1 **Overview** 68

4.2 **System Features** 68
- 4.2.1 Work Envelope 68
- 4.2.2 Load Capacity 68
- 4.2.3 Kinematic Skeleton 69

4.3 **Kinematics and Kinetics** 69
- 4.3.1 Robot Topology 69
- 4.3.2 Kinematics Equations 71
- 4.3.3 Configuration Space 71
- 4.3.4 Speed Ratios 71
- 4.3.5 Mechanical Advantage 72

4.4 **Serial Robots** 72
- 4.4.1 Design Optimization 73
- 4.4.2 Speed Ratios 73

4.5 **Parallel Robots** 73
- 4.5.1 Workspace 74
- 4.5.2 Mechanical Advantage 74
- 4.5.3 Specialized Parallel Robots 75

4.6 **Mechanical Structure** 75
- 4.6.1 Links 75
- 4.6.2 Joints 76

4.7 **Joint Mechanisms** 76
- 4.7.1 Joint Axis Structures 76

4.8 **Actuators** 78
- 4.8.1 Electromagnetic Actuators 78
- 4.8.2 Hydraulic Actuators 80
- 4.8.3 Pneumatic Actuators 81
- 4.8.4 Other Actuators 82
- 4.8.5 Transmissions 82

4.9 **Robot Performance** 85
- 4.9.1 Robot Speed 85
- 4.9.2 Robot Acceleration 85
- 4.9.3 Repeatability 85
- 4.9.4 Resolution 86
- 4.9.5 Accuracy 86
- 4.9.6 Component Life and Duty Cycle 86
- 4.9.7 Collisions 86

4.10 **Conclusions and Further Reading** 87

Video-References 87

References 87

4.1 Overview

The physical structure such as the beams, links, castings, shafts, slides, and bearings of a robot that create its movable skeleton is termed the mechanical structure or mechanism of the robot. The electric, hydraulic and pneumatic motors and other elements that cause the links of the mechanism to move are called actuators. In this chapter we consider the variety of designs for the mechanisms and actuators that result in a machine system that transforms computer commands into versatile physical movement.

Early robots were designed with general motion capability under the assumption that they would find the largest market if they could perform the widest variety of tasks; this emphasis on flexibility proved to be expensive in both cost and performance. Robots are now often designed for specific applications and to perform limited sets of tasks.

Robot design focuses on the number of joints, physical size, payload capacity, and the movement requirements of the end-effector. The configuration of the movable skeleton and the overall size of the robot are determined by task requirements for reach, workspace, and reorientation ability. These features affect the precision of end-effector path control needed for applications such as for arc-welding and for the smooth movement of paint spraying. They also define the absolute positioning capability necessary for small part assembly, the repeatability needed for material and package handling, and the fine resolution that allows precise, real-time sensor-based motions.

A critical concern in robotic system design is the range of tasks the robot is expected to perform. The robot should be designed to have the flexibility it needs to perform the range of tasks for which it is intended. This determines the topology of the robot mechanism and the actuator system. The choices of geometry, material, sensors, and cable routing follow from these basic decisions.

4.2 System Features

The primary features that characterize a robot are its work envelope and load capacity.

4.2.1 Work Envelope

The space in which a robot can operate is its work envelope, which encloses its workspace. While the workspace of the robot defines positions and orientations that it can achieve to accomplish a task, the work envelope also includes the volume of space the robot itself occupies as it moves. This envelope is defined by the types of joints, their range of movement and the lengths of the links that connect them. The physical size of this envelope and the loads on the robot within this envelope are of primary consideration in the design of the mechanical structure of a robot.

Robot work envelope layouts must include considerations of regions of limited accessibility where the mechanical structure may experience movement limitations. These constraints arise from limited joint travel range, link lengths, the angles between axes, or a combination of these. Revolute joint manipulators generally work better in the middle of their work envelopes than at extremes (Fig. 4.1). Manipulator link lengths and joint travel should be chosen to leave margins for variable sensor-guided controlled path motions and for tool or end-effector changes, as offsets and length differences will often alter the work envelope.

4.2.2 Load Capacity

Load capacity, a primary robot specification, is closely coupled with acceleration and speed. For assembly robots, mechanism acceleration and stiffness (structure and drive stiffness) are often more important design parameters than peak velocity or maximum load capacity, as minimizing pick-and-place motion cycle time, while maintaining placement precision, is generally a top priority in small part assembly. In the case of arc-welding, where slow controlled-path motion is required, velocity

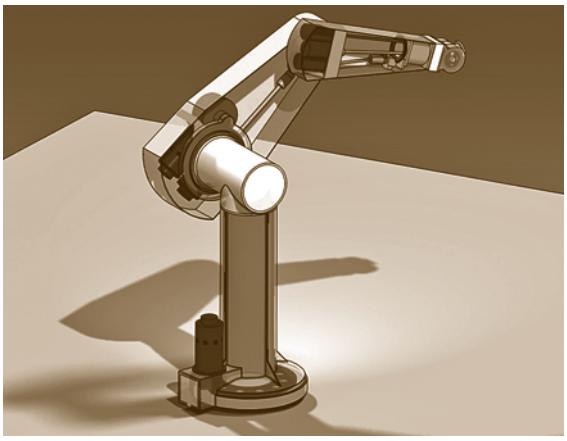


Fig. 4.1 The PUMA 560 robot

jitter and weld path-following accuracy are important. Load capacity should be seen as a variable. It is wise to design and specify a manipulator in terms of useful payload as a function of performance rather than just in terms of maximum capacity.

Load capacity specifications must take into account gravity and inertial loading seen at the end-effector. These factors strongly affect wrist, end-effector design and drive selection. In general, load capacity is more a function of manipulator acceleration and wrist torque than any other factor. The load rating also affects manipulator static structural deflection, steady-state motor torque, system natural frequency, damping, and the choice of control system parameters for best performance and stability.

4.2.3 Kinematic Skeleton

Manipulator shape and size is determined by requirements on its workspace shape and layout, the precision of its movement, its acceleration and speed, and its construction. Cartesian manipulators have the simplest transform and control equation solutions. Their prismatic (straight-line motion) and orthogonal (perpendicular) axes make motion planning and computation relatively straightforward. Because their major motion axes do not couple dynamically (to a first order), their control equations are also simplified. Manipulators with all revolute (rotary) joints are generally harder to control, but they feature a more compact and efficient structure for a given working volume. It is generally easier to design and build a good revolute joint than a long motion prismatic joint. The workspaces of revolute joint manipulators can easily overlap for coordinated multi-robot installations, in contrast to the more exclusive useful workspace of gantry style robots.

Final selection of the robot configuration should capitalize on specific kinematic, structural or performance requirements. For example, a requirement for a very precise vertical straight-line motion may dictate the choice of a simple prismatic vertical axis rather than two or three revolute joints requiring coordinated control.

Six degrees of freedom (DOFs) are the minimum required to place the end-effector or tool of a robotic manipulator at any arbitrary location (position and orientation) within its accessible workspace. Most simple or preplanned tasks can be performed with fewer than six DOFs by careful task setup to eliminate certain axis motions, or because the tool or task does not require full specification of location. An example of this is vertical assembly using a powered screwdriver, where all operations can be achieved with three degrees of freedom.

Some applications require the use of manipulators with more than six DOFs, in particular when mobility or obstacle avoidance are necessary. For example, a pipe-crawling maintenance robot requires control of the robot shape as well as precise positioning of its end-effector. Generally, adding degrees of freedom increases cycle time and reduces load capacity and accuracy for a given manipulator configuration and drive system.

4.3 Kinematics and Kinetics

The dynamics of a robot can be separated into the properties of the movement that depend upon the geometry of its mechanical structure, termed *kinematics*, and those that depend on forces that act on the system, known as *kinetics*. It is a law of dynamics that the difference between the change in energy of the moving robot and the work performed by the forces acting on it does not change over small variations of its trajectory. This is called the *principle of virtual work* and states that variations in work and energy must cancel for all virtual displacements [4.1, 2].

Because machines such as robots are designed to minimize energy losses, often due to joint friction and material strain losses, we can assume that the variation in energy is small. This means that the work input of the actuators is nearly equal to the work of the output forces.

If we consider this relationship over a small duration of time, we have that the time rate of input work, or input power is nearly equal to the associated output power. Because power is force times velocity, we obtain the fundamental relationship that the ratio of input to output forces is the reciprocal of the ratio of input to output speeds. Another way of saying this is that, in the ideal machine, *the mechanical advantage of a machine is the inverse of its speed ratio.*

4.3.1 Robot Topology

The kinematic skeleton of a robot is modeled as a series of links connected by either hinged or sliding joints forming a serial chain. This skeleton has two basic forms, that of a single serial chain called a *serial robot* (Fig. 4.1) and as a set of serial chains supporting a single end-effector, called a *parallel robot*, such as the platform shown in Fig. 4.2. Robots can be configured to work in parallel such as the individual legs of walking machines (Figs. 4.3 and 4.4) [4.3], as

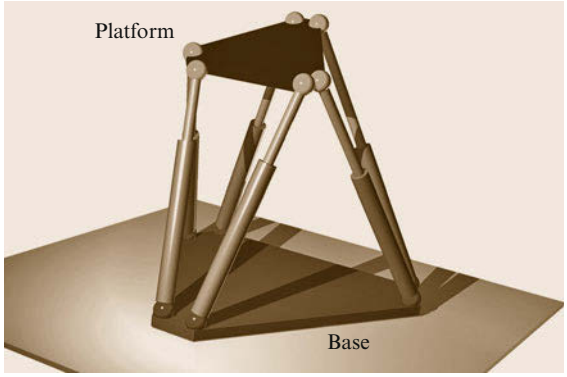


Fig. 4.2 A parallel robot can have as many as six serial chains that connect a platform to the base frame (VIDEO 640)

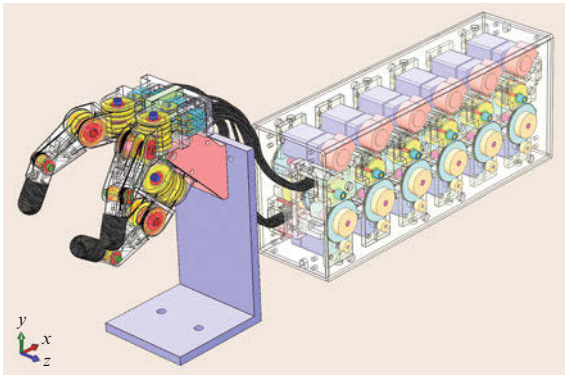


Fig. 4.5 The Salisbury three-fingered robot hand with its cable drive system

Fig. 4.3 A photograph of the adaptive suspension vehicle (ASV)walking machine

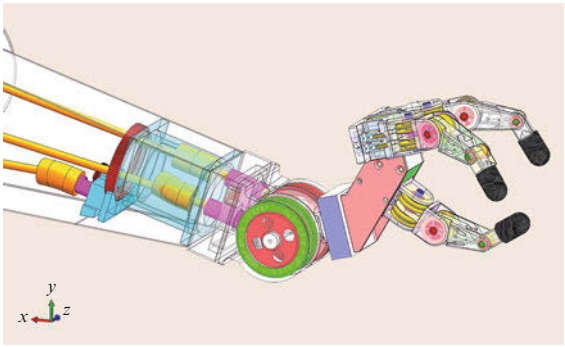


Fig. 4.6 The Salisbury hand as the end-effector of a PUMA robot (the drive system is not shown)

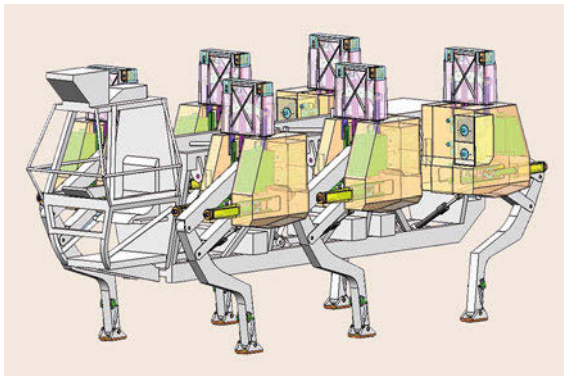


Fig. 4.4 The adaptive suspension vehicle walking machine

Fig. 4.7 A photograph of the Salisbury three-fingered hand grasping a block

well as the fingers of mechanical hands (Figs. 4.5–4.7 and VIDEO 642) [4.4].

The robot end-effector is the preferred tool for interaction with the environment, and the ability to position and orient this end-effector is defined by the skeleton of the robot. In a general serial robot, a chain of six joints provides full control over the end-effector. In a general parallel robot, there are more than six joints and the six actuators may be applied to these joints in a variety of ways to control the movement of the end-effector.

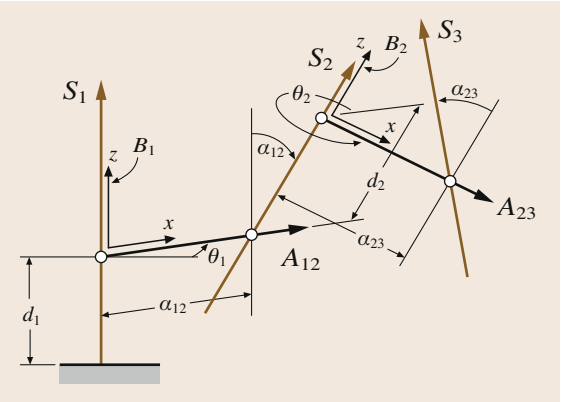


Fig. 4.8 The kinematic skeleton defined by the join axes and their common normals

4.3.2 Kinematics Equations

A robot is designed so that specifying the values of local joint parameters, such as the angle of rotary joints and the travel of sliding joints, specifies the position of every component of a machine using its kinematics equations. To do this the robot is described by a sequence of lines representing the axes $\hat{z}_j$ of equivalent revolute or prismatic joints and the common normal lines $\hat{x}_j$ which form the kinematic skeleton of the chain (Fig. 4.8). This construction allows the specification of the location of each link of the robot relative to the base by the matrix equation

$$\mathbf{T} = \mathbf{Z}(\theta_1, d_1)\mathbf{X}(\alpha_1, a_1)\mathbf{Z}(\theta_2, d_2)\ldots \times \mathbf{X}(\alpha_{m-1}, a_{m-1})\mathbf{Z}(\theta_m, d_m)\,, \tag{4.1}$$

known as the *kinematics equations* of the chain [4.5, 6] (Chap. 2; (2.46)). The set of all positions $\mathbf{T}$ that the end-effector can reach for all values of the joint parameters is called the *workspace* of the robot.

The matrices $\mathbf{Z}(\theta_j, d_j)$ and $\mathbf{X}(\alpha_j, a_j)$ are 4×4 matrices that define screw displacements around and along the joint axes $\hat{z}_j$ and $\hat{x}_j$, respectively [4.7]. The parameters α_j and a_j define the dimensions of the links in the chain. The parameter θ_j is the joint variable for revolute joints and d_j is the variable for prismatic joints. The trajectory ${}^F\boldsymbol{p}(t)$ of a point ${}^M\boldsymbol{p}$ in the end-effector is obtained from the joint trajectory,

$$\boldsymbol{q}(t) = (q_1(t), \ldots, q_m(t))^{\mathrm{T}}\,,$$

where q_i is either θ_i or d_i depending on the joint, given by

$${}^F\boldsymbol{p}(t) = \mathbf{T}(\boldsymbol{q}(t)){}^M\boldsymbol{p}\,. \tag{4.2}$$

If the end-effector is connected to the base frame by more than one serial chain (Fig. 4.2) then we have a set of kinematics equations for each chain,

$$\mathbf{T} = \mathbf{B}_j\mathbf{T}(\boldsymbol{q}_j)\mathbf{E}_j, \quad j = 1, \ldots, n\,, \tag{4.3}$$

where $\mathbf{B}_j$ locates the base of the j-th chain and $\mathbf{E}_j$ defines the position of its attachment to the part, or end-effector. The set of positions $\mathbf{T}$ that simultaneously satisfy all of these equations is the workspace of the part. This imposes constraints on the joint variables that must be determined to define its workspace completely [4.8, 9].

4.3.3 Configuration Space

The kinematics equations of the robot relate the range of values available for the joint parameters, called the configuration space of the robot, to the workspace of the end-effector. This configuration space is a fundamental tool in robot path planning for obstacle avoidance [4.10]. Though any link in the chain forming a robot may hit an obstacle, it is the end-effector that is intended to approach and move around obstacles such as the table supporting the robot and the fixtures for parts it is to pick up. Obstacles define forbidden positions and orientations in the workspace which map back to forbidden joint angles in the configuration space of the robot. Robot path planners seek trajectories to a goal position through the free space around these joint space obstacles [4.11].

4.3.4 Speed Ratios

The speed ratios of a robot relate the velocity ${}^F\dot{\boldsymbol{p}}$ of a point ${}^F\boldsymbol{p}$ in the end-effector to the joint rates

$$\dot{\boldsymbol{q}} = (\dot{q}_1, \ldots, \dot{q}_m)^{\mathrm{T}}\,,$$

that is

$${}^F\dot{\boldsymbol{p}} = \boldsymbol{v} + \boldsymbol{\omega} \times \left({}^F\boldsymbol{p} - \boldsymbol{d}\right), \tag{4.4}$$

where $\boldsymbol{d}$ and $\boldsymbol{v}$ are the position and velocity of a reference point, respectively, and $\boldsymbol{\omega}$ is the angular velocity of the end-effector.

The vectors $\boldsymbol{v}$ and $\boldsymbol{\omega}$ depend on the joint rates $\dot{q}_j$ through the formula

$$\begin{pmatrix}\boldsymbol{v}\\ \boldsymbol{\omega}\end{pmatrix} = \begin{pmatrix}\frac{\partial \boldsymbol{v}}{\partial \dot{q}_1} & \frac{\partial \boldsymbol{v}}{\partial \dot{q}_2} & \cdots & \frac{\partial \boldsymbol{v}}{\partial \dot{q}_m}\\ \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_1} & \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_2} & \cdots & \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_m}\end{pmatrix}\begin{pmatrix}\dot{q}_1\\ \vdots\\ \dot{q}_m\end{pmatrix}, \tag{4.5}$$

or

$$\mathbf{v} = \mathbf{J}\dot{\boldsymbol{q}} \,. \tag{4.6}$$

The coefficient matrix $\mathbf{J}$ in this equation is called the *Jacobian* and is a matrix of speed ratios relating the velocity of the tool to the input joint rotation rates [4.6, 9].

4.3.5 Mechanical Advantage

If the end-effector of the robot exerts a force $\boldsymbol{f}$ at the point ${}^F\boldsymbol{p}$, then the power output is

$$P_{\text{out}} = \boldsymbol{f} \cdot {}^F\dot{\boldsymbol{p}} = \sum_{j=1}^{m} \boldsymbol{f} \cdot \left[\frac{\partial \boldsymbol{v}}{\partial \dot{q}_j} + \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_j} \times \left({}^F\boldsymbol{p} - \boldsymbol{d}\right) \right] \dot{q}_j \,. \tag{4.7}$$

Each term in this sum is the portion of the output power that can be associated with an actuator at joint S_j, if one exists.

The power input at joint S_j is the product $\tau_j \dot{q}_j$ of the torque τ_j and joint angular velocity $\dot{q}_j$. Using the principle of virtual work for each joint we can compute

$$\tau_j = \boldsymbol{f} \cdot \frac{\partial \boldsymbol{v}}{\partial \dot{q}_j} + \left({}^F\boldsymbol{p} - \boldsymbol{d}\right) \times \boldsymbol{f} \cdot \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_j}, \quad j = 1, \ldots, m \,. \tag{4.8}$$

We have arranged this equation to introduce the force-torque vector

$$\boldsymbol{f} = (\boldsymbol{f}, ({}^F\boldsymbol{p} - \boldsymbol{d}) \times \boldsymbol{f})^{\mathrm{T}}$$

at the reference point $\boldsymbol{d}$.

The equations (4.8) can be assembled into the matrix equation

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}} \boldsymbol{f} \,, \tag{4.9}$$

where $\mathbf{J}$ is the *Jacobian* defined above in (4.5). For a chain with six joints this equation can be solved for the output force-torque vector $\boldsymbol{f}$,

$$\boldsymbol{f} = \left(\mathbf{J}^{\mathrm{T}}\right)^{-1} \boldsymbol{\tau} \,. \tag{4.10}$$

Thus, the matrix that defines the mechanical advantage for this system is the inverse of the matrix of speed ratios.

4.4 Serial Robots

A serial chain robot is a sequence of links and joints that begins at a base and ends with an end-effector (Fig. 4.9). The links and joints of a robot are often configured to provide separate translation and orientation structures. Usually, the first three joints are used to position a reference point in space and the last three form the *wrist* which orients the end-effector around this point [4.12, 13]. This reference point is called the *wrist center*. The volume of space in which the wrist center can be placed is called the *reachable workspace* of the robot. The rotations available at each of these points is called the *dexterous workspace*.

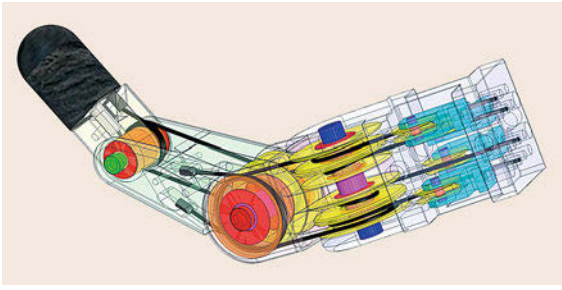


Fig. 4.9 A single finger of the Salisbury hand is a serial chain robot

The design of a robot is often based on the symmetry of its reachable workspace. From this point of view there are three basic shapes: rectangular, cylindrical, and spherical [4.6].A rectangular workspace is provided by three mutually perpendicular prismatic (P) joints which form a PPPS chain called a *Cartesian* robot – S denotes a spherical wrist which allows all rotations about its center point. A rotary base joint combined with two prismatic joints forms a CPS chain with a cylindrical workspace – C denotes a rotary (R) and sliding (P) joint with the same axis. The P-joint can be replaced by a revolute (R) joint that acts as an elbow in order to provide the same radial movement. Finally, two rotary joints at right angles form a T-joint at the base of the robot that supports rotations about a vertical and horizontal axes. Radial movement is provided either by a P-joint, or by an R-joint configured as an elbow. The result is a TPS or TRS chain with a spherical workspace.

It is rare that the workspace is completely symmetrical because joint axes are often offset to avoid link collisions and there are limits to joint travel which combine to distort the shape of the workspace.

4.4.1 Design Optimization

Another approach to robot design uses a direct specification of the workspace as a set of positions for the end-effector of a robotic system [4.14–17] which we call the *task space*. A general serial robot arm has two design parameters, link offset and twist, for each of five links combined with four parameters each that locate the base of the robot and the workpiece in its end-effector, making a total of 18 design variables. The link parameters are often specified so the chain has a spherical wrist and specific workspace shape. The design goal is usually to determine the workspace volume and locate the base and workpiece frames so that the workspace encloses the specified task space.

The task space is defined by a set of 4×4 transformations $\mathbf{D}_i, i=1,\ldots,k$. The problem is solved iteratively by selecting a design and using the associated kinematics equations $\mathbf{T}(\boldsymbol{q})$ to evaluate relative displacements in the objective function

$$f(\boldsymbol{r}) = \sum_{i=1}^{k} \left\| \mathbf{D}_i \mathbf{T}^{-1}(\boldsymbol{q}_i) \right\| \ . \tag{4.11}$$

Optimization techniques yield the design parameter vector $\boldsymbol{r}$ that minimizes this objective function.

This optimization depends on the definition of the distance measure between the positions reached by the end-effector and the desired workspace. *Park* [4.18], *Martinez* and *Duffy* [4.19], *Zefran* et al. [4.20], *Lin* and *Burdick* [4.21], and others have shown that there is no distance metric that is coordinate frame invariant. This means that, unless this objective function can be forced to zero so the workspace completely contains the task space, the resulting design will not be *geometric* in the sense that the design is not independent of the choice of coordinates.

4.4.2 Speed Ratios

A six-axis robot has a 6×6 Jacobian $\mathbf{J}$ obtained from (4.5) that is an array of speed ratios relating the components of the velocity $\boldsymbol{v}$ of the wrist center and the angular velocity $\boldsymbol{\omega}$ of the end-effector to each of the joint velocities. Equation (4.9) shows that this Jacobian defines the force-torque vector $\boldsymbol{f}$ exerted at the wrist center in terms of the torque applied by each of the actuators. The link parameters of the robot can be selected to provide a Jacobian $\mathbf{J}$ with specific properties.

The sum of the squares of the actuator torques of a robot is often used as a measure of *effort* [4.22, 23]. From (4.9) we have

$$\boldsymbol{\tau}^{\mathrm{T}}\boldsymbol{\tau} = \boldsymbol{f}^{\mathrm{T}}\mathbf{J}\mathbf{J}^{\mathrm{T}}\boldsymbol{f} \ . \tag{4.12}$$

The matrix $\mathbf{J}\mathbf{J}^{\mathrm{T}}$ is square and positive definite. Therefore, it can be viewed as defining a hyperellipsoid in six-dimensional space [4.24]. The lengths of the semi-diameters of this ellipsoid are the inverse of the absolute value of the eigenvalues of the Jacobian $\mathbf{J}$. These eigenvalues may be viewed as *modal* speed ratios that define the amplification associated with each joint velocity. Their reciprocals are the associated *modal* mechanical advantages, so the shape of this ellipsoid illustrates the force amplification properties of the robot.

The ratio of the largest of these eigenvalues to the smallest, called the condition number, gives a measure of the anisotropy or *out-of-roundness* of the ellipsoid. A sphere has a condition number of one and is termed *isotropic*. When the end-effector of a robot is in a position with an isotropic Jacobian there is no amplification of the speed ratios or mechanical advantage. This is considered to provide high-fidelity coupling between the input and output because errors are not amplified [4.25, 26]. Thus, the condition number is used as a criterion in a robot design [4.27].

In this case, it is assumed that the basic design of the robot provides a workspace that includes the task space. Parameter optimization finds the internal link parameters that yield the desired properties for the Jacobian. As in minimizing the distance to a desired workspace, optimization based on the Jacobian depends on a careful formulation to avoid coordinate dependency.

4.5 Parallel Robots

A robotic system in which two or more serial chain robots support an end-effector is called a *parallel robot*. For example, the adaptive suspension vehicle (ASV) leg (Fig. 4.10), is a pantograph mechanism driven by parallel actuation. Each supporting chain of a parallel robot may have as manyas six degrees of freedom, however, in general only a total of six joints in the entire system are actuated. A good example is the Stewart platform formed from six TPS robots in which usually only the prismatic joint (P-joint) in each chain is actuated (Fig. 4.2) [4.9, 28, 29].

The kinematics equations of the TPS legs are

$$\mathbf{T} = \mathbf{B}_j \mathbf{T}(\boldsymbol{\theta}_j)\mathbf{E}_j, \quad j=1,\ldots,6 \ , \tag{4.13}$$

where $\mathbf{B}_j$ locates the base of the leg and $\mathbf{E}_j$ defines the position of its attachment to the end-effector. The set of positions $\mathbf{T}$ that simultaneously satisfy all of these equations is the workspace of the parallel robot.

Often the workspace of an individual chain of a parallel robot can be defined by geometric constraints, for example, a position $\mathbf{T}$ is in the workspace of the j-th supporting TPS chain if it satisfies the constraint equation

$$(\mathbf{T}\boldsymbol{x}_j - \boldsymbol{p}_j) \cdot (\mathbf{T}\boldsymbol{x}_j - \boldsymbol{p}_j) = \rho_j^2 \,. \tag{4.14}$$

This equation defines the distance between the base joint $\boldsymbol{p}_j$ and the point of attachment ${}^F\boldsymbol{x}_j = \mathbf{T}\boldsymbol{x}_j$ to the platform as the length ρ_j is controlled by the actuated prismatic joint. In this case the workspace is the set of positions $\mathbf{T}$ that satisfy all six equations, one for each leg.

4.5.1 Workspace

The workspace of a parallel robot is the intersection of the workspaces of the individual supporting chains. However, it is not the intersection of the reachable and dexterous workspaces separately. These workspaces are intimately combined in parallel robots. The dexterous workspace is usually largest near the center of the reachable workspace and shrinks as the reference point moves toward the edge. A focus on the symmetry of movement allowed by supporting leg designs has been an important design tool resulting in many novel parallel designs [4.30, 31]. Simulation of the system is used to evaluate its workspace in terms of design parameters.

Another approach is to specify directly the positions and orientations that are to lie in the workspace and solve the algebraic equations that define the leg constraints to determine the design parameters [4.32, 33]. This is known as kinematic synthesis and yields parallel robots that are asymmetric but have specified reachable and dexterous workspaces, see *McCarthy* and *Soh* [4.34].

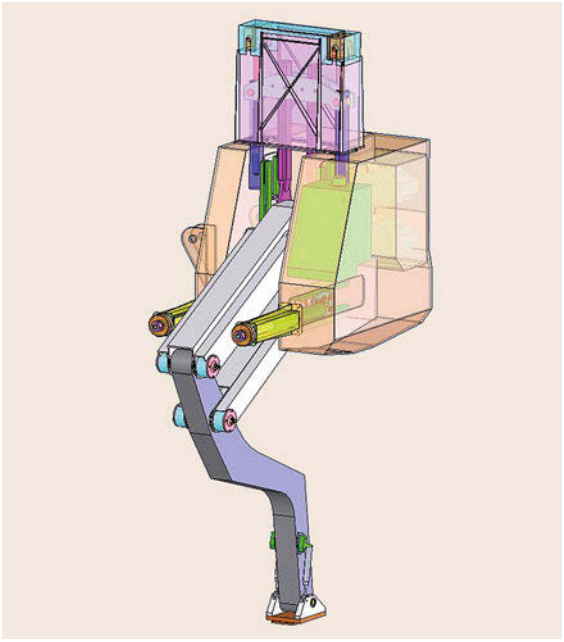


Fig. 4.10 One leg of the ASV walking machine is a parallel robot

4.5.2 Mechanical Advantage

The force amplification properties of a parallel robot are obtained by considering the Jacobians of the individual supporting chains. Let the linear and angular velocity of the platform be defined by the six-vector $\mathbf{v} = (\boldsymbol{v}, \boldsymbol{\omega})^{\mathrm{T}}$, then from the kinematics equations of each of the support legs we have

$$\mathbf{v} = \mathbf{J}_1\dot{\boldsymbol{\rho}}_1 = \mathbf{J}_2\dot{\boldsymbol{\rho}}_2 = \cdots = \mathbf{J}_6\dot{\boldsymbol{\rho}}_6 \,. \tag{4.15}$$

Here we assume that the platform is supported by six chains, but it can be fewer, such as when the fingers of a mechanical hand grasp an object [4.4].

The force on the platform applied by each chain is obtained from the principle of virtual work as

$$\boldsymbol{f}_j = \left(\mathbf{J}_j^{\mathrm{T}}\right)^{-1}\boldsymbol{\tau}_j, \quad j = 1, \ldots, 6 \,. \tag{4.16}$$

There are only six actuated joints in the system so we assemble the associated joint torques into the vector $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_6)^{\mathrm{T}}$. If $\boldsymbol{f}_i$ is the force-torque vector obtained from (4.16) for $\tau_i = 1$ and the remaining torques to zero, then the resultant force-torque $\mathbf{w}$ applied to the platform is

$$\mathbf{w} = (\boldsymbol{f}_1, \boldsymbol{f}_2, \cdots, \boldsymbol{f}_6)\boldsymbol{\tau} \,, \tag{4.17}$$

or

$$\mathbf{w} = \boldsymbol{\Gamma}\boldsymbol{\tau} \,. \tag{4.18}$$

The elements of the coefficient matrix $\boldsymbol{\Gamma}$ define the mechanical advantage for each of the actuated joints. In the case of a Stewart platform the columns of this matrix are the Plücker coordinates of the lines along each leg [4.29].

The principle of virtual work yields the velocity of the platform in terms of the joints rates $\dot{\boldsymbol{\rho}}$ as

$$\boldsymbol{\Gamma}^{\mathrm{T}}\mathbf{v} = \dot{\boldsymbol{\rho}} \,. \tag{4.19}$$

Thus, the inverse of $\mathbf{\Gamma}$ defines the speed ratios between the actuated joints and the end-effector. The same equation can be obtained by computing the derivative of the geometric constraint equations (4.14), and $\mathbf{\Gamma}$ is the *Jacobian* of the parallel robot system [4.35].

The Jacobian $\mathbf{\Gamma}$ is used in parameter optimization algorithms to design parallel robots [4.36] with isotropic mechanical advantage. The square root of the determinant $\left|\mathbf{\Gamma}\mathbf{\Gamma}^{\mathrm{T}}\right|$ measures the six-dimensional volume spanned by the column vectors $\boldsymbol{f}_j$. The distribution of the percentage of this volume compared to its maximum within the workspace is also used as a measure of the overall performance [4.37, 38]. A similar performance measure normalizes this Jacobian by the maximum joint torques available and the maximum component of force and torque desired, and then seeks an isotropic design [4.39].

4.5.3 Specialized Parallel Robots

Another approach to the design of parallel robots has been to separate their functionality into orientation and translational platforms. *Tsai* and *Joshi* [4.40] and *Jin* and *Yang* [4.41] survey designs for a class of parallel chains that generate pure translation. *Kong* and *Gosselin* [4.42] and *Hess-Coelho* [4.43] do the same for parallel chains that provide rotational movement in space.

4.6 Mechanical Structure

For the purposes of dynamic modeling, the links of a robot are generally considered to be rigid. However, a robot is not a rigid structure. Like all structures it deflects under applied loads, such as its own weight and the weight of the payload, termed gravity loading; see Figs. 4.11 and 4.12. The issue is a matter of degree. The more force that is needed to cause a deflection in the links, the more the robot moves like a connected set of rigid bodies. Rigid robots have links designed to be stiff so the deflections under load are less than the positioning accuracy required for their range of tasks. This allows the dynamic model and control algorithms to ignore link deflection. Most commercially available robot arms are of this type (*Rivin* [4.44]).

It is possible to improve the positioning accuracy of a rigid robot by augmenting a control algorithm that includes a model of link deflection resulting from gravity loading. It is also possible to use strain sensors to measure loads and deflections. These *semirigid* robots assume small structural deflections that are linearly related to known applied loads.

Flexible robots require that the dynamic model include the deflection of its various links under gravity loading as well as under the forces associated with link acceleration, called inertia loading. The robot control algorithms must control the vibration of the system as well its gross motion. Management of vibration may be required even in rigid robots to reduce cycle time when subjected to high dynamic loading at high speeds or when manipulating large payloads.

4.6.1 Links

For industrial robots a critical concern is the link stiffness in bending and in torsion. To provide this stiffness, robot links are designed either as beams or shell (monocoque) structures. Monocoque structures have lower weight or higher strength-to-weight ratios, but are more costly and generally more difficult to manufacture. Cast, extruded, or machined beam-based links are often more cost effective; *Juvinall* and *Marshek* [4.45], and *Shigley* and *Mischke* [4.46, 47]

Another important consideration is whether the link structure includes bolted, welded, or adhesive bonded assemblies of cast, machined, and fabricated elements. Screw and bolted connections may seem straightforward, inexpensive, and easily maintained, but the inevitable deflection of a link even in the manufacturing process introduces creep in these multiple element assemblies that changes the dimensions and performance of the robot. Welded and cast structures are much less susceptible to creep and the associated hysteresis deformation, though in many cases they require secondary manufacturing operations such as thermal stress relieving and finish machining.

The minimum practical wall or web thickness for castings may be thicker than necessary for stiffness. Thin walls can be achieved with structural skin (mono-

Fig. 4.11 The hydraulic Skywash aircraft cleaning robot

Fig. 4.12 DeLaval Cow Milking System features hydraulic robot with machine vision guided positioning (VIDEO 643)

coque) structures but this is offset by the potential for denting, permanent deformation, and damage in the event of slight collisions. Therefore, both the performance and application requirements must be considered when selecting the construction and fabrication details of the robot.

Performance- and application-specific materials and geometry are used to reduce the weight of the links and therefore the associated gravity and inertial loading. For structures that move in a straight line, aluminum or magnesium alloy extrusions of constant cross section are convenient. Carbon and glass-fiber composites provide lower mass for robots that require high acceleration (painting robots). Thermoplastic materials provide low-cost link structures though at reduced load capacity. Stainless steel is often used in robots for medical and food service applications. The longer links on serial chain robots often are designed to taper in cross section or wall thickness to reduce the associated inertial loading.

4.6.2 Joints

Joints for most robots allow either rotary or linear movement, termed revolute and prismatic joints. Other joints that are available are the ball-in-socket, or spherical joint, and the Hooke-type universal joint.

Integration of the mechanical structure of the robot with its joint mechanism, which includes the actuator and joint motion sensor, is a source of structural flexibility. Precision is reduced when deformation in the joint at its bearings can reduce gear and shaft preloads, allowing undesirable backlash and free play. Structural flexibility can also introduce changes in gear center spacing, introducing forces and torques and associated deflection, binding, jamming, and wear.

4.7 Joint Mechanisms

A robot joint mechanism consists of at least four major components: the joint axis structure, an actuator, transmission, and state sensor (usually for position and velocity feedback, but force sensors are also common).

For low-performance manipulators that accelerate the payload at less than a peak of 0.5 g, system inertia is not as important as gravity forces and torques. This means the actuators can be placed near the joints, and their suspended weight compensated by using counterbalancing masses, springs, or gas pressure.

In high-performance robots where peak payload accelerations reach 3–10 g or more, minimizing system inertia is important. The actuators are placed near the first joint axis of a serial link manipulator to minimize its inertial contribution, and drive linkages, belts, cables or spaced gear stages are used to drive the joints.

While a longer transmission distance can reduce mass and gravity moments and inertia, it introduces flexibility and thus reduces the system stiffness. The design of the actuator placement and transmission for each joint is a trade-off between weight, inertia, stiffness, and complexity. This choice dictates the major physical characteristics of a manipulator design. To illustrate this point, consider the Adept 1 assembly robot with four degrees of freedom, each a different structure. The first axis has a direct motor drive. The second axis is driven by a steel band drive, and the third by a synchronous belt drive. Finally, the fourth axis uses a linear ball–screw drive. For a variety of useful joint mechanisms *Sclater* and *Chironis* [4.48].

4.7.1 Joint Axis Structures

Revolute Joints

Revolute or rotary motion joints are designed to perform pure rotation while minimizing other displacements and motions. The most important measure of the

quality of a revolute joint is its stiffness or resistance to all undesired motion. Key factors to be considered in design for stiffness are shaft diameter, mounting configuration and preloading of the bearings and proper clearances and tolerances. Shaft diameter and bearing size are not always based on load-carrying capacity; rather, they often will be selected to be compatible with a rigid mounting configuration and also have a bore large enough to pass cables, hoses, and even drive elements for other joints. Because joint shafts will frequently be torque-transmitting members, they and their supporting structure must be designed both for bending and torsional stiffness. The first axis of the PUMA robot is an example of such a joint with its large-diameter tubular configuration.

An important factor in maintaining stiffness in a revolute joint is the choice of bearing-mounting configuration. The mounting arrangement and mount must be designed to accommodate manufacturing tolerances, thermal expansion and bearing preload. Axial preloading of ball or tapered roller bearings improves system accuracy and stiffness by minimizing both axial and radial play. Preloads can be achieved through selective assembly or elastic (spring) elements, shim spacers, threaded collars, four-point contact bearings, duplex bearing arrangements, and tight manufacturing tolerances.

Prismatic Joints

There are two basic types of prismatic or linear motion joints: single-stage and telescoping joints. Often a prismatic joint and its associated link and actuator are combined as a linear actuator. Single-stage joints are made up of a moving surface that slides linearly along a fixed surface. Telescoping joints are essentially sets of nested or stacked single-stage joints. Single-stage joints feature simplicity and high stiffness, whereas the primary advantage of telescoping joints is their retracted-state compactness and large extension ratio. Telescoping joints have a lower effective joint inertia for some configurations and motions because part of the joint may remain stationary or move with reduced acceleration.

The primary functions of bearings in prismatic joints are to facilitate motion in a single direction and to prevent motion in all other directions. Preventing these unwanted motions poses the more challenging design problem. Deformations in the structure can distort the bearing. In severe cases, ball or roller deflection under load may cause binding, which precludes motion. For high-precision prismatic joints, ways must be straight and precise along their entire length which may be several meters. The required precision grinding on multiple surfaces can be costly. Bulky covers may be required to shield and seal a prismatic bearing and way assembly.

The primary criterion for evaluating higher number (in or near the wrist or end-effector) linear motion joints or axes is the stiffness-to-weight ratio. Achieving a good stiffness-to-weight ratio requires the use of a hollow or thin-walled structure rather than solid members for the moving elements.

Bearing spacing is extremely important in design for stiffness. If this spacing is too short, system stiffness will be inadequate no matter how great the bearing stiffness. A major cause for failure in prismatic joints is surface fatigue wear (brinelling) of the ways caused by excessive ball loading due to high preload, moment loads and shock loads.

The large exposed precision surfaces in most prismatic joints make them much more sensitive than revolute joints to contamination, improper handling and environmental effects. They are also significantly more difficult to manufacture, properly assemble, and align.

Common types of sliding elements for prismatic motion are bronze or thermoplastic impregnated bushings. These bushings have the advantage of being low in cost, of having relatively high load capacity, and of working with unhardened or superficially hardened (i. e., plated or anodized) surfaces. Because the local or contact stress on the moving element is distributed and is low this element may be made of thin tube or an extruded shape. Another type of bushing in common use is the ball bushing. Ball bushings have the advantages of lower friction and greater precision than plain bushings. However, they require that the contacting surface of the joint be heat treated or hardened (generally to Rc 55 or greater) and of sufficient case and wall thickness to support the ball point contact loads and resulting high stresses.

Ball and roller slide assemblies are commonly used in robot prismatic joints. There are two basic categories of these slides; recirculating and non-recirculating. Non-recirculating ball and roller slides are used primarily for short-travel applications. They feature high precision and very low friction at the expense of being more sensitive to shock and relatively poor at accommodating moment loading. Recirculating ball slides are somewhat less precise but can carry higher loads than non-recirculating ball slides. They can also be set up to handle relatively large moment loads. Travel range can be up to several meters. Commercial recirculating ball slides and ways have greatly simplified the design and construction of linear axes, particularly in gantry and track mounted manipulators.

Another common type of prismatic robot joint is made up of cam followers, rollers, or wheels rolling on extruded, drawn, machined, or ground surfaces. In high-load applications the surfaces must be hardened before they are finish ground. Cam followers can be purchased

Fig. 4.13 RobotWorld is an integrated workcell with multiple robot modules that move on air bearings

with eccentric mounting shafts to facilitate setup and adjustment. Elastomer rollers provide quiet, smooth operation.

Two less common types of linear or prismatic joints feature flexures and air bearings. Flexure-based joints, whose motion results from elastic bending deformations of beam support elements, are used primarily for small, high-resolution, quasilinear motions. Air bearings require smooth surfaces and close control of tolerances as well as a constant supply of filtered, oil-free compressed air. Two- and three-degree-of-freedom air bearings (x, y, θ) can enable multiaxis motion with few moving parts (Fig. 4.13).

Joint Travel

For revolute joint configurations, the shoulder and elbow joints and links determine the gross volume of the work envelope (reachable workspace) of a robot manipulator arm. The wrist joints generally determine the orientation range (dexterous workspace) about a point within this work envelope. Larger joint travel may increase the number of possible manipulator configurations that will reach a particular location (increased task space). Wrist joint travel in excess of 360° and up to 720° can be useful for situations requiring controlled-path (e.g., straight-line) motion, synchronized motion such as small part assembly on a moving conveyor belt, or sensor-modified motions as in using machine vision to select and guide picking jumbled items from a bin. Continuous last-joint rotation is desirable in certain cases like loading or unloading a rotating machine or mating threaded parts.

Fig. 4.14 Wrist for surgery has no singularities within range of motion. Features tungsten cable actuation (after [4.49])

Additional joints and links, sometimes in the robot but more often in the end-effector, and specialized tooling also serve to increase the task space of a robot. Continuous and controlled-path robot motion requires planning to avoid singularities (regions where two or more joints may become aligned or nearly aligned) and the resulting unstable end-effector motion in these regions. Manipulator design coordinated with a well-planned workcell layout can improve the useful task space by placing critical motions well away from singularity regions. For example, a standard three-axis robot wrist has singularities 180° apart, which can be increased to 360° by implementing a somewhat more complex reduced singularity wrist. Such a wrist is used in a sheep-shearing robot to achieve multiple, long, continuous, smooth, constant-velocity, sensor-guided passes over the contoured body of a sheep to shear its wool [4.50, 51]. Figure 4.14 shows a wrist used for minimally-invasive surgery that has no singularities within its range of motion [4.49].

4.8 Actuators

Actuators supply the motive power for robots. Most robot actuators are commercially available components, which are adapted or modified, as necessary, for a specific robot application. Three commonly used actuator types are electromagnetic, hydraulic and pneumatic.

4.8.1 Electromagnetic Actuators

The most common types of actuators in robots today are electromagnetic actuators.

Electric Servomotors

Most robot manipulators use servomotors as a power source. Servomotors are designed to accurately follow the desired position, velocity and torque which change frequently and sometimes abruptly. They have structures similar to ordinary electric motors, but with low inertia and large torque capably for high accelerations. Typical servomotors used for robotic applications are permanent magnet (PM) DC motors and brushless DC (BLDC) motors.

PM DC motors are widely used as a servomotor because of high torque, speed controllability over a wide range, well-behaved torque-speed characteristics, and adaptability to various types of control methods. The DC motor converts electrical energy into rotational or linear mechanical energy. It comes in many different types and configurations. The lowest-cost PM motors use ceramic (ferrite) magnets and robot toys and hobby robots often use this type of motor. A PM motor with a rare-earth (neodymium-iron-boron), NEO magnet stator produces the most torque and power for its size.

Brushless motors, also called AC servomotors or brushless DC motors, are widely used in industrial robots (Figs. 4.15 and 4.16). They substitute magnetic or optical sensors and electronic switching circuitry for the graphite brushes and copper bar commutator used in brush-type DC motors, thus eliminating the friction, spark-ing, and wear of commutating parts. Brushless motors generally have good performance at low cost because of the decreased complexity of the motor. However, the controllers for these motors are more complex and expensive than brush-type motor controllers. A passive multi-pole neodymium magnet rotor and a wire-wound iron stator of a brushless motor provide good heat dissipation and excellent reliability. Linear brushless motors function like unrolled rotary motors. They typically have a long, heavy, multiple magnet passive stator and a short, lightweight, electronically commutated wire-wound forcer (slider).

Fig. 4.15 The Baldor AC servomotor

Fig. 4.16 The Anorad brushless linear motor

Modeling of DC Motors and BLDC Motors. Both DC and BLDC motors can be described by almost identical equations although they have different structures, *de Silva* [4.52]. Fig. 4.17. The motor torque τ_m generated at the motor is given by,

$$\tau_m = K_t i_a \,, \tag{4.20}$$

where K_t is the torque constant [N m/A] and i_a is the armature current [A].

When the rotor is rotating, the back electromotive force (emf) v_b is induced as follows,

$$v_b = K_b \omega_m \,, \tag{4.21}$$

where K_b is the back emf constant [V/(rad/s)] and ω_m is the angular speed of the motor [rad/s].

The circuit equation is given by

$$v_a = R_a\, i_a + L_a \frac{\mathrm{d}i_a}{\mathrm{d}t} + v_b \,, \tag{4.22}$$

where v_a is the armature voltage (supply voltage to the armature) [V], R_a and L_a are the resistance [Ω] and inductance [H] of the armature winding, respectively.

Lastly, the mechanical equation is described by

$$J\frac{\mathrm{d}\omega_m}{\mathrm{d}t} + B\omega_m = \tau_m - \frac{\tau_l}{r} \,, \tag{4.23}$$

where J is the equivalent moment of inertia referred to the motor shaft [kg m^2], B is the equivalent viscous-friction coefficient referred to the motor shaft [N m/rad/s], τ_l is the load torque [N m], and r is the gear ratio. The quantities J and B are given by

$$J = J_m + \frac{J_l}{r^2}, \quad B = B_m + \frac{B_l}{r^2} \,, \tag{4.24}$$

where J_m and B_m are related to the motor including the gear connected to the motor shaft and J_l and B_l are referred to the load including the gear connected to the load shaft.

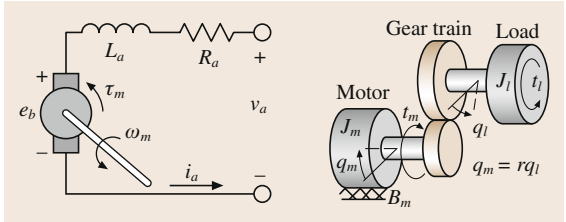


Fig. 4.17 Servomotor models

The four equations above can be represented as the block diagram in Fig. 4.18.

In the servomotors, the electrical time constant $\tau_e = L_a/R_a$ and mechanical time constant $\tau_m = J/B$ have effects on their performance. The electrical time constant is much smaller than mechanical time constant because of the armature inductance is usually negligible. Considering this factor, the angular speed of the motor is described in terms of the armature voltage v_a and load torque τ_l as follows

$$\omega_m(s) = \frac{1}{JR_a\, s + (R_a\, B + K_t K_b)} \left(K_t v_a(s) - R_a \frac{\tau_l(s)}{r} \right). \tag{4.25}$$

The above equation is frequently used in simulations of both DC motor and BLDC motors.

Stepper Motors. Small, simple robots, such as benchtop adhesive dispensing robots, frequently use stepper or pulse motors of the permanent magnet (PM) hybrid type or sometimes the variable reluctance (VR) type (Fig. 4.19). These robots use open-loop position and velocity control. They are relatively low in cost and interface easily to electronic drive circuits. Microstep control can produce 10 000 or more discrete robot joint positions. In open-loop step mode the motors and robot motions have a significant settling time, which can be damped either mechanically or through the application of control algorithms. Power-to-weight ratios are lower for stepper motors than for other types of electric motors. Stepper motors operated with closed-loop control function similarly to direct-current (DC) or alternating-current (AC) servomotors (Fig. 4.20).

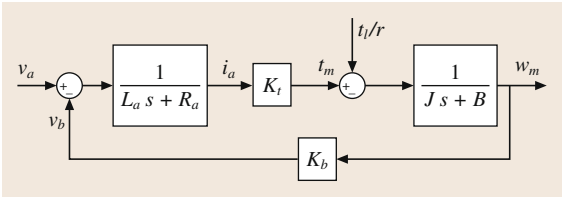


Fig. 4.18 Block diagram for servomotors

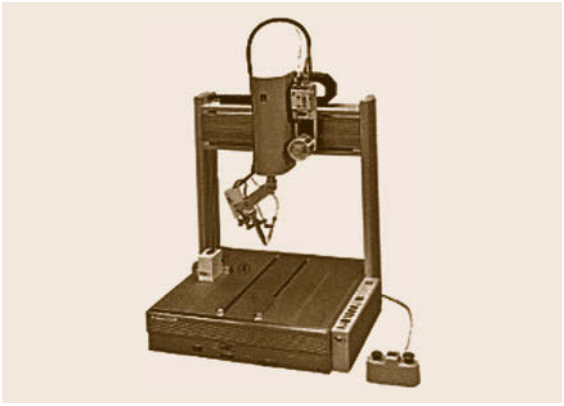


Fig. 4.19 The Sony robot uses open-loop permanent-magnet stepper motors

Permanent-Magnet DC Motor. The permanent-magnet, direct-current, brush-commutated motor is widely available and comes in many different types and configurations. The lowest-cost permanent-magnet motors use ceramic (ferrite) magnets. Robot toys and hobby robots often use this type of motor. Neodymium (NEO) magnet motors have the highest energy-product magnets, and in general produce the most torque and power for their size.

Ironless rotor motors, often used in small robots, typically have copper wire conductors molded into epoxy or composite cup or disk rotor structures. The advantages of these motors include low inductance, low friction, and no cogging torque. Disk armature motors have several advantages. They have short overall lengths, and because their rotors have many commutation segments, they produce a smooth output with low torque ripple. A disadvantage of ironless armature motors is that they have a low thermal capacity due to low mass and limited thermal paths to their case. As a result, when driven at high power levels they have rigid duty-cycle limitations or require forced-air cooling.

4.8.2 Hydraulic Actuators

Hydraulic actuators, chosen as power sources for the earliest industrial robots, offer very large force capability and high power-to-weight ratios. They convert hydraulic power into useful mechanical energy. The mechanical motion produced may be linear for hy-

Fig. 4.20 The Adept robot uses closed-loop control and variable-reluctance motors (VIDEO 644)

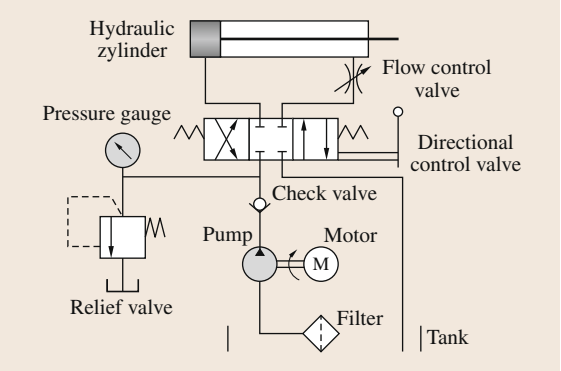


Fig. 4.21 General hydraulic circuit

draulic cylinders or rotary for hydraulic motors and vane actuators.

Figure 4.21 shows the basic hydraulic circuit composed of oil tank, hydraulic pump, check valve, relief valve, directional control valve and hydraulic actuator. Hydraulic fluid pressurized by the hydraulic pump is conveyed to the hydraulic machine to perform the specified task. To this end, the pressure, flow, and direction of hydraulic fluid is changed by the hydraulic control valves such as a pressure control valve, flow control valve, and directional control valve, respectively. The directional control valve changes the flow direction of the hydraulic fluid which is pressurized in the hydraulic pump and thus the direction of actuator motion. The flow control valve changes the flow rate of hydraulic fluid flow and thus the speed of the hydraulic actuator. For safety, the highest pressure is limited by the pressure control valve such as a relief valve.

Hydraulic actuators have several advantages and disadvantages due to the use of high pressure fluid. They can offer very large force or torque and high power-to-weight ratios. Both linear and rotary motions are readily available with small inertia of the moving part. However, the hydraulic power supply is bulky and the cost of the proportional, fast-response servovalves are high. Leaks and maintenance issues have limited the use and applica-tion of hydraulically powered robots.

There have been many applications of hydraulic actuators to robotics. For high forces or torques and speeds, hydraulic servo actuators out-perform current electromagnetic actuators. With proper design, leakage can be virtually eliminated, *Hollerbach* et al. [4.53]. Figure 4.22 shows some typical applications of hydraulic actuators, [4.54–56].

4.8.3 Pneumatic Actuators

Pneumatic actuators are similar to hydraulic actuators. Pneumatic actuators convert energy (in the form of compressed air) into mechanical motion, which may be linear or rotary. Pneumatic actuators are primarily found in simple manipulators. Typically they provide uncontrolled motion between mechanical limit stops. These actuators provide good performance in point-to-point motion. They are simple to control and are low in cost. Pneumatic motors have several advantages over electric motors. They are relatively safe in the explosive environment. They are also less affected by ambient temperature and humidity than electric motors. Although a few small actuators may be run with typical factory air supplies, extensive use of pneumatic-actuated robots requires the purchase and installation of a costly dedicated compressed-air source. Pneumatic actuators have low energy efficiency.

Pneumatic systems consist of pneumatic generator, pneumatic valves, pneumatic actuator and pipes. A pneumatic generator produces compressed air using an air compressor. Pneumatic valves are used to control the pressure, flow rate and direction. The mechanical motion produced may be linear for pneumatic cylinders or rotary for pneumatic motors.

Pneumatic actuators are not used for applications requiring large forces or torques since they produce less power than hydraulic actuators or electric actuators. However, they are used in robot hands or artificial muscles, which require high power-to-weight ratios. Pneumatic artificial muscles are contractile or extensional devices operated by pressurized air filling a pneumatic bladder. They are usually grouped in the agonist

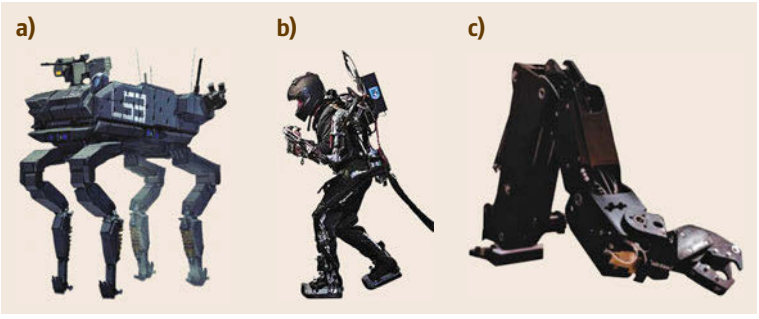


Fig.4.22a–c Applications of hydraulic actuators to robot: **(a)** BigDog (Boston dynamics) (VIDEO 645), **(b)** Sarcos exoskeleton (VIDEO 646) (Raytheon) **(c)** Magnum 7 (International Submarine Engineering)

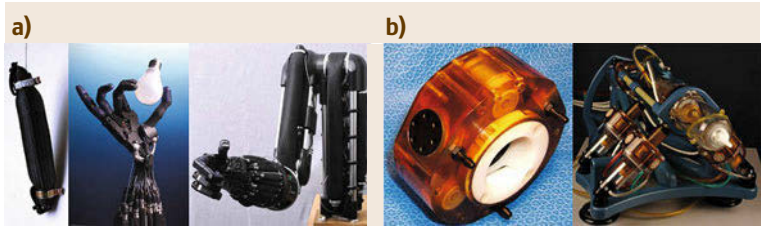


Fig.4.23a,b Applications of pneumatic actuator: **(a)** robot hand and arm with artificial muscle (Shadow robot) and **(b)** pneumatic step motor and MrBot (Urobotics, Johns Hopkins)

and antagonist pairs. Furthermore, they can be used for medical robots since they are not affected by magnetic field, and for robots in explosive environments because there is no electrical arcing that exists in electromagnetic actuators. Figure 4.23 shows some robotic applications of pneumatic actuators [4.57, 58].

4.8.4 Other Actuators

A wide variety of other types of actuators have been applied to robots. A sampling of these include, thermal, shape-memory alloy (SMA), bimetallic, chemical, piezoelectric, magnetostrictive, electroactive polymer (EAP), bladder, and micro-electromechanical system (MEMS) actuators (see Figs. 4.24 and 4.25). Most of these actuators have been applied to research and special application robots rather than volume production industrial robots. An example of a piezoelectric actuator powered robot is the six-axis PI (Physik Instrumente) piezo hexapod with sub-nanometer resolution shown in Fig. 4.26.

Fig. 4.24 The artificial muscle EAP motor

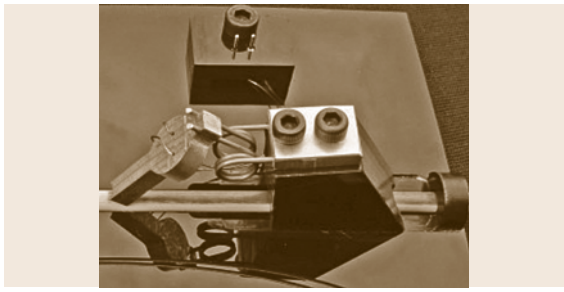


Fig. 4.25 The Elliptec piezoelectric motor

4.8.5 Transmissions

The purpose of a transmission or drive mechanism is to transfer mechanical power from a source to a load. The design and selection of a robot drive mechanism requires consideration of motion, load, and power requirements and the placement of the actuator with respect to the joint. The primary considerations in transmission design are stiffness, efficiency, and cost. Backlash and windup impact drive stiffness especially in robot applications where motion is constantly reversing and loading is highly variable. High transmission stiffness and low or no backlash result in increased friction losses. Most robot transmission elements have good efficiencies when they are operating at or near their rated power levels but not necessarily when lightly loaded. Larger than necessary drives add weight, inertia and friction loss to the system. Underdesigned drives have lower stiffness, can wear rapidly in continuous or in high duty cycle operation or fail due to accidental overloads.

Joint actuation in robots is generally performed by drive mechanisms which interface the actuator (mechanical work source) to the robot links through the joints in an energy-efficient manner. A variety of drive mechanisms are incorporated in practical robots. The transmission ratio of the drive mechanism sets the torque, speed, and inertia relationship of the actuator to the link. Proper placement, sizing, and design of the drive mechanisms set the stiffness, mass, and overall operational performance of the robot. Most modern robots incorporate efficient, overload damage resistant, back-driveable drives.

Fig. 4.26 A six-axis Physik Instrumente (PI) piezo hexapod with sub-nanometer resolution (VIDEO 648)

Direct Drives

The direct drive is kinematically the simplest drive mechanism. In the case of pneumatic or hydraulic actuated robots, the actuator is directly connected between the links. Electric direct-drive robots employ high-torque, low-speed rotary or linear motors directly interfaced to the links. The complete elimination of free play and smooth torque transmission are features of a direct drive. However, there is often a poor dynamic (inertia ratio) match of the actuator to the link requiring a larger, less energy efficient, actuator.

Band Drives

A variant of direct drive is band drive. A thin alloy steel or titanium band is fixed between the actuator shaft and the driven link to produce limited rotary or linear motion. Drive ratios in the order of up to 10 : 1 (10 actuator revolutions for 1 revolution of the joint) can be obtained. Actuator mass is also moved away from the joint – usually toward the base, to reduce robot inertia and gravity loading. It is a smoother and generally stiffer drive than a cable or belt drive.

Belt Drives

Synchronous (toothed) belts are often employed in drive mechanisms of smaller robots and some axes of larger robots. These function much the same as band drives, but have the ability to drive continuously. Multiple stages (two or three) are occasionally used to produce large drive ratios (up to 100 : 1). Belt tension is controlled with idlers or center adjustment. The elasticity and mass of long belts can cause drive instability and thus increased robot settling time.

Gear Drives

Spur or helical gear drives provide reliable, sealed, low-maintenance power transmission in robots. They are used in robot wrists where multiple axes intersect and compact drive arrangements are required. Large-diameter gears are used in the base joints of larger robots to handle high torques with high stiffness. Gears are often used in stages and often with long drive shafts, enabling large physical separation between actuator and driven joint. For example, the actuator and one stage of reduction may be located near the elbow driving another stage of gearing or differential in a wrist through a long hollow drive shaft (Fig. 4.1).

Planetary gear drives are often integrated into compact gearmotors (Fig. 4.27). Minimizing backlash (free play) in a joint gear drive requires careful design, high-precision and rigid support to produce a drive mechanism which does not sacrifice stiffness, efficiency and accuracy for low backlash. Backlash in robots is controlled by a number of methods including selective assembly, gear center adjustment, and proprietary anti-backlash designs.

Worm Gear Drives

Worm gear drives are occasionally used in low-speed robot manipulator applications. They feature right-angle and offset drive capability, high ratios, simplicity, good stiffness and load capacity. They have poor efficiency, which makes them non-back-driveable at high ratios. This causes the joints to hold their position when unpowered but also makes them prone to damage by attempts to manually reposition the robot.

Proprietary Drives

Proprietary drives are widely used in standard industrial manipulators. The harmonic drive and the rotary vector (RV) drive are two examples of compact, low-backlash, high-torque-capability drives using special gears, cams, and bearings (Figs. 4.28 and 4.29).

Harmonic drives are frequently used in very small to medium-sized robots. These drives have low backlash, but its flexspline allows elastic windup and low stiffness during small reversing movements. RV drives are

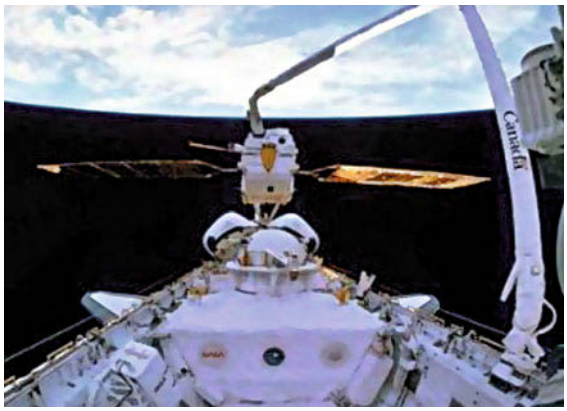


Fig. 4.27 The Space Shuttle robot arm has planetary gear joint drives

Fig. 4.28 The harmonic drive (VIDEO 649)

usually used in larger robots, especially those subject to overloads and shock loading.

Linear Drives

Direct-drive linear actuators incorporate a linear motor with a linkage to a linear axis. This linkage is often merely a rigid or flexure connection between the actuator forcer and the robot link. Alternatively, a packaged linear motor with its own guideways is mechanically connected directly to a linear axis. Direct linear electromagnetic drives feature zero backlash, high stiffness, high speeds, and excellent performance but are heavy, have poor energy efficiency, and cost more than other types of linear drives.

Ball Screws

Ball–screw-based linear drives efficiently and smoothly convert rotary actuator motion into linear motion. Typically, a recirculating ball nut mates with a ground and hardened alloy steel screw to convert rotary motion into linear motion. Ball screws can be easily integrated into linear axes. Compact actuator/drive packages are available, as well as components for custom integration. Stiffness is good for short and medium travel, however it is lower for long motions because the screw can only be supported at its ends. Low or zero backlash can be obtained with precision-ground screws. Speeds are limited by screw dynamic stability so rotating nuts enable higher speeds. Low-cost robots may employ plain screw drives featuring thermoplastic nuts on smooth rolled thread screws.

Fig. 4.29 The Nabtesco RV drive

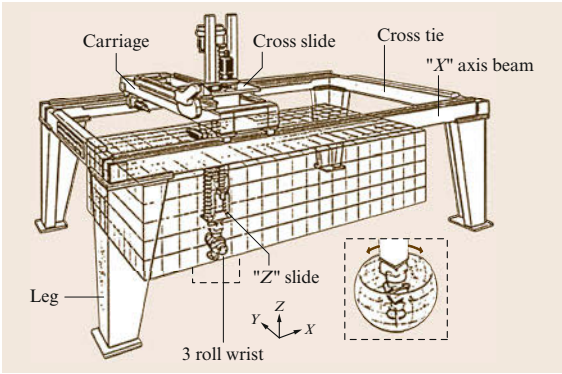


Fig. 4.30 The NASA gantry robot

Rack-and-Pinion Drives

These traditional components are useful for long motions where the guideways are straight or even curved. Stiffness is determined by the gear/rack interface and independent of length of travel. Backlash can be diffi-

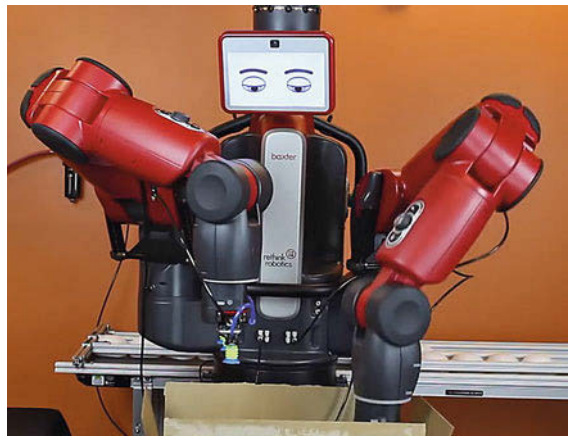


Fig. 4.31 Two-arm collaborative robot with Series Elastic Actuators (after [4.59])

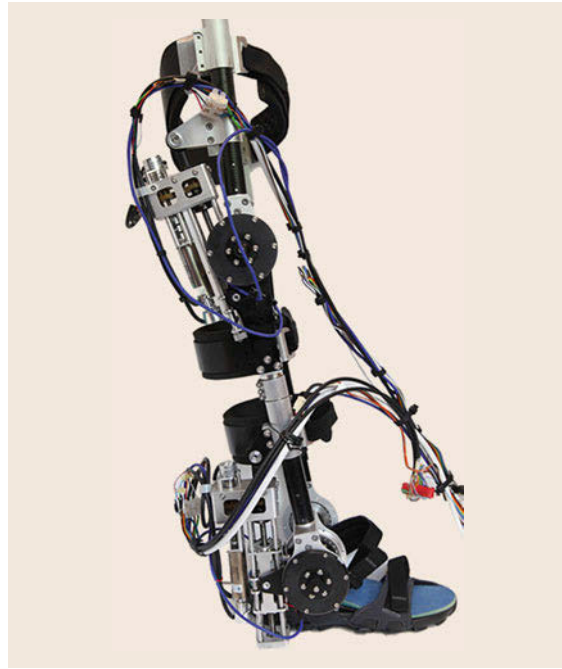


Fig. 4.32 Series Elastic Actuator assembly in knee-ankle-foot gait rehabilitation robot (after [4.60])

cult to control as rack-to-pinion center tolerances must be held over the entire length of travel. Dual pinion drives are sometimes employed to deal with backlash by providing active preload. Forces are generally lower than with screws due to lower ratios. Small-diameter (low teeth count) pinions have poor contact ratios, resulting in vibration. Sliding involute tooth contact requires lubrication to minimize wear. These catalog stock drive components are often used on large gantry robots and track-mounted manipulators (Fig. 4.30).

Compliant Actuators. Compliance in the form of elasticity in a drive can be an asset or a defect in a robot. High overall robot stiffness traditionally produces fast response, high positioning accuracy and simplified control. However the contact and interaction forces developed due to unexpected errors in dealing with the workpieces, tools, workplace and the environment can cause damage to the robot, the surroundings, and injury to persons. By adding controlled and measureable compliance to actuators, the elasticity of the robot can be usefully increased.

One class of reduced stiffness actuator is the Series Elastic Actuator (SEA). It features an elastic output element (spring) with a displacement sensor (measures spring stretch) in series with a stiff actuator and transmission component. With an appropriate controller, the stiff classical position control actuator acts as a force actuator ($F = -kx$), effectively isolating the drive inertia from the load inertia. This limits the forces resulting from collisions or forced compliance often encountered when working in unstructured environments and around persons. SEAs and similar compliant actuators and compliant actuator configurations are discussed in Chap. 69. Collaborative robots are designed to safely work next to or in contact with persons, Fig. 4.31 [4.59], and are an example of robots featuring compliant actuators. Figure 4.32 shows compliant actuators used in an exoskeleton for gait rehabilitation [4.60].

Other Drive Components

Splined shafts, kinematic linkages (four-bar, slider-crank mechanisms, etc.) chains, cables, flex couplings, clutches, brakes, and limit stops are some examples of other mechanical components used in robot drive mechanisms (Fig. 4.8). The Yaskawa RobotWorld assembly and process automation robot system features a magnetically suspended direct electromagnetic drive planar (2 axis) motor with no internal moving parts floating on a virtually frictionless planar air bearing. (Fig. 4.12).

4.9 Robot Performance

Industrial robot performance is often specified in terms of functional operations and cycle time. For assembly robots the specification is often the number of typical pick-and-place cycles per minute. Arc-welding robots are specified with a slow weld pattern and weave speed as well as by a fast repositioning speed. For painting robots, the deposition or coverage rate and spray pattern speed are important. Peak robot velocity and acceleration catalog data are generally just calculated numbers and will vary due to dynamic (inertia) and static (gravity) coupling between robot joints due to configuration changes as a robot moves.

4.9.1 Robot Speed

Maximum joint velocity (angular or linear) is not an independent value. For longer motions it is often limited by servomotor bus voltage or maximum allowable motor speed. For manipulators with high accelerations, even short point-to-point motions may be velocity limited. For low-acceleration robots, only gross motions will be velocity limited. Typical peak end-effector speeds can range up to 20 m/s for large or fast manipulators.

4.9.2 Robot Acceleration

In most modern manipulators, because the payload mass is small when compared with the manipulator mass, more power is spent accelerating the manipulator than the load. Acceleration affects gross motion time as well as cycle time (gross motion time plus settling time). Manipulators capable of greater acceleration tend to be stiffer manipulators. In high-performance robot manipulators, acceleration and settling time are more important design parameters than velocity or load capacity. Maximum acceleration for some assembly and material handling robots is in excess of 10 g with light payloads.

4.9.3 Repeatability

This specification represents the ability of the manipulator to return repeatedly to the same location. Depending on the method of teaching or programming the manipulator, most manufacturers intend this figure to indicate the radius of a sphere enclosing the set of locations to which the arm returns when sent from the same origin by the same program with the

same load and setup conditions. This sphere may not include the target point because calculation round-off errors, simplified calibration, precision limitations, and differences during the teaching and execution modes can cause significantly larger errors than those just due to friction, unresolved joint and drive backlash, servo system gain, and structural and mechanical assembly clearances and play. The designer must seriously consider the real meaning of the required repeatability specification. Repeatability is important when performing repetitive tasks such as blind assembly or machine loading. Typical repeatability specifications range from 1–2 mm for large spot-welding robots to 0.005 mm (5 μm) for precise micropositioning manipulators.

4.9.4 Resolution

This specification represents the smallest incremental motion that can be produced by the manipulator. Resolution is important in sensor-controlled robot motion and in fine positioning. Although most manufacturers calculate system resolution from the resolution of the joint position encoders, or from servomotor and drive step size, this calculation is misleading because system friction, windup, backlash, and kinematic configuration affect the system resolution. Typical encoder or resolver resolution is 10^{14}–10^{25} counts for full-axis or joint travel, but actual physical resolution may be in the range 0.001–0.5 mm. The useful resolution of a multijoint serial-link manipulator is worse than that of its individual joints.

4.9.5 Accuracy

This specification covers the ability of a robot to position its end-effector at a preprogrammed location in space. Robot accuracy is important in the performance of nonrepetitive types of tasks programmed from a database, or for taught tasks that have been remapped or offset owing to measured changes in the installation.

Accuracy is a function of the precision of the arm kinematic model (joint type, link lengths, angles between joints, any accounting for link or joint deflections under load, etc.), the precision of the world, tool, and fixture models, and the completeness and accuracy of the arm solution routine. Although most higher-level robot programming languages support arm solutions, these solutions usually model only simplified rigid-body kinematic configurations. Thus, manipulator accuracy becomes a matter of matching the robot geometry to the robot solution in use by precisely measuring and calibrating link lengths, joint angles, and mounting positions.

Typical accuracies for industrial manipulators range from ±10 mm for uncalibrated manipulators that have poor computer models to ±0.01 mm for machine-tool-like manipulators that have controllers with accurate kinematic models and solutions and precisely manufactured and measured kinematic elements.

4.9.6 Component Life and Duty Cycle

The three subassemblies in an electrically powered robot with the greatest failure problems are the actuators (servomotors), transmissions, and power and signal cables. Mean time between failures (MTBF) should be a minimum of 5000 h on line, and ideally at least 10 000 operating hours should pass between major component preventive maintenance replacement schedules.

Worst-case motion cycles must be assumed as most current robot installations are used in generally repetitive tasks. Small-motion design-cycle life (less than 5% of joint travel range) for assembly robots should be 20–100 million full bidirectional cycles. Large-motion cycle life (greater than 50% of full joint range) should typically be 5–40 million cycles.

Short-term peak performance is frequently limited by maximum drive loading, whereas long-term, continuous, performance is limited by motor heating. Rather than design for equal levels of short- and long-term performance, cost savings and performance improvements can result from designing for an anticipated duty cycle. This allows the use of smaller, lower-inertia, lighter motors. Industrial robots usually become obsolete and are replaced before they reach their design cycle life.

4.9.7 Collisions

In the course of operation, unforeseen or unexpected situations may occasionally result in a collision involving the manipulator, its tools, the workpiece, or other objects in the workplace. These accidents may result in no, little, or extensive damage, depending in large part on the design of the manipulator. Crash-resistant design options should be considered early in the design process if the time lost or cost of such accidents could be significant. Typical damage due to accidents include fracture or shear failures of gear teeth or shafts, dented or bent link structures, slipping of gears or pulleys on shafts, cut or severely abraded or deformed wires, cables or hoses, and broken connectors, fittings, limit stops or switches. Compliant elements such as overload (slip) clutches, elastic members, and padded surfaces can be incorporated to reduce shock loads and help decouple or isolate the actuators and drive components in the event of such collisions.

4.10 Conclusions and Further Reading

The mechanical design of a robot is an iterative process involving engineering, technical, and application-specific considerations evaluations, and choices. The final design should reflect consideration of detailed task requirements rather than simply broad specifications. Proper identification and understanding of these requirements is a key to achieving the design goals.

Design and choice of specific components involves tradeoffs. A purely static, rigid-body approach to manipulator design is often used, but is not always sufficient. Mechanical system stiffness, natural frequencies, control system compatibility, and intended robot applications and installation requirements must be considered.

There are many opportunities for further reading on the design of the mechanisms and actuation that form the core of a robotic system. A well-known and useful reference for robot design is *Rivin* [4.44].

Craig [4.6] and *Tsai* [4.9] provide the mathematical relations between the mechanical structure of a robot and its workspace and mechanical advantage. *Sclater* and *Chironis* [4.48] is a reprint of a valuable compendium of devices useful for a variety of applications, such as joint drives and transmissions. See *McCarthy* and *Soh* [4.34] for geometric techniques to design specialized mechanisms.

Juvinall and *Marshek* [4.45] and *Shigley* and *Mischke* [4.46, 47] are important references for the design of the components such as the link structure, bearings, and transmissions that are central to the effective mechanical performance of robotic systems.

Although many design decisions can be made through the application of straightforward algorithms and equations, a multitude of other important considerations transform the challenge of robot design into one requiring good engineering judgment.

Video-References

VIDEO 640 A parallel robot
available from http://handbookofrobotics.org/view-chapter/04/videodetails/640

VIDEO 642 Three-fingered robot hand
available from http://handbookofrobotics.org/view-chapter/04/videodetails/642

VIDEO 643 Robotics milking system
available from http://handbookofrobotics.org/view-chapter/04/videodetails/643

VIDEO 644 SCARA robots
available from http://handbookofrobotics.org/view-chapter/04/videodetails/644

VIDEO 645 Big Dog –Applications of hydraulic actuators
available from http://handbookofrobotics.org/view-chapter/04/videodetails/645

VIDEO 646 Raytheon Sarcos exoskeleton
available from http://handbookofrobotics.org/view-chapter/04/videodetails/646

VIDEO 648 PI piezo hexapod
available from http://handbookofrobotics.org/view-chapter/04/videodetails/648

VIDEO 649 Harmonic drive
available from http://handbookofrobotics.org/view-chapter/04/videodetails/649

References

4.1 D.T. Greenwood: *Classical Dynamics* (Prentice Hall, Upper Saddle River 1977)
4.2 F.C. Moon: *Applied Dynamics* (Wiley, New York 1998)
4.3 S.-M. Song, K.J. Waldron: *Machines that Walk: The Adaptive Suspension Vehicle* (MIT Press, Cambridge 1988)
4.4 M.T. Mason, J.K. Salisbury: *Robot Hands and the Mechanics of Manipulation* (MIT Press, Cambridge 1985)
4.5 R.P. Paul: *Robot Manipulators: Mathematics, Programming, and Control* (MIT Press, Cambridge 1981)
4.6 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Addison-Wesley, Reading 1989)
4.7 O. Bottema, B. Roth: *Theoretical Kinematics* (North-Holland, Amsterdam 1979)
4.8 J.M. McCarthy: *An Introduction to Theoretical Kinematics* (MIT Press, Cambridge 2013)
4.9 L.W. Tsai: *Robot Analysis. The Mechanics of Serial and Parallel Manipulators* (Wiley, New York 1999)
4.10 T. Lozano-Perez: Spatial Planning: A configuration space approach, IEEE Trans. Comput. **32**(2), 108–120 (1983)
4.11 J.C. Latombe: *Robot Motion Planning* (Kluwer, Boston 1991)
4.12 R. Vijaykumar, K. Waldron, M.J. Tsai: Geometric optimization of manipulator structures for working vol-

ume and dexterity. In: *Kinematics of Robot Manipulators*, ed. by J.M. McCarthy (MIT Press, Cambridge 1987) pp. 99–111
4.13 K. Gupta: On the nature of robot workspace. In: *Kinematics of Robot Manipulators*, ed. by J.M. McCarthy (MIT Press, Cambridge 1987) pp. 120–129
4.14 I. Chen, J. Burdick: Determining task optimal modular robot assembly configurations, Proc. IEEE Robotics Autom. Conf. (1995) pp. 132–137
4.15 P. Chedmail, E. Ramstei: Robot mechanisms synthesis and genetic algorithms, Proc. IEEE Robotics Autom. Conf. (1996) pp. 3466–3471
4.16 P. Chedmail: Optimization of multi-DOF mechanisms. In: *Computational Methods in Mechanical Systems*, ed. by J. Angeles, E. Zakhariev (Springer, Berlin, Heidlberg 1998) pp. 97–129
4.17 C. Leger, J. Bares: Automated Synthesis and Optimization of Robot Configurations, Proc. ASME Design Tech. Conf., Atlanta (1998), paper no. DETC98/Mech-5945 CD-ROM
4.18 F.C. Park: Distance metrics on the rigid body motions with applications to mechanism design, ASME J. Mech. Des. **117**(1), 48–54 (1995)
4.19 J.M.R. Martinez, J. Duffy: On the metrics of rigid body displacements for infinite and finite bodies, ASME J. Mech. Des. **117**(1), 41–47 (1995)
4.20 M. Zefran, V. Kumar, C. Croke: Choice of Riemannian metrics for rigid body kinematics, Proc. ASME Design Tech. Conf., Irvine (1996), paper no. DETC96/Mech-1148
4.21 Q. Lin, J.W. Burdick: On well-defined kinematic metric functions, Proc. Int. Conf. Robotics Autom., San Francisco (2000) pp. 170–177
4.22 C. Gosseli: On the design of efficient parallel mechanisms. In: *Computational Methods in Mechanical Systems*, ed. by J. Angeles, E. Zakhariev (Springer, Berlin, Heidelberg 1998) pp. 68–96
4.23 J.V. Albro, G.A. Sohl, J.E. Bobrow, F. Park: On the computation of optimal high-dives, Proc. Int. Conf. Robotics Autom., San Francisco (2000) pp. 3959–3964
4.24 G.E. Shilov: *An Introduction to the Theory of Linear Spaces* (Dover, New York 1974)
4.25 J.K. Salisbury, J.J. Craig: Articulated hands: Force control and kinematic issues, Int. J. Robotics Res. **1**(1), 4–17 (1982)
4.26 J. Angeles, C.S. Lopez-Cajun: Kinematic isotropy and the conditioning index of serial manipulators, Int. J. Robotics Res. **11**(6), 560–571 (1992)
4.27 J. Angeles, D. Chabla: On isotropic sets of points in the plane. Application to the design of robot architectures. In: *Advances in Robot Kinematics*, ed. by J. Lenarčič, M.M. Stanišić (Kluwer, Boston 2000) pp. 73–82
4.28 E.F. Fichter: A Stewart platform-based manipulator: General theory and practical construction. In: *Kinematics of Robot Manipulators*, ed. by J.M. McCarthy (MIT Press, Cambridge 1987) pp. 165–190
4.29 J.P. Merlet: *Parallel Robots* (Kluwer, Boston 1999)
4.30 J.M. Hervé: Analyse structurelle des méchanismes par groupe des déplacements, Mech. Mach. Theory **13**(4), 437–450 (1978)
4.31 J.M. Hervé: The Lie group of rigid body displacements, a fundamental tool for mechanism design, Mech. Mach. Theory **34**, 719–730 (1999)
4.32 A.P. Murray, F. Pierrot, P. Dauchez, J.M. McCarthy: A planar quaternion approach to the kinematic synthesis of a parallel manipulator, Robotica **15**(4), 361–365 (1997)
4.33 A. Murray, M. Hanchak: Kinematic synthesis of planar platforms with RPR, PRR, and RRR chains. In: *Advances in Robot Kinematics*, ed. by J. Lenarčič, M.M. Stanišić (Kluwer, Boston 2000) pp. 119–126
4.34 J.M. McCarthy, G.S. Soh: *Geometric Design of Linkages*, 2nd edn. (Springer, Berlin, Heidelberg 2010)
4.35 V. Kumar: Instantaneous kinematics of parallel-chain robotic mechanisms, J. Mech. Des. **114**(3), 349–358 (1992)
4.36 C. Gosselin, J. Angeles: The optimum kinematic design of a planar three-degree-of-freedom parallel manipulator, ASME J. Mech. Transmiss. Autom. Des. **110**(3), 35–41 (1988)
4.37 J. Lee, J. Duffy, M. Keler: The optimum quality index for the stability of in-parallel planar platform devices, Proc. ASME Design Eng. Tech. Conf., Irvine (1996), paper no. 96-DETC/MECH-1135
4.38 J. Lee, J. Duffy, K. Hunt: A practical quality index based on the octahedral manipulator, Int. J. Robotics Res. **17**(10), 1081–1090 (1998)
4.39 S.E. Salcudean, L. Stocco: Isotropy and actuator optimization in haptic interface design, Proc. Int. Conf. Robotics Autom., San Francisco (2000) pp. 763–769
4.40 L.-W. Tsai, S. Joshi: Kinematics and optimization of a spatial 3-UPU parallel manipulator, J. Mech. Des. **122**, 439–446 (2000)
4.41 Q. Jin, T.-L. Yang: Theory for topology synthesis of parallel manipulators and its application to three-dimension-translation parallel manipulators, J. Mech. Des. **126**(3), 625–639 (2004)
4.42 X. Kong, C.M. Gosselin: Type synthesis of three-degree-of-freedom spherical parallel manipulators, Int. J. Robotics Res. **23**, 237–245 (2004)
4.43 T.A. Hess-Coelho: Topological synthesis of a parallel wrist mechanism, J. Mech. Des. **128**(1), 230–235 (2006)
4.44 E.I. Rivin: *Mechanical Design of Robots* (McGraw-Hill, New York 1988) p. 368
4.45 R.C. Juvinall, K.M. Marshek: *Fundamentals of Machine Component Design*, 4th edn. (Wiley, New York 2005) p. 832
4.46 J.E. Shigley, C.R. Mischke: *Mechanical Engineering Design*, 7th edn. (McGraw-Hill, New York 2004) p. 1056
4.47 J.E. Shigley, C.R. Mischke: *Standard Handbook of Machine Design*, 2nd edn. (McGraw-Hill, New York 1996) p. 1700
4.48 N. Sclater, N. Chironis: *Mechanisms and Mechanical Devices Sourcebook*, 4th edn. (McGraw Hill, New York 2007) p. 512
4.49 Intuitive Surgical EndoWrist Insturments, http://www.intuitivesurgical.com/products/instruments/
4.50 J.P. Trevelyan: Sensing and control for shearing robots, IEEE Trans. Robotics Autom. **5**(6), 716–727 (1989)

4.51 J.P. Trevelyan, P.D. Kovesi, M. Ong, D. Elford: ET: A wrist mechanism without singular positions, Int. J. Robotics Res. **4**(4), 71–85 (1986)
4.52 C. de Silva: *Sensors and Actuators: Control Systems Instrumentation* (CRC, Boca Raton 2007)
4.53 J. Hollerbach, I. Hunter, J. Ballantyne: A Comparative analysis of actuator technologies for robotics, Robotics Rev. **2**, 299–342 (1992)
4.54 Boston Dynamics, Waltham, MA, USA: Big Dog Robot, http://www.bostondynamics.com/robot_bigdog.html
4.55 Raytheon, Waltham MA, USA: Sarcos Exoskeleton, http://www.popsci.com/category/tags/raytheon-sarcos-xos
4.56 International Submarine Engineering, Port Coquitlam, BC, Canada: Magnum 7, http://www.ise.bc.ca/manips.html
4.57 Shadow Robot Company, London, UK: Air muscles and pneumatic hands, http://www.shadowrobot.com/tag/muscle-hand
4.58 Johns Hopkins, Baltimore, USA: Pneumatic Stepper Motor and MrBot, http://urobotics.urology.jhu.edu/projects/PneuStep
4.59 The Baxter Robot Rethink Robotics, http://www.rethinkrobotics.com/products/baxter/
4.60 Singapore Institute for Neurotechnology, NeuroRehabilitation Laboratory: http://www.sinapseinstitute.org/projects/neurorehabilitation

Multimedia Contents

5. Sensing and Estimation

Henrik I. Christensen, Gregory D. Hager

Sensing and estimation are essential aspects of the design of any robotic system. At a very basic level, the state of the robot itself must be estimated for feedback control. At a higher level, perception, which is defined here to be task-oriented interpretation of sensor data, allows the integration of sensor information across space and time to facilitate planning.

This chapter provides a brief overview of common sensing methods and estimation techniques that have found broad applicability in robotics. The presentation is structured according to a process model that includes sensing, feature extraction, data association, parameter estimation, and model integration. Several common sensing modalities are introduced and characterized. Common methods for estimation in linear and nonlinear systems are discussed, including statistical estimation, the Kalman filter, and sample-based methods. Strategies for robust estimation are also briefly described. Finally, several common representations for estimation are introduced.

5.1 **Introduction** 91
5.2 **The Perception Process** 92
5.3 **Sensors** 94
5.4 **Estimation Processes** 98
5.4.1 Point Estimation 98
5.4.2 Other Approaches to Estimation 101
5.4.3 Robust Estimation Methods 103
5.4.4 Data Association Techniques 105
5.4.5 Modeling Sensors 108
5.4.6 Other Uncertainty Management Methods 108
5.5 **Representations** 109
5.5.1 Raw Sensor Representations 109
5.5.2 Grid-Based Representations 109
5.5.3 Feature Representations 110
5.5.4 Symbolic/Graph-Based Models 110
5.6 **Conclusions and Further Readings** 111
References 111

5.1 Introduction

Controlling a robotic system would be relatively simple if a complete model of the environment was available, and if the robot actuators could execute motion commands perfectly relative to this model. Unfortunately, in most cases of interest, a complete world model is not available, and perfect control of mechanical structures is never a realistic assumption. Sensing and estimation are a means of compensating for this lack of complete information. Their role is to provide information about the state of the environment and the state of the robot system as a basis for control, decision making, and interaction with other agents in the environment, such as humans.

For the purposes of discussion, we will differentiate between sensing and estimation to recover the state of the robot itself, referred to as *proprioception*, versus sensing and estimation to recover the state of the external world, referred to as *exteroception*. In practice, most robot systems are designed to have the proprioception necessary to estimate and control their own physical state. On the other hand, recovering the state of the world from sensor data is usually a much larger and more complex problem.

Early work on computational perception for robotics assumed that one could recover a complete general-purpose model of the environment, use such

a model to make decisions, and subsequently act on them, as for example presented by [5.1]. More recently it has become apparent that such an approach is not realistic. Indeed, considering that sensor-based robots now appear in diverse applications such as mobile surveillance, high-performance manipulation, and medical interventions, it is clear that appropriate sensing and estimation for a given system must be highly task dependent. Consequently, the discussion here is organized along the lines of task-oriented sensing and estimation of the external world.

Sensing and estimation together can be viewed as the process of transforming a physical quantity into a computer representation that can be used for further processing. Sensing is thus closely tied to transducers that transform some physical entity into a signal that can be processed by a computer. Sensing is also intimately tied to perception, the process of representing the sensory information in an task-oriented model of the world. However, sensor data is usually corrupted in various ways that complicate this process. Statistical noise arises from the transducer, discretization is introduced in the digitization process, and ambiguity is introduced by poor sensor selectivity to name a few examples. Estimation methods are thus introduced to support appropriate integration of information into models of the environment and for improvement of the signal-to-noise ratio.

In this chapter the general characteristics of sensing and estimation are introduced, while more in-depth presentations of select topics are provided in Part C of the handbook. In Sect. 5.2 the overall sensing/perception process is introduced. In Sect. 5.3 different kinds of sensors are introduced and some key characteristics are presented. Estimation of world representations can utilize a number of different methods involving both parametric and nonparametric techniques as discussed in Sect. 5.4. For model-based integration a variety of different representations can be used, as described in Sect. 5.5.

5.2 The Perception Process

The input to the perception process is typically twofold: (1) digital data from a number of sensors/transducers, and (2) a partial model of the environment (a world model) that includes information about the state of the robot and other relevant entities in the external world. The sensor data itself can take on a number of different forms such as a scalar or vector value $\boldsymbol{x}(\alpha, \beta)$ acquired over a time series $x(t)$, a scan $x_t(\theta_i)$, a vector field $\boldsymbol{x}$ or a three-dimensional volume $x(\rho, \theta, \phi)$. In many cases, a system must integrate data from several disparate sensors, for example, an estimate of the position of a mobile robot may integrate data from axis encoders, vision, global positioning system (GPS) data, and inertial sensors.

To further structure the discussion in this chapter, we adopt a general model of the perception process as shown in Fig. 5.1. In this model, we have included the most common operations applied to integrate sensor data with a world model. Depending on the task in question, some of the included modules may be missing, and others may themselves take on a complicated structure. However, the supplied model suffices to illustrate many of the issues in sensing and estimation. In the remainder of this section, we discuss an example from mobile localization to illustrate this model.

The initial problem in sensory processing is data preprocessing and feature extraction. The role of preprocessing is to reduce noise from the transducer, to remove any systematic errors, and to enhance relevant aspects of the data. In some cases, sensory information might also have to be temporally or spatially aligned for subsequent integration. There are innumerable ways

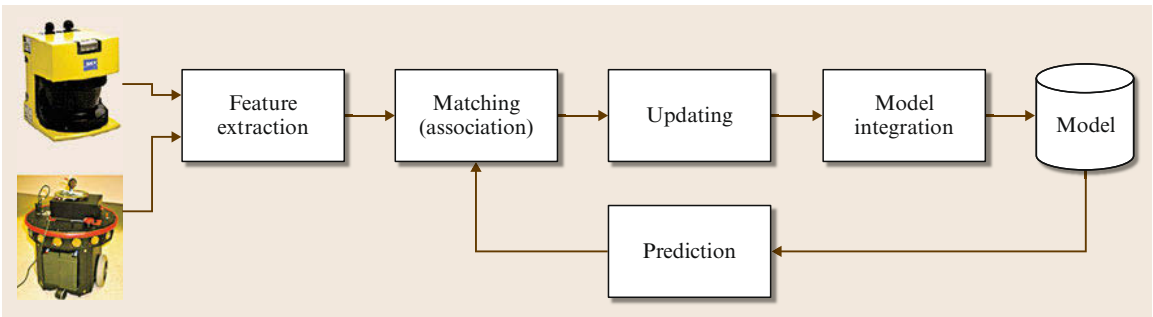


Fig. 5.1 Example of a perception process as discussed in this chapter

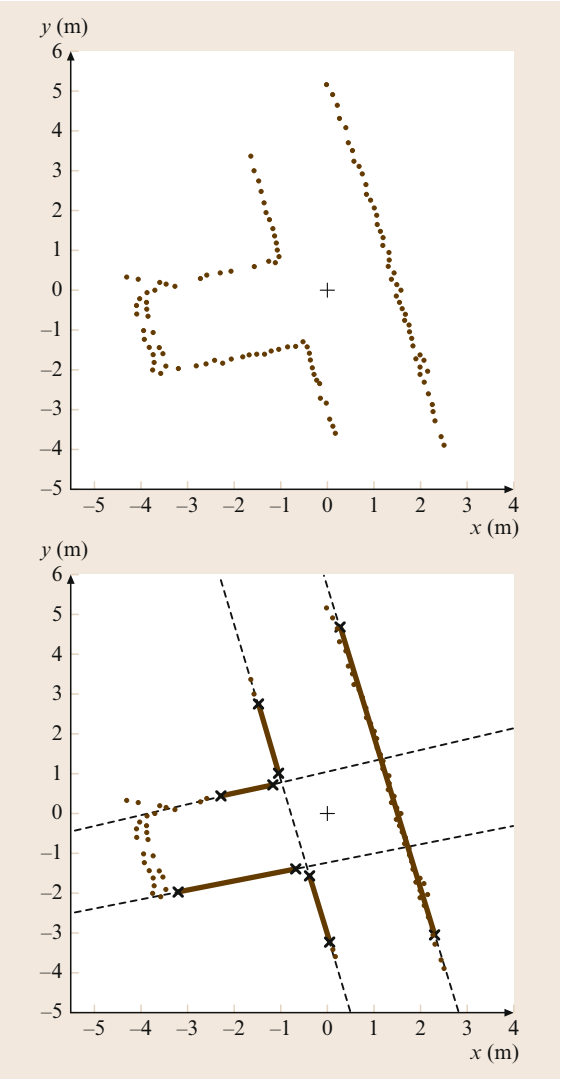


Fig. 5.2 An example of feature extraction from a laser scan (after [5.2])

that data can be preprocessed to enhance or extract features that are used in the integration. One common approach is model fitting, as illustrated for a laser scanner in Fig. 5.2. Once sensor information is available, it is often necessary to match the data with an existing model (Fig. 5.3). This model may be based on a priori known structure (e.g., a computer-aided design (CAD) model of the environment), or may have been built up from previously acquired data. Data association methods are commonly employed to estimate the relationship between sensor data and the model of the environment. In our mobile robot localization example, the extracted line features are matched against a polygonal world model. This matching process can be performed in several different ways, but in general it is an optimization that maximizes the alignment of features to the model.

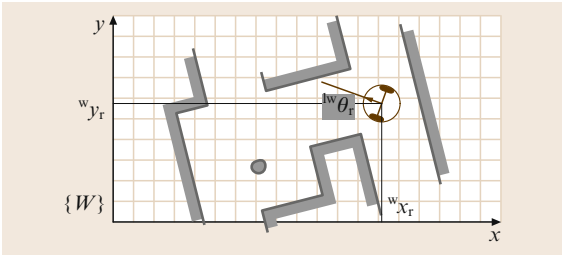


Fig. 5.3 An example environmental model for mobile robot localization (after [5.2])

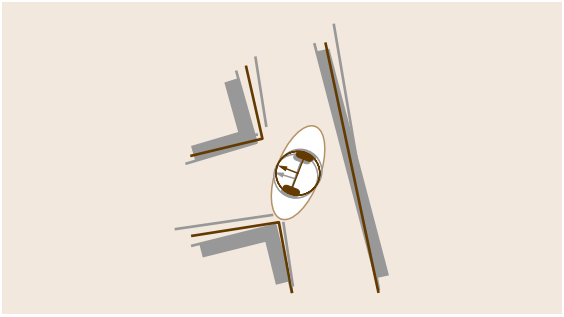


Fig. 5.4 Estimation of position and orientation for the example mobile robot (after [5.2])

Once sensory data has been matched against the world model it is possible to update the model with new information contained in the sensor data. In the example, the orientation and position of the robot relative to the world model can be updated (Fig. 5.4) from the matched line segments.

Finally, it may be possible to develop a dynamical system model of the underlying state being estimated. Using such a system model, it is possible to predict how the world changes over time until new sensory data is acquired. This can be used within a feed-forward prediction process, which in turn simplifies data association for new sensory readings, as shown in Fig. 5.1.

With this as a prologue, we now turn to discuss each step of the perception process in greater detail.

5.3 Sensors

There are a variety of ways to classify sensors depending on what they measure, and how they measure it. As noted previously, proprioceptive sensors are used to measure the internal state of a robot, which might include position of different degrees of freedom, temperature, voltage on key components, motor current, force applied to an effector, and so forth. Exteroceptive sensors, on the other hand, generate information about the external environment in terms of distance to an object, interaction forces, tissue density, and so forth.

Sensors may also be differentiated based on whether they are passive or active. In general, an active sensor is one that emits energy into the environment, and measures properties of the environment based on the response. A passive sensor is one that is not active. Active sensors are generally more robust than passive sensors since they exert some control over the measured signal. For example, a passive stereo camera system must rely on the appearance of viewed surfaces when performing feature matching for triangulation (Chap. 31), whereas structured light systems project a pattern onto the scene and are thus less sensitive to scene characteristics. Even so, absorbtion, scattering or interference of the emitted signal can affect the performance of active sensors.

Proprioceptive sensors are typically passive and usually measure physical properties of the robot such as joint position, velocity, or acceleration, motor torque, and so forth. Exteroceptive sensors, on the other hand, can be further divided into contact and noncontact sensing. The contact sensors are typically the same modalities as used for proprioception, while noncontact sensor sensors involve most of the modalities that can be used for estimation of physical properties at a distance including intensity, range, direction, size, and so forth.

A classification of typical sensors according to method and typical application is shown in Table 5.1. More detail on methods of sensing, characterization of sensors, and general applications can for example be found in the Handbook of Modern Sensors [5.3] and in Part C of this handbook.

Estimation of rotational motion is fundamental to control of robot manipulators and also for estimation of ego-motion for mobile systems. The most common sensor for measurement of rotation is the quadrature encoder. It is composed of a transparent disc, with two periodic patterns that are out of phase, as shown in Fig. 5.5. Through the use of counters it is possible

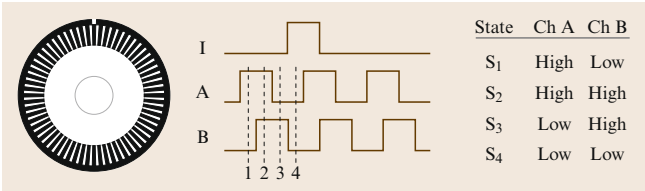


Fig. 5.5 Sketch of the quadrature encoder disc, and output from photodetectors placed over each of the two pattern. The corresponding state changes are shown on the *right*

Table 5.1 Classification of sensors frequently used in robotics according to sensing objective (proprioception (PC)/exteroception (EC)) and method (active/passive)

Classification	Sensor type	Sens	A/P
Tactile sensors	Switches/bumpers	EC	P
	Optical barriers	EC	A
	Proximity	EC	P/A
Haptic sensors	Contact arrays	EC	P
	Force/torque	PC/EC	P
	Resistive	EC	P
Motor/axis sensors	Brush encoders	PC	P
	Potentiometers	PC	P
	Resolvers	PC	A
	Optical encoders	PC	A
	Magnetic encoders	PC	A
	Inductive encoders	PC	A
	Capacity encoders	EC	A
Heading sensors	Compass	EC	P
	Gyroscopes	PC	P
	Inclinometers	EC	A/P
Beacon based (postion wrt an inertial frame)	GPS	EC	A
	Active optical	EC	A
	Radio frequency (RF) beacons	EC	A
	Ultrasound beacon	EC	A
	Reflective beacons	EC	A
Ranging	Capacitive sensor	EC	P
	Magnetic sensors	EC	P/A
	Camera	EC	P/A
	Sonar	EC	A
	Laser range	EC	A
	Structured light	EC	A
Speed/motion	Doppler radar	EC	A
	Doppler sound	EC	A
	Camera	EC	P
	Accelerometer	EC	P
Identification	Camera	EC	P
	Radio frequency identification RFID	EC	A
	Laser ranging	EC	A
	Radar	EC	A
	Ultrasound	EC	A
	Sound	EC	P

to directly compute the motion and its direction (the phasing between sensors A and B in Fig. 5.5). In addition the disc is frequently fitted with a single dot on the outer rim for indexing (specification of a zero index). The density of the pattern determines the resolution of the measurements. When fitting the sensor to a motor before a reduction gear it is easy to achieve accuracies beyond $1/1000°$.

For the estimation of force and torque at an end-effector it is possible to use piezoelectric elements. These elements generate a voltage that is proportional to the introduced deformation. Through careful placement it is possible to measure both force and torque. The sensors are used in robotic manipulation as part of assembly systems, deburring, etc. and also in medical applications for the estimation of stress and contact. Force/torque sensors are widely available in a range of sizes and dynamic ranges, including new flexible arrays that can be mounted on a variety of end-effectors (Fig. 5.6). These arrays are more commonly referred to as *tactile sensors* as they begin to simulate the human tactile sense. See Chap. 19 and [5.4, 5] for recent reviews of the state of the art in tactile sensing. Potential problems with force sensors are a dead band on initial contact, and noisy data from the basic sensing elements, which calls for signal processing to clean up the data.

Ego-motion estimation is an important part of almost all robotic systems. To this end it is possible to use inertial measurement units (IMU). An IMU typically includes both accelerometers and gyros. Accelerometers are sensitive to all types of acceleration, which implies that both translation motion and rotation (centripetal forces) are measured in combination. Joint IMU units allow the estimation of rotation and translation, and allow for double integration to estimation the velocity, orientation, and position of a system, as for example reported in [5.6]. One of the problems associated with the use of an IMU is the need for double integration. Small biases and noise can result in significant divergence in the final estimate, which calls for use of detailed models and careful calibration and identification of sensor characteristics. An example of data from a cross-bow DMU-6x unit for a car driving on an unpaved road is shown in Fig. 5.7.

Much early work on mobile robotics, underwater robots, and some medical robotics relies on ultrasonic ranging. The general class of sensors are often termed sound navigation and ranging (sonars). The general principle is that the system emits a sound pulse and awaits the return of echoes that have bounced off objects in the environment. Knowing the transmission speed in the medium and the time of flight it is possible to compute the distance. The method was widely used in early robotics due to the availability of low-cost sensors with adequate performance. In underwater robotics this is still a primary sensor. Sonar is discussed in detail in Chap. 30.

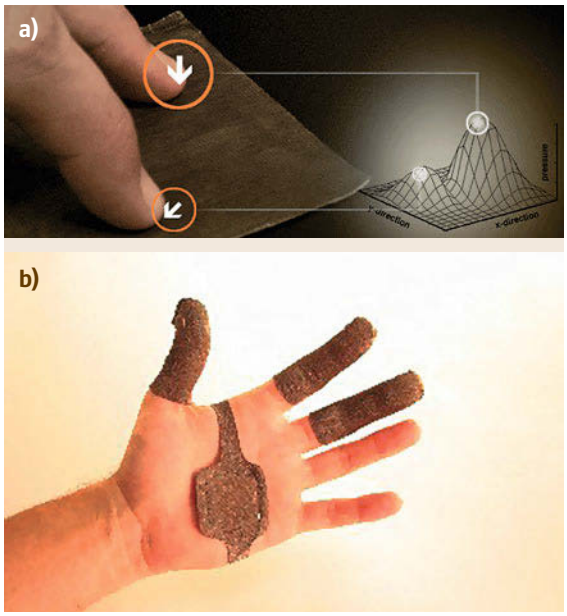


Fig. 5.6 **(a)** TactArray, a flexible capacitive array tactile sensor from Pressure Profile Systems, Inc., is appropriate for sensing contact locations and areas under sliding conditions. **(b)** Conformable TactArray sensors can fit on a human or robotic hand (courtesy Pressure Profile Systems, Inc.)

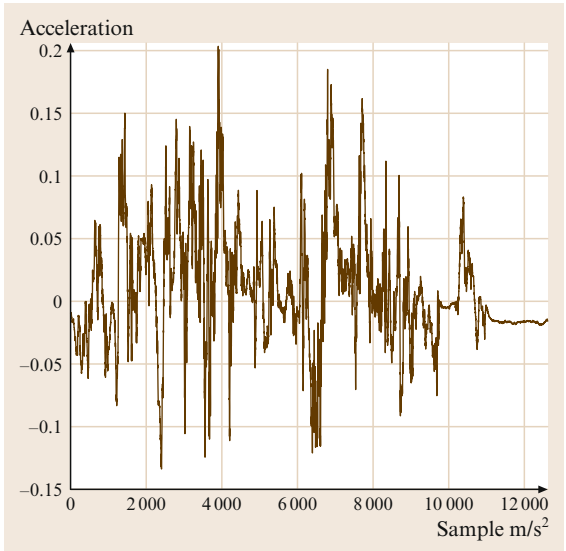


Fig. 5.7 Example data from an IMU unit for driving on an unpaved road

Fig. 5.8 Example laser ranging sensor (SICK LMS291) that is widely used in mobile robotics

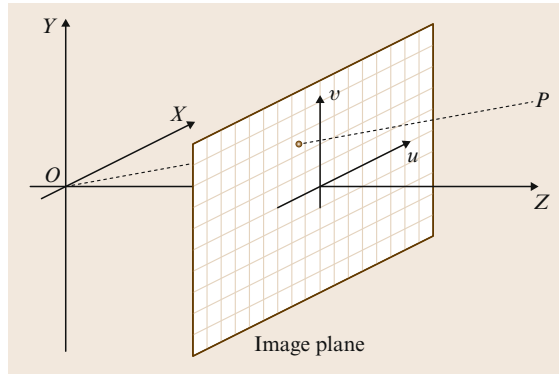


Fig. 5.9 The pinhole camera model

Recent progress on environmental modeling and navigation has, in many respects, been due to the emergence of low-cost high-fidelity laser scanning systems. The SICK series of laser scanners are time-of-flight scanners. The scanner sends out a pulse of light and measures the time to return. The standard scanner enables estimation of distances up to 80 m at centimeter or millimeter accuracy. The scanner measures distances in a plane with an angular resolution of 0.5–1°. The field of view is 180° resulting in 181–361 range measurements. The sensor data are contaminated by uniformly distributed noise, which must be considered in the detection of features or integration of data into a raw sensor map (Fig. 5.8).

Imaging sensors are a rich source of information for sensing and estimation. Imaging sensors come in a wide variety of configurations, varying according to imaging geometry, image resolution, sensor technology and the range of sensed spectral bands. Most readers are no doubt familiar with the traditional three-CCD, perspective color camera. In this case, there are three charge-coupled detector (CCD) arrays, each receiving a portion of the visible spectrum corresponding roughly to the human perception of red, green, and blue colors. A common and less expensive alternative is a so-called single-chip CCD camera. In this case, a special spatial array of color filters, usually referred to as a Bayer filter after its inventor Bryce Bayer, is employed. The resulting spatial array is subsequently processed (a process referred to as demosaicing) to provide color information for each pixel.

In the United States, image sensors traditionally contained 480 rows of 640 pixels according to the National Television System Committee (NTSC) standard created for analog transmission of television signals. The corresponding European standard, PAL, has 576 lines of 768 pixels. More recently, the advent of digital interfaces such as IEEE 1394 and

Fig. 5.10 Catadioptric image and the same image mapped to a cylindrical surface

USB 2.0 have allowed camera systems to be developed with significantly improved resolution ranging into the millions of pixels. At the same time, cost-effective infrared (IR) and ultraviolet (UV) cameras have become available, allowing the development of advanced multispectral image interpretation systems.

A traditional imaging sensor contains an optical system that focuses light on a planar imaging array. In most cases, this system can be modeled using the classical pinhole camera model (shown in Fig. 5.9). Given

a point $(x, y, z)^{\mathrm{T}}$ in Euclidean space, the corresponding camera pixel coordinates $(u, v)^{\mathrm{T}}$ are given by

$$(u - u_{\mathrm{c}}) = \frac{f}{s_x}\frac{x}{z}\,,$$
$$(v - v_{\mathrm{c}}) = \frac{f}{s_y}\frac{y}{z}\,, \qquad (5.1)$$

where f is the focal length of the lens system, u_{c} and v_{c} are the pixel coordinates of the center of projection, and s_x and s_y are the size of a single pixel on the imaging array. In practice, these models are also augmented with low-order models of image distortion. The values of these parameters for a given camera system can be determined experimentally using a variety of methods [5.7].

By combining a traditional perspective camera with a mirror, creating a so-called catadioptric system, it is possible to create imaging geometries that map fields of view as large as a hemisphere into a single image. Such systems are useful, for example, for surveillance, and their geometric properties provide for stable position referencing for mobile navigation [5.8]. An example image is shown in Fig. 5.10 together with the corresponding image when mapped to a cylindrical surface.

Active ranging cameras, which combine images from an optical camera with a dense range map, are now also widely available and quite cost effective. These systems perform triangulation between a light projector that throws a pattern with known structure into the scene, and a camera that views the scene and detects the pattern. By matching or correlating the pattern elements between the projector and the camera, and using the known relationship between them, it is possible to recover depth using standard methods. The camera/projector pair for depth recovery typically operate in the infrared to avoid the pattern being visible to the human eye. Thus, these systems typically include a third visible light camera to acquire a corresponding color image. Hence the name RGB-D (red–green–blue–depth) cameras. Fig. 5.11 shows two recent low-cost RGB-D products. In Fig. 5.12 is show the fused output for a tabletop scene.

The discussion above has touched on the most commonly employed robotic sensing devices. Many special-purpose sensors are employed for specific applications. In medicine (Chap. 63), ultrasound, X-ray, computed tomography, and magnetic resonance imaging are commonly employed. Underground mapping makes use of ground-penetrating radar [5.9]. Underwater robotics makes use of many variations on acoustical sensors. Further discussions of these more task-specific sensing modalities can be found in the application chapters in Part C of this handbook.

Fig. 5.11 Picture of proxim and kinect RGB-D sensors

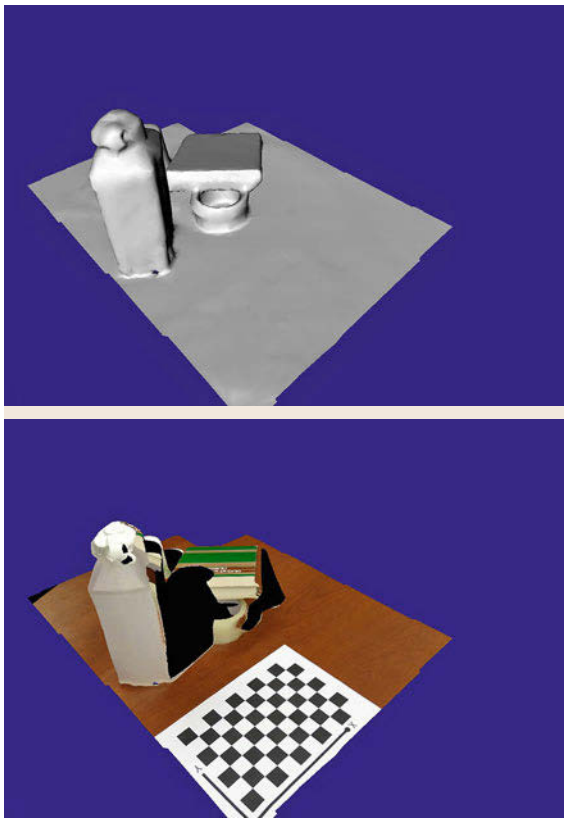


Fig. 5.12 Example of fused range images from RGB-D camera and the same scene as an intensity textured mesh

5.4 Estimation Processes

As discussed in the introduction, there are many different techniques for combining information from sensors. The appropriate set of techniques depends, to a great degree, on what is known a priori about the environment, what information is necessary for the task at hand, and what models for the sensing system are appropriate. Common methodologies include simple voting-based methods, parametric and nonparametric statistical estimation techniques, fuzzy logic-based systems, and Dempster–Shafer theory.

To illustrate this point, consider the robot localization problem introduced in Sect. 5.2. At the outset, if nothing is known about the environment, the robot may acquire a laser scan and try to produce an initial model of the environment using line segments. Since nothing is known a priori, the system must estimate:

1. The number of line segments
2. The data association between line segments and observed data values
3. The parameters of the line segments themselves.

This is a challenging problem that can be attacked by simple voting techniques such as the Hough transform [5.10] or random sample consensus (RANSAC) [5.11] or more sophisticated unsupervised clustering methods such as *k*-means [5.12], expectation maximization (EM) [5.13], or generalized principal component analysis (GPCA) [5.14]. In many cases, this is a computationally intensive, iterative process.

Conversely, if a prior CAD model for the environment is known, then the problem is to produce a small set of parameters (translation and rotation) of the model to match the data. This problem can be solved, using feature matching by aligning observed points to the model with iterative closest-point algorithms (ICP) [5.15] or other efficient combinatorial matching algorithms such as Monte Carlo methods [5.16]. The best method to apply again depends to a great degree on the structure of the environment and what is known a priori.

Once an initial registration is known, new data can take advantage of the fact that strong prior knowledge is available. In particular, as the robot moves, the sensor data should change in a predictable fashion. Thus is it possible to make use of predictor–corrector methods such as the Kalman filter [5.17, 18] or sequential importance sampling [5.19], provided appropriate statistical characterizations of the sensing system are available. The data association problem, if present, can be addressed using a variety of general techniques such as EM [5.12] or more specialized modifications to the previously cited predictor–corrector methods [5.20].

It is often the case that sensor data is corrupted by occasional nonsensical values. For example, the laser range finder in our example may occasionally return a spurious range value due to a reflection. Many commonly used estimation techniques are not robust to such so-called data outliers. Techniques from robust statistics [5.21] can be used to improve the performance of sensing and estimation systems in such cases.

Finally, we may want to consider what information is actually important for the task at hand. Most of the techniques above presume that the goal is to produce an accurate estimate of a set of continuous parameters closely related to the underlying data. However, in some tasks, the parameter values themselves may not be what is of interest. For example, suppose that the goal of our robot is to drive through a doorway. Although this clearly depends on an ability to estimate the width of the door (a continuous parameter), the decision is ultimately binary. This problem can be codified as a *decision problem*. Decision problems can be modeled using concepts from *decision theory* [5.22] including zero–one loss functions, likelihood ratios, or probability ratios. For example, in the case of fitting through a door, for a low-priority task there may be a low cost associated with not attempting to move through this particular door (necessitating replanning to find an alternative route) relative to attempting to navigate through an opening that is too small (risking damage to the robot or the door, or both). Conversely, if the task is urgent, more risky behavior may be warranted.

For any given task (or decision), the amount of information necessary to reach the decision may vary, for example, if the doorway is quite wide, it may require relatively little information to safely navigate through it. Conversely, a tight fit may require close inspection before a decision can be reached. The problem of determining the type and/or amount of information necessary to reach a decision is referred to variously as the *sequential sampling problem* [5.22], the sensor control problem, or the sensor planning problem [5.23–25].

5.4.1 Point Estimation

In our robot localization example we saw several cases where the key problem was to estimate an unknown quantity that can be represented as a point in a vector space. Examples include the location of a two-dimensional (2-D) or three-dimensional (3-D) point or the location of a robot. We also saw examples where the problem was to locate the pose (position and orien-

tation) of the robot, or parameters of a line segment. The latter differ in that the underlying parameter space is not a vector space. This introduces some additional unique problems. We refer the reader to [5.26, 27] for further discussion, and in the remainder of this chapter restrict our attention to *point estimation* problems on vector spaces. In our discussion, we assume the reader is familiar with multivariate Gaussian distributions as described in [5.28] and basic linear algebra [5.29].

In the remainder of this section, we consider the following general problem.

Given: an observation model

$$y = f(x, \eta) \,. \tag{5.2}$$

Estimate: $x \in \mathrm{Re}(n)$ from observations $y \in \mathrm{Re}(m)$ where η is an unknown disturbance taking values in $\mathrm{Re}(k)$ and f is a known mapping from $\mathrm{Re}(k+n)$ to $\mathrm{Re}(m)$.

We divide our discussion into two topical areas:

- Methods for performing estimation on batch and sequential data when f is linear,
- Methods for performing estimation on sequential data when f is nonlinear.

Estimation Techniques for Batch and Sequential Data with Linear Models

In this section, we discuss linear and linearized estimation techniques for sequential data, including the Kalman filter and extensions thereof. Our goal is to provide an overview of techniques available. The reader may also wish to consult more in-depth references such as [5.18, 30, 31] and (Chap. 35) for additional information.

We first consider the case when f in (5.2) is linear in its arguments. In this case, we can write

$$y = \mathbf{F}x + \mathbf{B}\eta \,, \tag{5.3}$$

where $\mathbf{F} \in \mathrm{Re}(m \times n)$ defines the (linear) relationship between the unknown x and the observation y and $\mathbf{B} \in \mathrm{Re}(m \times m)$. For the moment, we will drop $\mathbf{B}$ and assume that η represents the complete disturbance model of the system.

The *least-squares* method of estimating x from y proceeds by solving the optimization problem

$$\min_x \|\mathbf{F}x - y\|^2 \,. \tag{5.4}$$

This optimization has a unique solution $\hat{x}$ if and only if the matrix $\mathbf{F}$ has full column rank. In this case, the solution can be computed by solving the following linear system

$$\mathbf{F}^{\mathrm{T}}\mathbf{F}\hat{x} = \mathbf{F}^{\mathrm{T}}y \,. \tag{5.5}$$

In some cases, there may be reason to believe that some observed elements are more reliable than others, and hence should contribute more to the final estimate. This information can be incorporated by modifying (5.4) to include a diagonal positive-definite weighting matrix $\mathbf{W}$ as

$$\min_x (\mathbf{F}x - y)^{\mathrm{T}}\mathbf{W}(\mathbf{F}x - y) \,. \tag{5.6}$$

The solution is then given by solving

$$(\mathbf{F}^{\mathrm{T}}\mathbf{W}\mathbf{F})\hat{x} = \mathbf{F}^{\mathrm{T}}\mathbf{W}y \,. \tag{5.7}$$

Although (5.3) included a disturbance component (in the form of η), the parameter estimates computed in (5.5) or (5.7) made no explicit use of this quantity. However, we can often model the noise characteristics of the underlying sensor using a statistical model and recast our original estimation problem to incorporate this information. One common method is to compute the *maximum-likelihood estimate* (MLE), which is a value $\hat{x}$ such that

$$p(y|\hat{x}) = \max_x p(y|x) \,. \tag{5.8}$$

For the linear additive model of (5.3), the likelihood function can be expressed in a particularly simple form. Suppose that η is described by a fixed, known probability density function D. The likelihood function is then given by

$$p(y|x) = D(y - \mathbf{F}x) \,. \tag{5.9}$$

The MLE can be related to the previous least-squares method as follows. Suppose that $\eta \sim N(0, \mathbf{\Lambda})$, where N denotes a multivariate Gaussian density function with (mean) 0 and covariance $\mathbf{\Lambda}$. Upon observing that the maximizing the value of the likelihood function is equivalent to minimizing the negative log of the likelihood function, a short series of calculations shows that the optimal maximum-likelihood estimate is computed by weighted least squares with $\mathbf{W} = \mathbf{\Lambda}^{-1}$.

Finally, there is often a reason to include the idea that some parameters are more likely a priori to occur as others. For example, when observing a car driving on an expressway, a velocity of 60 mph is much more likely than either 20 or 300 mph. This information can be captured in *prior statistics* on the unknown value x.

Given a prior probably density on x, $p(x)$, *Bayes theorem* states that

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\int p(y|x)p(x)\,\mathrm{d}x} \,. \tag{5.10}$$

The *maximum a posteriori probability* (MAP) estimate is the value $\hat{x}$ such that

$$p(\hat{x}|y) = \max_x p(x|y) \,. \tag{5.11}$$

In general, the solution to this optimization problem can be quite complex. Rather than pursue this course further, we consider another alternative. Namely, provided the second moments of $p(x|y)$ exist, it is possible to produce a *least-squares* estimate, in a statistical sense, by solving the following optimization problem over an unknown function δ

$$\min_{\delta} E\,\|\delta(y) - x\|^2 \,. \tag{5.12}$$

That is, the best function δ is one that produces an estimate of x from y with *minimum mean-square error* (MMSE). Thus, the estimator δ is often referred to as an MMSE estimator.

It can be shown that, in the general case, the *optimal* decision rule δ^* is the conditional mean [5.22]

$$\delta^*(y) = E\,[x \mid y] \,. \tag{5.13}$$

Unfortunately, this expression, as with the MAP estimate defined above, can be extremely difficult to compute for the general case. Later we consider methods for computing approximations to (5.13). For now, we again consider our previous linear observation model (5.3) (without B). Additionally, we suppose that x and η are *independent* random variables with finite second moments, and both are *zero-mean* random variables. Note that the latter is not really a restriction since it can be accomplished by simply defining a new variable $x' = x - E[x]$. Finally, we will consider only *linear* functions δ, that is, we can write $\hat{x} = \delta(y) = \mathbf{K}y$.

With this, (5.12) can be expanded as

$$\begin{aligned} E\,\|\delta(y) - x\|^2 &= E\,\|\mathbf{K}y - x\|^2 \\ &= E\,\|\mathbf{K}(\mathbf{F}x + \eta) - x\|^2 \\ &= E\,\|(\mathbf{K}\mathbf{F} - I)x\|^2 + E\,\|\mathbf{K}\eta\|^2 \\ &= \mathrm{tr}\left[(\mathbf{K}\mathbf{F} - I)\mathbf{\Lambda}(\mathbf{K}\mathbf{F} - I)^{\mathrm{T}} + \mathbf{K}\mathbf{\Sigma}\mathbf{K}^{\mathrm{T}}\right] . \end{aligned} \tag{5.14}$$

Here, the independence of x and η and the fact that they are both zero mean has eliminated several terms. The final step makes use of the fact that $\|x\|^2 = \mathrm{tr}(xx^{\mathrm{T}})$.

Taking derivatives with respect to $\mathbf{K}$ and setting them equal to zero yields the solution

$$\mathbf{K} = \mathbf{\Lambda}\mathbf{F}^{\mathrm{T}}(\mathbf{F}\mathbf{\Lambda}\mathbf{F}^{\mathrm{T}} + \mathbf{\Sigma})^{-1} \,. \tag{5.15}$$

Thus, in this case the optimal estimate is given by a linear function of the observation, where the linear term depends only on the variance of the underlying random variables and the linear term defining the observation system.

If x is not zero-mean, but has mean μ, it is not hard to show that the optimal estimate is

$$\hat{x} = \mathbf{K}y + (\mathbf{I} - \mathbf{K}\mathbf{F})\mu \,, \tag{5.16}$$

and that the variance of the estimate Λ^+ is

$$\mathbf{\Lambda}^+ = (\mathbf{I} - \mathbf{K}\mathbf{F})\,\mathbf{\Lambda} \,. \tag{5.17}$$

The interested reader may wish to work this out for a few simple cases, for example, if $\mathbf{\Lambda} = \mathbf{\Sigma}$ and $\mathbf{F} = \mathbf{I}$, $\mathbf{K} = 1/2\mathbf{I}$ and thus $\hat{x} = y + \mu$ – a simple average – with variance $\mathbf{\Lambda}^+ = 1/2\mathbf{\Lambda}$.

When both the observation noise and prior statistics are Gaussian distributions, then it can be shown that the solution we have derived is also the MAP estimate for the unknown x [5.22].

The Kalman Filter

With this as background, we are now in a position to define the discrete-time Kalman–Bucy filter [5.32] for linear systems. Consider the following time-series model

$$x_{t+1} = \mathbf{G}x_t + w_t \,, \tag{5.18}$$

$$y_t = \mathbf{F}x_t + \eta_t \,, \tag{5.19}$$

where $\mathbf{G}$ is an $n \times n$ matrix describing the system time evolution and x_0 is distributed according to a Gaussian distribution with mean $\hat{x}_0$ and variance $\mathbf{\Lambda}_0$. In addition w_t and η_t are zero-mean Gaussian independent random variables for all t, w_t is independent of $w_{t'}$ for all $t \neq t'$, and likewise η_t is independent of $\eta_{t'}$ for all $t \neq t'$. Finally, η_t has variance $\mathbf{\Sigma}_t$ and w_t has variance $\mathbf{\Omega}_t$.

Given an observation y_1 it is possible, using the derivation of the previous section, to compute an updated estimate $\hat{x}_1$ with variance $\mathbf{\Lambda}_1$. Note, that the solution is a linear combination of two Gaussian random variables: the observation value y_1 and the prior estimate $\hat{x}_0$. As any linear combination of Gaussian random variables is also a Gaussian random variable, it follows that the updated estimate is also Gaussian.

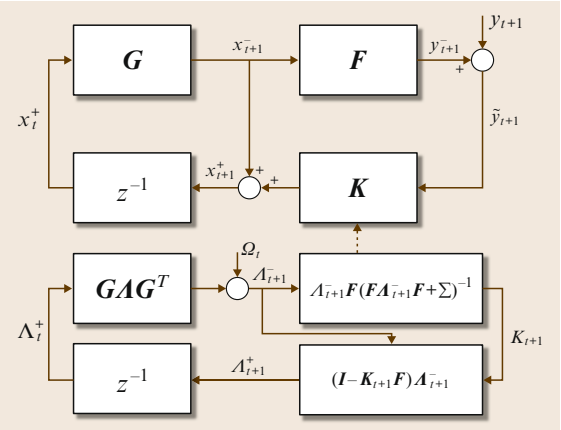


Fig. 5.13 A summary of the Kalman filter

Now, we add one additional step: projection through the dynamics model. To describe this, superscripts minus and plus will denote before and after the estimation step, respectively. Thus, given an estimate $\hat{x}_t^+$ with variance $\mathbf{\Lambda}_t^+$, the projection ahead one time step produces

$$\hat{x}_{t+1}^- = \mathbf{G}x_t^+ \,, \tag{5.20}$$

$$\mathbf{\Lambda}_{t+1}^- = \mathbf{G}\mathbf{\Lambda}_t^+\mathbf{G}^{\mathrm{T}} + \Omega_t \,. \tag{5.21}$$

At this point, a new observation y_{t+1} is acquired and the cycle repeats. The summarization of the complete Kalman filtering algorithm for linear systems is shown in Fig. 5.13.

It is possible to show that the Kalman filter is the *optimal* filter, under the stated assumptions, in the mean-square sense. It is also the optimal *linear* filter when either or both Gaussian assumptions do not hold.

Nonlinear Estimation Techniques for Sequential Data

The results of the previous subsection presume a linear form for the relationship between the observation and system state, additive noise, and a linear relationship describing the state evolution. Furthermore, the stated results are globally optimal for systems with Gaussian observation and driving noise, but are only the best *linear* estimator if the noise sources are non-Gaussian.

As noted at the outset, the more general nonlinear (discrete-time) system description is

$$\begin{aligned} x_{t+1} &= g_t(x_t) + w_t \,, \\ y_t &= f_t(x_t) + \eta_t \,, \end{aligned} \tag{5.22}$$

where, for the moment, the noise model continues to be additive.

Although this model contains nonlinear elements, it is still possible to apply a variant of the Kalman filter, the extended Kalman filter (EKF) by making use of the Taylor-series expansion of the nonlinear elements about the current estimates. Let $\mathbf{J}_f$ (resp. $\mathbf{J}_g$) denote the Jacobian matrix of the function f (resp. g). Supposing that an estimate at time step $t-1$ exists, the first-order expansion of (5.22) about this point yields

$$x_{t+1} = g_t(\hat{x}_{t-1}) + \mathbf{J}_g(\hat{x}_{t-1})(x_t - \hat{x}_{t+1}) + w_t \,, \tag{5.23}$$

$$y_t = f_t(\hat{x}_{-1}) + \mathbf{J}_f(\hat{x}_{t-1})(x_t - \hat{x}_{t-1}) + \eta_t \,. \tag{5.24}$$

Rearranging yields a linear form appropriate for the previously defined Kalman filter

$$\tilde{x}_{t+1} = x_{t+1} - g_t(\hat{x}_{t-1}) + \mathbf{J}_{g_t}\hat{x}_{t-1} = \mathbf{J}_{g_t}x_t + w_t \,, \tag{5.25}$$

$$\tilde{y}_t = y_t - f_t(\hat{x}_{t-1}) + \mathbf{J}_{f_t}\hat{x}_{t-1} = \mathbf{J}_{f_t}x_t + \eta_t \,. \tag{5.26}$$

In this form, $\tilde{x}$ and $\tilde{y}$ are new *synthetic* state and observation variables, $\mathbf{J}_g(\hat{x}_{t-1})$ plays the role of G, and $\mathbf{J}_f(\hat{x}_{t-1})$ plays the role of F.

It is worth noting that the EKF iterations are essentially a form of weighted Newton iterations (i. e., an iterative nonlinear estimation method). As a result, it is often useful to iterate more than once *on the same observation* while holding the variance terms fixed. This allows the estimator to converge to a solution in the presence of large disturbances or significant nonlinearities. Only after convergence are the variance terms updated. This version of the Kalman filter is referred to as the *iterated extended Kalman filter* (IEKF).

5.4.2 Other Approaches to Estimation

In the previous section, we reviewed a common and widely used estimation method. However, there are several alternative methodologies for solving parameter estimation problems. Here we briefly introduce two: sequential importance sampling and graphical models.

Sequential Importance Sampling

Much of the discussion heretofore has centered around the notion of approximating everything known about the system state using an estimated mean and covariance. An alternative presents itself by simply going back to Bayes theorem which states, in general, that

$$p(x_n|y_1, y_2 \ldots y_n) = \frac{p(y_1, y_2 \ldots y_n|x_n)p(x_n)}{p(y_1, y_2 \ldots y_n)} \,. \tag{5.27}$$

Assuming that y_n is independent of all prior observations and states given x_n, and that x_n is independent of x_{n-k} for $k > 1$ given x_{n-1}, this expression simplifies

to

$$p(x_n|x_{n-1}, y_n) = \frac{p(y_n|x_n)p(x_n \mid x_{n-1})}{p(y_n \mid x_{n-1})} . \tag{5.28}$$

Recall that the optimal mean-square estimate is given by the conditional mean which, in this case, is

$$\delta^*(y_n) = E\left[x_n \mid y_n\right] . \tag{5.29}$$

In fact, we essentially showed that the Kalman filter is a special case of this result for linear systems corrupted by Gaussian noise.

The difficulty in implementing this procedure in the general case ultimately comes down to the problem of representing and computing with the distributions that arise in the nonlinear, non-Gaussian case. However, suppose that the heretofore continuous variable x_n only took on a discrete set of values. In this case, computing Bayes theorem and other associated statistical quantities reduces to a straightforward set of computations on this discrete set of variables. This can be simply done for any distribution and any set of transformations.

Sequential important sampling (also known as particle filtering, condensation, and a variety of other names) is a way of applying this approach to continuous variables in a statistically sound manner. In order to perform sequential importance sampling, it is assumed that:

1. It is possible to *sample from* the likelihood function $P(y_n \mid x_n)$, and
2. It is possible to *sample from* the dynamical model $P(x_n \mid x_{n-1})$.

Note the emphasis on sampling – there is no need to explicitly exhibit an analytical form of the likelihood function or of the dynamical model.

Given this, sequential important sampling, in its simplest form, can be written as follows:

1. Let $\pi_{n-1} = \{\langle x^k_{n-1}, w^k_{n-1}\rangle, k = 1, 2, \ldots N\}$ represent a set of sample points x^k_{n-1} together with a set of weights w^k_{n-1} with $\sum w^k_{n-1} = 1$.
2. Compute a new set of N samples $\pi^-_{n-1} = \{\langle x^k_n, 1/N\rangle, k = 1, 2, \ldots N\}$ as:
   a) Choose a sample point x^{k-1}_{n-1} with probability proportional to its weight w^{k-1};
   b) Sample from $P(x_n \mid x^k_{n-1})$ given x^k_n with weight $1/N$;
3. Compute $\pi_n = \{\langle x^k_n, P(y_n \mid x^k_n)\rangle, k = 1, 2, \ldots N\}$.

It is easy to see that this set of steps is now in the form of a recursive filter. Furthermore, at any time any statistic of the associated distribution can be approximated from the set of samples and associated weights.

Sampling-based filters of this form have found wide applicability in a variety of challenging areas where linear estimation techniques do not suffice. These techniques have been particularly successful, for problems with low state dimension (typically $n \leq 3$) and well-constrained dynamics. For higher-dimensional problems or systems exhibiting high dynamic variability, the number of particles necessary to obtain good approximations can become prohibitively large. However, even in these cases, sampling-based systems can sometimes be engineered to produce acceptably good results.

Graphical Models

Graphical models are a class of models that represent dependence and independence relationships among a set of variables. Common examples of graphical models include Bayes nets, influence diagrams, and neural nets. Graphical models are quite general – indeed much of this chapter could have been written by first defining graphical models, and exploring specializations that lead to the Kalman Filter, for example. Here, for reasons of space, we focus on Bayes nets as a specific example of graphical models.

A Bayesian network is a directed acyclic graph consisting of nodes representing random variables, and directed arcs representing probabilistic relationships between pairs of random variables. Let parents(X) denote the set of nodes which have arcs terminating at X, and let $X_1, X_2, \ldots, X_N$ be the N random variables in the graph. Then we can write

$$P(X_1, X_2, \ldots, X_N) = \prod_{i=1}^{N} P(X_i \mid \text{parents}(X_i)) . \tag{5.30}$$

For example, a Bayesian network representing a mobile robot performing localization is shown in Fig. 5.14. This graphical model encodes the sequential form of the problem and is thus an example of a so-called *recurrent*

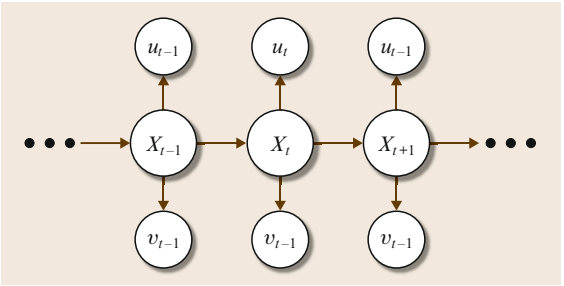


Fig. 5.14 An example of robot localization expressed as a graphical model

network. More discussion of such models can be found in [5.33].

The structure of a Bayesian network encodes various independence relationships among variables. By exploiting these independence relationships, it is possible to design efficient inference algorithms. In particular, graphs which are acyclic even in their undirected form (referred to as *polytrees*) admit linear time inference algorithms. More general graphs can be solved using various types of iterative methods. In particular, if the distributions in the network are of a continuous type, variations on sequential importance sampling can be used to solve problems in an approximate sense [5.34].

Conditional Random Fields

In many cases of interest, including many of the examples in this chapter, the end goal is to infer or predict a value or label from observed data. We might then frame the problem by exploring the joint distribution $P(X, Y)$ where X represents some data that is *observed* and Y is what we would like to infer. Recall that

$$P(X, Y) = P(Y|X)P(X) \, .$$

If we take $X = x$ for some observed values x, then we see that $P(X)$ becomes constant, and inferring a value for Y depends only on $P(Y|X)$. If we were to apply a Bayes Net to this problem, the model would represent the complete joint probability distribution on X and Y, what is referred to as a *generative model*. But, if we know X is always observed, then much of this structure is irrelevant to our problem – we don't care about the probabilistic structure of X. This observation has given rise to a specialization of graphical models referred to as *Conditional Random Fields*, or CRFs for short.

The immediate value of CRFs is their economy and expressivity compared to graphical models. This has immediate positive implications for the complexity of both learning and inference. Traditionally CRF models are learned using maximum likelihood-based methods using gradient descent or other unconstrained optimization techniques. However, recent methods like Cutting Planes [5.35] and Block Coordinate Frank Wolfe [5.36] pose it as a constrained optimization problem in the form of a Structural Support Vector Machine. These techniques tend to be more computationally efficient and are often more accurate.

CRFs have proven to be very general, and are now extremely widely used for image processing, natural language processing, video processing – nearly any problem where there is a series of data elements

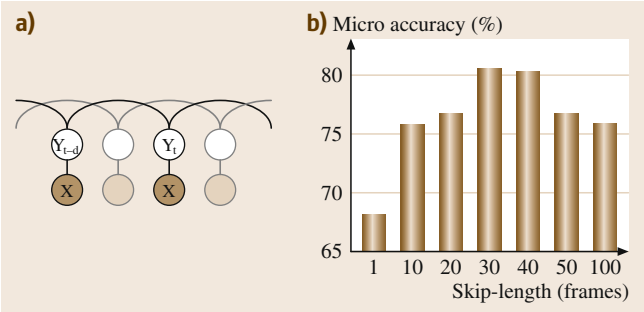


Fig. 5.15 **(a)** A skip chain CRF for inferring symbolic labels (Y) from robot kinematic and video data (X) acquired while a user performed a surgical training task. **(b)** The change in classification accuracy as a function of the skip length (images courtesy of Colin Lea)

from which some prediction is to be performed. For example, Fig. 5.15 shows the graphical structure of a skip-chain CRF designed to compute gesture labels from kinematic and video data acquired from a surgical robot [5.37]. This can be viewed as a discriminative generalization of a Hidden Markov Model (a generative model) that is designed to capture dependencies over a specified period of time (the *skip*). The right side shows the change in labeling performance as a function of the skip length which is now a *tunable* parameter of the model.

An in-depth discussion of CRFs goes well beyond this chapter. The interested reader is referred to [5.38, 39] to learn more about the underlying theory and application of CRFs. Because of their high interest, there are a number of open-source packages for developing and applying CRFs including PyStruct [5.40] for Python and CRF++ for C++.

5.4.3 Robust Estimation Methods

In our previous discussions, we generally assumed that all of the data was *good*, meaning that it was perhaps corrupted by noise but ultimately carried information about the problem at hand. However, in many cases, the data may contain so-called *outliers* – data points that are either much more highly corrupted than typical data, or which are completely spurious. For example, in our mapping application we might occasionally obtain range data through multiple reflections. Thus, while scanning a straight wall, most of the points would lie on a straight line, but occasionally we would have a data point that has a completely inconsistent range value.

Many common estimation methods are quite sensitive to data outliers. Consider a very simple case: estimating a single scalar value x by averaging a series of observations $X_1, X_2, \ldots X_N$. Then we can write our

estimate $\hat{x}$ as

$$\hat{x} = \sum_{i=1}^{N} X_i / N \,. \tag{5.31}$$

Now, without loss of generality, suppose that X_N is an outlier. We can rewrite the above as

$$\hat{x} = \sum_{i=1}^{N-1} X_i / n + X_N / n \,. \tag{5.32}$$

It is now easy to see that we can produce *any* value of $\hat{x}$ by manipulating X_n. In short, a single outlier can create an arbitrarily poor estimate. More generally, the solution to any least-squares problem, e.g., estimating a line from laser range data, takes the general form $\hat{x} = \mathbf{M}y$. By the same argument as above, it is easy to show that any least-squares solution is likewise susceptible to outliers.

The field of *robust statistics* studies the problem of estimation or decision making when the underlying data are contaminated by outliers. In robust statistics, there are two important concepts: the *breakdown point* and *influence function*. The breakdown point is the proportion of outliers (i. e., data with arbitrarily large errors) that an estimator can tolerate before producing arbitrarily large errors in an estimate. We argued above that least-squares methods have a breakdown point of 0% since the estimate can be perturbed arbitrarily far by a single observation. By comparison, we might compute an estimate by taking the median of the data, which has a breakdown point of 50% – up to half of the data can be outliers and meaningful results may still be produced.

Whereas the breakdown point quantifies *how many* outliers can be tolerated, the influence function quantifies *how much* an outlier affects an estimate. In the case of least squares, the influence function is linear. One way of creating new estimators with better robustness is the method of *M-estimators* [5.21]. To produce an M-estimate, we consider the following minimization problem

$$\min_{\hat{x}} \sum_{i=1}^{N} \rho(\hat{x}, y_i) \,. \tag{5.33}$$

Note that defining $\rho(a, b) = (a - b)^2$ leads to a least-squares solution. However, we can now choose other functions with better resistance to outliers. Fig. 5.16 shows three common examples.

Note that, in general, the optimization of (5.33) is nonlinear and the result will often not exist in closed form. Interestingly, it is often possible to solve this problem using the method of *iteratively reweighted least squares* (IRLS) [5.30, 41]. The idea behind IRLS is quite simple. Recall that in (5.7) we introduced a weighting matrix $\mathbf{W}$. Suppose that, through some means, we knew which data points were outliers. In this case, we could simply set the weights for those points to zero, and the result would be the least-squares estimate on the remaining (good) data.

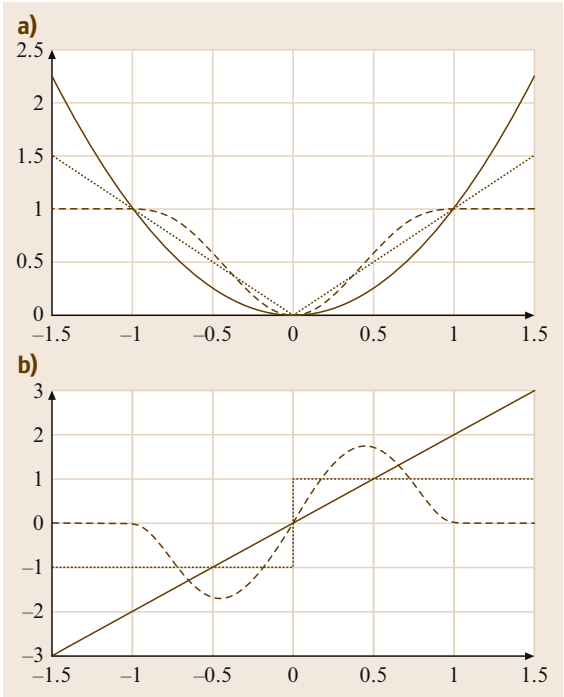


Fig. 5.16 **(a)** Three common robust M-estimation functions: the square function, the absolute value, and the Tukey biweight function. **(b)** The corresponding influence functions

In IRLS, we alternate between hypothesizing outliers (through reweighting) and solving to produce a solution (through least squares). Typically, the weight for a point depends on the residual error of the estimate. That is, suppose we compute

$$r = y - \mathbf{F}\hat{x} \,. \tag{5.34}$$

Let $\psi(y) = \mathrm{d}\rho / \mathrm{d}x \,|_{\hat{x}}$; then we can set $\mathbf{W}_{i,i} = \psi(y)/r_i$. It can be shown that in many cases this form of weighting will lead to convergence. An example of using IRLS techniques for video tracking is shown in Fig. 5.17.

Voting-Based Methods

Another common method for dealing with outliers is to choose a set of data and let it *vote* for a result. We dis-

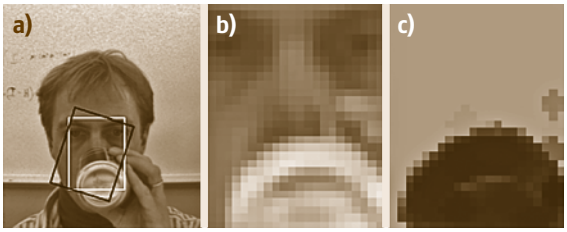


Fig.5.17a–c An example of using an M-estimate implemented via IRLS for visual tracking (after [5.42]). **(a)** Results of a face tracker in a single frame of video. The black frame corresponds to a tracking algorithm without outlier rejection and the white frame corresponds to the algorithm with outlier rejection. **(b)** Magnified view of the region in the white frame; **(c)** the corresponding weighting matrix in which darker areas mark outliers

cuss two common methods: RANSAC [5.11] and least median of squares (LMedS) [5.43].

In both cases, we start with the idea that, amidst all of the data (including outliers), there is an estimate that is consistent with the *good* data. The problem is to choose that estimate. Consider, however, our problem of estimating a line from laser data, and suppose we have 100 laser points. All we really need is to choose two points correctly, fit a line, and then count how many other points are consistent with this line. If we (conservatively) estimate that 3/4 of the data is good, then the odds of choosing two *good* points is 9/16, or equivalently, the odds of one or both points being outliers is 7/16. If we now repeat this process a few (e.g., ten) times, then the odds that *all* of our choices are bad is $(7/16)^{10} = 0.025\%$. To put it in other terms, there is a 99.975% chance we have chosen a *good* pair of points.

How do we decide to accept a sample? In RANSAC, we *vote* by counting the number of samples that are consistent with an estimate to within a given distance threshold. For example, we would choose points that are within a fixed distance to the line we estimated. We choose the candidate estimate with the largest number of votes. In LMedS, we instead compute the median distance of all of the samples to the line. We then choose the estimate with the least median value.

It is not hard to see that LMedS has a breakdown point of 50% of the data. RANSAC, on the other hand, can have a breakdown point that is potentially larger, but it requires the choice of a threshold. RANSAC also has the advantage that, once the inliers are identified, it is possible to compute a least-squares estimate from them, thus reducing the noise in the estimate.

Both RANSAC and LMedS can also provide good starting solutions for a robust iterative method such as IRLS.

5.4.4 Data Association Techniques

The previous section considered the case where there is a *known* relationship between observations and a quantity to be estimated. However, as was illustrated in our initial mobile robot mapping problem, it may be the case that we also have to compute this correspondence in conjunction with estimation. In this case, an essential step in estimation is the *data association* problem: producing a correspondence between the observed data and quantities to be estimated.

The literature on this problem is enormous; here we will focus on a few specific methods that have found wide use. We will also separate our discussion into *causal* (or sequential) association methods commonly used when filtering time-series data and *noncausal* (or batch) methods that can be used when the complete data set is available for processing. The latter is typically treated with methods for data *clustering*.

In both cases, we can extend our previous models and notation to include uncertainty as to the underlying source of the data. To this end, we will use a superscript on quantities to denote the observation model. Thus, our observation model becomes

$$x_{t+1}^k = g^k(x_t^k) + w_t^k \,, \tag{5.35}$$

$$y_t^k = f_t^k(x_t^k) + \eta_t^k \,, \tag{5.36}$$

where $k = 1 \ldots M$.

Clustering on Batch Data

Following the same course as our previous discussion on point estimation, let us first consider the case where we do not make any statistical assumptions about the data, and we have no system dynamics. Thus, we are simply given the observations $y_1, y_2, \ldots, y_M$. We have unknown underlying parameters $x^1, x^2, \ldots, x^N$ (for the moment, we take N as known). Our goal it to compute an *association mapping* π such that $\pi(j) = k$ if and only if y_j arose from the model parameters x^k.

k-Means Clustering

The k-means algorithm for clustering and data association is simple, well established, and forms a good starting point for our discussion. Here, we assume that $f(x) = x$ – that is, we are provided with noisy observations of the underlying state vectors. The k-means algorithm then proceeds as follows:

1. Pick N cluster centers $\{\hat{x}^i\}$.
2. For each observation y_j, associate it with the closest cluster center, that is, set $\pi(j) = i$, where

$$d(\hat{x}^i, y_j) = \min_k d(\hat{x}^k, y_j) \tag{5.37}$$

for some distance function d (typically the Euclidean distance).

3. Estimate the mean of the observation associated with each cluster center as

$$\hat{x}^i = \sum_{j,\pi(j)=i} y_j \,. \tag{5.38}$$

4. Repeat steps 2 and 3.

In many cases and with good initialization, k-means works quite well. However, it can also fail to produce good clusters, and there is no guarantee that it will even converge to a solution. It is common to repeat the algorithm several times from different initial conditions and take the result that has the *best* outcome. Note also that the extension to linear observation models is straightforward by including $\mathbf{F}$ in (5.3) by defining

$$d(\hat{x}^i, y_j) = \|\mathbf{F}\hat{x}^i - y_j\| \tag{5.39}$$

and replacing (5.38) with the corresponding least-squares estimator. Going a step further, if we have a statistical model for observed data, then we could make use of the likelihood function introduced earlier and define $d(\hat{x}^i, y_j) = p(y_j|\hat{x}^i)$ and make use of the MLE in (5.38).

One disadvantage of the k-means algorithm is that, even when we have known statistical models, it is not guaranteed to converge. However, a variation, known as *expectation maximization*, can be shown to converge.

Expectation Maximization for Data Association and Modeling

The expectation-maximization (EM) algorithm [5.44] is a general statistical technique for dealing with missing data. In previous discussion, we made use of maximum-likelihood estimation to maximize the conditional probability of observed data given a set of unknown parameters. However, our use of MLE presumed that we had complete knowledge of the data. In particular, we knew the association between the data elements and models.

Let use now assume that some of our data is missing. To this end, define $\mathcal{Y}_\mathrm{O}$ and $\mathcal{Y}_\mathrm{U}$ as the *observed* and *unobserved* data, respectively. We then note that we can write

$$p(\mathcal{Y}_\mathrm{O}, \mathcal{Y}_\mathrm{U}|x) = p(\mathcal{Y}_\mathrm{U}|\mathcal{Y}_\mathrm{O}, x)p(\mathcal{Y}_\mathrm{O}|x) \,. \tag{5.40}$$

Suppose now that we make a guess for $\hat{x}$, and we have a distribution over the unknown data $\mathcal{Y}_\mathrm{U}$ (where this comes from we will discuss in a minute). It follows that we could compute the *expected value* of the log-likelihood function (recall that maximizing the log likelihood is equivalent to maximizing the likelihood) as

$$Q(x, \hat{x}) = E_{\mathcal{Y}_\mathrm{U}}\left[\log p(\mathcal{Y}_\mathrm{O}, \mathcal{Y}_\mathrm{U}|x)|\mathcal{Y}_\mathrm{O}, \hat{x}\right] \,. \tag{5.41}$$

Note that we differentiate between the fixed value $\hat{x}$ that is usually needed to define the distribution over the unknown data and the unknown x of the log-likelihood function.

Ideally, we would then like to choose values for x that make Q large. Thus, we can choose a new value according to the iterative rule

$$\hat{x}_i = \arg\max_x Q(x, \hat{x}_{i-1}) \,. \tag{5.42}$$

What can be shown is that this iteration will converge to some local maximum of the objective function Q. It is important to note that there is no guarantee that this is, however, the *global* maximum.

How do we connect this with clustering? We consider the observed data to be just that, the data we have observed. Let the unobserved data be the *association values* $\pi(j)$, $j = 1, 2, \ldots M$ that determine which model the observed data items originate from. Note that this is a discrete random variable. Let us further assume that N underlying clusters are distributed according to a Gaussian distribution with mean x_i and covariance Λ_i. Let the unconditional probability that a particular data item y_j comes from cluster i be α_i. The unknown parameters are then $\theta = \{x_1, x_2, \ldots, x_N, \Lambda_1, \Lambda_2, \ldots, \Lambda_N, \alpha_1, \alpha_2, \ldots, \alpha_N\}$. We now use $^-$ and $^+$ to denote prior and updated parameter estimates, respectively. For conciseness, we also define $w_{i,j} = p(\pi_j = i|y_j, \theta)$ and we use a superscript $^+$ to denote updated parameter estimates. Then, after a series of calculations [5.44], the EM algorithm for data clustering becomes

E-Step:

$$w_{i,j} = \frac{p(y_j|\pi(j) = i, \theta)\alpha_i}{\sum_i p(y_j|\pi(j) = i, \theta)\alpha_i} \,. \tag{5.43}$$

M-Step:

$$\hat{x}_i^+ = \frac{\sum_j y_j w_{i,j}}{\sum_j w_{i,j}} \,, \tag{5.44}$$

$$\Lambda_i^+ = \frac{\sum_j y_j (y_j)^t w_{i,j}}{\sum_j w_{i,j}} \,, \tag{5.45}$$

$$\alpha_i^+ = \frac{\sum_j w_{i,j}}{\sum_i \sum_j w_{i,j}} \,. \tag{5.46}$$

From this, we can see that EM produces a type of soft clustering, as opposed to k-means which produces

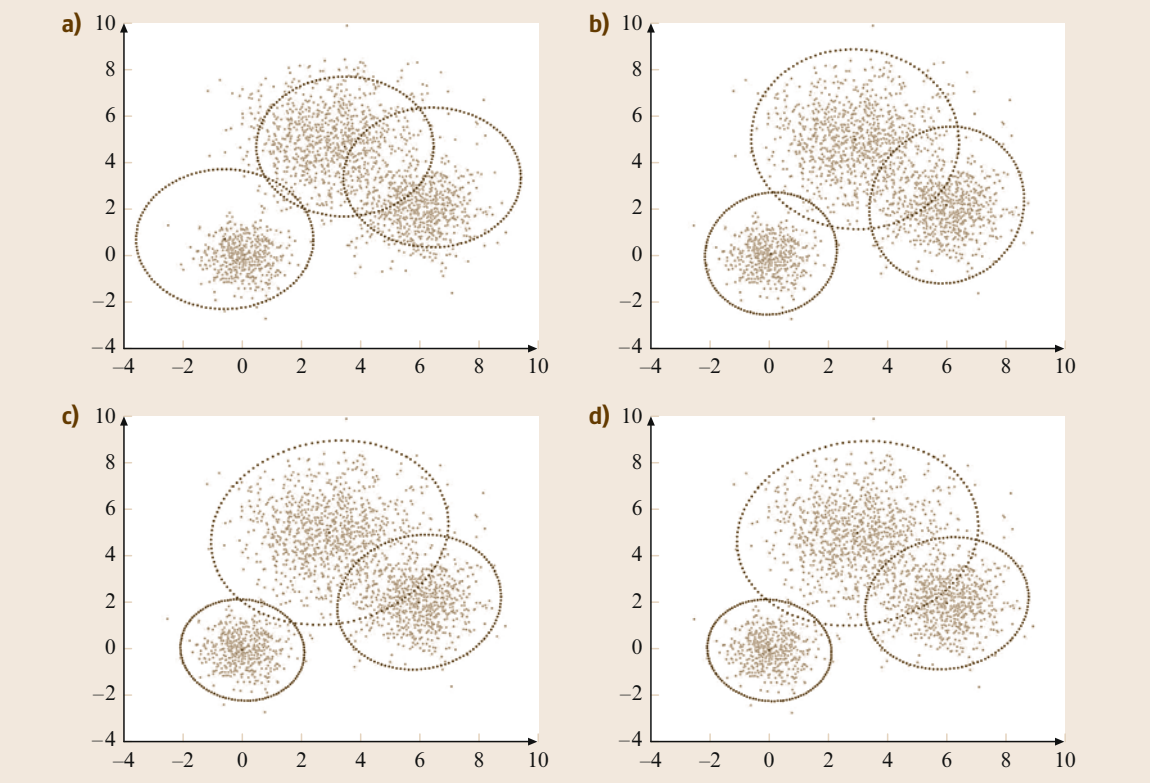


Fig.5.18a–d An example of clustering using expectation maximization. The figures are the results at iterations **(a)** 1, **(b)** 2, **(c)** 5, and **(d)** 10

specific decisions (in terms of $w_{i,j}$) as to which cluster an observation belongs. In fact, the result of estimation is the maximum-likelihood estimate of a *Gaussian mixture model* which has the form

$$p(y|\theta) = \sum_j \alpha_j N(y|\hat{x}_j, \mathbf{\Lambda}_j) \,, \tag{5.47}$$

where $N(\cdot)$ denotes a Gaussian density function. Fig. 5.18 shows the results of executing the EM algorithm on data sampled from a Gaussian mixture model.

Recursive Filtering

In the batch methods described above, we do not have a priori information on state parameters. In the case of recursive filtering, we have the advantage that prior state estimates, $\hat{x}_t^k$ and Λ_t^k, are available for processing at time $t+1$. As before, for data $y_t^i, i = 1 \ldots N$, the problem is to determine a mapping $\pi : \{1 \ldots N\} \rightarrow \{1 \ldots M\}$ which *associates* data element i to model $k = \pi(i)$. In some cases, it is also useful to include an *outlier process* to handle data that comes from no known model. For this purpose, we can include 0 in the range of the function, and use the mapping to zero as an outlier.

Nearest-Neighbor Association

Analogous to k-mean clustering, a simple way of producing a data association is to compute the data association value as

$$\pi(i) = \arg\min_j d(\mathbf{F}^j \hat{x}^j, \hat{y}^i) \,. \tag{5.48}$$

However, nearest-neighbor methods do not take into account what we know about either the sensor data or the estimate. That is, we may have a very very good estimate of some model i and a very very bad estimate for some other model j. If a sensor observation is equidistance between them, does it make sense to flip a coin? Odds are that it is more likely to come from j (with a larger variance) than i (with a smaller variance).

A commonly used measure that can take this into account is the *Mahalanobis* distance [5.45]. The idea is to weight each value by its variance as

$$m(y_1, y_2) = (y_1 - y_2)(\mathbf{\Lambda}_1 + \mathbf{\Lambda}_2)^{-1}(y_1 - y_2)^{\mathrm{T}} \,. \tag{5.49}$$

Thus, distances are scaled inversely with uncertainty. In the case above, the observation with a higher variance would produce the smaller distance, as desired.

Even using this as a weighting method, it is still possible that we will make an error in data association. From an estimation point of view, this will introduce an outlier in the estimation process with, as discussed above, potentially disastrous results. Another approach, analogous to IRLS, is instead to weight the data based on the distance to a model. This leads naturally to the notion of a *data association filter*. We refer the reader to [5.20] for extensive discussions of these techniques.

5.4.5 Modeling Sensors

To this point, we have introduced several sensing modalities, and we have discussed several methods for estimation. However, the latter often rely on having statistical models for the former. Thus, no chapter on sensing and estimation would be complete without a short discussion of modeling sensors.

Developing a sensor model potentially involves four major elements:

1. Creating a physical model
2. Determining a sensor calibration
3. Determining an error model
4. Identifying failure conditions.

The physical model is the relationship f between the underlying quantities of interest (x) and the available data (y). In many cases, this relationship is obvious, e.g., the distance from a laser sensor to a surface in the world. In others, it may be less so, e.g., what is the right model relating intensities in multiple camera images to the distance to an observed point? In some cases, it may be necessary to include computational processes, e.g., feature detection and correspondence, in the sensor model.

Once a physical model is determined, there is often a process of sensor calibration. Such procedures are typically specific to the sensor in question, for example, the imaging geometry of a perspective camera system requires identification of two scale parameters (governing image scale) and the location of the optical center (two additional parameters). There are also often lens distortion parameters. These parameters can only be determined by a careful calibration procedure [5.7].

Once a calibrated physical sensor model is available, determining an error model typically involves performing an identification of the statistical parameters. Ideally, the first step is to determine an empirical distribution on errors. However, this can often be difficult, as it requires knowing accurate *ground truth* for the underlying unknown parameters. This often requires the development of a laboratory setup that can simulate the expected sensing situation.

Given such an empirical distribution, there are several important questions, including:

1. Are observations statistically independent?
2. Is the error distribution unimodal?
3. Can the essential aspects of the empirical error be captured using common statistical quantities such as the data variance?

We refer the reader to books on statistics and data modeling [5.46] for further information on this topic.

Finally, it is important to understand when sensors can and cannot provide reliable data, for example, a laser sensor may be less accurate on dark surfaces than on light ones, cameras do not produce meaningful data if the lighting is too bright or too dark, and so forth. In some cases, there are simple clues to these conditions (e.g., simply looking at the intensity histogram of a camera image can quickly determine if conditions are suitable for processing). In some cases it is only possible to detect conditions in context (e.g., two range sensors disagree on the distance to a surface). In some cases failure is only detectable in retrospect, e.g., after a 3-D surface model is built it is apparent that a hypothesized surface would be occluded by another and must therefore be a multiple reflection. In a truly robust sensing system, all available possibilities for verifying sensor operation should be exploited.

5.4.6 Other Uncertainty Management Methods

Due to the limitations of space, we have necessarily limited our discussion to cover the most commonly used sensing and estimation methods. It is important to note that many other alternative uncertainty management methods have been proposed and employed with success.

For example, if it is known that sensing error is bounded, constraint-based methods can be quite effective at performing point estimation [5.47, 48]. Alternatively, if only partial probability models can be identified, Dempster–Shafer methods can be employed to make judgments [5.49].

Fuzzy logic allows graded membership of a set. With fuzzy set theory it is possible to have partial membership. As an example in classification of data it might be difficult to select between two categories such as *average* and *tall* and gradual shifts may make sense. Such methods have for example been used for situation assessment and navigation as reported by [5.50] for the DAMN architecture.

5.5 Representations

Sensor data can be used directly for control but it is also used for estimation of the state of the robot and/or the world. The definition of *state* and the appropriate methods for estimation are closely related to the representation adopted for the application.

There are a rich variety of possible world representations including most typical geometric elements such as points, curves, surfaces, and volumes. A fundamental aspect in robotics is the concept of rigid-body *pose*. The pose of a robot or an entity in the world is characterized by position and orientation with respect to a reference frame.

In general, pose is represented by the pair $(\mathbf{R}, \boldsymbol{H})$. Here $\mathbf{R}$ is the orientation of the object represented by a rotation matrix with respect to a reference frame. Similarly, $\boldsymbol{H}$ represents the translation of the object with respect the reference frame. There is a rich set of potential representations for the transformation between reference frames as detailed in the chapter on Kinematics (Chap. 2) and in [5.51].

Sensory data is acquired in a local sensor reference frame, for example, a sonar transducer, a laser scanner, and a stereo imaging system would all measure distances to surfaces in the world relative to their own frame. However, if the goal is to combine this information into a common world model, the data must be transformed into a robot-centered reference frame, or possibly into a fixed world (inertial) reference frame. In particular, the world-centered reference frame enable simple transfer across robot motions and communication to other robots and/or users.

For the purposes of discussion, most representations for the integration of sensor data can be categorized into four general classes of models:

- Raw sensor data models
- Grid-based models
- Feature-based models
- Symbolic or graphical models.

Naturally, it is also possible to combine elements of these four categories to achieve hybrid models of the environment.

5.5.1 Raw Sensor Representations

For simple feedback control [5.52] it is common to integrate raw sensory data directly into the control system, as in many cases it is unnecessary to have a *world model* for the control. For example, proprioceptive sensing is often used in this manner: basic trajectory control makes direct use of encoder information from joints, and force control operates directly from force or torque information from force sensors.

Raw sensor models are less common with exteroceptive sensing, but there are cases where it can be useful. One example is mobile robot mapping from dense point data. This approach has in particular been made popular for laser range sensors, where scan alignment is used for the generation of point-based world models. The work by [5.53, 54] demonstrates how a number of laser range scans can be combined into a joint model of the environment. More formally a scan of the environment at time t is represented as a point set

$$\mathcal{P}_t = \{p_i = (\rho_i, \theta_i) | i \in 1 \ldots N\} \,. \tag{5.50}$$

Two different scans $\mathcal{P}_t$ and $\mathcal{P}_{t+1}$ are then aligned through a standard rigid body transformation. The estimation of the transformation is typically achieved through use of the ICP algorithm [5.15]: assume that $\boldsymbol{H}^{[0]}$ is an initial estimate of the transformation between the two point sets and that $||\boldsymbol{p}_t - \boldsymbol{p}_{t+1}||$ is the Euclidean distance between a point from $\mathcal{P}_t$ and a point from $\mathcal{P}_{t+1}$. If furthermore CP is a function to locate the closest point from one set in the other set, then let C be the set of point correspondences between the two sets. Through iterations of the following algorithm,

1. Compute $C_k = \cup_{i=1}^{N} \{\boldsymbol{p}_i, CP[\boldsymbol{H}^{[k-1]}(\boldsymbol{p}_i, \mathcal{P}_{t+1})]\}$,
2. Estimate the $\boldsymbol{H}^{[k]}$ that minimizes the LSQ error between the points in C_k until the error has converged

an estimate of the scan alignment can be found and a joint model of the environment can be constructed.

The model is simple to construct and well suited for integration of sensor data from a single modality. Typically the model does not include information about uncertainty and, as the model grows the complexity, $O(\sum_t |\mathcal{P}_t|)$ becomes an issue.

5.5.2 Grid-Based Representations

In a grid-based representation the world is tessellated into a number cells. The cells can contain information about environmental features such as temperature, obstacles, force distribution, etc. The dimensionality of the grid is typically two or three, depending on the application. The tessellation can either be uniform or tree based using quad-tree or oct-trees [5.55]. The tree-based methods are in particular well suited for handling of inhomogeneous and large-scale data sets. In a grid model each cell contains a probability over the parameter set. As an example, when using

the grid model for representation of a physical environment, the cell specifies *occupied* (O) or *free* (F) and the cell encodes the probability P(occupied). Initially where there is no information the grid is initialized to $P(\mathrm{O}) = 0.5$ to indicate unknown. It is further assumed that sensor models are available that specify $P(R|S_{ij})$, i. e., the probability of detection objects for a given sensor and location. Using Bayes theorem (5.10) it is now possible to update the grid model according to

$$p_{ij}(t+1) = \frac{P(R|S_{ij}=O)p_{ij}(t)}{P(R|S_{ij}=O)p_{ij}(t) + P(R|S_{ij}=F)(1-p_{ij}(t))} ,$$

where p_{ij} is computed across the grid model whenever new data are acquired.

The grid-based model has been widely used in mobile robotics [5.56, 57] and in medical imaging where image volumes are quite common [5.58]. Volume models can be relative large. As an example a millimeter-resolution grid model of the human head requires 4 GB of storage, and thus demands significant computational resources for maintenance.

5.5.3 Feature Representations

Both the raw sensor representation and the grid-based models contain a minimum of abstraction for the sensory data. In many cases there is an interest in extracting features from the sensor data to reduce the storage requirement and only preserve data that are invariant across motion of the platform or external objects. Features span most standard geometric entities such as points ($\boldsymbol{p}$), lines ($\boldsymbol{l}$), planes ($\boldsymbol{N}, \boldsymbol{p}$), curves ($\boldsymbol{p}(s)$), and more general surfaces. For estimation of properties of the external world there is a need for a hybrid model in which collections of features are integrated into a unified model of state.

In general a point is represented in $\mathcal{R}(3)$. Sensors have associated noise and, consequently, in most cases points have an associated uncertainty, typically modeled as Gaussian with mean μ and standard deviation σ. The estimation of the statistics is achieved using first- and second-order moments.

Line features are more difficult to represent. The mathematical line can be represented by the vector pair $(\boldsymbol{p}, \boldsymbol{t})$, i. e., a point on the line and the tangent vector. In many practical applications the line has a finite extent, and there is a need to encode the length of the line, which can be achieved using end points, start point, tangent, and length. In some cases it is advantageous to have a redundant representation of the line model to simplify updating and matching. The relation between end-point uncertainties and other line parameters can be derived analytically, as described in [5.59]. The estimation of line parameters is often based on the previously describe RANSAC method through the use of the *Hough* transform [5.10], which is another voting-based method.

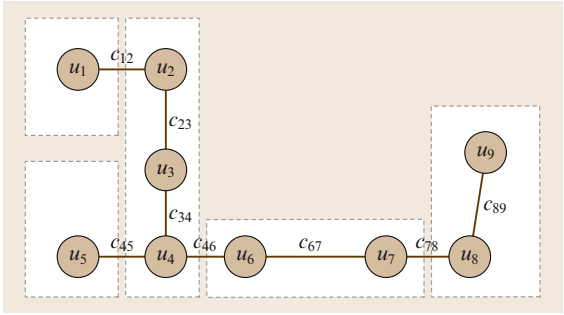


Fig. 5.19 A topological map of a spatial environment

For more complex feature models such as curves or surfaces there is a corresponding need to utilize detection methods that facilitate robust segmentation of features, and estimation of the associated uncertainty. A comprehensive description of such methods is available from [5.44].

5.5.4 Symbolic/Graph-Based Models

All of the representations presented in Sects. 5.5.1–5.5.3 are parametric in nature with limited associated semantics. Methods for the recognition of structures, spaces, locations, and objects have seen major recent progress in particular due to advances in statistical learning theory [5.12, 60]. Consequently, today there exist a variety of methods for the recognition of complex structures in sensor data, such as landmarks, road surfaces, body structures, etc. Given the availability of recognized structures it is possible to represent the environment using the previously discussed graphical models. In general a graph is composed of a set of nodes N and a set of edges E that connect nodes. Both nodes and edges can have attributes associated such as labels and distances. One example of a graph structure is a topological map of the environment as shown in Fig. 5.19. The graph representation could also be a semantic model of the environment (objects and places) or a representation of the composition of an object to assembled.

In terms of model updating semantic/graph-based representations can take advantage of recent advances in Bayesian reasoning as presented by *Pearl* [5.61], and exemplified in [5.62].

5.6 Conclusions and Further Readings

Sensing and estimation continues to be a challenging and very active area of robotics research. Several areas of sensing such as computer vision and medical imaging are themselves large and diverse research areas. At the same time, new fundamental and applied techniques in estimation continue to be developed. Indeed, it is fair to say that perception continues to be one of the most challenging areas of robotics research.

Given this wealth of activity, no single chapter can hope to cover all of the material that can be useful in the development of sensor-based robotics. However, the methods that have been presented here are representative of the most commonly used techniques in robotics. In particular, linear techniques such as the Kalman filter continue to form the backbone of perceptive robotics. Part C of the handbook provides more in-depth coverage of several of the key topics in sensing and estimation.

For the reader wishing to learn more, general discussion on the design, physics, and use of a rich variety of sensors can be found in the Handbook of Modern Sensors [5.3]. A discussion of sensors for mobile robots can be found in [5.63], though significant advances have been achieved since the book was published more than a decade ago. Sensing and estimation using computer vision is described in detail in [5.64] and [5.65].

The basic estimation theory is covered in a number of excellent text books. Much of the detection and linear estimation theory is covered in depth in [5.20] and [5.66]. General statistical estimation is covered in [5.12] and [5.13] and the more recently updated version [5.44]. Robust methods are described in detail in [5.21, 43]. In-depth coverage of estimation methods for mobile systems is also covered in [5.33].

References

5.1 D.C. Marr: *Vision* (Freeman, Bedford 1982)
5.2 R. Siegwart, I.R. Nourbakhsh, D. Scaramuzza: *Introduction to Autonomous Mobile Robots*, Intelligent Robotics and Autonomous Systems (MIT Press, Cambridge 2011)
5.3 J. Fraden: *Handbook of Modern Sensors: Physic, Design and Applications*, 2nd edn. (Springer, New York 1996)
5.4 H. Yousef, M. Boukallel, K. Althoefer: Tactile sensing for dexterous in-hand manipulation in robotics – A review, Sensors Actuators A: Physical **167**(2), 171–187 (2011)
5.5 M.I. Tiwana, S.J. Redmond, N.H. Lovell: A review of tactile sensing technologies with applications in biomedical engineering, Sensors Actuators A: Physical **179**, 17–31 (2012)
5.6 G. Dissanayaka, S. Sukkarieh, E. Nebot, H. Durrant-Whyte: The aiding of a low-cost strapdown inertial measurement unit using vehicle model constraints for land vehicle applications, IEEE Trans. Robot. Autom. **17**(5), 731–748 (2001)
5.7 Z. Zhang: A flexible new technique for camera calibration, IEEE Trans. Pattern. Anal. Mach. Intell. **22**(11), 1330–1334 (2000)
5.8 D. Burschka, J. Geiman, G.D. Hager: Optimal landmark configuration for vision-based control of mobile robots, Proc. Int. Conf. Robot. Autom. ICRA (2003) pp. 3917–3922
5.9 J. Baker, N. Anderson, P. Pilles: Ground-penetrating radar surveying in support of archeological site investigations, Comput. Geosci. **23**(10), 1093–1099 (1997)
5.10 P.V.C. Hough: A method and means for recognizing complex patterns, U.S. Patent 306 9654 (1962)
5.11 M.A. Fischler, R.C. Bolles: Random Sample concensus: A paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM **24**, 381–395 (1981)
5.12 T. Hastie, R. Tibshirani, J. Friedman: *The Elements of Statistical Learning*, Springer Series in Statistics (Springer, Berlin, Heidelberg 2002)
5.13 R.O. Duda, P.E. Hart: *Pattern Classification and Scene Analysis* (Wiley-Interscience, New York 1973)
5.14 R. Vidal, Y. Ma, J. Piazzi: A new GPCA algorithm for clustering subspaces by fitting, differentiating and dividing polynomials, Proc. Int. Conf. Cumput. Vis. Pattern Recog. **1**, 510–517 (2004)
5.15 P. Besl, N.D. McKay: A method for registration of 3-D shapes, IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 239–256 (1992)
5.16 F. Dellaert, S. Seitz, C. Thorpe, S. Thrun: Special issue on Markov chain Monte Carlo methods, Mach. Learn. **50**, 45–71 (2003)
5.17 A. Gelb (Ed.): *Applied Optimal Estimation* (MIT Press, Cambridge 1974)
5.18 D. Simon: *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches* (Wiley, New York 2006)
5.19 A. Doucet, N. de Freitas, N. Gordon: *Sequential Monte Carlo Methods in Practice* (Springer, Berlin, Heidelberg 2001)
5.20 Y. Bar-Shalom, T. Fortmann: *Tracking and Data Association* (Academic, New York 1988)
5.21 P.J. Huber: *Robust Statistics* (Wiley, New York 1981)
5.22 J.O. Berger: *Statistical Decision Theory and Bayesian Analysis*, 2nd edn. (Springer, New York 1985)
5.23 G.D. Hager: *Task-Directed Sensor Fusion and Planning* (Kluwer, Boston 1990)

5.24 S. Abrams, P.K. Allen, K. Tarabanis: Computing camera viewpoints in a robot work-cell, Int. J. Robot. Res. **18**(3), 267–285 (1999)

5.25 M. Suppa, P. Wang, K. Gupta, G. Hirzinger: *C-space exploration using noisy sensor models*, *Proc* (IEEE, Int. Conf. Robot. Autom 2004) pp. 1927–1932

5.26 G.S. Chirikjian, A.B. Kyatkin: *Engineering Applications of Noncommutative Harmonic Analysis* (CRC, Boca Raton 2000)

5.27 J.C. Kinsey, L.L. Whitcomb: Adaptive identification on the group of rigid body rotations and its application to precision underwater robot navigation, IEEE Trans. Robot. **23**, 124–136 (2007)

5.28 P.J. Bickel, K.A. Doksum: *Mathematical Statistics*, 2nd edn. (Prentice-Hall, Upper Saddle River 2006)

5.29 G. Strang: *Linear Algebra and its Applications*, 4th edn. (Brooks Cole, New York 2005)

5.30 P. McCullagh, J.A. Nelder: *Generalized Linear Models*, 2nd edn. (Chapman Hall, New York 1989)

5.31 E.L. Lehmann, G. Casella: *Theory of Point Estimation* (Springer, New York 1998)

5.32 R.E. Kalman: A new approach to linear filtering and prediction problems, Transactions of the ASME, J. Basic Eng. **82**, 35–45 (1960)

5.33 S. Thrun, D. Fox, W. Burgard: *Probabilistic Robotics, Autonomous Robotics and Intelligent Agents* (MIT Press, Cambridge 2005)

5.34 C. Bishop: *Pattern Recognition and Machine Learning* (Springer, New York 2006)

5.35 T. Joachims, T. Finley, C.-N.J. Yu: Cutting-plane training of structural SVMs, Mach. Learn. **77**(1), 27–59 (2009)

5.36 S. Lacoste-Julien, M. Jaggi, M. Schmidt, P. Pletscher: Block-coordinate Frank-Wolfe optimization for structural SVMs, Proc. Int. Conf. Mach. Learn. (2013) pp. 53–61

5.37 L. Tao, L. Zappella, G.D. Hager, R. Vidal: Surgical gesture segmentation and recognition, Med. Image Comput. Comput.-Assisted Intervent., MICCAI 2013 (2013) pp. 339–346

5.38 C. Sutton, A. McCallum: An introduction to conditional random fields for relational learning. In: *Introduction to Statistical Relational Learning*, ed. by L. Getoor, B. Taskar (MIT Press, Cambridge 2006) pp. 93–128

5.39 C. Sutton, A. McCallum: An introduction to conditional random fields, Found. Trends Mach. Learn. **1**, 2055–2060 (2010)

5.40 C.A. Müller, S. Behnke: PyStruct-learning structured prediction in python, J. Mach. Learn. Res. **1**, 2055–2060 (2013)

5.41 J.W. Hardin, J.M. Hilbe: *Generalized Linear Models and Extensions*, 2nd edn. (Stata, College Station 2007)

5.42 G.D. Hager, P.N. Belhumeur: Efficient region tracking of with parametric models of illumination and geometry, IEEE Trans. Pattern Anal. Mach. Intell. **20**(10), 1025–1039 (1998)

5.43 P.J. Rousseauw, A. Leroy: *Robust Regression and Outlier Detection* (Wiley, New York 1987)

5.44 R.O. Duda, P.E. Hart, D.G. Stork: *Pattern Classification*, 2nd edn. (Wiley, New York 2001)

5.45 P.C. Mahalanobis: On the generalised distance in statistics, Proc. Nat. Inst. Sci. India **12**, 49–55 (1936)

5.46 J. Hamilton: *Time Series Analysis* (Princeton Univ. Press, Princeton 1994)

5.47 S. Atiya, G.D. Hager: Real-time vision-based robot localization, IEEE Trans. Robot. Autom. **9**(6), 785–800 (1993)

5.48 G.D. Hager: Task-directed computation of qualitative decisions from sensor data, IEEE Trans. Robot. Autom. **10**(4), 415–429 (1994)

5.49 G. Shafer: *A Mathematical Theory of Evidence* (Princeton Univ. Press, Princeton 1976)

5.50 J. Rosenblatt: *DAMN: A distributed architecture for mobile navigation*, *AAAI 1995* (Spring, Symposium on Lessons Learned for Implementing Software Architectures for Physical Agents 1995) pp. 167–178

5.51 R.M. Murrey, Z. Li, S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1993)

5.52 K.J. Åström, B. Wittenmark: *Adaptive Control*, 2nd edn. (Addison-Wesley, Reading 1995)

5.53 S. Gutmann, C. Schlegel: AMOS: Comparison of scan-matching approaches for self-localization in indoor environments, 1st Euromicro Conf. Adv. Mobile Robotics (1996) pp. 61–67

5.54 S. Gutmann: *Robust Navigation for Autonomous Mobile Systems*, Ph.D. Thesis (Alfred Ludwig University, Freiburg 2000)

5.55 H. Samet: The quadtree and related hierarchical data structures, ACM Comput. Surv. **16**(2), 187–260 (1984)

5.56 A. Elfes: Sonar-based real-world mapping and navigation, IEEE Trans. Robot. Autom. **3**(3), 249–265 (1987)

5.57 A. Elfes: *A Probabilistic Framework for Robot Perception and Navigation*, Ph.D. Thesis (Carnegie Mellon University, Pittsburgh 1989)

5.58 M.R. Stytz, G. Frieder, O. Frieder: Three-dimensional medical imaging: Algorithms and computer systems, ACM Comput. Surv. **23**(4), 421–499 (1991)

5.59 R. Deriche, R. Vaillant, O. Faugeras: From Noisy Edges Points to 3D Reconstruction of a Scene: A robust approach and its uncertainty analysis. In: *Theory and Applications of Image Analysis*, (World Scientific, Singapore 1992) pp. 71–79

5.60 V.N. Vapnik: *Statistical Learning Theory* (Wiley, New York 1998)

5.61 J. Pearl: *Probabilistic Reasoning in Intelligent Systems* (Morgan Kaufmann, New York 1988)

5.62 M. Paskin: *Thin Junction Tree Filters for Simultaneous Localisation and Mapping*, Ph.D. Thesis (University of California, Berkley 2002)

5.63 H.R. Everett: *Sensors for Mobile Robots: Theory and Application* (Peters, London 1995)

5.64 D. Forsyth, J. Ponce: *Computer Vision – A Modern Approach* (Prentice-Hall, Upper Saddle River 2003)

5.65 R. Hartley, A. Zisserman: *Multiple View Geometry in Computer Vision* (Cambridge Univ. Press, Cambridge 2000)

5.66 S. Blackman, R. Popoli: *Design and Analysis of Modern Tracking Systems* (Artech House, London 1999)

Multimedia Contents

6. Model Identification

John Hollerbach, Wisama Khalil, Maxime Gautier

This chapter discusses how to determine the kinematic parameters and the inertial parameters of robot manipulators. Both instances of model identification are cast into a common framework of least-squares parameter estimation, and are shown to have common numerical issues relating to the identifiability of parameters, adequacy of the measurement sets, and numerical robustness. These discussions are generic to any parameter estimation problem, and can be applied in other contexts.

For kinematic calibration, the main aim is to identify the geometric Denavit–Hartenberg (DH) parameters, although joint-based parameters relating to the sensing and transmission elements can also be identified. Endpoint sensing or endpoint constraints can provide equivalent calibration equations. By casting all calibration methods as closed-loop calibration, the calibration index categorizes methods in terms of how many equations per pose are generated.

Inertial parameters may be estimated through the execution of a trajectory while sensing one or more components of force/torque at a joint. Load estimation of a handheld object is simplest because of full mobility and full wrist force-torque sensing. For link inertial parameter estimation, restricted mobility of links nearer the base as well as sensing only the joint torque means that not all inertial parameters can be identified. Those that can be identified are those that affect joint torque, although they may appear in complicated linear combinations.

6.1 **Overview** 113
6.2 **Kinematic Calibration** 115
6.2.1 Serial-Link Robot Manipulators 116
6.2.2 Parallel Manipulator Calibration 120
6.3 **Inertial Parameter Estimation** 122
6.3.1 Link Inertial Parameter Estimation. 122
6.3.2 Load Inertial Parameter Estimation 124
6.3.3 Identification of Total Joint Drive Gains 126
6.3.4 Link Parameter Estimation for More Complex Structures 126
6.4 **Identifiability and Numerical Conditioning** 127
6.4.1 Identifiability 128
6.4.2 Observability 131
6.4.3 Scaling 132
6.4.4 Recursive Least Squares and the Kalman Filter 134
6.5 **Conclusions and Further Reading** 135
6.5.1 Relation to Other Chapters 135
6.5.2 Further Reading 136
Video-References 136
References 137

6.1 Overview

There are many different kinds of models in robotics, whose accurate identification is required for precise control. Examples from the previous chapters include sensor models, actuator models, kinematic models, dynamic models, and flexibility models. System identification is the general field concerned with the process of identifying models from measurements. Generally speaking, there are two types of models: parametric and nonparametric models. Parametric models are described by a few parameters, which are adequate to

characterize the accuracy of a model throughout the working range. Examples include sensor gain and offset, the DH parameters for links, and rigid-body inertial parameters. Parametric models are particularly appropriate for robotics, whose components are manmade and whose properties are controlled and understood.

Nonparametric models include the impulse response and the Bode plot for linear systems, and Wiener and Volterra kernels for nonlinear systems [6.1]. A nonparametric model can be used as a stepping stone towards identifying a parametric model; for example, a Bode plot (graph of amplitude and phase versus input frequency) is often used to decide on model order, such as whether an actuator should be modeled as a second- or third-order system. Otherwise, nonparametric models are required when the system's properties are so complicated that a few lumped parameters do not suffice. This is particularly true for biological systems.

This chapter will describe the parametric calibration of the following kinds of models:

1. Kinematic parameters. Kinematic calibration is the process of locating coordinate systems of objects relative to each other. These objects may be isolated from each other, or they may be linked by joints. Examples include:
   - Locating a robot relative to a global reference frame.
   - Locating a stereo vision system relative to a robot.
   - Locating a grasped object relative to the manipulator's gripper frame.
   - Locating neighboring coordinate systems of links of a robot manipulator.
2. Rigid-body inertial parameters. These parameters are required to predict the driving forces and torques to move objects and manipulators.

Suppose there are N_{par} parameters combined into an $N_{\text{par}} \times 1$ parameter vector $\boldsymbol{\phi} = \{\phi_1, \ldots, \phi_{N_{\text{par}}}\}$. The parameters may appear linearly or nonlinearly in the model.

$$\text{Linear model:} \qquad \boldsymbol{y}^l = \mathbf{A}^l \boldsymbol{\phi} \,, \tag{6.1}$$

$$\text{Nonlinear model:} \qquad \boldsymbol{y}^l = \boldsymbol{f}(\boldsymbol{x}^l, \boldsymbol{\phi}) \,, \tag{6.2}$$

where $\boldsymbol{y}^l = \{y_1^l, \ldots, y_M^l\}$ is the $M \times 1$ vector of output variables and $\boldsymbol{x}^l = \{x_1^l, \ldots, x_n^l\}$ is the vector of input variables. For the linear model, $\mathbf{A}^l$ is an $M \times N_{\text{par}}$ matrix whose entries A_{ij}^l are functions of the input variables $\boldsymbol{x}^l$. Any entry A_{ij}^l may be a complicated nonlinear function of $\boldsymbol{x}^l$, but it evaluates to just a number. For the nonlinear model, an explicit equation is shown in which the input variables appear in a nonlinear function $\boldsymbol{f} = \{f_1, \ldots, f_M\}$. Implicit nonlinear models $\boldsymbol{f}(\boldsymbol{y}^l, \boldsymbol{x}^l, \boldsymbol{\phi}) = \mathbf{0}$ may also appear in calibration [6.2]; they are handled similarly to explicit nonlinear models (Sect. 6.2.2). There may be P different measurements; a particular measurement is indicated by the superscript $l = 1, \ldots, P$.

For linear models, information from different measurements is combined by stacking the P (6.1)

$$\boldsymbol{y} = \mathbf{A}\boldsymbol{\phi} \,, \tag{6.3}$$

where $\boldsymbol{y} = \{\boldsymbol{y}^1, \ldots, \boldsymbol{y}^l\}$ is an $MP \times 1$ vector of all output measurements and $\mathbf{A} = \{\mathbf{A}^1, \ldots, \mathbf{A}^l\}$ is $MP \times N_{\text{par}}$. The parameters are estimated by ordinary least squares

$$\boldsymbol{\phi} = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\boldsymbol{y} \,. \tag{6.4}$$

In statistics, the matrix $\mathbf{A}$ is called the regressor matrix and the least-squares solution is called regression [6.3]. An example of a linear model is the rigid-body model for inertial parameters.

The Gauss–Newton method [6.3] is typically employed to estimate the nonlinear model (6.2). First, absorb the input variables $\boldsymbol{x}^l$ (which can be considered as a number of constants) into the nonlinear function $\boldsymbol{f}^l$, now given a superscript. The model is linearized through a Taylor series expansion at a current estimate $\boldsymbol{\phi}^k$ of the parameters at iteration k

$$\begin{aligned} \boldsymbol{y}_{\text{c}}^l &= \boldsymbol{f}^l(\boldsymbol{\phi}^k + \Delta\boldsymbol{\phi}) \\ &= \boldsymbol{f}^l(\boldsymbol{\phi}^k) + \left.\frac{\partial \boldsymbol{f}^l(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}}\right|_{\boldsymbol{\phi}=\boldsymbol{\phi}^k} \Delta\boldsymbol{\phi} \\ &\quad + \text{higher order terms} \\ &\approx \boldsymbol{f}^l(\boldsymbol{\phi}^k) + \mathbf{A}^l\Delta\boldsymbol{\phi} \,, \end{aligned} \tag{6.5}$$

where $\boldsymbol{y}_{\text{c}}^l$ are the computed values of the output variables and $\mathbf{A}^l = \partial \boldsymbol{f}^l / \partial \boldsymbol{\phi}$ is a Jacobian matrix evaluated at $\boldsymbol{\phi}^k$. Higher-order terms in the Taylor series are ignored, yielding the linearized form (6.5). The bold assumption is now made that a correction $\Delta\boldsymbol{\phi}$ to the parameter estimate $\boldsymbol{\phi}^k$ causes the computed output variables to equal the measurements: $\boldsymbol{y}_{\text{c}}^l = \boldsymbol{y}^l$. Defining $\Delta\boldsymbol{y}^l = \boldsymbol{y}^l - \boldsymbol{f}^l(\boldsymbol{\phi}^k)$ as the error between the output measurement and the predicted output with the current model $\boldsymbol{\phi}^k$, the linearized (6.5) becomes

$$\Delta\boldsymbol{y}^l = \mathbf{A}^l\Delta\boldsymbol{\phi} \,. \tag{6.6}$$

The linearized equation is then stacked for P measurements for the estimation form

$$\Delta\boldsymbol{y} = \mathbf{A}\Delta\boldsymbol{\phi} \,. \tag{6.7}$$

A correction $\Delta\boldsymbol{\phi}$ to the parameter estimates is now found by ordinary least squares

$$\Delta\boldsymbol{\phi} = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\Delta\boldsymbol{y}\,. \tag{6.8}$$

This process is iterated with a new estimate $\boldsymbol{\phi}^{k+1} = \boldsymbol{\phi}^{k} + \Delta\boldsymbol{\phi}$ until the error $\Delta\boldsymbol{\phi}$ becomes sufficiently small. The Gauss–Newton method has quadratic convergence, which is very fast, provided that there is a good initial estimate $\boldsymbol{\phi}^{0}$ of the parameters and the nonlinearity is not too severe. An example of a nonlinear model is the kinematic model containing the DH parameters. The nonlinearity is mild because it is due to sines and cosines, so the Gauss–Newton method usually has good convergence properties.

Both for linear and nonlinear estimation, however, rank deficiencies and numerical ill-conditioning may cause problems in inverting $\mathbf{A}^{\mathrm{T}}\mathbf{A}$. Rank deficiencies may result from two problems:

1. Inadequate data. The quality of the data for estimating the parameters can be quantified by an observability index, such as the condition number [6.4] of the regressor matrix $\mathbf{A}$. A different choice of data to maximize the observability index might result in more robust estimates. Examples are choosing different poses for kinematic calibration, or different trajectories for inertial parameter estimation.
2. Unidentifiable parameters. Perhaps no set of data from an experiment can identify some of the parameters. A procedure has to be found to eliminate or circumvent the unidentifiable parameters. Parameter elimination is usually done by using the singular value decomposition of $\mathbf{A}$, while circumvention can be achieved by using a priori values and ridge regression. This does not mean that the parameters are intrinsically identifiable, only that the experimental setup precludes their determination; for example, only one of the 10 inertial parameters of the first link of a robot manipulator can be identified if the base of the manipulator is stationary. A different experimental setup involving accelerating the base of the manipulator and measuring the reaction forces and torques at the base would allow other parameters to be identified [6.5].

Ill-conditioning may result from poor scaling of measurements or parameters:

1. The least-squares estimation minimizes the error between the predicted and measured output vector. Components y_j^l of the output vector $\boldsymbol{y}^l$ may have different units and magnitudes, such as radians versus meters, in pose measurements for kinematic calibration. Moreover, not all components may be measured to the same level of accuracy. Normalizing the output vectors with an appropriately chosen weighting matrix may result in a better conditioned estimate.
2. The parameters may also have different units and different magnitudes. This can cause problems both in terms of convergence criteria and in terms of deciding which parameters to eliminate. Again, a weighting matrix for the parameters may be introduced to improve the conditioning.

These numerical issues are generic to any parameter estimation problem, and are discussed at the end of this chapter. The next sections discuss individual robot models and issues in putting them into parameter estimation form.

6.2 Kinematic Calibration

In general, the relative location between coordinate systems requires six geometric parameters to be calibrated (position plus orientation). If there are mechanical constraints between the relative movement of the coordinate systems, such as connection by a joint, fewer parameters are required. For a rotary joint connecting two links, whose axis is a line-bound vector, four geometric parameters are required. For a prismatic joint, whose axis is a free vector, only two geometric parameters describing orientation are required.

In addition, nongeometric parameters are required to model sensors and mechanical deflection:

- Joint angle sensors may require a gain and offset determination.
- Camera calibration using an undistorted pinhole camera model may require the determination of the focal length and an image sensor offset.
- Joint flexibility from gears leads to angle change due to gravity loading and the manipulator's own weight.
- Base flexibility results from nonrigid attachment of the robot to the environment. Depending on how the manipulator is outstretched, there will be a varying effect on endpoint location.

- Thermal effects and vibration may need to be modeled for very fine positioning control.

This section focuses on determining the geometric parameters and the sensor-based nongeometric parameters.

6.2.1 Serial-Link Robot Manipulators

The modified Denavit–Hartenberg (DH) parameters (Fig. 6.1) serve as the main geometric parameters (Sect. 2.4); the link transformation matrix is

$$
{}^{i-1}\mathbf{T}_i = \mathrm{Rot}(\boldsymbol{x}, \alpha_i)\mathrm{Trans}(\boldsymbol{x}, a_i)\mathrm{Trans}(\boldsymbol{z}, d_i)\mathrm{Rot}(\boldsymbol{z}, \theta_i) \; . \tag{6.9}
$$

- For an n-joint manipulator with rotary joint $i = 1, \ldots, n$, whose axis $\boldsymbol{z}_i$ is a line in space, all four parameters a_i, d_i, α_i, and θ_i have to be calibrated.
- For a prismatic joint i, whose axis $\boldsymbol{z}_i$ is a free vector, only two parameters describing its orientation (α_i and θ_i) are required. In principle, the axis $\boldsymbol{z}_i$ can be positioned anywhere in space. This means two DH parameters are arbitrary. One possibility is to intersect $\boldsymbol{z}_i$ with O_{i+1} [6.6, 7], which sets $d_{i+1} = 0$ and $a_{i+1} = 0$. While kinematically correct, this placement is nonintuitive in that it doesn't correspond to the physical location of the prismatic mechanism. It is possible to set a_{i+1} with a_i or θ_i to values that position $\boldsymbol{z}_i$ in the middle of the prismatic joint's mechanical structure.

In the case of nearly parallel neighboring axes, the common normal is poorly defined and the calibration is ill-conditioned. For this case, *Hayati* and *Mirmirani* [6.7] introduced an extra rotational parameter β_i about the $\boldsymbol{y}_{i-1}$ axis (Fig. 6.2).

Let $\boldsymbol{x}'_{i-1}$ lie along a line from O_i to axis $\boldsymbol{z}_{i-1}$, such that $\boldsymbol{x}'_{i-1}$ is perpendicular to $\boldsymbol{z}_i$; the intersection point defines the origin O_{i-1}. Two rotations are required to relate $\boldsymbol{z}_{i-1}$ to $\boldsymbol{z}_i = \boldsymbol{z}'_{i-1}$: a rotation α_i about $\boldsymbol{x}'_{i-1}$ maps $\boldsymbol{z}''_{i-1}$ to $\boldsymbol{z}_i$, and a rotation β_i about $\boldsymbol{y}_{i-1} = \boldsymbol{y}''_{i-1}$ maps $\boldsymbol{z}_{i-1}$ to $\boldsymbol{z}''_{i-1}$. The angle θ_i is now from $\boldsymbol{x}'_{i-1}$ to $\boldsymbol{x}_i$ about $\boldsymbol{z}_i$. The link transformation is

$$
{}^{i-1}\mathbf{T}_i = \mathrm{Rot}(\boldsymbol{y}, \beta_i)\mathrm{Rot}(\boldsymbol{x}, \alpha_i)\mathrm{Trans}(\boldsymbol{x}, a_i)\mathrm{Rot}(\boldsymbol{z}, \theta_i) \; . \tag{6.10}
$$

- For a rotary joint, the parameter β_i is calibrated instead of d_i.
- For a prismatic joint, the joint variable d_i has to be retained. As before, position $\boldsymbol{z}_i$ at some convenient location on link i, by specifying two parameters relative to the coordinate system $i+1$. Then proceed with the construction of the Hayati parameters for $d_i = 0$. As d_i changes, axis $\boldsymbol{x}_i$ is displaced along $\boldsymbol{z}_i$ relative to $\boldsymbol{x}'_{i-1}$ (not shown). The link transformation is

$$
{}^{i-1}\mathbf{T}_i = \mathrm{Rot}(\boldsymbol{y}, \beta_i)\mathrm{Rot}(\boldsymbol{x}, \alpha_i)\mathrm{Trans}(\boldsymbol{x}, a_i)\,\mathrm{Trans}(\boldsymbol{z}, d_i)\mathrm{Rot}(\boldsymbol{z}, \theta_i) \; . \tag{6.11}
$$

Although there are five parameters in this transformation, the process of setting the Hayati parameters for $d_i = 0$ is tantamount to setting the value of d_{i-1} to locate O_{i-1}, so there is no net increase in the number of parameters.

The procedures above set the coordinate systems in the intermediate links of a serial manipulator. Coordinate systems also have to be set in the base and end links, but the procedure for doing so depends on the external metrology system and on any physical constraints on the end link pose. The last frame may be n or $n+1$, while the first frame may be 0 or -1 (to keep consecutive numbers); examples appear below. Collect the unknown kinematic parameters into the vectors $\boldsymbol{a}$, $\boldsymbol{d}$, $\boldsymbol{\alpha}$, $\boldsymbol{\theta}$, and $\boldsymbol{\beta}$, and thence into the parameter vector $\boldsymbol{\phi} = \{\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{\alpha}, \boldsymbol{\theta}, \boldsymbol{\beta}\}$. The parameters $\boldsymbol{\phi}$ predict the

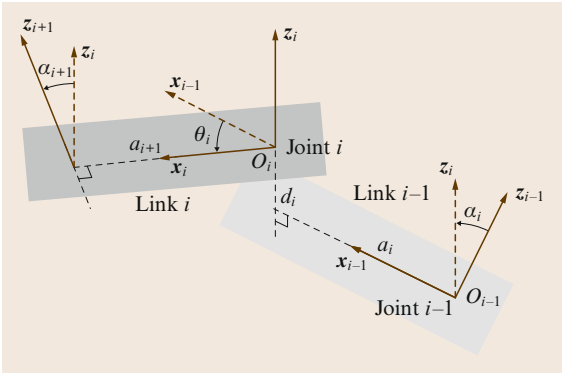


Fig. 6.1 The modified DH parameters

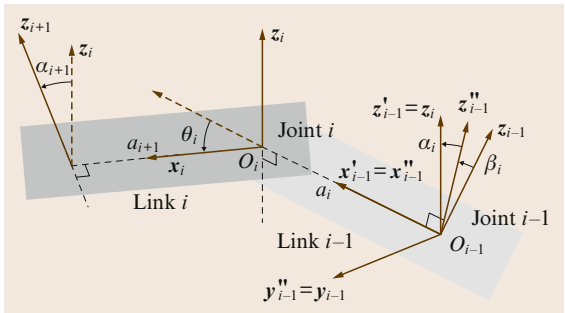


Fig. 6.2 The additional parameter β_i about $\boldsymbol{y}_{i-1}$ is employed for nearly parallel axes

position and orientation of the last coordinate system relative to the first, such as ${}^0\mathbf{T}_{n,c}$.

Not all six components of pose are necessarily used for calibration; the number can vary from one to six. Calibration proceeds by observing the error in the prediction of a certain number of pose components, then employing the nonlinear calibration method (6.7). There are two general methods for observing error:

- *Open-loop calibration* utilizes an external metrology system to measure the pose components. Because the manipulator is not in contact with the environment during this process, the method is termed open loop.
- *Closed-loop calibration* utilizes physical constraints on the end link pose to substitute for measurements. Deviations from the physical constraint represent the prediction error. Because of the contact with the physical constraint, the manipulator forms a closed loop with the ground.

Open-Loop Kinematic Calibration

The calibration literature contains a variety of metrology systems, which can be categorized based on the number of pose components measured [6.8]:

- *1 component*: The distance to a single point on the end link can be measured in a variety of ways, such as an instrumented ball bar [6.9], wire potentiometer [6.10], or laser displacement meter [6.11].
- *2 components*: A single theodolite has been employed to provide two orientation measurements [6.12]. A reference length had to be sighted for scaling.
- *3 components*: Laser tracking systems provide accurate three-dimensional (3-D) measurements by reflecting a beam off a retroreflector mounted on the end effector. The beam provides length information, while the gimbal drive for laser steering provides two orientation measurements [6.13]. Since the least precise part of this setup is the angle sensing, another approach is to employ three laser tracking systems and use only the length information. Commercial 3-D stereo-camera motion tracking systems also provide high-accuracy measurements of position.
- *5 components*: *Lau* et al. [6.14] presented a steerable laser interferometer with steerable reflector. With pitch and yaw measurements, the steerable interferometer yields all three components of position, while the steerable reflector yields two components of orientation.
- *6 components*: Full pose can be inferred from the 3-D position of multiple points on the last link measured with a stereo-camera system. A coordinate system fit to the cloud of points yields position plus orientation [6.15].
  Vincze et al. [6.16] measured full pose with a single-beam laser tracking system, by fitting the robot with a retroreflector mounted on a universal joint. Position is measured using interferometry, as usual. The novel aspect is orientation measurement, by imaging of the diffraction pattern of the edges of the retroreflector.

Examples are given for calibrating with three pose components and with all six pose components measured.

Point Measurement. The 3-D position of a particular point on the end link can be conveniently located by some form of stereo camera system. The camera system defines a global coordinate system relative to which frame 1 of the robot is located. To provide enough parameters, an intermediate coordinate system has to be introduced; this intermediate coordinate system has index 0, while the camera system is given index -1 to keep consecutive numbers. Two of the eight parameters are arbitrary. Figure 6.3 shows one possibility: z_0 is made parallel ($\alpha_0 = 0$) and coincident ($a_0 = 0$) with z_{-1}. The calibrated parameters are d_0, θ_0, a_1, d_1, α_1, and θ_1. In the case that z_{-1} is nearly parallel to z_0, the measurement coordinate system can be simply redefined to avoid using Hayati parameters; for example, y_{-1} can be redefined as z_{-1}.

In the end link, the origin O_n and axis x_n are unspecified, as are the associated parameters d_n and θ_n. Locating the measured point requires only three parameters, so the addition of a single coordinate system $n+1$ is required to provide an additional parameter.

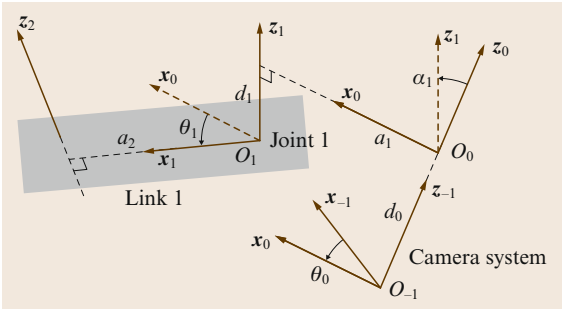


Fig. 6.3 A camera system (index−1) is located relative to the manipulator coordinate system 1 via an intermediate coordinate system 0

The measured point is defined as the origin O_{n+1} of the new coordinate system, and the normal from z_n that intersects O_{n+1} defines the x_n axis (Fig. 6.4). Three parameters are arbitrary; a simple choice is to make z_{n+1} parallel to z_n ($\alpha_{n+1} = 0$) and x_{n+1} collinear with x_n ($\theta_{n+1} = 0$ and $d_{n+1} = 0$). The calibrated parameters are a_{n+1}, d_n, and θ_n.

From the transformation

$$^{-1}\mathbf{T}_{n+1} = {}^{-1}\mathbf{T}_0 \dots {}^{n}\mathbf{T}_{n+1} ,$$

the position $^{-1}\boldsymbol{p}_{n+1}$ of the measured point relative to the camera frame is extracted. Collect the unknown kinematic parameters into vectors $\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{\alpha}, \boldsymbol{\theta}$, and $\boldsymbol{\beta}$, and thence into the parameter vector $\boldsymbol{\phi} = \{\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{\alpha}, \boldsymbol{\theta}, \boldsymbol{\beta}\}$. The nonlinear kinematic model analogous to (6.2) is

$$^{-1}\boldsymbol{p}^l_{n+1} = \boldsymbol{f}(\boldsymbol{q}^l, \boldsymbol{\phi}) , \quad l = 1, \dots, P , \tag{6.12}$$

where $\boldsymbol{q}^l$ is the vector of joint variables for pose l. To linearize this equation into estimation form (6.6), the associated Jacobians for each parameter type are calculated

$$\Delta^{-1}\boldsymbol{p}^l_{n+1} = {}^{-1}\boldsymbol{p}^l_{n+1} - {}^{-1}\boldsymbol{p}^l_{n+1,c} = \mathbf{J}^l \Delta\boldsymbol{\phi}$$
$$= \begin{pmatrix} \mathbf{J}^l_{\boldsymbol{a}} & \mathbf{J}^l_{\boldsymbol{d}} & \mathbf{J}^l_{\boldsymbol{\alpha}} & \mathbf{J}^l_{\boldsymbol{\theta}} & \mathbf{J}^l_{\boldsymbol{\beta}} \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{a} \\ \Delta\boldsymbol{d} \\ \Delta\boldsymbol{\alpha} \\ \Delta\boldsymbol{\theta} \\ \Delta\boldsymbol{\beta} \end{pmatrix} , \tag{6.13}$$

where a typical column i for each individual parameter Jacobian is derived from the screw parameters as if each parameter represented an active joint (Sect. 2.8.1)

$$\mathbf{J}^l_{a_i} = \begin{cases} {}^{-1}\boldsymbol{x}^l_{i-1} & \text{DH} \\ {}^{-1}\boldsymbol{x}'^l_{i-1} & \text{Hayati} \end{cases} , \tag{6.14}$$

$$\mathbf{J}^l_{d_i} = {}^{-1}\boldsymbol{z}^l_i , \tag{6.15}$$

$$\mathbf{J}^l_{\alpha_i} = \begin{cases} {}^{-1}\boldsymbol{x}^l_{i-1} \times {}^{-1}\boldsymbol{d}^l_{i-1,n+1} & \text{DH} \\ {}^{-1}\boldsymbol{x}'^l_{i-1} \times {}^{-1}\boldsymbol{d}^l_{i-1,n+1} & \text{Hayati} \end{cases} , \tag{6.16}$$

$$\mathbf{J}^l_{\theta_i} = {}^{-1}\boldsymbol{z}^l_i \times {}^{-1}\boldsymbol{d}^l_{i,n+1} , \tag{6.17}$$

$$\mathbf{J}^l_{\beta_i} = {}^{-1}\boldsymbol{y}^l_{i-1} \times {}^{-1}\boldsymbol{d}^l_{i-1,n+1} , \tag{6.18}$$

where $^{-1}\boldsymbol{d}_{i,n+1} = {}^{-1}\mathbf{R}_i {}^{i}\boldsymbol{p}_{n+1}$ refers the interorigin vector to coordinate system -1. Equation (6.13) is stacked for all the poses for the final estimation form

$$\Delta^{-1}\boldsymbol{p}_{n+1} = \mathbf{J}\Delta\boldsymbol{\phi} , \tag{6.19}$$

which corresponds to (6.7) and which is solved for $\Delta\boldsymbol{\phi}$ by least squares and for $\boldsymbol{\phi}$ by iteration.

Full Pose Measurement. Suppose that coordinate system $n+1$ in link n is measured (Fig. 6.5). Coordinate system n is completed in the usual way, and the six calibrated parameters to locate coordinate system $n+1$ are d_n, α_n, θ_n, a_{n+1}, d_{n+1}, and θ_{n+1}. If z_{n+1} is nearly parallel to z_n, this axis may simply be permuted with other axes, such as y_{n+1}, to avoid using the Hayati parameters.

In addition to the position (6.12), orientation equations of coordinate system $n+1$

$$^{-1}\mathbf{R}^l_{n+1} = \mathbf{F}(\boldsymbol{q}^l, \boldsymbol{\phi}) , \quad l = 1, \dots, P \tag{6.20}$$

are extracted from $^{-1}\mathbf{T}_{n+1}$, where $\mathbf{F}$ is a matrix function. Linearization of this equation yields

$$\Delta^{-1}\mathbf{R}^l_{n+1} = {}^{-1}\mathbf{R}^l_{n+1} - {}^{-1}\mathbf{R}^l_{n+1,c}$$
$$= \Delta^{-1}\boldsymbol{\rho}^l_{n+1} \times {}^{-1}\mathbf{R}^l_{n+1,c}$$
$$\mathbf{S}(\Delta^{-1}\boldsymbol{\rho}^l_{n+1}) = \left({}^{-1}\mathbf{R}^l_{n+1} - {}^{-1}\mathbf{R}^l_{n+1,c}\right)\left({}^{-1}\mathbf{R}^l_{n+1,c}\right)^{\mathrm{T}} , \tag{6.21}$$

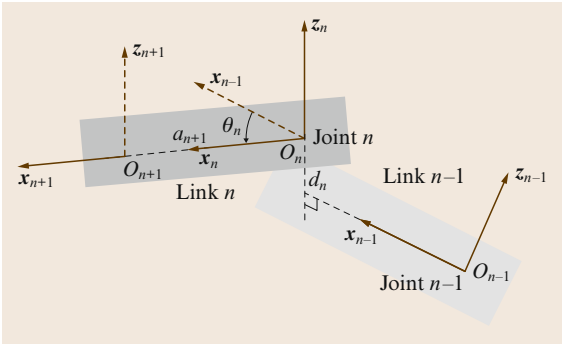


Fig. 6.4 A measured point O_{n+1} is located on the end link by adding a coordinate system $n+1$

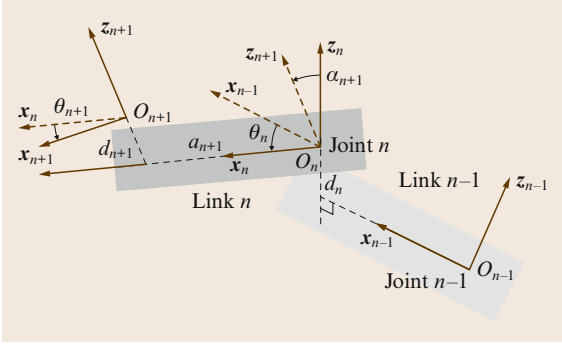


Fig. 6.5 Full pose measurement of coordinate system $n+1$ in the end link

where $\Delta^{-1}\boldsymbol{\rho}^l_{n+1}$ are differential orthogonal rotations which are the finite-difference counterpart to the angular velocity vector. Continuing with this analogy, a Jacobian $\mathbf{J}$ similar to that for spatial velocities (Sect. 2.8.1) is employed to represent the combined effect of parameter variations $\Delta\boldsymbol{\phi}$ on changes in position $\Delta^{-1}\boldsymbol{p}^l_{n+1}$ and orientation $\Delta^{-1}\boldsymbol{\rho}^l_{n+1}$

$$\begin{pmatrix}\Delta^{-1}\boldsymbol{p}^l_{n+1}\\ \Delta^{-1}\boldsymbol{\rho}^l_{n+1}\end{pmatrix} = \mathbf{J}^l\Delta\boldsymbol{\phi}\,. \tag{6.22}$$

Compared to (6.13), the Jacobian $\mathbf{J}^l$ now has six rows, as do the individual parameter Jacobians

$$\mathbf{J}^l_{a_i} = \begin{cases}\begin{pmatrix}{}^{-1}\boldsymbol{x}^l_{i-1}\\ \mathbf{0}\end{pmatrix} & \text{DH}\\ \begin{pmatrix}{}^{-1}\boldsymbol{x'}^l_{i-1}\\ \mathbf{0}\end{pmatrix} & \text{Hayati}\end{cases}, \tag{6.23}$$

$$\mathbf{J}^l_{d_i} = \begin{pmatrix}{}^{-1}\boldsymbol{z}^l_i\\ \mathbf{0}\end{pmatrix}, \tag{6.24}$$

$$\mathbf{J}^l_{\alpha_i} = \begin{cases}\begin{pmatrix}{}^{-1}\boldsymbol{x}^l_{i-1}\times{}^{-1}\boldsymbol{d}^l_{i-1,n+1}\\ {}^{-1}\boldsymbol{x}^l_{i-1}\end{pmatrix} & \text{DH}\\ \begin{pmatrix}{}^{-1}\boldsymbol{x'}^l_{i-1}\times{}^{-1}\boldsymbol{d}^l_{i-1,n+1}\\ {}^{-1}\boldsymbol{x'}^l_{i-1}\end{pmatrix} & \text{Hayati}\end{cases}, \tag{6.25}$$

$$\mathbf{J}^l_{\theta_i} = \begin{pmatrix}{}^{-1}\boldsymbol{z}^l_i\times{}^{-1}\boldsymbol{d}^l_{i,n+1}\\ {}^{-1}\boldsymbol{z}^l_i\end{pmatrix}, \tag{6.26}$$

$$\mathbf{J}^l_{\beta_i} = \begin{pmatrix}{}^{-1}\boldsymbol{y}^l_{i-1}\times{}^{-1}\boldsymbol{d}^l_{i-1,n+1}\\ {}^{-1}\boldsymbol{y}^l_{i-1}\end{pmatrix}. \tag{6.27}$$

As before, $\Delta\boldsymbol{\phi}$ is found by least squares and $\boldsymbol{\phi}$ by iteration.

Closed-Loop Calibration

Physical constraints on end-effector position or orientation can substitute for measurements. The location of a physical constraint defines the reference coordinate system; the *measurement* of endpoint position or orientation is therefore defined as zero. The deviation from the physical constraint due to an incorrect kinematic model is cast as a displacement from the reference coordinates. Analogous to point measurement and full pose measurement, there are closed-loop methods using point constraints and full pose constraints.

Point Constraint. Suppose that the end-effector holds a stylus that makes contact with a fixed point in the environment. The orientation of the stylus can be changed by varying the joint angles, as long as the point of contact is not changed. Before, the measurement system for point measurement defined the reference coordinate system -1 (Fig. 6.3), and there was an end effector coordinate system $n+1$ whose origin O_{n+1} was measured (Fig. 6.4). Now the reference coordinate origin O_0 has index 0 and is collocated with O_{n+1} (Fig. 6.6). An extra coordinate system -1 is not required because a point is located relative to coordinate system 1 with only three parameters: a_1, d_1, and θ_1. The arbitrary choice $\alpha_1 = 0$ is made, i. e., z_0 is made parallel to z_1.

Different poses while maintaining the point contact can be generated manually, or they can be generated automatically if there is a force control capability. In comparison to (6.13), the *measured* position ${}^0\boldsymbol{p}^l_{n+1} = 0$ by definition and the linearized calibration equation is simply

$$\Delta^{-1}\boldsymbol{p}^l_{n+1} = -{}^{-1}\boldsymbol{p}^l_{n+1,c} = \mathbf{J}^l\Delta\boldsymbol{\phi}\,. \tag{6.28}$$

The set of poses that may be generated is limited relative to open-loop calibration, which may affect identifiability.

Full Pose Constraint. Analogous to full pose measurement (Fig. 6.5), the end link may be fully constrained by rigidly gripping the environment. If the manipulator is redundant (Chap. 10), then poses can be generated through self motion. Generating such poses would require endpoint force/torque sensing or joint torque sensing to be accomplished.

The end link can be considered part of the ground due to rigid attachment, and therefore one fewer coordinate system is required than for the fixed point case. Fig. 6.7 shows one way of setting up coordinate systems 0 and n for calibration. Axis z_0 is set equal and coincident with axis z_n. The common normal between z_0 and z_1 sets the origin O_0 and the axis x_0. Coordinate system n is completed by defining $O_n = O_0$ and $x_n = x_0$. Six parameters result for calibration, as they must to re-

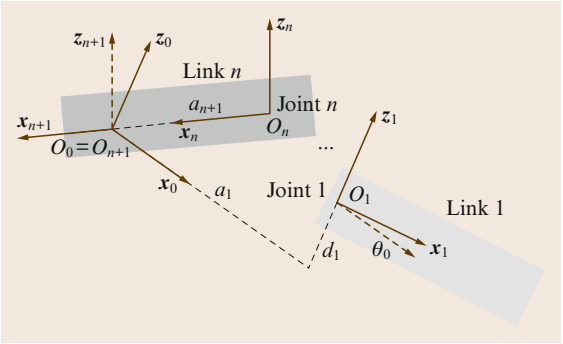


Fig. 6.6 Fixed point contact, setting $O_0 = O_{n+1}$

late coordinate system n to coordinate system 0: θ_n, d_n, α_1, a_1, d_1, and θ_1.

Similar to (6.28) after index adjustment, the *measured* position ${}^0\boldsymbol{p}_n^l = 0$ by definition and the linearized position calibration equation is

$$\Delta {}^0\boldsymbol{p}_n^l = -{}^0\boldsymbol{p}_{n,c}^l \,. \tag{6.29}$$

With regard to the error in orientation (6.21), the *measured* orientation ${}^0\mathbf{R}_n^l = \mathbf{I}$, the identity matrix, after index adjustment.

$$\mathbf{S}\left(\Delta {}^0\boldsymbol{\rho}_n^l\right) = \left(\mathbf{I} - {}^0\mathbf{R}_{n,c}^l\right)\left({}^0\mathbf{R}_{n,c}^l\right)^{\mathrm{T}} = \left({}^0\mathbf{R}_{n,c}^l\right)^{\mathrm{T}} - \mathbf{I} \,. \tag{6.30}$$

The error (6.22) is then applied as for full pose measurement, after index adjustment. The set of poses that may be generated is limited relative to open-loop calibration, which may affect identifiability.

6.2.2 Parallel Manipulator Calibration

Parallel manipulators are comprised of multiple closed loops. The methods of the previous section could readily be extended to calibrate parallel manipulators. Rather than treat parallel and serial manipulators differently, and continue to elaborate the calibration equations on a case-by-case basis that includes different loop arrangements and different sensing arrangements, *Hollerbach* and *Wampler* [6.8] presented a unifying methodology termed the *calibration index*, which views all calibration problems as closed-loop calibration problems.

Open-loop calibration is converted to closed-loop calibration by considering the end effector measurement to form a joint. All measurements, whether from joints or metrology systems, are put on an equal footing, as are all unsensed joints, whether from unsensed components of pose, passive environmental constraints, or joints in the chain without sensors. In the case of parallel linkages, sufficient numbers of loop-closure equations are formulated to characterize the kinematics and are combined at each pose. Because the loop-closure equations are implicit functions of all measurements, *Wampler* et al. [6.2] called this calibration method the *implicit loop method*.

Fig. 6.8 illustrates this idea for a calibrated stereo camera metrology system that measures the 3-D coordinates of a distinguished point on the end-effector of an uncalibrated robot. On the right this camera system has been replaced by a prismatic leg, which stylistically represents a six-degree-of-freedom (6-DOF) joint that provides equivalent 3-D coordinate measurements. The result is a closed-loop mechanism.

The kinematic loop closure equations $\boldsymbol{f}$ for the i-th pose ($i = 1, \ldots, P$) are formed as

$$\boldsymbol{f}^i(\boldsymbol{\phi}) \equiv \boldsymbol{f}(\boldsymbol{x}^i, \boldsymbol{\phi}) = \mathbf{0} \,, \tag{6.31}$$

where $\boldsymbol{\phi}$ is a vector of robot parameters to be calibrated, $\boldsymbol{x}^i$ is a vector of joint sensor readings and possibly external sensor readings, and $\boldsymbol{f}^i$ absorbs the sensor readings $\boldsymbol{x}^i$ and so is given an index. Combining (6.31) for the P poses into a single matrix equation

$$\boldsymbol{f}(\boldsymbol{\phi}) = \left(\boldsymbol{f}^{1\mathrm{T}} \quad \ldots \quad \boldsymbol{f}^{P\mathrm{T}}\right)^{\mathrm{T}} = \mathbf{0} \,. \tag{6.32}$$

Linearize (6.32) around the nominal values of the parameters

$$\Delta \boldsymbol{f} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\phi}} \Delta \boldsymbol{\phi} = \mathbf{A} \Delta \boldsymbol{\phi} \,, \tag{6.33}$$

where $\Delta \boldsymbol{f}$ is the deviation of the computed loop closure equations from zero, $\mathbf{J}$ is the identification Jacobian, and $\Delta \boldsymbol{\phi}$ is the correction to be applied to the current parameter estimate. The calibration problem is then solved by minimizing $\Delta \boldsymbol{f}$ via iterative least squares.

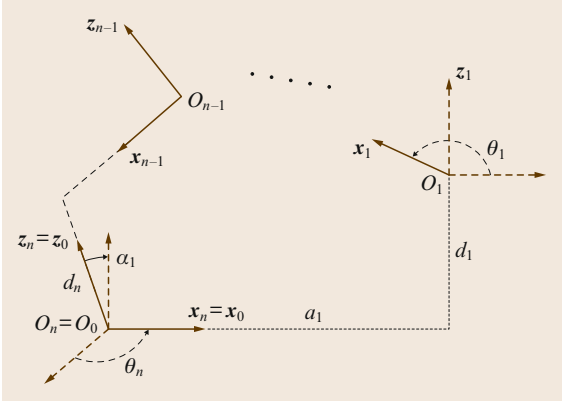


Fig. 6.7 Fully constrained end link, setting $O_0 = O_n$

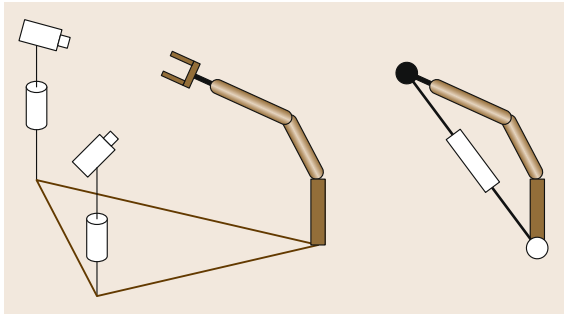


Fig. 6.8 External metrology system modeled as a 6-DOF joint

The Calibration Index

The basis for kinematic calibration is to compute the error in positioning a manipulator using the current kinematic model. The error may be relative to measurements by an external metrology system (open-loop methods), or relative to a physical constraint such as contact with a plane (closed-loop methods). An external metrology system may measure all six components of pose or fewer components such as a point on the end link (three components). Likewise, a physical constraint can limit between one and six components of pose. Constraints and measurements may be mixed. The number of measurements and constraints determine how many scalar equations per pose are available for calibration.

The calibration index C quantifies the number of equations per pose. While this analysis is rather straightforward for serial linkages, for parallel linkages it can be difficult to infer the number of independent equations per pose

$$C = S - M \,, \tag{6.34}$$

where S is the sensor index and M is the mobility index. The mobility index [6.17] characterizes the degrees of freedom in the calibration setup.

$$M = 6n - \sum_{i=1}^{N_J} n_i^{\mathrm{c}} \,, \tag{6.35}$$

where n is the number of links, N_J is the number of joints, and n_i^{c} is the number of constraints at joint i. n includes any extra links attached to the robot to constrain or measure its motion. N_J includes joints of any additional linkages added for calibration. For a rotational or prismatic joint $n_i^{\mathrm{c}} = 5$, while for a ball or spherical joint $n_i^{\mathrm{c}} = 3$. For an external measurement system for a freely moving end-effector $n_{N_J}^{\mathrm{c}} = 0$, while for rigid attachment of the endpoint $n_{N_J}^{\mathrm{c}} = 6$. While generally correct, there are exceptions to (6.35) due to special or degenerate mechanisms that have to be analyzed on a case-by-case basis.

The sensor index S is the total number of sensors in the joints

$$S = \sum_{i=1}^{N_J} S_i \,, \tag{6.36}$$

where S_i is the number of sensed degrees in joint i. Usually $S_i = 1$ for the lower-order pairs typical of actuated joints, while $S_{N_J} = 6$ for full pose measurement of the end-effector *joint* N_J. For an unsensed joint, such as in passive environment kinematics, $S_i = 0$.

If P is the number of poses, then CP is the total number of equations for the calibration procedure. Clearly a larger C means that fewer poses are required, other things being equal. For the single-loop case consisting of a series of sensed lower-order robot joints ($S_i = 1$, $n_i^{\mathrm{c}} = 5$, $i = 1, \ldots, N_J - 1$) with a final joint $(S_{N_J}, n_{N_J}^{\mathrm{c}})$ connecting the end-effector to ground, one has from (6.34–6.35)

$$C = S_{N_J} + n_{N_J}^{\mathrm{c}} \,. \tag{6.37}$$

According to the calibration index, using full endpoint constraints is an equivalent kinematic calibration method to using full pose measurements. There is potentially a problem with the smaller range of poses available with the endpoint constrained, but otherwise the mathematics are the same.

Categorization of Serial Link Calibration Methods

A great variety of calibration methods have been proposed, varying in the manner of pose measurement or endpoint constraints. These are categorized below according to the calibration index C and the values of $n_{N_J}^{\mathrm{c}}$ and S_{N_J}:

- $C = 6$: $n_{N_J}^{\mathrm{c}} = 0$ and $S_{N_J} = 6$ corresponds to full pose measurement. $n_{N_J}^{\mathrm{c}} = 6$ and $S_{N_J} = 0$ corresponds to rigid endpoint attachment.
- $C = 5$: $n_{N_J}^{\mathrm{c}} = 0$ and $S_{N_J} = 5$ corresponds to 5-DOF pose measurement [6.14]. $n_{N_J}^{\mathrm{c}} = 5$ and $S_{N_J} = 0$ corresponds to 5-DOF endpoint constraints, such as manipulation of an unsensed passive hinge joint [6.18].
- $C = 4$: No published method exists for $C = 4$.
- $C = 3$: $n_{N_J}^{\mathrm{c}} = 0$ and $S_{N_J} = 3$ corresponds to 3-DOF pose measurement. $n_{N_J}^{\mathrm{c}} = 3$ and $S_{N_J} = 0$ corresponds to 3-DOF endpoint constraints.
- $C = 2$: $n_{N_J}^{\mathrm{c}} = 0$ and $S_{N_J} = 2$ corresponds to 2-DOF pose measurements, such as is provided by a single theodolite [6.12]. $n_{N_J}^{\mathrm{c}} = 2$ and $S_{N_J} = 0$ corresponds to 2-DOF endpoint constraints. Motion along a line provides two constraints [6.19].
- $C = 1$: $n_{N_J}^{\mathrm{c}} = 0$ and $S_{N_J} = 1$ corresponds to measurement of just 1-DOF pose, provided by a linear transducer such as an linear variable differential transformer (LVDT) [6.9] or wire potentiometer [6.10]. $n_{N_J}^{\mathrm{c}} = 1$ and $S_{N_J} = 0$ corresponds to a plane constraint [6.20].

Categorization of Parallel-Link Calibration Methods

To calibrate parallel robots, a loop-closure equation is written for each loop j

$$\mathbf{0} = \boldsymbol{f}_j(\boldsymbol{\phi}) \,. \tag{6.38}$$

The equations for all the loops are combined. One problem is to eliminate unsensed degrees of freedom. The calibration index is now applied based on the number of loops:

- *2 loops*: A two-loop mechanism has three arms or legs, attached to a common platform. An example is the RSI Research Ltd. Hand Controller [6.21], which employs three 6-DOF arms with three sensed joints each. The mobility of this mechanism is $M = 6$. As $S = 9$, therefore $C = 3$ and closed-loop calibration is possible.
- *4 loops*: *Nahvi* et al. [6.22] calibrated a spherical shoulder joint, driven redundantly by four prismatic legs (Fig. 6.9). In addition, the platform is constrained to rotate about a spherical joint; with the four legs, four kinematic loops are formed. For this system, $M = 3$ and $S = 4$, so that $C = 1$. Hence self-calibration is possible. Without the extra leg and the sensing it provides, one would have $C = 0$ and calibration would not be possible.
- *5 loops*: *Wampler* et al. [6.2] calibrated the six-legged Stewart platform ($M = 6$) via a closed-loop procedure. In addition to leg length measurements, all angles on one of the legs were measured ($S = 11$). The reason for the extra sensors is to yield a unique forward kinematics solution, but a side benefit is that, with $C = 5$, closed-loop calibration is possible. For a regular Stewart platform without the instrumented leg, $S = 6$ and hence $C = 0$. External pose measurement is required; for example, with full pose measurement $S = 12$ and $C = 6$; full pose measurement was utilized in [6.23].

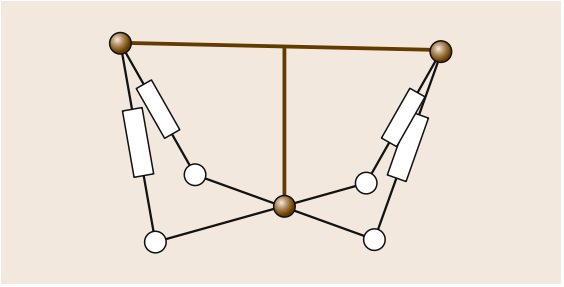


Fig. 6.9 Redundant parallel drive shoulder joint

6.3 Inertial Parameter Estimation

A rigid body i has 10 inertial parameters: mass m_i, center of mass $\boldsymbol{r}_{0i}$ relative to an origin O_0, and symmetric inertia matrix $\bar{\mathbf{I}}_i$ referred here to the origin O_i (Fig. 6.10). The rigid body may be a load held by the end effector, or it can be one of the manipulator's own links. By generating a trajectory and measuring forces or torques in combination with velocity and acceleration, some or all of the inertial parameters can be estimated.

6.3.1 Link Inertial Parameter Estimation

The procedure involves:

1. Formulating the Newton–Euler equations of the load dynamics to reveal a linear dependence on the inertial parameters.
2. Estimating the parameters using ordinary least squares.

Begin with the Newton-Euler equations of the last link, and assume that suitable filters have been designed to estimate the velocity $\dot{\theta}_i$ and acceleration $\ddot{\theta}_i$ at each joint i.

The center of mass of link n is defined as C_n, located relative to the base origin O_0 by $\boldsymbol{r}_n = C_n - O_0$ and relative to link n's origin O_n by $\boldsymbol{c}_n = C_n - O_n$. All vectors and matrices are expressed in coordinate system n, in which the center of mass location $\boldsymbol{c}_n$ and the inertia matrix $\bar{\mathbf{I}}_n$ about origin O_n are constant.

From (3.35) in Chap. 3, the Newton-Euler equations referred to O_n are

$$\boldsymbol{f}_n = \mathbf{I}_n \boldsymbol{a}_n + \boldsymbol{v}_n \times \mathbf{I}_n \boldsymbol{v}_n \,. \tag{6.39}$$

Substituting for the spatial inertia $\mathbf{I}_n$, the spatial acceleration $\boldsymbol{a}_n$, and the spatial velocity $\boldsymbol{v}_n$, the first term on

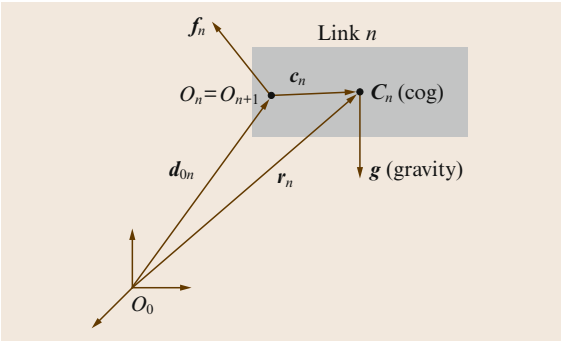


Fig. 6.10 Location of the center of gravity for an intermediate link i, and the constraint forces and torques

the right evaluates to

$$\mathbf{I}_n \boldsymbol{a}_n = \begin{pmatrix} \bar{\mathbf{I}}_n & m_n \mathbf{S}(\boldsymbol{c}_n) \\ m_n \mathbf{S}(\boldsymbol{c}_n)^{\mathrm{T}} & m_n \mathbf{1} \end{pmatrix} \begin{pmatrix} \dot{\boldsymbol{\omega}}_n \\ \dot{\boldsymbol{v}}_n \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{I}}_n \dot{\boldsymbol{\omega}}_n + m_n \mathbf{S}(\boldsymbol{c}_n) \left(\ddot{\boldsymbol{d}}_{0n} - \boldsymbol{\omega}_n \times \boldsymbol{v}_n \right) \\ m_n \mathbf{S}(\boldsymbol{c}_n)^{\mathrm{T}} \dot{\boldsymbol{\omega}}_n + m_n \left(\ddot{\boldsymbol{d}}_{0n} - \boldsymbol{\omega}_n \times \boldsymbol{v}_n \right) \end{pmatrix},$$

where $\dot{\boldsymbol{v}}_n = \ddot{\boldsymbol{d}}_{0n} - \boldsymbol{\omega}_n \times \boldsymbol{v}_n$ has been substituted. The second term on the right evaluates to

$$\boldsymbol{v}_n \times \mathbf{I}_n \boldsymbol{v}_n = \begin{pmatrix} \mathbf{S}(\boldsymbol{\omega}_n) & \mathbf{S}(\boldsymbol{v}_n) \\ \mathbf{0} & \mathbf{S}(\boldsymbol{\omega}_n) \end{pmatrix} \times \begin{pmatrix} \bar{\mathbf{I}}_n & m_n \mathbf{S}(\boldsymbol{c}_n) \\ m_n \mathbf{S}(\boldsymbol{c}_n)^{\mathrm{T}} & m_n \mathbf{1} \end{pmatrix} \begin{pmatrix} \boldsymbol{\omega}_n \\ \boldsymbol{v}_n \end{pmatrix} = \begin{pmatrix} \mathbf{S}(\boldsymbol{\omega}_n) \bar{\mathbf{I}}_n \boldsymbol{\omega}_n + m_n \mathbf{S}(\boldsymbol{c}_n) \mathbf{S}(\boldsymbol{\omega}_n) \boldsymbol{v}_n \\ \mathbf{S}(\boldsymbol{\omega}_n) m_n \mathbf{S}(\boldsymbol{c}_n)^{\mathrm{T}} \boldsymbol{\omega}_n + \mathbf{S}(\boldsymbol{\omega}_n) m_n \boldsymbol{v}_n \end{pmatrix}.$$

Combining and simplifying,

$$\boldsymbol{f}_n = \begin{pmatrix} \bar{\mathbf{I}}_n \dot{\boldsymbol{\omega}}_n + \mathbf{S}(\boldsymbol{\omega}_n) \bar{\mathbf{I}}_n \boldsymbol{\omega}_n - \mathbf{S}\left(\ddot{\boldsymbol{d}}_{0n} \right) m_n \boldsymbol{c}_n \\ m_n \ddot{\boldsymbol{d}}_{0n} + \mathbf{S}(\dot{\boldsymbol{\omega}}_n) m_n \boldsymbol{c}_n + \mathbf{S}(\boldsymbol{\omega}_n) \mathbf{S}(\boldsymbol{\omega}_n) m_n \boldsymbol{c}_n \end{pmatrix}, \tag{6.40}$$

where the mass moment $m_n \boldsymbol{c}_n$ appears as a quantity to be estimated in combination. However, since the mass m_n is separately estimated from the $m_n \ddot{\boldsymbol{d}}_{0n}$ term, the center of mass $\boldsymbol{c}_n$ can be extracted. To account explicitly for gravity $\boldsymbol{g}$, we substitute $\ddot{\boldsymbol{d}}_{0n} - \boldsymbol{g}$ for $\ddot{\boldsymbol{d}}_{0n}$ subsequently.

To formulate an estimation algorithm, the force and torque measured by the wrist sensor must be expressed in terms of the product of known geometric parameters and the unknown inertial parameters. Elements of the inertia matrix are vectorized as $\boldsymbol{l}(\bar{\mathbf{I}}_n)$ according to the following notation

$$\bar{\mathbf{I}}_n \boldsymbol{\omega}_n = \begin{pmatrix} \omega_1 & \omega_2 & \omega_3 & 0 & 0 & 0 \\ 0 & \omega_1 & 0 & \omega_2 & \omega_3 & 0 \\ 0 & 0 & \omega_1 & 0 & \omega_2 & \omega_3 \end{pmatrix} \begin{pmatrix} I_{11} \\ I_{12} \\ I_{13} \\ I_{22} \\ I_{23} \\ I_{33} \end{pmatrix} \equiv \mathbf{L}(\boldsymbol{\omega}_n) \boldsymbol{l}\left(\bar{\mathbf{I}}_n \right),$$

where $\mathbf{L}(\boldsymbol{\omega}_n)$ is a 3×6 matrix of angular velocity elements and

$$\bar{\mathbf{I}}_n = \begin{pmatrix} I_{11} & I_{12} & I_{13} \\ I_{12} & I_{22} & I_{23} \\ I_{13} & I_{23} & I_{33} \end{pmatrix}.$$

Using these expressions, (6.40) can be written as

$$\boldsymbol{f}_n = \begin{pmatrix} \mathbf{0} & -\mathbf{S}\left(\ddot{\boldsymbol{d}}_{0n} \right) & \mathbf{L}(\dot{\boldsymbol{\omega}}_n) + \mathbf{S}(\boldsymbol{\omega}_n) \mathbf{L}(\boldsymbol{\omega}_n) \\ \ddot{\boldsymbol{d}}_{0n} & \mathbf{S}(\dot{\boldsymbol{\omega}}_n) + \mathbf{S}(\boldsymbol{\omega}_n) \mathbf{S}(\boldsymbol{\omega}_n) & \mathbf{0} \end{pmatrix} \times \begin{pmatrix} m_n \\ m_n \boldsymbol{c}_n \\ \boldsymbol{l}\left(\bar{\mathbf{I}}_n \right) \end{pmatrix}$$

or more compactly,

$$\boldsymbol{f}_n = \mathbf{A}_n \boldsymbol{\phi}_n, \tag{6.41}$$

where $\mathbf{A}_n$ is a 6×10 matrix, and $\boldsymbol{\phi}_n$ is the vector of the 10 unknown inertial parameters which appear linearly.

The previous formulation can be extended to all the links of a manipulator with n joints (Fig. 6.10). (Only manipulators with revolute joints will be considered, since handling prismatic joints requires only trivial modifications to the algorithm.) Define $\boldsymbol{f}_{ij}$ as the spatial force at joint i due to movement of link j alone. Then $\boldsymbol{f}_{ii}$ is the spatial force at joint i due to movement of its own link, and is the same as (6.41) with i substituted for n in the $\mathbf{A}_n$ matrix

$${}^i\boldsymbol{f}_{ii} = {}^i\mathbf{A}_i \boldsymbol{\phi}_i, \tag{6.42}$$

where $\boldsymbol{\phi}_i$ is the vector of unknown link i inertial parameters. Superscript i has been added to indicate that vectors are expressed in terms of link i coordinates, so that the center of mass ${}^i\boldsymbol{c}_i$ and the inertia matrix ${}^i\bar{\mathbf{I}}_i$ are constant.

The total spatial force ${}^i\boldsymbol{f}_i$ at joint i is the sum of the spatial forces ${}^i\boldsymbol{f}_{ij}$ for all links j distal to joint i

$${}^i\boldsymbol{f}_i = \sum_{j=i}^{n} {}^i\boldsymbol{f}_{ij}. \tag{6.43}$$

Each spatial force ${}^i\boldsymbol{f}_{ij}$ at joint i is determined by transmitting the distal spatial force ${}^j\boldsymbol{f}_{jj}$ across intermediate joints. Using the spatial force transform matrix ${}^i\mathbf{X}_j^{\mathrm{F}}$,

$${}^i\boldsymbol{f}_{i,i+1} = {}^i\mathbf{X}_{i+1}^{\mathrm{F}} \, {}^{i+1}\boldsymbol{f}_{i+1,i+1} = {}^i\mathbf{X}_{i+1}^{\mathrm{F}} \, {}^{i+1}\mathbf{A}_{i+1} \boldsymbol{\phi}_{i+1}. \tag{6.44}$$

For convenience, we note that ${}^i\mathbf{X}_i^{\mathrm{F}} = \mathbf{1}_{6\times 6}$. To obtain the forces and torques at the i-th joint due to the movements of the j-th link, these matrices can be cascaded

$$\begin{aligned} {}^i\boldsymbol{f}_{ij} &= {}^i\mathbf{X}_{i+1}^{\mathrm{F}} \quad {}^{i+1}\mathbf{X}_{i+2}^{\mathrm{F}} \quad \cdots \quad {}^{j-1}\mathbf{X}_j^{\mathrm{F}} \quad {}^j\boldsymbol{f}_{jj} \\ &= {}^i\mathbf{X}_j^{\mathrm{F}j} \mathbf{A}_j \boldsymbol{\phi}_j. \end{aligned} \tag{6.45}$$

An upper-diagonal matrix expression for a serial kinematic chain can be derived from (6.43) and (6.45)

$$\begin{pmatrix} {}^1\boldsymbol{f}_1 \\ {}^2\boldsymbol{f}_2 \\ \vdots \\ {}^n\boldsymbol{f}_n \end{pmatrix} = \begin{pmatrix} {}^1\mathbf{X}_1^{\mathrm{F}1}\mathbf{A}_1 & {}^1\mathbf{X}_2^{\mathrm{F}2}\mathbf{A}_2 & \cdots & {}^1\mathbf{X}_n^{\mathrm{F}n}\mathbf{A}_n \\ \mathbf{0} & {}^2\mathbf{X}_2^{\mathrm{F}2}\mathbf{A}_2 & \cdots & {}^2\mathbf{X}_n^{\mathrm{F}n}\mathbf{A}_n \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & {}^n\mathbf{X}_n^{\mathrm{F}n}\mathbf{A}_n \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi}_1 \\ \boldsymbol{\phi}_2 \\ \vdots \\ \boldsymbol{\phi}_n \end{pmatrix} . \tag{6.46}$$

This equation is linear in the unknown parameters, but the left-hand side is composed of a full force-torque vector at each joint. Since only the torque τ_i about the joint rotation axis z_i can usually be measured, each spatial force ${}^i\boldsymbol{f}_i$ must be projected onto the joint rotation axis, reducing (6.46) to

$$\boldsymbol{\tau} = \mathbf{K}\boldsymbol{\phi} , \tag{6.47}$$

where

$$\tau_i = \begin{pmatrix} z_i \\ \mathbf{0} \end{pmatrix} \cdot \boldsymbol{f}_i ,$$

$$\mathbf{K}_{ij} = \begin{pmatrix} z_i \\ \mathbf{0} \end{pmatrix} \cdot {}^i\mathbf{X}_j^{\mathrm{F}j}\mathbf{A}_j , \quad \boldsymbol{\phi} = \begin{pmatrix} \boldsymbol{\phi}_1 \\ \cdot \\ \cdot \\ \boldsymbol{\phi}_n \end{pmatrix}$$

and $\mathbf{K}_{ij} = \mathbf{0}_{1\times 10}$ if $i > j$. For an n-link manipulator, $\boldsymbol{\tau}$ is an $n\times 1$ vector, $\boldsymbol{\phi}$ is a $10n\times 1$ vector, and $\mathbf{K}$ is an $n\times 10n$ matrix.

For geared electric drives, joint torque may either be measured by joint torque sensors (Chap. 28) or estimated from the motor current by using an electric motor model (Chap. 8). Most robots do not have joint torque sensors, in which case joint friction needs to be taken into account. Joint friction typically consumes a large fraction of the torque that the motor produces. Coulomb and viscous friction are the most important components of a friction model, although Stribeck friction may need to be modeled for low joint velocities [6.24, 25]. A friction model is estimated through a process of moving one joint at a time, and relating motor torque to velocity. Ripple torque, due either to uncompensated inhomogeneities of the magnetic field in the motor [6.26, 27], or to positional dependencies of gear tooth interaction [6.24], may need to be modeled.

Equation 6.47 represents the dynamics of the manipulator for one sample point. It can be rewritten using P data points as with geometric calibration

$$\boldsymbol{\tau} = \mathbf{K}\boldsymbol{\phi} .$$

$$\mathbf{K} = \begin{pmatrix} \mathbf{K}^1 \\ \vdots \\ \mathbf{K}^P \end{pmatrix} , \quad \boldsymbol{\tau} = \begin{pmatrix} \boldsymbol{\tau}^1 \\ \vdots \\ \boldsymbol{\tau}^P \end{pmatrix} , \tag{6.48}$$

where $\boldsymbol{\tau}$ is now an $nP\times 1$ vector and $\mathbf{K}$ is now $nP\times 10n$.

Unfortunately, one cannot apply simple least-squares estimation because $\mathbf{K}^{\mathrm{T}}\mathbf{K}$ is not invertible due to the loss of rank from restricted degrees of freedom at the proximal links and the lack of full force-torque sensing. Some inertial parameters are completely unidentifiable, while some others can only be identified in linear combinations. The general topic of identifiability of parameters, and how to handle unidentifiable parameters, is discussed next.

An issue with geared electric drives is the rotor inertia. If not known, the rotor inertia can be added to the list of 10 inertial parameters to be identified for a link [6.28]. For large gear ratios, the rotor inertia can dominate the other components of link inertia.

6.3.2 Load Inertial Parameter Estimation

The load is considered as a rigid body held by the end effector. It has 10 inertial parameters: mass m_{L}, first moments $m_{\mathrm{L}}c_{\mathrm{L}}$ relative to O_n and symmetric inertia matrix $\bar{\mathbf{I}}_{\mathrm{L}}$ referred here to origin O_n (Fig. 6.11). The estimation of these parameters can be used to tune the control law parameters in order to improve the dynamic accuracy of the robot, or to compensate for its dynamics by the control. They can also be exploited to verify the load transported by the robot. Two procedures are presented to estimate the load inertia parameters: the first one assumes that there is a wrist-mounted 6-axis force/torque sensor, whereas the second one makes use of the joint torques.

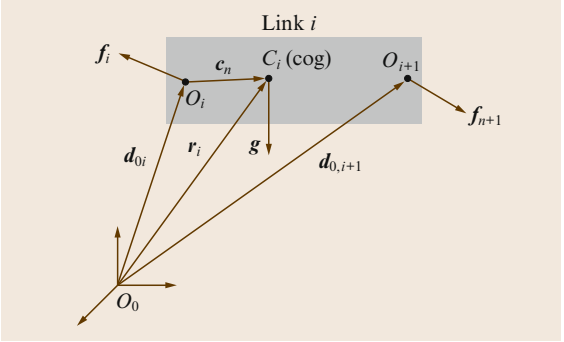


Fig. 6.11 Dynamics of the last link

Using 6-Axis Forces/Torque Sensor

Kinematically, the force-torque sensor is part of the last link n, mounted near the joint axis z_n and origin O_n. The sensor provides readings relative to its own coordinate system, called $n+1$ (Fig. 6.11). The remainder of the last link is attached to the shaft of the force-torque sensor, and so the force-torque sensor is measuring the loads on the last link excluding itself. The acceleration of the force-torque sensor reference frame would have to be calculated based on the amount of offset from O_n, but we will ignore that complication here and assume that the sensor origin O_{n+1} is coincident with O_n.

The estimation of load inertial parameters can be carried out using (6.41).

The quantities inside the $\mathbf{A}_n$ matrix are computed by direct kinematics computation from the measured joint angles, and from the estimated joint velocities and accelerations. The estimation is typically done by bandpass filtering of the joint angle data [6.29]. The elements of the $\boldsymbol{f}_n$ vector are measured directly by the wrist force sensor. For robust estimates in the presence of noise, a larger number of P data points are obtained by moving the manipulator through a suitable trajectory. Augment $\boldsymbol{f}_n$ and $\mathbf{A}_n$ as

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_n^1 \\ \vdots \\ \mathbf{A}_n^P \end{pmatrix}, \quad \boldsymbol{f} = \begin{pmatrix} \boldsymbol{f}_n^1 \\ \vdots \\ \boldsymbol{f}_n^P \end{pmatrix}, \tag{6.49}$$

where P is the number of data points. The least-squares estimate for $\boldsymbol{\phi}_\mathrm{L}$ is given by

$$\hat{\boldsymbol{\phi}}_\mathrm{L} = (\mathbf{A}^\mathrm{T}\mathbf{A})^{-1}\mathbf{A}^\mathrm{T}\boldsymbol{f} \,. \tag{6.50}$$

For object recognition, it may be desired to derive the inertia about the center of mass, which can be done with the parallel axis theorem. An eigenvalue analysis can diagonalize the inertia matrix to reveal the principal axes and inertia.

Using Joint Torques

When the robot is holding a payload, the dynamic identification (6.48) can be rewritten as follows

$$\boldsymbol{\tau} = \mathbf{K}\boldsymbol{\phi} + \mathbf{K}_\mathrm{L}\boldsymbol{\phi}_\mathrm{L} \,, \tag{6.51}$$

where $\boldsymbol{\tau}$ contains the measured joint torques when the robot is holding a load for P points. Using (6.51) in the identification will result in grouping $\boldsymbol{\phi}_\mathrm{L}$ with $\boldsymbol{\phi}_n$ and with other link parameters. The load parameters can be deduced by examining how the load parameters have modified the values of the grouped inertial parameters using the expressions of grouping relations [6.30]. In the following we present three methods to identify $\boldsymbol{\phi}_\mathrm{L}$ parameters without using the grouping expressions.

Using a Priori Link Parameters Estimates. In this case we suppose that the robot parameters without load have been estimated as given in Sec. 6.3.1. By substituting the robot parameters into (6.51) the 10 inertial parameters of the load are identifiable and can be estimated using the least squares solution of [6.31]

$$(\boldsymbol{\tau} - \mathbf{K}\boldsymbol{\phi}) = \mathbf{K}_\mathrm{L}\boldsymbol{\phi}_\mathrm{L} \,. \tag{6.52}$$

In this method we suppose that the friction parameters are invariant with respect to the payload. If this hypothesis is not true the friction parameters must be re-estimated at the same step with $\boldsymbol{\phi}_\mathrm{L}$.

Using Joint Torques on the Same Trajectory Twice with and Without the Load. Since the vector $\mathbf{K}\boldsymbol{\phi}$ of (6.52) is equal to the joint torques without load, consequently, the payload inertial parameters could be identified by (6.52) after replacing $\mathbf{K}\boldsymbol{\phi}$ by $\boldsymbol{\tau}_0$ representing the joint torques on the same trajectory without load

$$(\boldsymbol{\tau} - \boldsymbol{\tau}_0) = \mathbf{K}_\mathrm{L}\boldsymbol{\phi}_\mathrm{L} \,. \tag{6.53}$$

This method assumes that the difference between the joint variables with and without payload is negligible and that the friction parameters are the same with and without load. If the friction parameters vary with respect to the load we can estimate this variation at the same time as the load inertial parameter.

Estimating the Robot Parameters and the Load Parameters in One Step. In this method the identification model is constructed by grouping two sets of equations: the first using a trajectory without load, and the second using a trajectory when the robot is holding the load. In this case the two sets of trajectories could be different. The link parameters and the payload inertial parameters are estimated using the least squares solution of

$$\begin{pmatrix} \boldsymbol{\tau}_a \\ \boldsymbol{\tau}_b \end{pmatrix} = \begin{pmatrix} \mathbf{K}_a & \mathbf{0} \\ \mathbf{K}_b & \mathbf{K}_\mathrm{L} \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi} \\ \boldsymbol{\phi}_\mathrm{L} \end{pmatrix}, \tag{6.54}$$

where the indices a and b indicate the trajectories without and with load, respectively.

This method has the advantage of using a global identification procedure that can avoid the accumulation of errors of previous methods.

These three methods have been validated experimentally on the 6-DOF robot RX-90 of Staubli. They give similar results [6.30].

6.3.3 Identification of Total Joint Drive Gains

For gear electric drive of joint j, the joint torque τ_j is estimated as $\tau_j = v_{rj} g_j$, where v_{rj} is the control signal computed by the numerical controller of the robot (the input references of the motor current loop) and g_j is the total joint j drive gain, given by the gear ratio, the current amplifier gain, and the motor torque constant.

Because of large values of the gear ratio for industrial robots, (> 50), joint drive gain is very sensitive to errors and must be accurately measured from special, time consuming, heavy tests, on the drive chain [6.32, 33].

The methods used to identify the load inertial parameters using joint torques can be inversed to identify the drive gains. In this case some load inertial parameters ϕ_{kL} must be known while the other unknown parameters ϕ_{uL} are identified with the gain $\boldsymbol{g}$.

The vector $\boldsymbol{\tau}$ of sampled joint torques in (6.52) or (6.54) is rewritten as

$$\boldsymbol{\tau} = \mathbf{V}_r \boldsymbol{g} \,, \tag{6.55}$$

where: $\boldsymbol{g}$ is the $(n \times 1)$ vector of the joint drive gains, $\mathbf{V}_r$ regroups the samples of v_{rj}.

Using the Same Set of Trajectory Twice, with and Without the Load

Introducing (6.55) and the known load inertial parameters in (6.53) with the expression

$$\mathbf{K}_L \phi_L = \mathbf{K}_{kL} \boldsymbol{\phi}_{kL} + \mathbf{K}_{uL} \boldsymbol{\phi}_{uL} \tag{6.56}$$

gives

$$(\mathbf{V}_r - \mathbf{V}_{ruL}) \boldsymbol{g} = \mathbf{K}_{kL} \boldsymbol{\phi}_{kL} + \mathbf{K}_{uL} \boldsymbol{\phi}_{uL} \,. \tag{6.57}$$

This equation can be used to identify $\boldsymbol{\phi}_{uL}$ and $\boldsymbol{g}$ as the *LS* solution of the system

$$[\mathbf{K}_{kL} \boldsymbol{\phi}_{kL}] = \begin{pmatrix} (\mathbf{V}_r - \mathbf{V}_{ruL}) & -\mathbf{K}_{uL} \end{pmatrix} \begin{pmatrix} \boldsymbol{g}^T & \boldsymbol{\phi}_{uL}^T \end{pmatrix}^T \,. \tag{6.58}$$

Using Two Sets of Trajectories (Same or Different), the First Set Without Load and the Second Set with the Load

Introducing (6.55) and the known load inertial parameters (6.56) in (6.54) gives

$$\begin{pmatrix} \mathbf{V}_{ra} \\ \mathbf{V}_{rb} \end{pmatrix} \boldsymbol{g} = \begin{pmatrix} \mathbf{K}_a & 0 & 0 \\ \mathbf{K}_b & \mathbf{K}_{uL} & \mathbf{K}_{kL} \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi}^T & \boldsymbol{\phi}_{uL}^T & \boldsymbol{\phi}_{kL}^T \end{pmatrix}^T \,. \tag{6.59}$$

Equation (6.59) can be used to identify all the parameters, $\boldsymbol{g}$, $\boldsymbol{\phi}$ and $\boldsymbol{\phi}_{uL}$, as the single global *LS* solution of the system

$$\begin{pmatrix} 0 \\ \mathbf{K}_{kL} \boldsymbol{\phi}_{kL} \end{pmatrix} = \begin{pmatrix} \mathbf{V}_{ra} & -\mathbf{K}_a & 0 \\ \mathbf{V}_{rb} & -\mathbf{K}_b & -\mathbf{K}_{uL} \end{pmatrix} \begin{pmatrix} \boldsymbol{g}^T & \boldsymbol{\phi}^T & \boldsymbol{\phi}_{uL}^T \end{pmatrix}^T \,. \tag{6.60}$$

But owing to the fact that $\mathbf{K}_{kL}$ and $\mathbf{K}_{uL}$ are correlated by the same noisy data $(q, \dot{q}, \ddot{q})$, the total least squares (TLS) solution is used to avoid biais errors of the ordinary *LS* solution [6.34]. Rewritting (6.60) as

$$\mathbf{K}_{tot} \boldsymbol{\phi}_{tot} = \mathbf{0} \,, \tag{6.61}$$

where

$$\mathbf{K}_{tot} = \begin{pmatrix} \mathbf{V}_{ra} & -\mathbf{K}_a & 0 & 0 \\ \mathbf{V}_{rb} & -\mathbf{K}_b & -\mathbf{K}_{uL} & -\mathbf{K}_{kL} \boldsymbol{\phi}_{kL} \end{pmatrix} , \tag{6.62}$$

$$\boldsymbol{\phi}_{tot} = \begin{pmatrix} \boldsymbol{g}^T & \boldsymbol{\phi}^T & \boldsymbol{\phi}_{uL}^T & 1 \end{pmatrix}^T \,. \tag{6.63}$$

The singular value decomposition (*SVD*) of $\mathbf{K}_{tot}$ is given by

$$\mathbf{K}_{tot} = \mathbf{U} \mathbf{S} \mathbf{V}^T \,, \tag{6.64}$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthonormal matrices, and $\mathbf{S} = \mathrm{diag}(s_i)$ is a diagonal matrix with singular values s_i of $\mathbf{K}_{tot}$ sorted in decreasing order. There are infinity of vectors $\hat{\boldsymbol{\phi}}_{tot} = \lambda \mathbf{V}_c$ which are the *TLS* solutions of (6.61), depending on a scale factor λ and on the normalized *TLS* solution given by the last column $\mathbf{V}_c$ of $\mathbf{V}$, $||\mathbf{V}_c|| = 1$, corresponding to the smallest singular value s_c [6.34]. The unique solution $\hat{\boldsymbol{\phi}}_{tot}^*$ assumes that its last coefficient $\hat{\boldsymbol{\phi}}_{tot}^*(end)$ must be equal to 1 according to (6.63) such as

$$\hat{\boldsymbol{\phi}}_{tot}^* = \frac{\mathbf{V}_c}{\mathbf{V}_c(\mathrm{end})} \tag{6.65}$$

The methods has been successfully experimentally validated on the 6-DOF Staubli TX-40 industrial robot [6.35].

This approach is very simple to perform and the experimental results have shown its effectiveness: for the global identification of the total joint drive gains and the dynamic parameters, it is only necessary to accurately weigh a payload mass and to carry standard trajectories of industrial robot.

6.3.4 Link Parameter Estimation for More Complex Structures

In this section we present the dynamic identification model for kinematic tree (spanning tree) robots and

kinematic closed-loop robots including parallel robots. These models are linear in the inertial parameters and can be represented by a model similar to (6.47), thus these parameters can be identified using the same techniques.

Tree-Structured Robots

For an n-link manipulator with a kinematic tree structure, the dynamic identification model (6.46) must be modified to take into account that the inertial parameters of link j have no effect on the dynamics of links which do not belong to the chain from the base to that link. For such structures, the link on which link j is articulated is denoted by $p(j)$ (parent of j – see Sect. 3.4). It could be a link with any number i, such that $i < j$. Thus the nonzero elements of the column of the matrix representing the coefficients of the inertial parameters of link j in (6.46) would be

$$ {}^{j}\mathbf{X}_{j}^{\mathrm{F}j}\mathbf{A}_{j}, {}^{p(j)}\mathbf{X}_{j}^{\mathrm{F}j}\mathbf{A}_{j}, {}^{p(p(j))}\mathbf{X}_{j}^{\mathrm{F}j}\mathbf{A}_{j}, \ldots, {}^{b}\mathbf{X}_{j}^{\mathrm{F}j}\mathbf{A}_{j} , \qquad (6.66) $$

where b is the first link of the chain connecting link 0 to link j, thus $p(b) = 0$.

Consequently, in (6.47) of a tree robot we obtain: $\mathbf{K}_{ij} = \mathbf{0}_{1\times 10}$ if $i > j$ or if link i does not belong to the branch connecting link 0 to link j. This means that some elements of the upper-right submatrix of $\mathbf{K}$ will be zero.

We note that a serial structure is a particular case of a tree structure where $p(j) = j - 1$. Thus any link m such that $m < j$ will belong to the chain from the base to link j.

Closed-Loop Robots

The dynamic model of a closed-loop structure can be obtained by considering first an equivalent spanning tree by opening one joint at each closed loop, and then using the principle of virtual work such that

$$ \boldsymbol{\tau} = \mathbf{G}^{\mathrm{T}}\mathbf{K}_{\mathrm{tr}}\boldsymbol{\phi} , \quad \mathbf{G} = \left(\frac{\partial \boldsymbol{q}_{\mathrm{tr}}}{\partial \boldsymbol{q}_{\mathrm{a}}}\right) , \qquad (6.67) $$

with $\boldsymbol{q}_{\mathrm{a}}$ an $N \times 1$ vector of N active joint angles (N differs from n, the total number of joints), and $\boldsymbol{q}_{\mathrm{tr}}$ an $n \times 1$ vector of the joints of the spanning tree structure.

Parallel Robots

A parallel robot is a particular case of closed-loop robots (Chap. 18). It is composed of a moving platform, representing the terminal link, that is connected to the base by m parallel legs. The dynamic model of parallel robots can be given by [6.36]

$$ \boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\mathbf{A}_{p}\boldsymbol{\phi}_{p} + \sum_{i=1}^{m}\left(\frac{\partial \boldsymbol{q}_{i}}{\partial \boldsymbol{q}_{\mathrm{a}}}\right)^{\mathrm{T}}\mathbf{K}_{i}\boldsymbol{\phi}_{i} , \qquad (6.68) $$

where $\mathbf{K}_i\boldsymbol{\phi}_i$ represents the dynamic model of leg i with $\mathbf{K}_i$ a function of $(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_i)$, $\boldsymbol{q}_i$ is the vector of joint angles of leg i, $\mathbf{A}_p\boldsymbol{\phi}_p$ is the Newton–Euler spatial force of the platform calculated in terms of the Cartesian variables of the mobile platform using (6.41), and $\mathbf{J}$ is the Jacobian matrix of the parallel manipulator. We can obtain (6.68) from (6.67) by supposing that the spanning tree structure is obtained by separating the mobile platform from the legs.

Equation 6.68 can be rewritten as

$$ \begin{aligned} \boldsymbol{\tau} &= \left[\mathbf{J}^{\mathrm{T}}\mathbf{A}_{p} \quad (\partial \boldsymbol{q}_{1}/\partial \boldsymbol{q}_{\mathrm{a}})^{\mathrm{T}}\mathbf{K}_{1} \quad \cdots \quad (\partial \boldsymbol{q}_{m}/\partial \boldsymbol{q}_{\mathrm{a}})^{\mathrm{T}}\mathbf{K}_{m}\right] \\ &\quad \boldsymbol{\phi}_{\mathrm{par}} \\ \boldsymbol{\tau} &= \mathbf{K}_{\mathrm{par}}\left(\boldsymbol{\omega}_{p}, \dot{\boldsymbol{v}}_{p}, \dot{\boldsymbol{\omega}}_{p}, \boldsymbol{q}_{i}, \dot{\boldsymbol{q}}_{i}, \ddot{\boldsymbol{q}}_{i}\right)\boldsymbol{\phi}_{\mathrm{par}} , \end{aligned} \qquad (6.69) $$

where $\boldsymbol{\phi}_{\mathrm{par}}$ is the vector of the inertial parameters of the robot (the legs and the platform).

$$ \boldsymbol{\phi}_{\mathrm{par}} = \begin{pmatrix} \boldsymbol{\phi}_{p} \\ \boldsymbol{\phi}_{1} \\ \vdots \\ \boldsymbol{\phi}_{m} \end{pmatrix} . $$

In the common case where the legs are identical and their inertial parameters are denoted by $\boldsymbol{\phi}_{\mathrm{leg}}$, the identification model can be rewritten by the following equation, which considerably reduces the number of inertial parameters to be identified

$$ \boldsymbol{\tau} = \left[\mathbf{J}^{\mathrm{T}}\mathbf{A}_{p} \quad \sum_{i=1}^{m}(\partial \boldsymbol{q}_{i}/\partial \boldsymbol{q}_{\mathrm{a}})^{\mathrm{T}}\mathbf{K}_{i}\right]\begin{pmatrix} \boldsymbol{\phi}_{p} \\ \boldsymbol{\phi}_{\mathrm{leg}} \end{pmatrix} . \qquad (6.70) $$

The identification of the orthoglide parallel robot is given in [6.37].

6.4 Identifiability and Numerical Conditioning

That some inertial parameters in (6.47) are unidentifiable does not mean that they are intrinsically identifiable, only that the experimental setup does not allow them to be identified. Limited motion near the base could be fixed by placing the whole manipulator on a six-axis moving platform such as a Stewart–Gough platform. In fact, for mobile manipulators mounted on high-mobility vehicles such as satellites, it may be necessary to know the full inertial model of the manipulator. Additional sensors could be added, for ex-

ample a six-axis force-torque sensor to the base of the robot [6.5], to identify some additional (but not all) inertial parameters.

A similar situation may arise in kinematic calibration, where for example joint models might be augmented to include gear eccentricities, transmission and coupling coefficients of gears, joint elasticity, link elasticity, and base deflection [6.38]. By instrumenting the robot with additional sensors, for example, by placing joint angle sensors before and after the gears to measure joint deflection, it may be possible to identify additional parameters of interest. Additional sensing with a theodolite was done to measure base deflection in [6.12].

Given that the experimental setup is what it is, there may be no recourse but to deal with unidentifiable parameters, treated in the next section. Another problem is that parameters might in principle be identifiable, but numerical conditioning prevents their accurate determination. Issues such as adequacy of the collected data, and the effect of different units and magnitudes of parameters, are treated subsequently.

6.4.1 Identifiability

There are two main approaches towards handling unidentifiable parameters, depending on whether the goal is a structural model or a prediction model. For a structural model, the goal is to find the minimum parameter set that provides a meaningful physical description of the system by eliminating parameters until all are identifiable. This is done by careful evaluation of the effect of each parameter of the original model.

For a prediction model, the goal is to match the outputs to the inputs as closely as possible, and is more of a curve-fitting approach. The resulting parameter values are not necessarily physically meaningful or accurate.

Structural Model

It may be possible in the initial model formation to avoid including redundant or unidentifiable parameters. At other times, it is not possible to determine a priori what the minimal parameter set is, because of system complexity or because of numerical difficulties such as measurement error or limitations in the collected data.

A Priori Parameter Elimination. In the initial model formation, the choice of representation can immediately dictate whether the model is redundant or minimal. For kinematic calibration, the minimal parameter set includes four parameters for a rotary joint and two parameters for a prismatic joint. The detailed presentation in Sect. 6.2.1 of how to set up the DH/Hayati parameters for different kinematic and sensing arrangements has the advantage that the parameter set is minimal (excepting joint models). Once done, the application to any manipulator is somewhat formulaic.

Five- and six-parameter joint models have also been proposed, either for the convenience of locating link coordinate systems or for the ease of model formation [6.13, 28, 39]. With such redundant parameter sets, extra steps must be taken to handle the numerical difficulties caused by the redundancy, such as by reducing the number of parameters in a postprocessing step. When there are complicated joint models that include gearing effects as mentioned above, the resultant large numbers of parameters make the parameter elimination problem more difficult. Determining the minimal set of inertial parameters offers a different sort of complexity, because large numbers of parameters are not identifiable or are identifiable only in linear combinations. Again, model reduction is one approach to handle this problem.

Two approaches towards determining a minimal or base parameter set are:

1. Numerical identification of the unidentifiable parameters or linear combinations of parameters.
2. Symbolic determination.

Numerical identification involves rank reduction through matrix manipulation of the regressor, either through QR decomposition or singular value decomposition [6.40]. If the kinematic or dynamic model is known exactly, then a simulation using exactly generated *data* will yield a noise-free regressor matrix $\mathbf{A}$ in (6.3). For QR decomposition, the regressor matrix is factored as

$$\mathbf{A} = \mathbf{Q}\begin{pmatrix}\mathbf{R}\\ \mathbf{0}_{MP-N_{\text{par}},N_{\text{par}}}\end{pmatrix}, \tag{6.71}$$

where $\mathbf{Q}$ is an $M \times MP$ orthogonal matrix, $\mathbf{R}$ is an $N_{\text{par}} \times N_{\text{par}}$ upper-triangular matrix, and $\mathbf{0}_{MP-N_{\text{par}},N_{\text{par}}}$ is the zero matrix with dimensions $MP - N_{\text{par}} \times N_{\text{par}}$. Theoretically, the unidentifiable parameters ϕ_i are those whose corresponding diagonal elements R_{ii} of matrix $\mathbf{R}$ are zero. In practice, $|R_{ii}|$ is considered to be zero if it is less than the numerical zero ζ [6.41]

$$\zeta = MP\epsilon \max_i |R_{ii}| , \tag{6.72}$$

where ϵ is the computer precision [6.41]. Other parameters may appear in linear combinations depending on how many nonzero elements there are in row j of $\mathbf{R}$. Resolving these linear combinations is arbitrary. One approach is to zero all elements in the linear combination save one. The result is a prediction model rather than a structural model.

Symbolic determination of the base inertial parameters has been proposed in [6.40, 42]. Using an energy formulation, the total energy of link j is expressed as $\boldsymbol{h}_j\phi_j$, where $\boldsymbol{h}_j$ is a row vector termed the total energy function whose elements are kinematic expressions of the angular and linear velocity of link j plus gravity. A recursive relation between neighboring links is developed as

$$\boldsymbol{h}_j = \boldsymbol{h}_{j-1}{}^{j-1}\boldsymbol{\lambda}_j + \dot{q}_j\boldsymbol{\eta}_j \,, \tag{6.73}$$

where the elements of the 10×10 matrix ${}^{j-1}\boldsymbol{\lambda}_j$ are functions of the DH parameters defining frame j, and the elements of the 1×10 row vector $\boldsymbol{\eta}_j$ depend on the linear and angular velocities. The details of these expressions can be found in [6.40, 42]. Grouping rules are then developed to find the precise linear combinations of parameters.

The minimum inertial parameters for a tree structure can be obtained using a closed-form solution similar to the serial structure case [6.42].

The term $\mathbf{K}_{\mathrm{tr}}\boldsymbol{\phi}$ in (6.67) shows that the minimum parameters of the spanning tree structure can be used to calculate the dynamics of the closed structure, giving by this a first reduction and grouping of the inertial parameters. However extra parameters can be eliminated or grouped when considering the matrix $\mathbf{G}$. The case of parallelogram loops can be treated symbolically using closed-form relations [6.42]. For general closed loops, the minimum parameters must be determined using numerical methods such as QR decomposition [6.40].

In the case of parallel robots, we deduce from (6.68) that the minimum parameters of the legs can be used to calculate $\mathbf{K}_i\boldsymbol{\phi}_i$, giving a first reduction of the inertial parameters. However, some other parameters can be grouped with the parameters of the platform. The minimum parameters of the Gough–Stewart robot are given in [6.43].

Data-Driven Parameter Elimination. The singular value decomposition of the regressor matrix can show which parameters are unidentifiable, weakly identifiable, or identifiable only in linear combination. For N_{par} parameters, P data points, and M-dimensional output measurements at each data point, the regressor matrix $\mathbf{A}$ (6.3) or (6.7) can be decomposed as

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}} \,, \tag{6.74}$$

where $\mathbf{U}$ is an $MP\times MP$ orthogonal matrix, $\mathbf{V}$ is an $N_{\mathrm{par}}\times N_{\mathrm{par}}$ orthogonal matrix, and $\boldsymbol{\Sigma}$ is the $MP\times N_{\mathrm{par}}$ matrix of singular values

$$\boldsymbol{\Sigma} = \begin{pmatrix} \mathbf{S} \\ \mathbf{0}_{MP-N_{\mathrm{par}},N_{\mathrm{par}}} \end{pmatrix} , \tag{6.75}$$

where $\mathbf{S} = \mathrm{diag}(\mu_1,\ldots,\mu_r,0,\ldots,0)$ is the $N_{\mathrm{par}}\times N_{\mathrm{par}}$ matrix of ordered singular values, with μ_1 the largest and μ_r the smallest nonzero singular value. There may be $N_{\mathrm{par}}-r$ zero singular values $\mu_{r+1}=\cdots=\mu_{N_{\mathrm{par}}}=0$.

Especially when complex joint models are assumed that include flexibility, backlash, and gear eccentricity, it is not clear that all parameters can be identified. Retaining poorly identifiable parameters will degrade the robustness of the calibration; such parameters are indicated by zero or very small singular values. The expansion of (6.7) in terms of (6.74) is

$$\Delta\boldsymbol{y} = \sum_{j=1}^{r} \mu_j(\boldsymbol{v}_j^{\mathrm{T}}\Delta\boldsymbol{\phi})\boldsymbol{u}_j \,, \tag{6.76}$$

where $\boldsymbol{u}_j$ and $\boldsymbol{v}_j$ are the j-th columns of $\mathbf{U}$ and $\mathbf{V}$. For zero or small singular values μ_j, the projection $\boldsymbol{v}_j^{\mathrm{T}}\Delta\boldsymbol{\phi}$ of the parameters onto the column vector $\boldsymbol{v}_j$ in general represents a linear combination of parameters. It is also possible that only a single parameter results from the projection.

To proceed, it is first necessary to scale the parameters and the output measurements so that the singular values are comparable. Scaling is discussed in Sect. 6.4.3. Small singular values signal that there are parameters that are poorly identifiable and that should be eliminated. By small singular values, *Schröer* [6.38] suggests the heuristic that the *condition number* of a well-conditioned regressor matrix should be less than 100

$$\kappa(\mathbf{A}) = \frac{\mu_1}{\mu_r} < 100 \,. \tag{6.77}$$

This heuristic derives from experience of their statistical community. If the condition number is above 100, the singular values are examined, beginning with the smallest one, which may be a zero singular value.

If the condition number is above 100, the linear sums (6.76) corresponding to the smallest singular value μ_r are examined. The elements of column $\boldsymbol{v}_r$ are in one-to-one correspondence with the elements of $\Delta\boldsymbol{\phi}$. If there is an element j of $\boldsymbol{v}_r$ that is much larger than the others, then the parameter ϕ_j corresponding to that column element is a candidate for elimination. This procedure will tend to pinpoint parameters that are totally unidentifiable. Isolating the largest element of $\boldsymbol{v}_r$ is once again only meaningful if the parameters have been previously scaled.

Once the parameter is eliminated, the condition number of the reduced regressor is computed again. The procedure iterates until the condition number of the regressor is below 100.

The previous procedure can be carried out using QR decomposition as in (6.71), by replacing ϵ representing

the computer precision in (6.72) by a greater value function of the noise level.

Prediction Model

If there are several largest elements of $\boldsymbol{v}_r$ that are roughly comparable in magnitude, then these parameters may only be identifiable in linear combination. There will be the same number of singular values which are too small. By examining several columns $\boldsymbol{v}_j$ corresponding to the small singular values, these linear combinations can become apparent. The linear combinations can be resolved arbitrarily, i. e., all parameter elements in the linear combination save one can be set to zero. The result of zeroing some parameters is that the model is no longer a structural model, but rather a prediction model.

One may also proceed without first removing parameters. It can be shown by substituting the singular value decomposition that

$$(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}} = \mathbf{V}\left(\mathbf{S}^{-1}\ \mathbf{0}_{N_{\mathrm{par}},MP-N_{\mathrm{par}}}\right)\mathbf{U}^{\mathrm{T}}\,. \tag{6.78}$$

Consequently the solution to (6.8) can be expressed as [6.44]

$$\Delta\boldsymbol{\phi} = \sum_{j=1}^{N_{\mathrm{par}}} \frac{\boldsymbol{u}_j^{\mathrm{T}}\Delta\boldsymbol{y}}{\mu_j}\boldsymbol{v}_j\,. \tag{6.79}$$

One can see that poorly identifiable parameters corresponding to small singular values μ_j greatly perturb the estimates, because the weighting is $1/\mu_j$. The strategy is to remove their influence. If μ_j is zero or very small relative to the largest singular value μ_1, then set $1/\mu_j = 0$.

Parameters that cannot be identified well are simply ignored in this procedure, which automatically converges to an identifiable parameter set. The resulting parameters can then be used for the model. A disadvantage is that the resulting parameters may not correspond to real model parameters. A procedure to project these parameters on the real ones is given in [6.40].

Incorporating A Priori Parameter Estimates

Least squares treats parameter values as completely unknown, i. e., they could be anywhere in the range from $-\infty$ to $+\infty$. Yet often a fairly good initial estimate of the parameters is available, for example, from a manufacturer's specifications or in the case of a recalibration. It makes sense to incorporate such an a priori preference into the least-squares optimization [6.45].

Suppose there is a preference for a solution near $\boldsymbol{\phi} = \boldsymbol{\phi}_0$. Express the preference as $\mathbf{I}\boldsymbol{\phi} = \boldsymbol{\phi}_0$, where $\mathbf{I}$ is the identity matrix, and append it as additional rows to (6.3) that reflect this preference

$$\begin{pmatrix}\mathbf{A}\\ \mathbf{I}\end{pmatrix}\boldsymbol{\phi} = \begin{pmatrix}\boldsymbol{y}\\ \boldsymbol{\phi}_0\end{pmatrix}. \tag{6.80}$$

To proceed with the solution, we can treat $\boldsymbol{\phi}_0$ as a constant. Redefine the parameter vector as $\tilde{\boldsymbol{\phi}} = \boldsymbol{\phi} - \boldsymbol{\phi}_0$, which we expect to be close to zero. Then

$$\begin{pmatrix}\mathbf{A}\\ \mathbf{I}\end{pmatrix}\tilde{\boldsymbol{\phi}} = \begin{pmatrix}\tilde{\boldsymbol{y}}\\ \mathbf{0}\end{pmatrix}, \tag{6.81}$$

where $\tilde{\boldsymbol{y}} = \boldsymbol{y} - \mathbf{A}\boldsymbol{\phi}_0$. We may not know $\boldsymbol{\phi}_0$ perfectly well, so we add a weighting parameter λ to express the confidence in this value

$$\begin{pmatrix}\mathbf{A}\\ \lambda\mathbf{I}\end{pmatrix}\tilde{\boldsymbol{\phi}} = \begin{pmatrix}\tilde{\boldsymbol{y}}\\ \mathbf{0}\end{pmatrix}. \tag{6.82}$$

The larger is λ, the more confidence we have in our a priori estimate. The least-squares solution is therefore

$$\begin{aligned}\tilde{\boldsymbol{\phi}} &= \left(\begin{pmatrix}\mathbf{A}^{\mathrm{T}} & \lambda\mathbf{I}\end{pmatrix}\begin{pmatrix}\mathbf{A}\\ \lambda\mathbf{I}\end{pmatrix}\right)^{-1}\begin{pmatrix}\mathbf{A}^{\mathrm{T}} & \lambda\mathbf{I}\end{pmatrix}^{\mathrm{T}}\begin{pmatrix}\tilde{\boldsymbol{y}}\\ \mathbf{0}\end{pmatrix}\\ &= (\mathbf{A}^{\mathrm{T}}\mathbf{A} + \lambda^2\mathbf{I})^{-1}\mathbf{A}^{\mathrm{T}}\tilde{\boldsymbol{y}}\,.\end{aligned} \tag{6.83}$$

This solution is called *damped least squares*, where λ is the damping factor. Expanding this solution in terms of the singular value decomposition,

$$\tilde{\boldsymbol{\phi}} = \sum_{j=1}^{N_{\mathrm{par}}}(\boldsymbol{u}_j^{\mathrm{T}}\tilde{\boldsymbol{y}})\frac{\mu_j}{\mu_j^2+\lambda^2}\boldsymbol{v}_j\,. \tag{6.84}$$

Hence a very small μ_j is counteracted by the larger λ value; then the a priori information about parameter values dominates the information from the data, i. e., the data are ignored. Thus for damped least squares, no explicit action on the singular values is required, because the damping factor modifies the singular values. The solution will be perturbed over the normal least-squares solution by the magnitude of the λ choice.

Confidence Measure of Parameter Estimates

After calibration, an estimate $\hat{\mathbf{M}}$ of the covariance of the parameter estimates can be derived from the data [6.3]. Assume that the task variables $\Delta\boldsymbol{y}$ have previously been scaled for equal uncertainty, that there is no bias, and that the errors are uncorrelated. Then

$$\hat{\mathbf{M}} = \sigma^2(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\,. \tag{6.85}$$

An estimate of the standard deviation σ after running the calibration procedure is obtained from the χ^2 statistic [6.44, 46]

$$\chi^2 = (\Delta\boldsymbol{y} - \mathbf{A}\Delta\hat{\boldsymbol{\phi}})^{\mathrm{T}}(\Delta\boldsymbol{y} - \mathbf{A}\Delta\hat{\boldsymbol{\phi}})\,. \tag{6.86}$$

An unbiased estimator for σ^2 is $\hat{\sigma}^2 = \chi^2/\nu$, where $\nu = MP - N_{\text{par}}$ is called the statistical *degrees of freedom*; ν subtracts from the total number of measurements MP the number N_{par} of parameters estimated, since some of the measurements went into determining $\boldsymbol{\phi}$.

The estimate $\hat{\mathbf{M}}$ can be used as a basis for eliminating parameters, by choosing those with the largest variances.

6.4.2 Observability

The measurements that are taken will influence the accuracy of parameter estimation. In kinematic calibration, the quality of the pose set has been measured by an *observability index*. In inertial parameter estimation, the quality of the identification trajectory has been termed *persistent excitation* [6.47]. Regardless of whether data is collected statically as in kinematic calibration or dynamically as in inertial parameter estimation, the result is just a bunch of numbers that go into the regressor matrix, and so common terminology is appropriate.

In statistics, optimal experimental design theory has given rise to several data measures termed *alphabet optimalities* [6.48]. Some of the most prominent are the following:

- *A-optimality* minimizes the trace of $(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}$ to obtain regression designs.
- *D-optimality* maximizes the determinant of $(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}$.
- *E-optimality* maximizes the minimum singular value of $(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}$.
- *G-optimality* minimizes the maximum prediction variance, and does not have a simple expression in terms of singular values.

Although ties to the experimental design literature have not typically been made [6.49], several of the proposed observability indexes for robot calibration have an alphabet-optimality counterpart. A-optimality does not have a counterpart in the robot calibration literature, and conversely a few proposed observability indexes do not have an alphabet-optimality counterpart. In [6.49], it is proved that E-optimality and G-optimality are equivalent for exact design.

Borm et al. [6.50, 51] proposed an observability index (here termed O_1 and numbered consecutively below) that maximizes the product of all of the singular values

$$O_1 = \frac{\sqrt[r]{\mu_1 \cdots \mu_r}}{\sqrt{P}} \,. \tag{6.87}$$

This is similar to D-optimality. The rationale is that O_1 represents the volume of a hyperellipsoid in $\Delta\boldsymbol{y}$, defined by (6.7) when $\Delta\boldsymbol{\phi}$ defines a hypersphere, and the singular values represent the lengths of axes. Therefore maximizing O_1 gives the largest hyperellipsoid volume, and hence a good aggregate increase of the singular values. One can also derive O_1 from the well-known relation $\sqrt{\det(\mathbf{A}^{\mathrm{T}}\mathbf{A})} = \mu_1 \cdots \mu_r$.

Minimizing the condition number of $\mathbf{A}$ as a measure of observability has been proposed in [6.40, 52, 53]

$$O_2 = \frac{\mu_1}{\mu_r} \,. \tag{6.88}$$

O_2 measures the eccentricity of the hyperellipsoid rather than its size. The intermediate singular values are not considered to be that pertinent, because minimizing the condition number automatically makes all singular values become more similar in magnitude and makes the hyperellipsoid closer to a hypersphere.

Nahvi et al. [6.22] argue for maximizing the minimum singular value μ_r as an observability measure

$$O_3 = \mu_r \,. \tag{6.89}$$

This is similar to E-optimality. The rationale is to make the shortest axis as long as possible, regardless of the other axes, that is to say, to improve the worst case. Consider the following standard result [6.4]

$$\mu_r \le \frac{\|\Delta\boldsymbol{y}\|}{\|\Delta\boldsymbol{\phi}\|} \le \mu_1 \,, \tag{6.90}$$

or more particularly

$$\mu_r \|\Delta\boldsymbol{\phi}\| \le \|\Delta\boldsymbol{y}\| \,. \tag{6.91}$$

Then maximizing μ_r ensures that a given size of parameter errors $\|\Delta\boldsymbol{\phi}\|$ has the largest possible influence on the pose errors $\|\Delta\boldsymbol{y}\|$.

Nahvi and *Hollerbach* [6.54] proposed the noise amplification index O_4, which can be viewed as combining the condition number O_2 with the minimum singular value O_3

$$O_4 = \frac{\sigma_r^2}{\sigma_1} \,. \tag{6.92}$$

The rationale is to measure both the eccentricity of the ellipse through O_2 as well as the size of the ellipse through O_3. The noise amplification index was argued to be the most sensitive to measurement error and modeling error.

Hollerbach and *Lokhorst* [6.21] found experimentally that the condition number and the minimum singular value gave about the same good result: their

relative magnitudes were almost proportional to the root-mean-square (RMS) errors of the final parameter errors. The observability index O_1 was not as sensitive and not directly related to parameter errors. *Sun* and *Hollerbach* [6.49] derived general relations among the observability indexes and alphabet optimalities

$$O_1 \geq \text{A-optimality} \geq O_3 \,. \tag{6.93}$$

They further showed that if $\mu_1 \geq 1$, then $O_3 \geq O_2$, and if $\mu_r \leq 1$, then $O_2 \geq O_4$. They also argued that O_3 (D-optimality) is in general the best index, because it minimizes the variance of the parameters and also minimizes the uncertainty of the end-effector pose.

Optimal Experimental Design

An observability index is typically used to decide how many data points to collect. As one begins to add data points, the observability increases, then plateaus. Adding additional data does not then improve the quality of the estimates. For kinematic calibration, data may be added through random selection, or optimal design methods can be employed that can drastically reduce the amount of data required [6.55, 56]. Optimal experimental designs work by measuring the effect of adding or exchanging data points [6.57].

Exciting Trajectories

For inertial parameter estimation, data points are not independent because they result from a trajectory rather than isolated poses. Therefore the issue is what type of trajectory to generate. Industrial robots usually have joint position point-to-point trajectory generators. A continuous and smooth trajectory is calculated by interpolating between these points, assuming zero initial and final velocity and acceleration at each point, and using polynomial interpolators. Excitation trajectories are obtained by minimizing an observability index, using nonlinear optimization techniques to calculate the polynomial coefficients, under the constraints of the joint positions, velocities, and accelerations limits [6.52].

It is possible to facilitate the optimization by proceeding to a sequential excitation procedure. Specific trajectories, which structurally excite a small number of parameters, are easier to optimize. For example, moving one joint at a time with constant velocities excites friction and gravity parameters. In this approach, sequential identification is avoided. However, it is better to collect all the data and to proceed to a globally weighted least-squares estimation [6.29]. This procedure avoids the accumulation of estimation errors and allows the calculation of the confidence intervals (6.85).

Specific trajectories have been proposed like sinusoidal interpolators [6.58], or periodic trajectories obtained from a spectral analysis of the contribution function of the parameters [6.59]. This is a general planning trajectory strategy, which is very important to get accurate experimental identification [6.60].

6.4.3 Scaling

The numerical conditioning of parameter estimation can be improved by scaling both the output measurements (task variable scaling) and the parameters.

Task Variable Scaling

When performing a least-squares analysis on the endpoint pose error, position errors and orientation errors have to be combined (6.22)

$$||\Delta \boldsymbol{y}^i||^2 = ||\Delta^{-1} \boldsymbol{p}^l_{n+1}||^2 + ||\Delta^{-1} \boldsymbol{\rho}^l_{n+1}||^2 \,. \tag{6.94}$$

However, position error and orientation error have different units, and so are not comparable. Furthermore, not all position or orientation components may be measured with equal accuracy.

Ordinary least squares (6.8) weighs all task variables equally. To weigh these variables differently, the general solution is left multiplication of (6.7) by a scaling matrix $\mathbf{G}$ [6.45],

$$\begin{aligned} \mathbf{G}\Delta \boldsymbol{y} &= \mathbf{G}\mathbf{A}\Delta \boldsymbol{\phi} \,, \\ \Delta \tilde{\boldsymbol{y}} &\equiv \tilde{\mathbf{A}}\Delta \boldsymbol{\phi} \,, \end{aligned} \tag{6.95}$$

where $\Delta \tilde{\boldsymbol{y}} = \mathbf{G}\Delta \boldsymbol{y}$ is the scaled output vector and $\tilde{\mathbf{A}} = \mathbf{G}\mathbf{A}$ is the scaled regressor matrix. The weighted least-squares solution is

$$\Delta \boldsymbol{\phi} = (\tilde{\mathbf{A}}^{\mathrm{T}}\tilde{\mathbf{A}})^{-1}\tilde{\mathbf{A}}^{\mathrm{T}}\Delta \tilde{\boldsymbol{y}} = (\mathbf{A}^{\mathrm{T}}\mathbf{W}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{W}\Delta \boldsymbol{y} \,, \tag{6.96}$$

where $\mathbf{W} = \mathbf{G}^{\mathrm{T}}\mathbf{G}$. Often $\mathbf{W}$ is a diagonal matrix.

One approach to scaling position error versus orientation error is to equalize the effect of a parameter error $\Delta\phi_i$ on either position error or orientation error. Curiously, for human-sized robot arms the effect is equal without scaling due to an accident of metric units. If θ is the joint angle resolution, then $s = r\theta$ is the resulting endpoint position resolution. For human-sized arms, $r = 1$ m and hence $s = \theta$. Thus meters and radians are directly comparable after all. Thus using no scaling factors for the pose parameters makes some sense, and may explain why the general disregard for scaling in the robotics community has not had more consequences. If linkages are much smaller (like fingers) or much larger (like excavators) then the situation is different.

A more general way of choosing the weighting matrix $\mathbf{W}$ is to use a priori information about acceptable relative errors. Such information might arise from specifications of the measurement apparatus. Suppose that the output variables are subject to independent Gaussian noise, such that σ_j^y is the standard deviation in task variable measurement component Δy_j^i, for $j = 1, \ldots, m$ pose components. Then the individual diagonal weights are $w_{jj} = 1/\sigma_j^y$, and define

$$\mathbf{R}^i = \text{diag}\left[\left(\sigma_1^y\right)^2, \ldots, \left(\sigma_m^y\right)^2\right] ,$$
$$\mathbf{R} = \text{diag}(\mathbf{R}^1, \ldots, \mathbf{R}^P) ,$$

where the weighting matrix $\mathbf{W} = \mathbf{R}^{-1}$, and $\mathbf{R}$ is called the *covariance matrix*.

The weighted least-squares estimate is

$$\Delta\boldsymbol{\phi} = (\mathbf{A}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{R}^{-1}\Delta\boldsymbol{y} . \tag{6.97}$$

The resulting scaled output variable $\Delta\tilde{y}_j^l = \Delta y_j^l/\sigma_j^y$ is dimensionless. The larger the uncertainty σ_j^y, the less this variable influences the least-squares solution relative to other variables. The standard deviations σ_j^y are not necessarily the same as endpoint measurement accuracy, because model errors and input noise also contribute to output errors.

The weighted least-squares solution using standard deviations is variously called the Gauss–Markov estimate, the generalized least-squares estimate, or the best linear unbiased estimator (BLUE) [6.3]. It is the minimum covariance estimate (on the parameter error) of all unbiased estimators. A significant point is that the standard deviations of the scaled components of $\Delta\tilde{\boldsymbol{y}}$ are all about the same size, or the covariance $\tilde{\mathbf{R}} = \text{cov}(\Delta\tilde{\boldsymbol{y}}) = \mathbf{I}$, the identity matrix. Hence the Euclidean norm of the error vector $\Delta\tilde{\boldsymbol{y}}$ is a reasonable measure of its size.

Often we do not know the covariance matrix $\mathbf{R}$ that well. An estimate of the standard deviations after running the calibration procedure is obtained from the χ^2 statistic [6.44, 46]

$$\chi^2 = (\Delta\boldsymbol{y} - \mathbf{A}\Delta\boldsymbol{\phi})^{\mathrm{T}}\mathbf{R}^{-1}(\Delta\boldsymbol{y} - \mathbf{A}\Delta\boldsymbol{\phi}) . \tag{6.98}$$

This equation is the same as the residual error (6.95), with substitution for $\mathbf{W} = \mathbf{R}^{-1}$. χ^2 is nothing more than the weighted residuals after calibration. The expected value of χ^2 is

$$E(\chi^2) \equiv \nu = PK - R , \tag{6.99}$$

where E is the expectation operator. That is to say, the unweighted residuals $(\Delta\boldsymbol{y} - \mathbf{A}\Delta\boldsymbol{\phi})^2$ should, with enough measurements, approximate the true covariance. We may now uniformly scale an initial estimate of $\mathbf{R}$ after a preliminary calibration, based on the value of χ^2,

$$\hat{\mathbf{R}} = \frac{\chi^2}{\nu}\mathbf{R} , \tag{6.100}$$

where $\hat{\mathbf{R}}$ is the revised estimate of the covariance matrix.

Parameter Scaling

Scaling of parameters is important for proper convergence in nonlinear optimization and for singular value decomposition. If parameters have vastly different magnitudes, then the singular values are not directly comparable. Also, parameter scaling can improve the conditioning of the regressor $\mathbf{A}$ and help to avoid invertibility problems.

Whereas left multiplication of $\mathbf{A}$ in (6.95) results in task variable scaling, right multiplication of $\mathbf{A}$ by a weighting matrix $\mathbf{H}$ results in parameter scaling [6.45]

$$\Delta\boldsymbol{y} = (\mathbf{AH})(\mathbf{H}^{-1}\Delta\boldsymbol{\phi}) \equiv \bar{\mathbf{A}}\Delta\tilde{\boldsymbol{\phi}} , \tag{6.101}$$

where the scaled Jacobian and parameters are $\bar{\mathbf{A}} = \mathbf{AH}$ and $\Delta\tilde{\boldsymbol{\phi}} = \mathbf{H}^{-1}\Delta\boldsymbol{\phi}$. The least-squares solution is not changed by parameter scaling, whereas it is by task variable scaling.

The most common approach towards parameter weighting is column scaling, which does not require a priori statistical information. Define a diagonal matrix $\mathbf{H} = \text{diag}(h_1, \ldots, h_{N_{\text{par}}})$ with elements

$$h_j = \begin{cases} \|\boldsymbol{c}_j\|^{-1} & \text{if } \|\boldsymbol{c}_j\| \neq 0 \\ 1 & \text{if } \|\boldsymbol{c}_j\| = 0 \end{cases} \tag{6.102}$$

where $\boldsymbol{a}_j$ is the j-th column of $\mathbf{A}$. Then (6.101) becomes

$$\Delta\boldsymbol{y} = \sum_{j=1}^{N_{\text{par}}} \frac{\boldsymbol{a}_j}{\|\boldsymbol{a}_j\|}\Delta\phi_j\|\boldsymbol{a}_j\| . \tag{6.103}$$

Suppose that $\Delta\boldsymbol{y}$ has been previously normalized; then its length is meaningful. Each $\boldsymbol{a}_j/\|\boldsymbol{a}_j\|$ is a unit vector, and hence each scaled parameter $\Delta\phi_j\|\boldsymbol{a}_j\|$ is about the same size and has the same effect on $\Delta\boldsymbol{y}$.

Schröer [6.38] identified a problem with column scaling, namely that poor identifiability of parameters can result in very small Euclidean norms, which result in very large scaling factors. This results in strong amplification of uncertainties of $\mathbf{A}$. Instead, Schröer proposed scaling based on the effect on the anticipated error of the robot (as previously discussed under task variable scaling).

In an ideal case, one would have a priori knowledge of the expected value $\boldsymbol{\phi}_0$ of the parameter vector and the standard deviation σ_j^{ϕ} of each of the parameter vector components. More generally, the parameter distribution would be described by a covariance matrix $\mathbf{M}$, but usually this information is not available. Instead, the estimate $\hat{\mathbf{M}}$ of the covariance from (6.85) can be used.

If the output measurement covariance $\mathbf{R}^{-1}$ and parameter error covariance $\mathbf{M}$ are both available, one may define a new least-squares optimization criterion that combines output errors with parameter errors to yield a new χ^2 statistic

$$\chi^2 = (\Delta \boldsymbol{y} - \mathbf{A}\Delta\boldsymbol{\phi})^{\mathrm{T}}\mathbf{R}^{-1}(\Delta \boldsymbol{y} - \mathbf{A}\Delta\boldsymbol{\phi}) + \Delta\boldsymbol{\phi}^{\mathrm{T}}\mathbf{M}^{-1}\Delta\boldsymbol{\phi} \,. \tag{6.104}$$

Its solution is the minimum-variance estimate, which unlike (6.97) is biased

$$\Delta\boldsymbol{\phi} = (\mathbf{A}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{A} + \mathbf{M}^{-1})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{R}^{-1}\Delta\boldsymbol{y} \,. \tag{6.105}$$

The Kalman filter recursively solves the same problem [6.61, 62]; when the state is nonvarying, there is a constant process, and there is no process noise [6.13, 63]. The Gauss–Markov estimate is the limiting case when $\mathbf{M}^{-1}$ is zero, i. e., there is no a priori information about the parameters. Again, there is an issue of determining the covariances. As for the Gauss–Markov estimate, the expected value of χ^2 can be used to uniformly scale $\mathbf{R}$ and $\mathbf{M}$ *post facto* [6.64].

6.4.4 Recursive Least Squares and the Kalman Filter

In a control setting, it may be necessary to update the estimate $\hat{\boldsymbol{\phi}}$ at time step k based upon all the readings up to time step $k-1$. It is possible to recast least squares recursively to perform this updating [6.3], which is strongly analogous to the Kalman Filter. At time step k, there is a new reading

$$\boldsymbol{y}^k = \mathbf{A}^k\,\boldsymbol{\phi} + \boldsymbol{e}^k \,, \tag{6.106}$$

where $\boldsymbol{y}^k$ is of dimension $M \times 1$, $\mathbf{A}^k$ is of dimension $M \times N$, and $\boldsymbol{\phi}$ is of dimension $N \times 1$. Let $\mathbf{R}^k$ be the covariance of $\boldsymbol{e}^k$. This equation is stacked for all readings up to k, and the Gauss–Markov estimate is derived

$$\begin{aligned} \boldsymbol{y}_k &= \mathbf{A}_k\boldsymbol{\phi} + \boldsymbol{e}_k \\ \hat{\boldsymbol{\phi}}_k &= (\mathbf{A}_k^T\mathbf{R}_k^{-1}\mathbf{A}_k)^{-1}\mathbf{A}_k^T\mathbf{R}_k^{-1}\boldsymbol{y}_k \,, \end{aligned} \tag{6.107}$$

where $\boldsymbol{y}_k$ is of dimension $MK \times 1$, and $\mathbf{A}_k$ is of dimension $MK \times N$. $\mathbf{R}_k$ is the covariance of $\boldsymbol{e}_k$, and is a block-diagonal matrix with elements $\mathbf{R}^k$.

To make an analogy to the Kalman filter, define the covariance of $\hat{\boldsymbol{\phi}}$ as $\mathbf{P}_k = (\mathbf{A}_k^T\mathbf{R}_k^{-1}\mathbf{A}_k)^{-1}$ of dimension $N \times N$. Inverting this relation eventually yields a recursive reltionship in terms of the covariance up to the previous time, step, plus a new contribution

$$\mathbf{P}_k^{-1} = \mathbf{P}_{k-1}^{-1} + (\mathbf{A}^k)^T(\mathbf{R}^k)^{-1}\mathbf{A}^k \,. \tag{6.108}$$

Similarly,

$$\mathbf{A}_k^T\mathbf{R}_k^{-1}\boldsymbol{y}_k = \mathbf{A}_{k-1}^T\mathbf{R}_{k-1}^{-1}\boldsymbol{y}_{k-1} + (\mathbf{A}^k)^T(\mathbf{R}^k)^{-1}\boldsymbol{y}^k \,. \tag{6.109}$$

Substituting the definition of $\mathbf{P}_k$ and (6.109) into (6.107) eventually yields

$$\hat{\boldsymbol{\phi}}_k = \hat{\boldsymbol{\phi}}_{k-1} + \mathbf{P}_k(\mathbf{A}^k)^T(\mathbf{R}^k)^{-1}(\boldsymbol{y}^k - \mathbf{A}^k\,\hat{\boldsymbol{\phi}}_{k-1}) \,. \tag{6.110}$$

In the covariance updating matrix (6.108), the inverse $\mathbf{P}_{k-1}^{-1}$ has to be taken. The updating relationship can be reworked into a computationally friendlier form using the binomial inverse theorem (see Wikipedia entry)

$$(\mathbf{A} + \mathbf{UBV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{B}^{-1} + \mathbf{VA}^{-1}\mathbf{U})^{-1}\mathbf{VA}^{-1} \tag{6.111}$$

Making the identification $\mathbf{A} = \mathbf{P}_{k-1}^{-1}$, $\mathbf{U} = (\mathbf{A}^k)^T$, $\mathbf{B} = (\mathbf{R}^k)^{-1}$, and $\mathbf{V} = \mathbf{A}^k$, after inversion (6.108) becomes

$$\mathbf{P}_k = \mathbf{P}_{k-1} - \mathbf{P}_{k-1}(\mathbf{A}^k)^T(\mathbf{R}^k + \mathbf{A}^k\mathbf{P}_{k-1}(\mathbf{A}^k)^T)^{-1}\mathbf{A}^k\,\mathbf{P}_{k-1} \tag{6.112}$$

The resulting matrix in parentheses is of size $M \times M$, and is typically of much lower dimension than the $N \times N$ dimension of $\mathbf{P}_k$. That is to say, there are usually many more parameters than measurement components. Hence the inversion is easier.

Equations (6.110) and (6.112) comprise a large part of the Kalman filter. Thus the Kalman filter is fundamentally recursive least squares [6.3, 61, 62]. There are many variants to the Kalman filter, including one based on the mmse estimate.

Gautier and *Poignet* [6.65] applied an extended Kalman filter to estimate a robot's inertial parameters, in which the filter was reworked to include velocity and acceleration estimation. They found that there was no advantage to using a Kalman filter over least squares, and in fact there was a disadvantage in terms of sensitivity to initial conditions and slower

convergence. One issue in on-line versus off-line estimation is the accuracy of velocity and acceleration estimates. In off-line estimation, two sided (or acausal) filters can be used, and they typically estimate time derivatives more accurately than one-side (or causal) filters.

6.5 Conclusions and Further Reading

This chapter has presented methods for calibrating the kinematic parameters and the inertial parameters of robot manipulators. Both methods are instances of parameter estimation using least squares. Inertial parameters appear linearly in the equations of motion, and ordinary least squares can be used. Kinematic parameters appear nonlinearly because of sines and cosines, and so nonlinear estimation via the Gauss–Newton method can be applied.

There are domain-specific issues for setting up the calibration equations. For kinematic calibration, Hayati parameters have to be mixed with Denavit–Hartenberg parameters to handle the case of nearly parallel joint axes. The calibration equations have to take into account how the endpoint is measured or constrained. By examining in a detailed fashion the possible joint sequences, including parallel or prismatic joints, a minimal parameterization is achieved that obviates problems of identifiability.

The calibration index has been presented as categorizing all kinematic calibration methods. This index counts the number of equations generated per pose, by calculating the excess of sensing over mobility. A key is that all calibration methods can be viewed as closed-loop methods, where any endpoint sensing system is considered as a joint. Parallel robots are handled by incorporating multiple closed loops.

For inertial parameter estimation,the recursive Newton–Euler equations lead to a formulation of the regressor matrix as an upper-diagonal matrix. A minimal parameterization is straightforward for serial and spanning robots. The load inertial parameters estimation has been treated by using either the joint torques or the terminal link force-torque sensor measurements. The proposed models are exploited to identify the joint torques gains supposing known load parameters.

Numerical methods were presented to handle unidentifiable parameters if a minimal parameterization has not been achieved. These methods hinge on the singular value or QR decomposition of the regressor matrix. The singular values can be examined to decide which parameters are not identifiable and should be eliminated. Alternatively, a small singular value can simply be zeroed to eliminate the effects of poorly identifiable parameters without explicitly eliminating them. The former yields a structural model, while the latter yields a prediction model. A priori estimates of parameters can be taken into account by using damped least squares.

The adequacy of the measurement set for parameter estimation is captured by different proposals for an observability index. This index can be related to the literature on alphabet optimalities in experimental design.

Finally, the scaling of measurements or of parameters is important for a well-conditioned numerical estimation, and is essential in order to compare singular values. When uncertainties in measurements and in parameters are included as weights, the optimal minimum variance estimate can be found, which is related to the Kalman filter. If not known beforehand, these uncertainties can be estimated from the data.

6.5.1 Relation to Other Chapters

Estimation involving least squares and Kalman filtering is discussed in Chap. 5 in a similar context. The problem of estimating properties of the world through sensors is very similar to model identification. Recursive estimation techniques are particularly appropriate when a robot needs to update its world model incrementally. For model identification, employing a recursive formulation is not particularly helpful, as the machinery of recursive updating can obscure numerical issues with the total data.

The singular value decomposition appears in other contexts. Chapter 10 and 16 analyze equal motion capability in different directions by measures similar to the observability indexes: O_1 has a counterpart in the manipulability measure, and O_2 the condition number and O_3 the minimum singular value appear again. By contrast, the concern in calibration is good data in all directions as captured by the singular values. Chap. 10 employs the singular value decomposition to analyze redundant mechanisms. Whereas parameter estimation generally is an overconstrained least-squares problem (many more measurements than parameters), redundant mechanisms are underconstrained (more joint angles than task variables). Instead of signalling identifiability problems, zero singular values indicate the null space of the Jacobian. Damped least squares is used in Chap. 10 to avoid singularities. Just as the true parameters are perturbed by damped least squares in calibration, the

trajectory is perturbed in order to get around a numerical conditioning issue.

Sensors involved in positioning a robot can have aspects of their sensor models calibrated as well, such as the gain of a potentiometer. Camera calibration is discussed in Chap. 32. Camera models can be determined at the same time as the kinematic models of robot manipulators [6.39, 66], including intrinsic camera parameters such as the pinhole model discussed in Chap. 5 as well as the extrinsic parameters of where the camera is located.

6.5.2 Further Reading

Screw Axis Measurement

An alternative to nonlinear least squares to estimate kinematic parameters is a class of methods which measure the joint axes as lines in space, termed *screw axis measurement* in [6.8]. One approach is *circle point analysis*, which involves moving one joint at a time to generate a circle at a distal measurement point [6.13]. Other approaches measure the Jacobian matrix, which contains as columns the joint screws [6.67]. With knowledge of the joint axes, the kinematic parameters are straightforwardly extracted without nonlinear search. The accuracy of this class of methods may not be as good as the nonlinear least-squares methods.

Total Least Squares

Ordinary least squares assumes that there is only noise in the output measurements, but often there is noise in the inputs as well. Input noise is known to lead to bias errors [6.3]. A framework for handling both input and output noise is total least squares [6.34], also known as orthogonal distance regression [6.68] or errors-in-variables regression [6.69]. Nonlinear total least squares has been applied to robot calibration in [6.2, 70, 71]. In the implicit loop method of [6.2], by treating endpoint measurements equally with joint measurements, no distinction is made between input and output noise.

Methods with Direct and Inverse Dynamic Models

The identification of dynamic parameters presented in this chapter is based on the Inverse Dynamic Identification Model and linear LS techniques. A common procedure to validate the identification result is to simulate the direct dynamic model with the identified parameters and compare the output with the real system.

Recently two methods which use both the direct and the inverse dynamic models have been developed to improve the noise immunity of estimates with respect to low rate or corrupted data in the observation matrix resulting from noisy measurements and/or bad tuning of joint positions band-pass filtering. The first method needs only the joint torques, while the velocity and acceleration are obtained from the direct dynamic model. It carries out in a combined procedure the identification and the validation. Current work aims to use this technique to identify robots with flexible joints where the flexible position variables are not measurable. The second method uses an instrumental variable (IV) approach. Both methods have been validated on a 6-DOF industrial robot [6.72, 73].

Video-References

VIDEO 422 Calibration of ABB's IRB 120 industrial robot
available from http://handbookofrobotics.org/view-chapter/06/videodetails/422

VIDEO 425 Robot calibration using a touch probe
available from http://handbookofrobotics.org/view-chapter/06/videodetails/425

VIDEO 430 Calibration and accuracy validation of a FANUC LR Mate 200iC industrial robot
available from http://handbookofrobotics.org/view-chapter/06/videodetails/430

VIDEO 480 Dynamic identification of Staubli TX40: Trajectory without load
available from http://handbookofrobotics.org/view-chapter/06/videodetails/480

VIDEO 481 Dynamic identification of Staubli TX40: Trajectory with load
available from http://handbookofrobotics.org/view-chapter/06/videodetails/481

VIDEO 482 Dynamic identification of Kuka LWR: Trajectory without load
available from http://handbookofrobotics.org/view-chapter/06/videodetails/482

VIDEO 483 Dynamic identification of Kuka LWR: Trajectory with load
available from http://handbookofrobotics.org/view-chapter/06/videodetails/483

VIDEO 485 Dynamic identification of a parallel robot: Trajectory with load
available from http://handbookofrobotics.org/view-chapter/06/videodetails/485

VIDEO 486 Dynamic identification of Kuka KR270: Trajectory without load
available from http://handbookofrobotics.org/view-chapter/06/videodetails/486

VIDEO 487 Dynamic identification of Kuka KR270: trajectory with load
available from http://handbookofrobotics.org/view-chapter/06/videodetails/487
VIDEO 488 Dynamic identification of a parallel robot: Trajectory without load
available from http://handbookofrobotics.org/view-chapter/06/videodetails/488

References

6.1 P.Z. Marmarelis, V.Z. Marmarelis: *Analysis of Physiological Systems* (Plenum, London 1978)
6.2 C.W. Wampler, J.M. Hollerbach, T. Arai: An implicit loop method for kinematic calibration and its application to closed-chain mechanisms, IEEE Trans. Robotics Autom. **11**, 710–724 (1995)
6.3 J.P. Norton: *An Introduction to Identification* (Academic, London 1986)
6.4 G.H. Golub, C.F. Van Loan: *Matrix Computations* (Johns Hopkins Univ. Press, Baltimore 1989)
6.5 H. West, E. Papadopoulos, S. Dubowsky, H. Cheah: A method for estimating the mass properties of a manipulator by measuring the reaction moments at its base, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 1510–1516
6.6 R.P. Paul: *Robot Manipulators: Mathematics, Programming, and Control* (MIT Press, Cambridge 1981)
6.7 S.A. Hayati, M. Mirmirani: Improving the absolute positioning accuracy of robot manipulators, J. Robotics Syst. **2**, 397–413 (1985)
6.8 J.M. Hollerbach, C.W. Wampler: The calibration index and taxonomy of kinematic calibration methods, Int. J. Robotics Res. **15**, 573–591 (1996)
6.9 A. Goswami, A. Quaid, M. Peshkin: Identifying robot parameters using partial pose information, IEEE Control Syst. **13**, 6–14 (1993)
6.10 M.R. Driels, W.E. Swayze: Automated partial pose measurement system for manipulator calibration experiments, IEEE Trans. Robotics Autom. **10**, 430–440 (1994)
6.11 G.-R. Tang, L.-S. Liu: Robot calibration using a single laser displacement meter, Mechatronics **3**, 503–516 (1993)
6.12 D.E. Whitney, C.A. Lozinski, J.M. Rourke: Industrial robot forward calibration method and results, ASME J. Dyn. Syst. Meas. Control **108**, 1–8 (1986)
6.13 B.W. Mooring, Z.S. Roth, M.R. Driels: *Fundamentals of Manipulator Calibration* (Wiley, New York 1991)
6.14 K. Lau, R. Hocken, L. Haynes: Robot performance measurements using automatic laser tracking techniques, Robotics Comput. Manuf. **2**, 227–236 (1985)
6.15 C.H. An, C.H. Atkeson, J.M. Hollerbach: *Model-Based Control of a Robot Manipulator* (MIT Press, Cambridge 1988)
6.16 M. Vincze, J.P. Prenninger, H. Gander: A laser tracking system to measure position and orientation of robot end effectors under motion, Int. J. Robotics Res. **13**, 305–314 (1994)
6.17 J.M. McCarthy: *Introduction to Theoretical Kinematics* (MIT Press, Cambridge 1990)
6.18 D.J. Bennet, J.M. Hollerbach: Autonomous calibration of single-loop closed kinematic chains formed by manipulators with passive endpoint constraints, IEEE Trans. Robotics Autom. **7**, 597–606 (1991)
6.19 W.S. Newman, D.W. Osborn: A new method for kinematic parameter calibration via laser line tracking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 160–165
6.20 X.-L. Zhong, J.M. Lewis: A new method for autonomous robot calibration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 1790–1795
6.21 J.M. Hollerbach, D.M. Lokhorst: Closed-loop kinematic calibration of the RSI 6-DOF hand controller, IEEE Trans. Robotics Autom. **11**, 352–359 (1995)
6.22 A. Nahvi, J.M. Hollerbach, V. Hayward: Closed-loop kinematic calibration of a parallel-drive shoulder joint, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 407–412
6.23 O. Masory, J. Wang, H. Zhuang: On the accuracy of a Stewart platform – Part II Kinematic calibration and compensation, Proc. IEEE Int. Conf. Robotics Autom., Piscataway (1994) pp. 725–731
6.24 B. Armstrong-Helouvry: *Control of Machines with Friction* (Kluwer, Boston 1991)
6.25 B. Armstrong-Helouvry, P. Dupont, C. de Canudas Wit: A survey of models, analysis tools and compensation methods for the control of machines with friction, Automatica **30**, 1083–1138 (1994)
6.26 F. Aghili, J.M. Hollerbach, M. Buehler: A modular and high-precision motion control system with an integrated motor, IEEE/ASME Trans. Mechatron. **12**, 317–329 (2007)
6.27 W.S. Newman, J.J. Patel: Experiments in torque control of the Adept One robot, Proc. IEEE Int. Conf. Robotics Autom., Piscataway (1991) pp. 1867–1872
6.28 W. Khalil, E. Dombre: *Modeling, Identification and Control of Robots* (Taylor Francis, New York 2002)
6.29 M. Gautier: Dynamic identification of robots with power model, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 1922–1927
6.30 W. Khalil, M. Gautier, P. Lemoine: Identification of the payload inertial parameters of industrial manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 4943–4948
6.31 J. Swevers, W. Verdonck, B. Naumer, S. Pieters, E. Biber: An Experimental Robot Load Identification Method for Industrial Application, Int. J. Robotics Res. **21(8)**, 701–712 (2002)
6.32 P.P. Restrepo, M. Gautier: Calibration of drive chain of robot joints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 526–531
6.33 P. Corke: In situ measurement of robot motor electrical constants, Robotica **23**(14), 433–436 (1996)
6.34 S. Van Huffel, J. Vandewalle: *The Total Least Squares Problem: Computational Aspects and Analysis* (SIAM, Philadelphia 1991)
6.35 M. Gautier, S. Briot: Global identification of drive gains parameters of robots using a known pay-

load, ASME J. Dyn. Syst. Meas. Control **136**(5), 051026 (2014)
6.36 W. Khalil, O. Ibrahim: General solution for the dynamic modeling of parallel robots, J. Intell. Robotics Syst. **49**, 19–37 (2007)
6.37 S. Guegan, W. Khalil, P. Lemoine: Identification of the dynamic parameters of the Orthoglide, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3272–3277
6.38 K. Schröer: Theory of kinematic modelling and numerical procedures for robot calibration. In: *Robot Calibration*, ed. by R. Bernhardt, S.L. Albright (Chapman Hall, London 1993) pp. 157–196
6.39 H. Zhuang, Z.S. Roth: *Camera-Aided Robot Calibration* (CRC, Boca Raton 1996)
6.40 M. Gautier: Numerical calculation of the base inertial parameters, J. Robotics Syst. **8**, 485–506 (1991)
6.41 J.J. Dongarra, C.B. Mohler, J.R. Bunch, G.W. Stewart: *LINPACK User's Guide* (SIAM, Philadelphia 1979)
6.42 W. Khalil, F. Bennis: Symbolic calculation of the base inertial parameters of closed-loop robots, Int. J. Robotics Res. **14**, 112–128 (1995)
6.43 W. Khalil, S. Guegan: Inverse and direct dynamic modeling of Gough–Stewart robots, IEEE Trans. Robotics Autom. **20**, 754–762 (2004)
6.44 W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery: *Numerical Recipes in C* (Cambridge Univ. Press, Cambridge 1992)
6.45 C.L. Lawson, R.J. Hanson: *Solving Least Squares Problems* (Prentice Hall, Englewood Cliffs 1974)
6.46 P.R. Bevington, D.K. Robinson: *Data Reduction and Error Analysis for the Physical Sciences* (McGraw-Hill, New York 1992)
6.47 B. Armstrong: On finding exciting trajectories for identification experiments involving systems with nonlinear dynamics, Int. J. Robotics Res. **8**, 28–48 (1989)
6.48 J. Fiefer, J. Wolfowitz: Optimum designs in regression problems, Ann. Math. Stat. **30**, 271–294 (1959)
6.49 Y. Sun, J.M. Hollerbach: Observability index selection for robot calibration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Piscataway (2008) pp. 831–836
6.50 J.H. Borm, C.H. Menq: Determination of optimal measurement configurations for robot calibration based on observability measure, Int. J. Robotics Res. **10**, 51–63 (1991)
6.51 C.H. Menq, J.H. Borm, J.Z. Lai: Identification and observability measure of a basis set of error parameters in robot calibration, ASME J. Mech. Autom. Des. **111**(4), 513–518 (1989)
6.52 M. Gautier, W. Khalil: Exciting trajectories for inertial parameter identification, Int. J. Robotics Res. **11**, 362–375 (1992)
6.53 M.R. Driels, U.S. Pathre: Significance of observation strategy on the design of robot, J. Robotics Syst. **7**, 197–223 (1990)
6.54 A. Nahvi, J.M. Hollerbach: The noise amplification index for optimal pose selection in robot calibration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 647–654
6.55 D. Daney, B. Madeline, Y. Papegay: Choosing measurement poses for robot calibration with local convergence method and Tabu search, Int. J. Robotics Res. **24**(6), 501–518 (2005)
6.56 Y. Sun, J.M. Hollerbach: Active robot calibration algorithm, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Piscataway (2008) pp. 1276–1281
6.57 T.J. Mitchell: An algorithm for the construction of D-Optimal experimental designs, Technometrics **16**(2), 203–210 (1974)
6.58 J. Swevers, C. Ganseman, D.B. Tukel, J. De Schutter, H. Van Brussel: Optimal robot excitation and identification, IEEE Trans. Robotics Autom. **13**, 730–740 (1997)
6.59 P.O. Vandanjon, M. Gautier, P. Desbats: Identification of robots inertial parameters by means of spectrum analysis, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 3033–3038
6.60 E. Walter, L. Pronzato: *Identification of Parametric Models from Experimental Data* (Springer, London 1997)
6.61 D.G. Luenberger: *Optimization by Vector Space Methods* (Wiley, New York 1969)
6.62 H.W. Sorenson: Least-squares estimation: from Gauss to Kalman, IEEE Spectr. **7**, 63–68 (1970)
6.63 Z. Roth, B.W. Mooring, B. Ravani: An overview of robot calibration, IEEE J. Robotics Autom. **3**, 377–386 (1987)
6.64 A.E. Bryson Jr., Y.-C. Ho: *Applied Optimal Control* (Hemisphere, Washington 1975)
6.65 M. Gautier, P. Poignet: Extended Kalman fitering and weighted least squares dynamic identification of robot, Control Eng. Pract. **9**, 1361–1372 (2001)
6.66 D.J. Bennet, J.M. Hollerbach, D. Geiger: Autonomous robot calibration for hand-eye coordination, Int. J. Robotics Res. **10**, 550–559 (1991)
6.67 D.J. Bennet, J.M. Hollerbach, P.D. Henri: Kinematic calibration by direct estimation of the Jacobian matrix, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1992) pp. 351–357
6.68 P.T. Boggs, R.H. Byrd, R.B. Schnabel: A stable and efficient algorithm for nonlinear orthogonal distance regression, SIAM J. Sci. Stat. Comput. **8**, 1052–1078 (1987)
6.69 W.A. Fuller: *Measurement Error Models* (Wiley, New York 1987)
6.70 J.-M. Renders, E. Rossignol, M. Becquet, R. Hanus: Kinematic calibration and geometrical parameter identification for robots, IEEE Trans. Robotics Autom. **7**, 721–732 (1991)
6.71 G. Zak, B. Benhabib, R.G. Fenton, I. Saban: Application of the weighted least squares parameter estimation method for robot calibration, J. Mech. Des. **116**, 890–893 (1994)
6.72 M. Gautier, A. Janot, P.O. Vandanjon: A New Closed-Loop Output Error Method for Parameter Identification of Robot Dynamics, IEEE Trans. Control Syst. Techn. **21**, 428–444 (2013)
6.73 A. Janot, P.O. Vandanjon, M. Gautier: A Generic Instrumental Variable Approach for Industrial Robots Identification, IEEE Trans. Control Syst. Techn. **22**, 132–145 (2014)

Multimedia Contents

7. Motion Planning

Lydia E. Kavraki, Steven M. LaValle

This chapter first provides a formulation of the geometric path planning problem in Sect. 7.2 and then introduces sampling-based planning in Sect. 7.3. Sampling-based planners are general techniques applicable to a wide set of problems and have been successful in dealing with hard planning instances. For specific, often simpler, planning instances, alternative approaches exist and are presented in Sect. 7.4. These approaches provide theoretical guarantees and for simple planning instances they outperform sampling-based planners. Section 7.5 considers problems that involve differential constraints, while Sect. 7.6 overviews several other extensions of the basic problem formulation and proposed solutions. Finally, Sect. 7.8 addresses some important and more advanced topics related to motion planning.

7.1 **Robotics Motion Planning** 139
7.2 **Motion Planning Concepts** 140
7.2.1 Configuration Space 140
7.2.2 Geometric Path Planning Problem . 141
7.2.3 Complexity of Motion Planning 141
7.3 **Sampling-Based Planning** 141
7.3.1 Multi-Query Planners: Mapping the Connectivity of C_{free} ... 142
7.3.2 Single-Query Planners: Incremental Search 143
7.4 **Alternative Approaches** 144
7.4.1 Combinatorial Roadmaps 145
7.4.2 Roadmaps in Higher Dimensions ... 146
7.4.3 Potential Fields 146
7.5 **Differential Constraints** 148
7.5.1 Concepts and Terminology 148
7.5.2 Discretization of Constraints 149
7.5.3 Decoupled Approach 149
7.5.4 Kinodynamic Planning 150
7.6 **Extensions and Variations** 151
7.6.1 Closed Kinematic Chains 151
7.6.2 Manipulation Planning 151
7.6.3 Time-Varying Problems 151
7.6.4 Multiple Robots 152
7.6.5 Uncertainty in Predictability 153
7.6.6 Sensing Uncertainty 153
7.6.7 Optimal Planning 154
7.7 **Advanced Issues** 154
7.7.1 Topology of Configuration Spaces .. 154
7.7.2 Sampling Theory 155
7.7.3 Computational Algebraic Geometry Techniques 156
7.8 **Conclusions and Further Reading** 157
Video-References 158
References 158

7.1 Robotics Motion Planning

A fundamental robotics task is to plan collision-free motions for complex bodies from a start to a goal position among a collection of static obstacles. Although relatively simple, this geometric path planning problem is computationally hard [7.1]. Extensions of this formulation take into account additional problems that are inherited from mechanical and sensor limitations of real robots such as uncertainties, feedback, and differential constraints, which further complicate the development of automated planners. Modern algorithms have been fairly successful in addressing hard instances of the basic geometric problem and

a lot of effort has been devoted to extend their capabilities to more challenging instances. These algorithms have had widespread success in applications beyond robotics, such as computer animation, virtual prototyping, and computational biology. There are many available surveys [7.2–4] and books [7.5–7] that cover modern motion planning techniques and applications.

7.2 Motion Planning Concepts

This section provides a description of the fundamental motion planning problem or the geometric path planning problem. Extensions of this basic formulation to more complicated instances will be discussed later in the chapter and will be revisited throughout this book.

7.2.1 Configuration Space

In path planning, a complete description of the geometry of a robot $\mathcal{A}$ and of a workspace $\mathcal{W}$ is provided. The workspace $\mathcal{W} = \mathbb{R}^N$, in which $N = 2$ or $N = 3$, is a static environment populated with obstacles. The goal is to find a collision-free path for $\mathcal{A}$ to move from an initial position and orientation to a goal position and orientation.

To achieve this, a complete specification of the location of every point on the robot geometry, or a *configuration* $\boldsymbol{q}$, must be provided. The *configuration space*, or C-space ($\boldsymbol{q} \in C$), is the space of all possible configurations. The C-space represents the set of all transformations that can be applied to a robot given its kinematics as described in Chap. 2 (Kinematics). It was recognized early on in motion planning research [7.8, 9] that the C-space is a useful way to abstract planning problems in a unified way. The advantage of this abstraction is that a robot with a complex geometric shape is mapped to a single point in the C-space. The number of degrees of freedom of a robot system is the dimension of the C-space, or the minimum number of parameters needed to specify a configuration.

Let the closed set $\mathcal{O} \subset \mathcal{W}$ represent the *(workspace) obstacle region*, which is usually expressed as a collection of polyhedra, three-dimensional (3-D) triangles, or piecewise-algebraic surfaces. Let the closed set $\mathcal{A}(\boldsymbol{q}) \subset \mathcal{W}$ denote the set of points occupied by the robot when at configuration $\boldsymbol{q} \in C$; this set is usually modeled using the same primitives as used for $\mathcal{O}$. The *C-space obstacle region*, C_{obs}, is defined as

$$C_{\text{obs}} = \{\boldsymbol{q} \in C \mid \mathcal{A}(\boldsymbol{q}) \cap \mathcal{O} \neq \emptyset\}. \tag{7.1}$$

Since $\mathcal{O}$ and $\mathcal{A}(\boldsymbol{q})$ are closed sets in $\mathcal{W}$, the obstacle region is a closed set in C. The set of configurations that avoid collision is $C_{\text{free}} = C \setminus C_{\text{obs}}$, and is called the *free space*.

Simple Examples of C-spaces

Translating Planar Rigid Bodies. The robot's configuration can be specified by a reference point (x, y) on the planar rigid body relative to some fixed coordinate frame. Therefore the C-space is equivalent to $\mathbb{R}^2$. Figure 7.1 gives an example of a C-space for a triangular robot and a single polygonal obstacle. The obstacle region in the C-space can be traced by sliding the robot around the workspace obstacle to find the constraints on all $\boldsymbol{q} \in C$. Motion planning for the robot is now equivalent to motion planning for a point in the C-space.

Planar Arms. Figure 7.2 gives an example of a two-joint planar arm. The bases of both links are pinned, so that they can only rotate around the joints, and there are no joint limits. For this arm, specifying the rotational parameters θ_1 and θ_2 provides the configuration. Each joint angle θ_i corresponds to a point on the unit circle $\mathbb{S}^1$ and the C-space is $\mathbb{S}^1 \times \mathbb{S}^1 = T^2$, the two-dimensional (2-D) torus shown in Fig. 7.2. For a higher number of

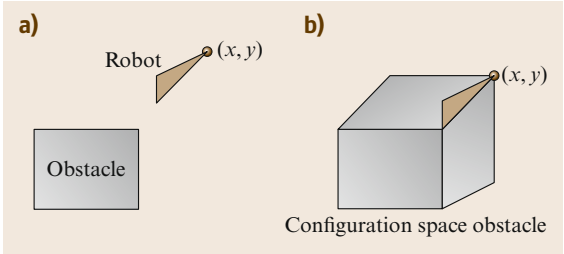


Fig.7.1a,b A robot translating in the plane: **(a)** a triangular robot moves in a workspace with a single rectangular obstacle. **(b)** The C-space obstacle

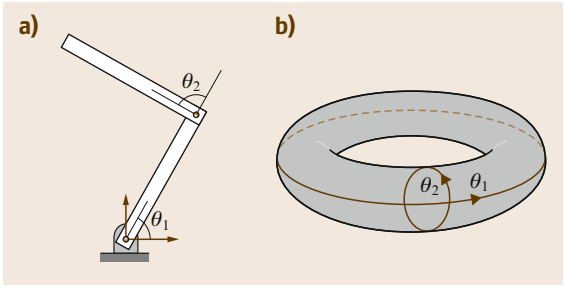


Fig. 7.2 **(a)** A two-joint planar arm in which the links are pinned and there are no joint limits. **(b)** The C-space

links without joint limits, the C-space can be similarly defined as

$$\mathcal{C} = \mathbb{S}^1 \times \mathbb{S}^1 \times \cdots \times \mathbb{S}^1 \,. \tag{7.2}$$

If a joint has limits, then each corresponding $\mathbb{S}^1$ is often replaced with $\mathbb{R}$, even though it is a finite interval. If the base of the planar arm is mobile and not pinned, then the additional translation parameters must also be considered in the arm's configuration

$$\mathcal{C} = \mathbb{R}^2 \times \mathbb{S}^1 \times \mathbb{S}^1 \times \cdots \times \mathbb{S}^1 \,. \tag{7.3}$$

Additional examples of C-spaces are provided in Sect. 7.7.1, where topological properties of configuration spaces are discussed.

7.2.2 Geometric Path Planning Problem

The basic motion planning problem, also known as the *piano mover's problem* [7.1], is defined as follows.
Given:

1. A *workspace* $\mathcal{W}$, where either $\mathcal{W} = \mathbb{R}^2$ or $\mathcal{W} = \mathbb{R}^3$.
2. An *obstacle region* $\mathcal{O} \subset \mathcal{W}$.
3. A *robot* defined in $\mathcal{W}$. Either a rigid body $\mathcal{A}$ or a collection of m links: $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m$.
4. The *configuration space* $\mathcal{C}$ ($\mathcal{C}_{\text{obs}}$ and $\mathcal{C}_{\text{free}}$ are then defined).
5. An *initial configuration* $\boldsymbol{q}_{\text{I}} \in \mathcal{C}_{\text{free}}$.
6. A *goal configuration* $\boldsymbol{q}_{\text{G}} \in \mathcal{C}_{\text{free}}$. The initial and goal configuration are often called a *query* $(\boldsymbol{q}_{\text{I}}, \boldsymbol{q}_{\text{G}})$.

Compute a (continuous) path, $\tau : [0, 1] \to \mathcal{C}_{\text{free}}$, *such that* $\tau(0) = \boldsymbol{q}_{\text{I}}$ *and* $\tau(1) = \boldsymbol{q}_{\text{G}}$.

7.2.3 Complexity of Motion Planning

The main complications in motion planning are that it is not easy to directly compute $\mathcal{C}_{\text{obs}}$ and $\mathcal{C}_{\text{free}}$, and the dimensionality of the C-space is often quite high. In terms of computational complexity, the piano mover's problem was studied early on and it was shown to be PSPACE-hard by *Reif* [7.1]. A series of polynomial-time algorithms for problems with fixed dimension suggested an exponential dependence on the problem dimensionality [7.10, 11]. A single exponential-time algorithm in the C-space dimensionality was proposed by *Canny* and showed that the problem is PSPACE-complete [7.12]. Although impractical, the algorithm serves as an upper bound on the general version of the basic motion planning problem. It applies computational algebraic geometry techniques for modeling the C-space in order to construct a *roadmap*, a one-dimensional (1-D) subspace that captures the connectivity of $\mathcal{C}_{\text{free}}$. Additional details about such techniques can be found in Sect. 7.7.3.

The complexity of the problem motivated work in path planning research. One direction was to study subclasses of the general problem for which polynomial time algorithms exist [7.13]. Even some simpler, special cases of motion planning, however, are at least as challenging, for example, the case of a finite number of translating, axis-aligned rectangles in $\mathbb{R}^2$ is PSPACE-hard as well [7.14]. Some extensions of motion planning are even harder. For example, a certain form of planning under uncertainty in 3-D polyhedral environment is NEXPTIME-hard [7.15]. The hardest problems in NEXPTIME are believed to require doubly exponential time to solve.

A different direction was the development of alternative planning paradigms that were practical under realistic assumptions. Many combinatorial approaches can efficiently construct 1-D roadmaps for specific 2-D or 3-D problems. Potential field-based approaches define vector fields which can be followed by a robot towards the goal. Both approaches, however, do not scale well in the general case. They will be described in Sect. 7.4. An alternative paradigm, sampling-based planning, is a general approach that has been shown to be successful in practice for many challenging problems. It avoids the exact geometric modeling of the C-space but it cannot provide the guarantees of a complete algorithm. Complete and exact algorithms are able to detect that no path can be found. Instead sampling-based planning offers a lower level of completeness guarantee. This paradigm is described in the following section.

7.3 Sampling-Based Planning

Sampling-based planners are described first because they are the method of choice for a very general class of problems. The following section will describe other planners, some of which were developed before the sampling-based framework. The key idea in sampling-based planning is to exploit advances in collision detection algorithms that compute whether a single configuration is collision free. Given this simple primitive, a planner samples different configurations to construct a data structure that stores 1-D C-space curves, which represent collision-free paths. In this way, sampling-based planners do not access the C-space obstacles directly but only through the collision detector and the constructed data structure. Using this level of abstrac-

tion, the planners are applicable to a wide range of problems by tailoring the collision detector to specific robots and applications.

A standard for sampling-based planners is to provide a weaker, but still interesting, form of completeness: *if a solution path exists, the planner will eventually find it*. Giving up on the stronger form of completeness, which also requires failure to be reported in finite time, these techniques are able to solve in practice problems with more than three degrees of freedom that complete approaches cannot address. More details on this weaker form of completeness are provided in Sect. 7.7.2.

Different planners follow different approaches on how to sample configurations and what kind of data structures they construct. Section 7.7.2 provides a deeper insight on sampling issues. A typical classification of sampling-based planners is between multi-query and single-query approaches:

- In the first category, the planners construct a roadmap, an undirected graph G that is precomputed once so as to *map the connectivity properties of* C_{free}. After this step, multiple queries in the same environment can be answered using only the constructed roadmap. Such planners are described in Sect. 7.3.1.
- Planners in the second category build tree data structures on the fly given a planning query. They attempt to focus on exploring the part of the C-space that will lead to solving a specific query as fast as possible. They are described in Sect. 7.3.2.

Both approaches, however, make similar use of a collision checking primitive. The objective of a collision detector is to report all geometric contacts between objects given their geometries and transformations [7.16–18]. The availability of packages that were able to answer collision queries in a fraction of a second was critical to the development of sampling-based planners. Modern planners use collision detectors as a *black box*. Initially the planner provides the geometries of all the involved objects and specifies which of them are mobile. Then, in order to validate a robot configuration, a planner provides the corresponding robot transformation and a collision detector responds on whether the objects collide or not. Many packages represent the geometric models hierarchically, avoid computing all-pairwise interactions, and conduct a binary search to evaluate collisions. Except from configurations, a planner must also validate entire paths. Some collision detectors return distance-from-collision information, which can be used to infer that entire neighborhoods in C are valid. It is often more expensive, however, to extract this information; instead paths are usually validated point by point using a small stepping size either incrementally or by employing binary search. Some collision detectors are incremental by design, which means that they can be faster by reusing information from a previous query [7.16]. Examples of problems solved by sampling-based planners are shown in VIDEO 24 and VIDEO 17.

7.3.1 Multi-Query Planners: Mapping the Connectivity of C_{free}

Planners that aim to answer multiple queries for a certain static environment use a preprocessing phase during which they attempt to map the connectivity properties of C_{free} onto a roadmap. This roadmap has the form of a graph G, with vertices as configurations and edges as paths. A union of 1-D curves is a roadmap G if it satisfies the following properties:

1. *Accessibility*: From any $\boldsymbol{q} \in C_{\text{free}}$, it is simple and efficient to compute a path $\tau : [0,1] \to C_{\text{free}}$ such that $\tau(0) = \boldsymbol{q}$ and $\tau(1) = \boldsymbol{s}$, in which $\boldsymbol{s}$ may be any point in $S(G)$. $S(G)$ is the *swath* of G, the union of all configurations reached by all edges and vertices. This means that it is always possible to connect a planning query pair $\boldsymbol{q}_{\text{I}}$ and $\boldsymbol{q}_{\text{G}}$ to some $\boldsymbol{s}_{\text{I}}$ and $\boldsymbol{s}_{\text{G}}$, respectively, in $S(G)$.
2. *Connectivity preserving*: The second condition requires that, if there exists a path $\tau : [0,1] \to C_{\text{free}}$ such that $\tau(0) = \boldsymbol{q}_{\text{I}}$ and $\tau(1) = \boldsymbol{q}_{\text{G}}$, then there also exists a path $\tau' : [0,1] \to S(G)$, such that $\tau'(0) = \boldsymbol{s}_{\text{I}}$ and $\tau'(1) = \boldsymbol{s}_{\text{G}}$. Thus, solutions are not missed because G fails to capture the connectivity of C_{free}.

The probabilistic roadmap method (PRM) approach [7.19] attempts to approximate such a roadmap G in a computationally efficient way. The preprocessing phase of PRM, which can be extended to sampling-based roadmaps in general, follows these steps:

1. *Initialization*: Let $G(V, E)$ represent an undirected *graph*, which is initially empty. Vertices of G will correspond to collision-free configurations, and edges to collision-free paths that connect vertices.
2. *Configuration sampling*: A configuration $\alpha(i)$ is sampled from C_{free} and added to the vertex set V. $\alpha(\cdot)$ is an infinite, dense sample sequence and $\alpha(i)$ is the i-th point in that sequence.
3. *Neighborhood computation*: Usually, a *metric* is defined in the C-space, $\rho : C \times C \to \mathbb{R}$. Vertices $\boldsymbol{q}$ already in V are then selected as part of $\alpha(i)$'s neighborhood if they have small distance according to ρ.
4. *Edge consideration*: For those vertices $\boldsymbol{q}$ that do not belong in the same connected component of G with

$\alpha(i)$ the algorithm attempts to connect them with an edge.

5. *Local planning method*: Given $\alpha(i)$ and $\boldsymbol{q} \in C_{\text{free}}$ a module is used that attempts to construct a path $\tau_s : [0,1] \rightarrow C_{\text{free}}$ such that $\tau(0) = \alpha(i)$ and $\tau(1) = \boldsymbol{q}$. Using collision detection, τ_s must be checked to ensure that it does not cause a collision.
6. *Edge insertion*: Insert τ_s into E, as an edge from $\alpha(i)$ to $\boldsymbol{q}$.
7. *Termination*: The algorithm is typically terminated when a predefined number of collision-free vertices N has been added in the roadmap.

The algorithm is incremental in nature. Computation can be repeated by starting from an already existing graph. A general sampling-based roadmap is summarized in Algorithm 7.1.

Algorithm 7.1 Sampling-Based Roadmap

N: number of nodes to include in the roadmap

```
G.init(); i ← 0;
while i < N do
  if α(i) ∈ C_free then
    G.add_ vertex(α(i)); i ← i + 1;
    for q ∈ NEIGHBORHOOD(α(i),G) do
      if CONNECT (α(i), q) then
        G.add_ edge (α(i), q);
      end if
    end for
  end if
end while
```

An illustration of the algorithm's behavior is depicted in Fig. 7.3. To solve a query, $\boldsymbol{q}_{\text{I}}$ and $\boldsymbol{q}_{\text{G}}$ are connected to the roadmap, and graph search is performed.

For the original PRM [7.19], the configuration $\alpha(i)$ was produced using random sampling. For the connection step between $\boldsymbol{q}$ and $\alpha(i)$, the algorithm used straight line paths in the C-space. In some cases a connection was attempted if $\boldsymbol{q}$ and $\alpha(i)$ were in the same connected component in order to improve path quality. There have been many subsequent works that try to improve the roadmap quality while using fewer samples. Methods for concentrating samples at or near the boundary of C_{free} are presented in [7.20, 21]. Methods that move samples as far from the boundary as possible appear in [7.22, 23]. Deterministic sampling techniques, including grids, appear in [7.24]. A method of pruning vertices based on mutual visibility that leads to a dramatic reduction in the number of roadmap vertices appears in [7.25]. Theoretical analysis of sampling-based roadmaps appears in [7.24, 26, 27] and is briefly discussed in Sect. 7.7.2. An experimental comparison of sampling-based roadmap variants appears in [7.28]. One difficulty in these roadmap approaches is identifying narrow passages. One proposal is to use the *bridge test* for identifying these [7.29]. For other PRM-based works, see [7.30–34]. Extended discussion of the topic can be found in [7.5, 7].

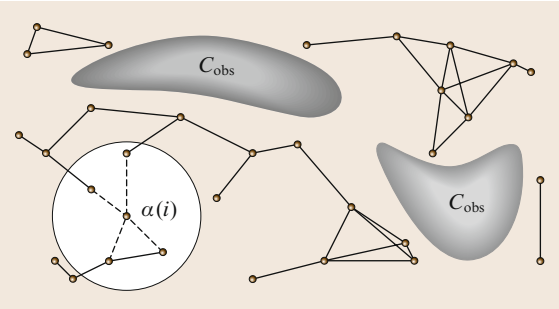


Fig. 7.3 The sampling-based roadmap is constructed incrementally by attempting to connect each new sample, $\alpha(i)$, to nearby vertices in the roadmap

7.3.2 Single-Query Planners: Incremental Search

Single-query planning methods focus on a single initial–goal configuration pair. They probe and search the continuous C-space by extending tree data structures initialized at these known configurations and eventually connecting them. Most single-query methods conform to the following template:

1. *Initialization*: Let $G(V, E)$ represent an undirected *search graph*, for which the vertex set V contains a vertex for one (usually $\boldsymbol{q}_{\text{I}}$) or more configurations in C_{free}, and the edge set E is empty. Vertices of G are collision-free configurations, and edges are collision-free paths that connect vertices.
2. *Vertex selection method*: Choose a vertex $\boldsymbol{q}_{\text{cur}} \in V$ for expansion.
3. *Local planning method*: For some $\boldsymbol{q}_{\text{new}} \in C_{\text{free}}$, which may correspond to an existing vertex in V but on a different tree or a sampled configuration, attempt to construct a path $\tau_s : [0,1] \rightarrow C_{\text{free}}$ such that $\tau(0) = \boldsymbol{q}_{\text{cur}}$ and $\tau(1) = \boldsymbol{q}_{\text{new}}$. Using collision detection, τ_s must be checked to ensure that it does not cause a collision. If this step fails to produce a collision-free path segment, then go to Step 2.
4. *Insert an edge in the graph*: Insert τ_s into E, as an edge from $\boldsymbol{q}_{\text{cur}}$ to $\boldsymbol{q}_{\text{new}}$. If $\boldsymbol{q}_{\text{new}}$ is not already in V, then it is inserted.
5. *Check for a solution*: Determine whether G encodes a solution path.

6. *Return to Step 2*: Iterate unless a solution has been found or some termination condition is satisfied, in which case the algorithm reports failure.

During execution, G may be organized into one or more trees. This leads to:

1. *Unidirectional* methods, which involve a single tree, usually rooted at $\boldsymbol{q}_{\mathrm{I}}$ [7.35],
2. *Bidirectional* methods, which involve two trees, typically rooted at $\boldsymbol{q}_{\mathrm{I}}$ and $\boldsymbol{q}_{\mathrm{G}}$ [7.35], and
3. *Multidirectional* methods, which may have more than two trees [7.36, 37].

The motivation for using more than one tree is that a single tree may become trapped trying to find an exit through a narrow opening. Traveling in the opposite direction, however, may be easier. As more trees are considered it becomes more complicated to determine which connections should be made between trees.

Rapidly Exploring Dense Trees

The important idea with this family of techniques is that the algorithm must incrementally explore the properties of the C-space. An algorithm that achieves this objective is the rapidly exploring random tree (RRT) [7.35], which can be generalized to the rapidly exploring dense tree (RDT) for any dense sampling, deterministic or random [7.7]. The basic idea is to induce a *Voronoi bias* in the exploration process by selecting for expansion the point in the tree that is closest to $\alpha(i)$ in each iteration. Using random samples, the probability that a vertex is chosen is proportional to the volume of its Voronoi region. The tree construction is outlined as:

Algorithm 7.2 Rapidly Exploring Dense Trees

k: the exploration steps of the algorithm

```
1: G.init(q_I);
2: for i = 1 to k do
3:     G.add_vertex(α(i));
4:     q_n ← NEAREST(S(G), α(i));
5:     G.add_edge(q_n, α(i));
6: end for
```

The tree starts at $\boldsymbol{q}_{\mathrm{I}}$, and in each iteration, an edge and vertex are added (Fig. 7.4).

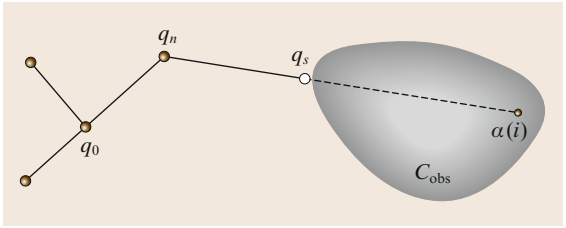


Fig. 7.4 If there is an obstacle, the edge travels up to the obstacle boundary, as far as allowed by the collision detection algorithm

So far, the problem of reaching $\boldsymbol{q}_{\mathrm{G}}$ has not been explained. There are several ways to use RDTs in a planning algorithm. One approach is to bias $\alpha(i)$ so that $\boldsymbol{q}_{\mathrm{G}}$ is frequently chosen (perhaps once every 50 iterations). A more efficient approach is to develop a bidirectional search by growing two trees, one from each of $\boldsymbol{q}_{\mathrm{I}}$ and $\boldsymbol{q}_{\mathrm{G}}$. Roughly half of the time is spent expanding each tree in the usual way, while the other half is spend attempting to connect the trees. The simplest way to connect trees is to let the newest vertex of one tree be a substitute for $\alpha(i)$ in extending the other. This tricks one RDT into attempting to connect to the other while using the basic expansion algorithm [7.38]. Several works have extended, adapted, or applied RDTs in various applications [7.37, 39–42]. Detailed descriptions can be found in [7.5, 7].

Other Tree Algorithms

Planners based on the idea of expansive spaces are presented in [7.43–45]. In this case, the algorithm forces exploration by choosing vertices for expansion that have fewer points in a neighborhood around them. In [7.46], additional performance is obtained by self-tuning random walks, which focus virtually all of their effort on exploration. Other successful tree-based algorithms include the path-directed subdivision tree algorithm [7.47] and some of its variants [7.48]. In the literature, it is sometimes hard to locate tree-based planners for ordinary path planning problems as many of them (including RRT) were designed and/or applied to more complex problems (Sect. 7.5.4). Their performance is nevertheless excellent for a variety of path planing problems.

7.4 Alternative Approaches

Alternative approaches to the sampling-based paradigm include potential-field-based techniques and combinatorial methods that also produce roadmaps, such as cell decompositions. These algorithms are able to elegantly and efficiently solve a narrow class of problems, and are much preferred over the algorithms of Sect. 7.3 in

these cases. Most of the combinatorial algorithms are of theoretical interest, whereas sampling-based algorithms are motivated primarily by performance issues in challenging applications. Nevertheless, given some abstractions, the combinatorial algorithms can be used to solve practical problems such as autonomous navigation of mobile planar robots.

7.4.1 Combinatorial Roadmaps

Several algorithms exist for the case in which $C = \mathbb{R}^2$ and C_{obs} is polygonal. Most of these cannot be directly extended to higher dimensions; however, some of the general principles remain the same. The *maximum clearance roadmap* (or *retraction method* [7.49]) constructs a roadmap that keeps paths as far from the obstacles as possible. Paths are contributed to the roadmap from the three cases shown in Fig. 7.5, which correspond to all ways to pair together polygon features. The roadmap can be made naively in time $O(n^4)$ by generating all curves shown in Fig. 7.5 for all possible pairs, computing their intersections, and tracing out the roadmap. Several algorithms exist that provide better asymptotic running time [7.50], but they are considerably more difficult to implement. The best-known algorithm runs in $O(n \lg n)$ time in which n is the number of roadmap curves [7.51].

An alternative is to compute a *shortest-path roadmap* [7.52], as shown in Fig. 7.6. This is different than the roadmap presented in the previous section because paths may actually touch the obstacles, which must be allowed for paths to be optimal. The roadmap vertices are the reflex vertices of C_{obs}, which are vertices for which the interior angle is greater than π. An edge exists in the roadmap if and only if a pair of vertices is mutually visible and the line through them pokes into C_{free} when extended outward from each vertex (such lines are called *bitangents*). An $O(n^2 \lg n)$-time construction algorithm can be formed by using a radial sweep algorithm from each reflex vertex. It can theoretically be computed in time $O(n^2 + m)$, in which m is the total number of edges in the roadmap [7.53].

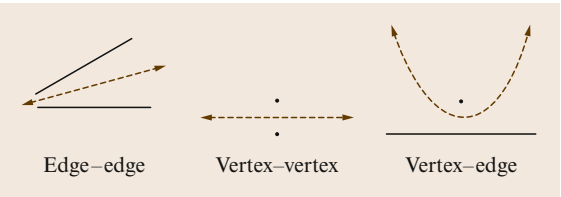


Fig. 7.5 Voronoi roadmap pieces are generated in one of three possible cases. The third case leads to a quadratic curve

Figure 7.7 illustrates the *vertical cell decomposition* approach. The idea is to decompose C_{free} into cells that are trapezoids or triangles. Planning in each cell is trivial because it is convex. A roadmap is made by placing a point in the center of each cell and each boundary between cells. Any graph search algorithm can be used to find a collision-free path quickly. The cell decomposition can be constructed in $O(n \lg n)$ time using the *plane-sweep principle* [7.54, 55]. Imagine that a vertical line sweeps from $x = -\infty$ to $x = \infty$, stopping at places where a polygon vertex is encountered. In these cases, a cell boundary may be necessary above and/or below the vertex. The order in which segments stab the vertical line is maintained in a balanced search tree. This enables the determination of the vertical cell boundary limits in time $O(\lg n)$. The whole algorithm runs in time $O(n \lg n)$ because there are $O(n)$ vertices at which the sweep line can stop (also, the vertices need to be sorted at the outset, which requires time $O(n \lg n)$).

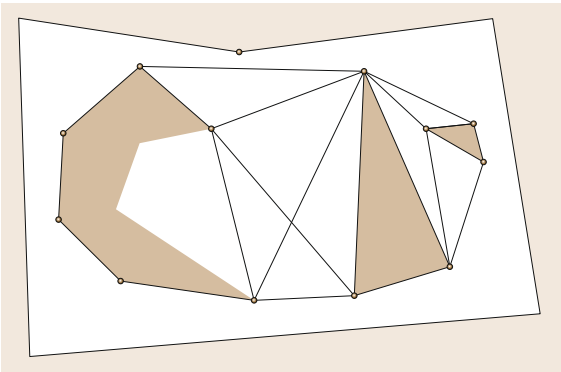


Fig. 7.6 The shortest-path roadmap includes edges between consecutive reflex vertices on C_{obs} and also bitangent edges

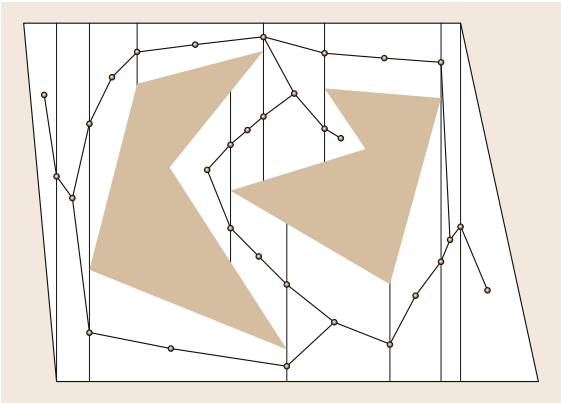


Fig. 7.7 The roadmap derived from the vertical cell decomposition

7.4.2 Roadmaps in Higher Dimensions

It would be convenient if the methods of Sect. 7.4.1 directly extend into higher dimensions. Although this unfortunately does not occur, some of the general ideas extend. To consider a cell decomposition in higher dimensions, there are two main requirements: (1) each cell should be simple enough that motion planning within a cell is trivial; (2) the cells should fit together nicely. A sufficient condition for the first requirement is that cells are convex; more general shapes may be allowed; however, the cells should not contain holes under any circumstances. For the second requirement, a sufficient condition is that the cells can be organized into a *singular complex*. This means that for any two d-dimensional cells for $d \leq n$, if the boundaries of the cells intersect, then the common boundary must itself be a complete cell (of lower dimension).

In two-dimensional polygonal C-spaces, triangulation methods define nice cell decompositions that are appropriate for motion planning. Finding good triangulations, which for example means trying to avoid thin triangles, is given considerable attention in computational geometry [7.55]. Determining the decomposition of a polygonal obstacle region with holes that uses the smallest number of convex cells is *NP*-hard [7.56]. Therefore, we are willing to tolerate nonoptimal decompositions.

In three-dimensional C-spaces, if C_{obs} is polyhedral, then the vertical decomposition method directly extends by applying the plane sweep recursively, for example, the critical events may occur at each z coordinate, at which point changes a 2-D vertical decomposition over the x and y coordinates are maintained. The polyhedral case is obtained for a translating polyhedral robot among polyhedral obstacles in $\mathbb{R}^3$; however, for most interesting problems, C_{obs} becomes nonlinear. Suppose $C = \mathbb{R}^2 \times \mathbb{S}^1$, which corresponds to a robot that can translate and rotate in the plane. Suppose the robot and obstacles are polygonal. For the case of a line-segment robot, an $O(n^5)$ algorithm that is not too difficult to implement is given in [7.57]. The approaches for more general models and C-spaces are extremely difficult to use in practice; they are mainly of theoretical interest and are summarized in Sect. 7.7.3.

7.4.3 Potential Fields

A different approach for motion planning is inspired from obstacle avoidance techniques [7.58]. It does not explicitly construct a roadmap, but instead constructs a differentiable real-valued function $U : \mathbb{R}^m \to \mathbb{R}$, called a potential function, that guides the motion of the moving object. The potential is typically constructed so that it consists of an attractive component $U_{\text{a}}(\boldsymbol{q})$, which pulls the robot towards the goal, and a repulsive component $U_{\text{r}}(\boldsymbol{q})$, which pushes the robot away from the obstacles, as shown in Fig. 7.8. The gradient of the potential function is the vector $\nabla U(\boldsymbol{q}) = DU(\boldsymbol{q})^{\text{T}} = \left[\frac{\partial U}{\partial q_1}(\boldsymbol{q}), \ldots, \frac{\partial U}{\partial q_m}(\boldsymbol{q})\right]^{\text{T}}$, which

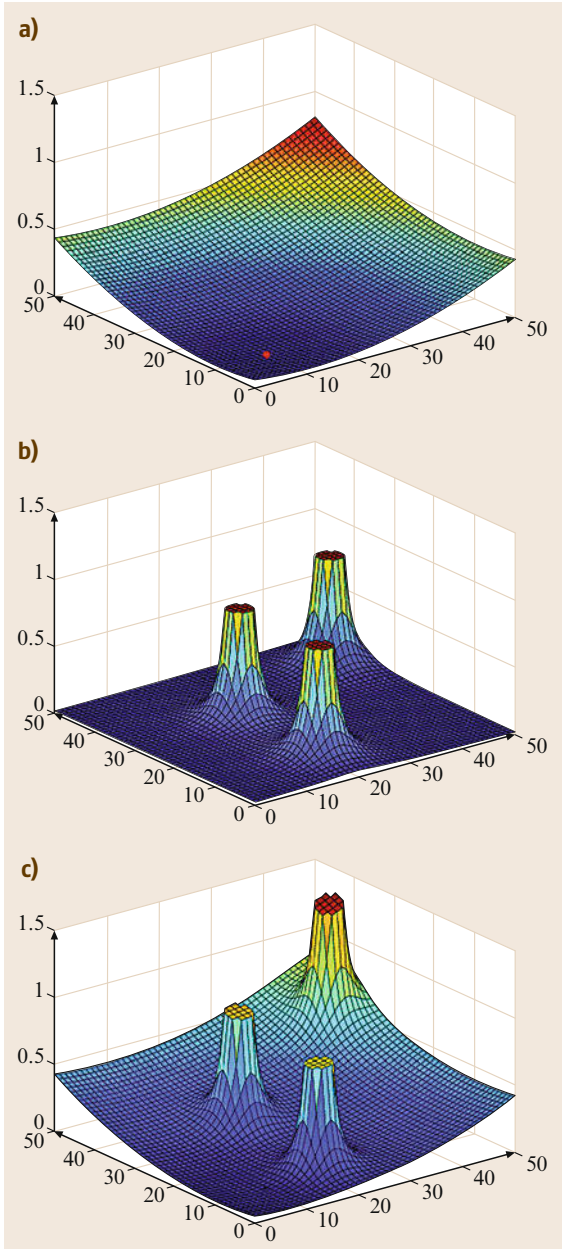


Fig.7.8a–c An attractive and a repulsive component define a potential function. **(a)** An attractive potential, **(b)** a repulsive potential, **(c)** an attractive and repulsive component define a potential function

points in the direction that locally maximally increases U. After the definition of U, a path can be computed by starting from $\boldsymbol{q}_\mathrm{I}$ and applying *gradient descent*:

1. $\boldsymbol{q}(0) = \boldsymbol{q}_\mathrm{I}$; $i = 0$;
2. **while** $\nabla U(\boldsymbol{q}(i)) \neq 0$ **do**
3. $\quad \boldsymbol{q}(i+1) = \boldsymbol{q}(i) + \nabla U(\boldsymbol{q}(i))$
4. $\quad i = i + 1$

However, this gradient-descent approach does not guarantee a solution to the problem. Gradient descent can only reach a local minimum of $U(\boldsymbol{q})$, which may not correspond to the goal state $\boldsymbol{q}_\mathrm{G}$, as shown in Fig. 7.9.

A planner that makes uses of potential functions and attempts to avoid the issue of local minima is the *randomized potential planner* [7.59]. The idea is to combine potential functions with random walks by employing multiple planning modes. In one mode, gradient descent is applied until a local minimum is reached. Another mode uses random walks to try to escape local minima. A third mode performs backtracking whenever several attempts to escape a local minimum have failed. In many ways, this approach can be considered as a sampling-based planner. It also provides the weaker completeness guarantee but it requires parameter tuning. Recent sampling-based methods achieve better performance by spending more time exploring the space, rather than focusing heavily on a potential function.

The gradient of the potential function can be also used to define a *vector field*, which assigns a motion for the robot at any arbitrary configuration $\boldsymbol{q} \in C$. This is an important advantage of the approach, beyond its computational efficiency, since it does not only compute a single path, but also a *feedback control* strategy. This makes the approach more robust against control and sensing errors. Most of the techniques in feedback motion planning are based on the idea of *navigation functions* [7.60], which are potential functions properly constructed so as to have a single minimum. A function $\phi : C_\mathrm{free} \to [0, 1]$ is called a navigation function if it:

- Is smooth (or at least C^k for $k \geq 2$),
- Has a unique minimum at $\boldsymbol{q}_\mathrm{G}$ in the connected component of the free space that contains $\boldsymbol{q}_\mathrm{G}$,
- Is uniformly maximal on the free-space boundaries,
- and is Morse, which means that all its critical points, such as saddle points, are isolated and can be avoided with small random perturbations.

Navigation functions can be constructed for sphere boundary spaces centered at $\boldsymbol{q}_\mathrm{I}$ that contain only spherical obstacles, as illustrated in Fig. 7.10. Then they can be extended to a large family of C-spaces that are diffeomorphic to sphere spaces, such as star-shaped spaces, as shown in Fig. 7.10. A more elaborate description of strategies for feedback motion planning will be presented in Chap. 47.

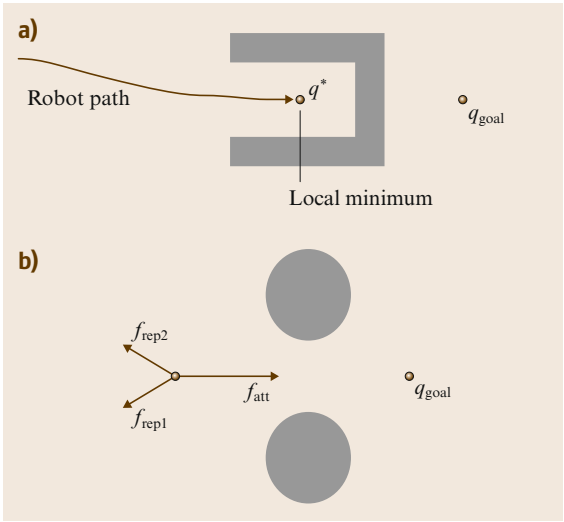


Fig.7.9a,b Two examples of the local minimum problem with potential functions

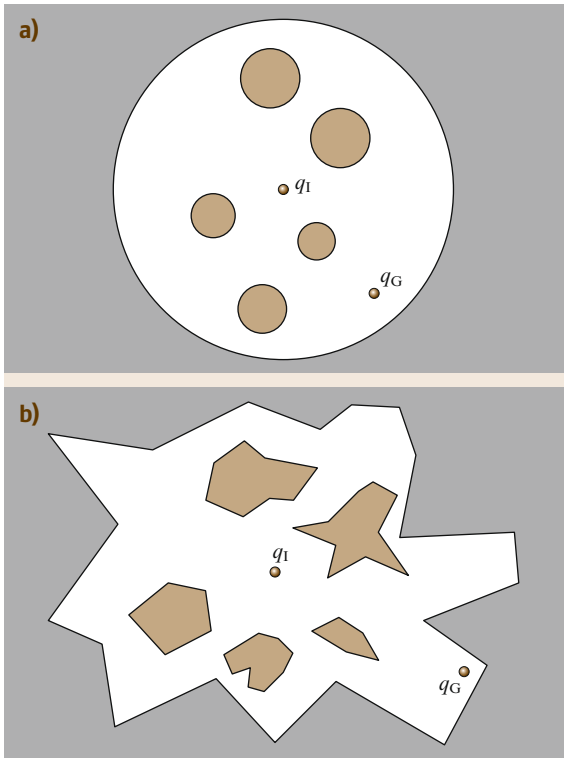


Fig.7.10a,b Examples of (a) sphere and (b) star spaces

Putting the issue of local minima aside, another major challenge for such potential function based approaches is constructing and representing the C-space in the first place. This issue makes the applications of these techniques too complicated for high-dimensional problems.

7.5 Differential Constraints

Robot motions must usually conform to both global and local constraints. Global constraints on C have been considered in the form of obstacles and possibly joint limits. Local constraints are modeled with differential equations, and are therefore called *differential constraints*. These limit the velocities, and possibly accelerations, at every point due to kinematic considerations, such as wheels in contact, and dynamical considerations, such as the conservation of angular momentum.

7.5.1 Concepts and Terminology

Let $\dot{\boldsymbol{q}}$ denote a velocity vector. Differential constraints on C can be expressed either *implicitly* in the form $g_i(\boldsymbol{q}, \dot{\boldsymbol{q}}) = 0$ or *parametrically* in the form $\dot{\boldsymbol{x}} = f(\boldsymbol{q}, \boldsymbol{u})$. The implicit form is more general but often more difficult to understand and utilize. In the parametric form, a vector-valued equation indicates the velocity that is obtained for a given $\boldsymbol{q}$ and $\boldsymbol{u}$, in which $\boldsymbol{u}$ is an *input*, chosen from some *input space*, U. Let T denote an interval of time, starting at $t = 0$.

To model dynamics, the concepts are extended into a *phase space* X of the C-space. Usually each point $\boldsymbol{x} \in X$ represents both a configuration and velocity, $\boldsymbol{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}})$. Both implicit and parametric representations are possible, yielding $g_i(\boldsymbol{x}, \dot{\boldsymbol{x}}) = 0$ and $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$, respectively. The latter is a common *control system* definition. Note that $\dot{\boldsymbol{x}} = (\dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$, which implies that acceleration constraints and full system dynamics can be expressed.

Planning in the state space X could lead to a straightforward definition of X_{obs} by declaring $\boldsymbol{x} \in X_{\text{obs}}$ if and only if $\boldsymbol{q} \in C_{\text{obs}}$ for $\boldsymbol{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}})$. However, another interesting possibility exists which provides some intuition about the difficulty of planning with dynamics. This possibility is based on the notion of a *region of inevitable collision*, which is defined as

$$X_{\text{ric}} = \{\boldsymbol{x}(0) \in X \mid \text{for any } \tilde{u} \in \mathcal{U}_\infty, \exists t > 0 \text{ such that } \boldsymbol{x}(t) \in X_{\text{obs}}\} , \tag{7.4}$$

in which $\boldsymbol{x}(t)$ is the state at time t obtained by integrating the control function $\tilde{u} : T \to U$ from $\boldsymbol{x}(0)$. The set $\mathcal{U}_\infty$ is a predefined set of all possible control functions. X_{ric} denotes the set of states in which the robot is either in collision or, because of momentum, it cannot do anything to avoid collision. It can be considered as an invisible obstacle region that grows with speed (Fig. 7.11).

Under the general heading of *planning under differential constraints*, there are many important categories of problems that have received considerable attention in research literature. The term *nonholonomic planning* was introduced for wheeled mobile robots [7.61]. A simple example is that a car cannot move sideways, thereby making parallel parking more difficult. In general, a *nonholonomic constraint* is a differential equality constraint that cannot be integrated into a constraint that involves no derivatives. Typically, nonholonomic constraints that appear in robotics are *kinematic*, and arise from wheels in contact [7.62]. Nonholonomic constraints may also arise from dynamics.

If a planning problem involves constraints on at least velocity and acceleration, the problem is often referred to as *kinodynamic planning* [7.63]. Usually, the model expresses a *fully actuated system*, which means that it can be expressed as $\ddot{\boldsymbol{q}} = h(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u})$, in which U contains an open set that includes the origin of $\mathbb{R}^n$ (here, n is the dimension of both U and C). It is possible for a problem to be nonholonomic, kinodynamic, both, or neither; however, in recent times, the terms are not used with much precision.

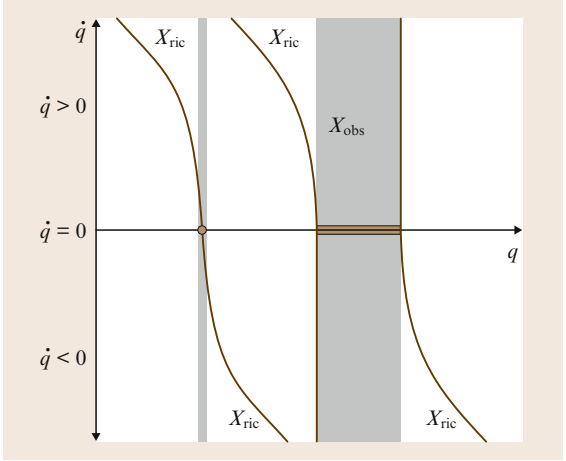


Fig. 7.11 The region of inevitable collision grows quadratically with the speed

Trajectory planning is another important term, which has referred mainly to the problem of determining both a path and velocity function for a robot arm (e.g., PUMA 560). In the treatment below, all of these will be referred to as *planning under differential constraints.*

7.5.2 Discretization of Constraints

The only known methods for complete and optimal planning under differential constraints in the presence of obstacles are for the double integrator system with $X = \mathbb{R}$ [7.64] and $X = \mathbb{R}^2$ [7.65]. To develop planning algorithms in this context, several discretizations are often needed. For ordinary motion planning, only C needed to be discretized; with differential constraints, T and possibly also U require discretization, in addition to C (or X).

Discretization of the differential constraints is one of the most important issues. To solve challenging planning problems efficiently, it is often necessary to define *motion primitives* for the particular dynamical system [7.40, 66, 67]. One of the simplest ways to discretize the differential constraints is to construct a *discrete-time model*, which is characterized by three aspects:

1. The time interval T is partitioned into intervals of length Δt. This enables stages to be assigned, in which stage k indicates that $(k-1)\Delta t$ time has elapsed.
2. A finite subset U_{d} of the action space U is chosen. If U is already finite, then this selection may be $U_{\mathrm{d}} = U$.
3. The action $\boldsymbol{u}(t)$ must remain constant over each time interval.

From an initial state, $\boldsymbol{x}$, a *reachability tree* can be formed by applying all sequences of discretized actions. Figure 7.12 shows the path of this tree for the *Dubins car*, which is a kinematic model of a car that drives in the plane at unit speed and cannot move in reverse. The edges of the tree are circular arcs and line segments. For general systems, each trajectory segment in the tree is determined by numerical integration of $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$ for a given $\boldsymbol{u}$. In general, this can be viewed as an *incremental simulator* that takes an input and produces a trajectory segment according to $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$.

7.5.3 Decoupled Approach

A popular paradigm for trajectory planning and other problems that involve dynamics is to decouple the problem into first planning a path and then computing a timing function along the path by performing a search in the space spanned by $(s, \dot{s})$, in which s is the path

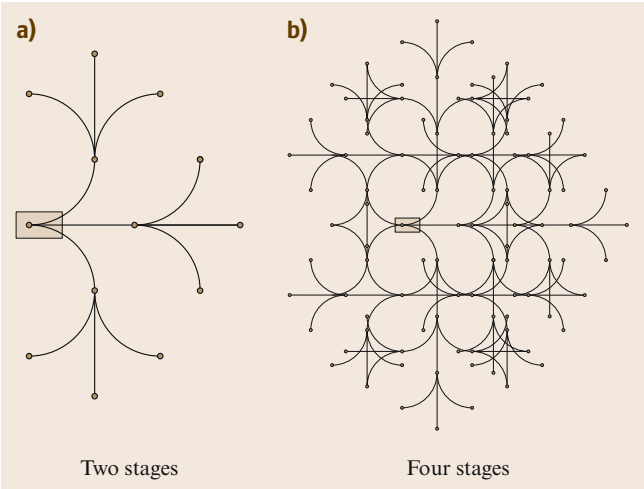


Fig. 7.12 **(a)** A reachability tree for the Dubins car with three actions. **(b)** A 2-stage tree is shown. The k-th stage produces 3^k new vertices

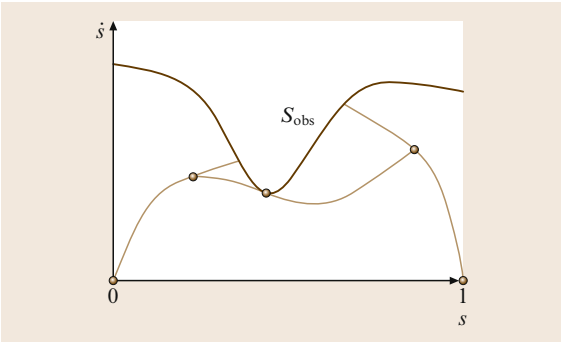


Fig. 7.13 An illustration of the bang–bang approach to computing a time-optimal trajectory. The solution trajectory is obtained by connecting the dots

parameter and $\dot{s}$ is its first derivative. This leads to a diagram such as the one shown in Fig. 7.13, in which the upper region S_{obs} must be avoided because the corresponding motion of the mechanical system violates the differential constraints. Most methods are based on early work in [7.68, 69], and determine a *bang–bang -control*, which means that they switch between accelerating and decelerating at full speed. This applies to determining time-optimal trajectories (optimal once constrained to the path). Dynamic programming can be used for more general problems [7.70].

For some problems and nonholonomic systems, *steering methods* have been developed to solve the two-point boundary value problem efficiently [7.62, 71]. This means that, for any pair of states, a trajectory that ignores obstacles but satisfies the differential constraints can be obtained. Moreover, for some systems,

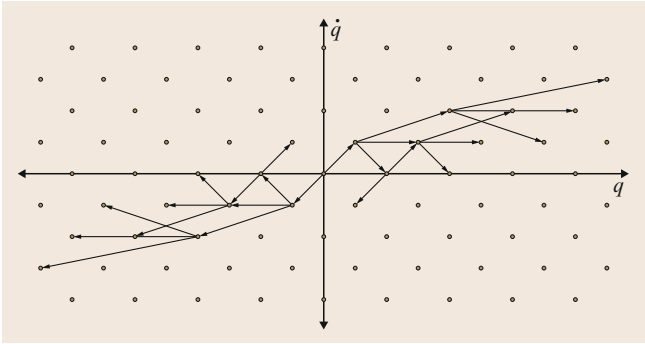


Fig. 7.14 Reachability graph from the origin, shown after three stages (the true edges are actually parabolic arcs when acceleration or deceleration occurs). Note that a lattice is obtained, but the distance traveled in one stage increases as $|\dot{\boldsymbol{q}}|$ increases

the complete set of optimal trajectories has been characterized [7.72, 73]. These control-based approaches enable straightforward adaptation of the sampling-based roadmap approach [7.74, 75]. One decoupled approach is to first plan a path that ignores differential constraints, and then incrementally transform it into one that obeys the constraints [7.62, 76].

7.5.4 Kinodynamic Planning

Due to the great difficulty of planning under differential constraints, many successful planning algorithms that address kinodynamic problems directly in the phase space X are sampling based.

Sampling-based planning algorithms proceed by exploring one or more reachability trees. Many parallels can be drawn with searching on a grid; however, reachability trees are more complicated because they do not necessarily involve a regular lattice structure. The vertex set of reachability trees is dense in most cases. It is therefore not clear how to search a bounded region exhaustively at a fixed resolution. It is also difficult to design approaches that behave like a multiresolution grid, in which refinements can be made arbitrarily to ensure resolution completeness.

Many algorithms attempt to convert the reachability tree into a lattice. This is the basis of the original kinodynamic planning work [7.63], in which the discrete-time approximation to the double integrator, $\ddot{\boldsymbol{q}} = \boldsymbol{u}$, is forced onto a lattice as shown in Fig. 7.14. This enables an approximation algorithm to be developed that solves the kinodynamic planning problem in time polynomial in the approximation quality $1/\epsilon$ and the number of primitives that define the obstacles. Generalizations of the methods to fully actuated systems are described in [7.7]. Surprisingly, it is even possible to obtain a lattice for some underactuated, nonholonomic systems [7.77].

If the reachability tree does not form a lattice, then one approach is to force it to behave as a lattice by imposing a regular cell decomposition over X (or C), and allowing no more than one vertex per cell to be expanded in the reachability graph (Fig. 7.15). This idea was introduced in [7.78]. In their version of this approach, the reachability graph is expanded by dynamic programming. Each cell is initially marked as being in collision or being collision free, but not yet visited. As cells are visited during the search, they become marked as such. If a potential new vertex lands in a visited cell, it is not saved. This has the effect of pruning the reachability tree.

Other related approaches do not try to force the reachability tree onto a lattice. RRTs were designed to expand the tree in a way that is biased toward covering as much new territory as possible in each iteration [7.79]. Planners that are based on the concept of expansive trees attempt to control the density of ver-

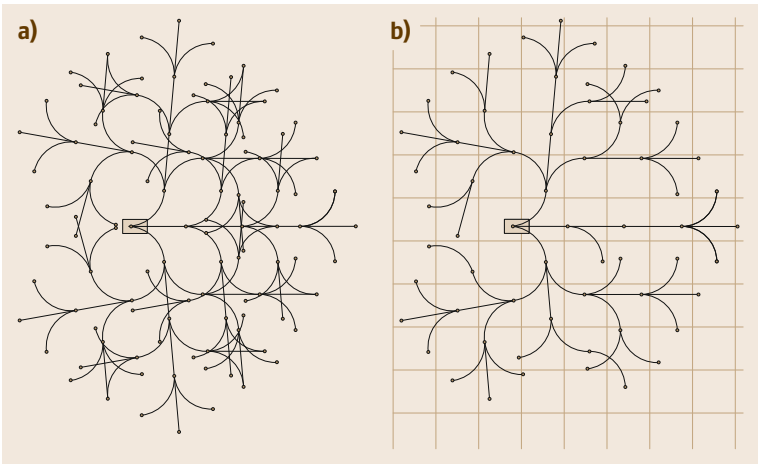


Fig. 7.15 **(a)** The first four stages of a dense reachability graph for the Dubins car. **(b)** One possible search graph, obtained by allowing at most one vertex per cell. Many branches are pruned away. In this simple example there are no cell divisions along the θ-axis

tices in the tree by analyzing neighborhoods [7.44]. The path-directed subdivision tree planner expands a tree, while building an adaptive subdivision of the state space, so as to avoid resampling the same regions of the space [7.47, 80]. Such approaches can be biased to accelerate the expansion of the tree towards a goal, while still providing the weaker probabilistic completeness guarantee [7.48]. VIDEO 24 provides an example of the use of a tree-based planner together with a physics-engine which accounts for the constraints.

7.6 Extensions and Variations

A brief overview of other important extensions to the basic motion planning problem are presented in this section.

7.6.1 Closed Kinematic Chains

In many cases, the robot may be consist of links that form closed loops. This arises in many important applications, for example, if two arms grasp an object then a loop is formed and a humanoid robot forms a loop if both legs touch the ground. For *parallel robots*, loops are intentionally designed into the robot [7.81]; a classic example is the Stewart–Gough platform. To model closed-chain problems, the loops are broken so that a kinematic tree of links is obtained. The main complication is that constraints on C of the form $h(\boldsymbol{q}) = 0$ are introduced, which require that the loops are maintained. This causes great trouble for most planning algorithms because without loops a parameterization of C was available. The closure constraints restrict the planning to a lower-dimensional subset of C for which no parameterization is given. Computing a parameterization is generally difficult or impossible [7.82], although there has been recent progress for some special cases [7.83].

Sampling-based approaches can generally be adapted to handle closed chains. The main difficulty is that the samples $\alpha(i)$ over C are unlikely to be configurations that satisfy closure. In [7.84], both RRTs and PRMs were adapted to closed chains. RRTs performed much better because a costly optimization was required in the PRM to move samples onto the closure subspace; RRTs on the other hand do not require samples to lie in this subspace. By decomposing chains into active and passive links, followed by inverse kinematics computations, performance was dramatically improved for PRMs in [7.85]. This idea was further improved by the introduction of the *random loop generator* (RLG). Based on this, some of the most challenging closed-chain planning problems ever solved appear in [7.86].

7.6.2 Manipulation Planning

In most forms of motion planning, the robot is not allowed to touch obstacles. Suppose instead that it is expected to interact with its environment by manipulating objects. The goal may be to bring an object from one place to another, or to rearrange a collection of objects. This leads to a kind of hybrid motion planning problem, which mixes discrete and continuous spaces. There are discrete modes that correspond to whether the robot is carrying a part [7.87]. In the *transit mode*, the robot moves toward a part. In the *transfer mode*, it carries the part. Transitions between modes require meeting specific grasping and stability requirement. One important variant of manipulation planning is *assembly planning*, in which the goal is to fit a collection of pieces together to make an assembled product [7.88]. Most motion planning work makes limiting assumptions on the kinds of interaction that can occur between the robot and the objects. For richer models of manipulation, see [7.89].

7.6.3 Time-Varying Problems

Suppose that the workspace contains moving obstacles whose trajectories are specified as a function of time. Let $T \subset \mathbb{R}$ denote the *time interval*, which may be *bounded* or *unbounded*. A state X is defined as $X = C \times T$, in which C is the usual C-space of the robot. The obstacle region in X is characterized as

$$X_{\text{obs}} = \{(\boldsymbol{q}, t) \in X \mid \mathcal{A}(\boldsymbol{q}) \cap \mathcal{O}(t) \neq \emptyset\} , \tag{7.5}$$

in which $\mathcal{O}(t)$ is a time-varying obstacle. Many planning algorithms can be adapted to X, which has only one more dimension than C. The main complication is that time must always increase along a path through X.

For the easiest version of the problem, there is no bound on the robot speed. In this case, virtually any sampling-based algorithm can be adapted. Incremental searching and sampling methods apply with little modification, except that paths are directed so that forward time progress is made. Using bidirectional approaches is more difficult for time-varying problems because the goal is usually not a single point due to the time dependency. Sampling-based roadmaps can be adapted; however, a *directed roadmap* is needed, in which every edge must be directed to yield a time-monotonic path.

If the motion model is *algebraic* (i. e., expressed with polynomials) then X_{obs} is semi-algebraic. This enables cylindrical algebraic decomposition to apply. If X_{obs} is polyhedral, as depicted in Fig. 7.16, then vertical decomposition can be used. It is best to first sweep the plane along the T-axis, stopping at the critical times when the linear motion changes.

There has been no consideration so far of the speed at which the robot must move to avoid obstacles. It is obviously impractical in many applications if the solution requires the robot to move arbitrarily fast. One step towards making a realistic model is to enforce a bound on the speed of the robot. Unfortunately, the problem is considerably more difficult. Even for piecewise-linear motions of obstacles in the plane, the problem has been established to be PSPACE-hard [7.90]. A complete algorithm based on the shortest-path roadmap is presented in [7.91].

An alternative to defining the problem in $C \times T$ is to decouple it into a *path planning* part and a *motion timing* part. A collision-free path in the absence of obstacles is first computed. A search in a 2-D space is then performed to determine the *timing function* (or *time scaling*) for the path.

7.6.4 Multiple Robots

A simple extension to the basic motion planning problem can be made to handle multibody robots by including robot self-intersections; however, it is important to specify the pairs of bodies for which collision is unacceptable, for example, consecutive links in a robot arm are allowed to touch.

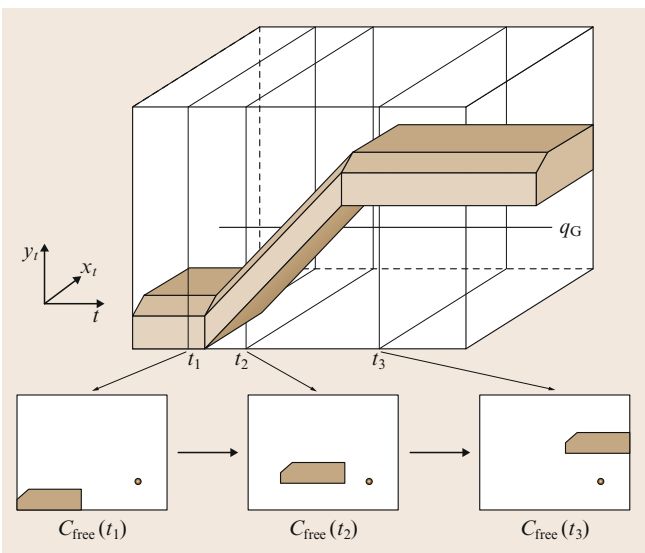


Fig. 7.16 A time-varying example with linear obstacle motion

Substantial attention has been devoted to the problem of planning for multiple robots (VIDEO 21 and VIDEO 22). Suppose there are m robots. A state space is defined that considers the configurations of all robots simultaneously,

$$X = C^1 \times C^2 \times \cdots \times C^m. \tag{7.6}$$

A state $\boldsymbol{x} \in X$ specifies all robot configurations, and may be expressed as $\boldsymbol{x} = (\boldsymbol{q}^1, \boldsymbol{q}^2, \ldots, \boldsymbol{q}^m)$. The dimension of X is N, which is $N = \sum_{i=1}^{m} \dim(C^i)$.

There are two sources of obstacle regions in the state space: (1) *robot–obstacle* collisions, and (2) *robot–robot* collisions. For each i such that $1 \leq i \leq m$, the subset of X that corresponds to robot $\mathcal{A}^i$ in collision with the obstacle region $\mathcal{O}$ is

$$X^i_{\text{obs}} = \{\boldsymbol{x} \in X \mid \mathcal{A}^i(\boldsymbol{q}^i) \cap \mathcal{O} \neq \emptyset\}. \tag{7.7}$$

This models the robot–obstacle collisions.

For each pair, $\mathcal{A}^i$ and $\mathcal{A}^j$, of robots, the subset of X that corresponds to $\mathcal{A}^i$ in collision with $\mathcal{A}^j$ is

$$X^{ij}_{\text{obs}} = \{\boldsymbol{x} \in X \mid \mathcal{A}^i(\boldsymbol{q}^i) \cap \mathcal{A}^j(\boldsymbol{q}^j) \neq \emptyset\}. \tag{7.8}$$

Both (7.7) and (7.8) will be combined in (7.9) to yield X_{obs}. The obstacle region in X is

$$X_{\text{obs}} = \left(\bigcup_{i=1}^{m} X^i_{\text{obs}}\right) \cup \left(\bigcup_{ij,\ i \neq j} X^{ij}_{\text{obs}}\right). \tag{7.9}$$

Once these definitions have been made, any general-purpose planning algorithm can be applied because X and X_{obs} appear no different from C and C_{obs}, except that the dimension N may be very high. Approaches that plan directly in X are called *centralized*. The high dimensionality of X motivates the development of *decoupled* approaches that handle some aspects of the planning independently for each robot. Decoupled approaches are usually more efficient, but this usually comes at the expense of sacrificing completeness. An early decoupled approach is *prioritized planning* [7.92, 93], in which a path and timing function is computed for the i-th robot while treating the first $i-1$ robots as moving obstacles as they follow their paths. Another decoupled approach is *fixed-path* coordination [7.94], in which the paths are planned independently for each robot, and then their timing functions are determined by computing a collision-free path through an m-dimensional *coordination space*. Each axis in this space corresponds to the domain of the path of one robot. Fig. 7.17 shows an example. The idea has been generalized to coordination on roadmaps [7.95, 96].

7.6.5 Uncertainty in Predictability

If the execution of the plan is not predictable, then feedback is needed. The uncertainty may be modeled either *implicitly*, which means that the plan is able to respond to unexpected future configurations, or *explicitly*, which means that the uncertainty is precisely characterized and analyzed in the development of a plan. Potential-function-based approaches are one way of achieving feedback motion planning.

A plan can be represented as a vector field over C_{free}, in which each vector indicates the required velocity. The integral curves of the field should flow into the goal without leaving C_{obs}. If dynamics are a concern, then the vector field can be tracked by an *acceleration-based control model*

$$\boldsymbol{u} = K(f(\boldsymbol{q}) - \dot{\boldsymbol{q}}) + \nabla_{\dot{\boldsymbol{q}}} f(\boldsymbol{q}) \,, \tag{7.10}$$

in which K is a scalar *gain constant*. Alternatively, a vector field may be designed directly on the phase space, X; however, there are not methods to compute such fields efficiently under general conditions. This can also be considered as a feedback control problem with implicit, nonlinear constraints on X.

If the uncertainty is modeled explicitly, then a *game against nature* is obtained, in which the uncertainty is caused by a special decision maker called *nature*. The decisions of nature can either be modeled *nondeterministically*, which means that a set of possible actions is specified, or *probabilistically*, which means that a probability distribution or density is specified over the nature actions. Under nondeterministic uncertainty, *worst-case analysis* is usually performed to select a plan; under probabilistic uncertainty, *expected-case analysis* is usually performed. Numerous approaches exist for such problems, including value iteration, Dijkstra-like algorithms, and reinforcement learning algorithms [7.7].

7.6.6 Sensing Uncertainty

Consider solving tasks such as localization, map building, manipulation, target tracking, and pursuit-evasion (hide-and-seek) with limited sensing. If the current configuration or state is not known during execution, then the problem may appear quite different. Information is obtained from sensors, and the planning problem naturally lives in an *information space* or I-*space* [7.7, Chap. 11]. The state may include the configuration, velocities, or even the map of the environment (e.g., obstacles). The most basic I-space is the set of all histories that can be obtained during execution, based on all sensing observations, actions previously applied,

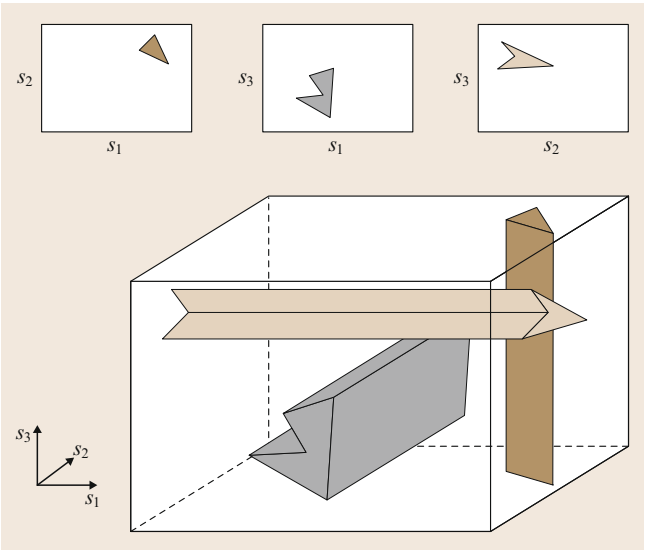


Fig. 7.17 The obstacles that arise from coordinating m robots are always cylindrical. The set of all $\frac{1}{2}m(m-1)$ axis-aligned 2-D projections completely characterizes X_{obs}

and the initial conditions. The goal in developing efficient algorithms in this context is to determine information mappings that reduce the I-space size or complexity so that plans that can be computed that use *information feedback*. The traditional way to use the information state is for estimating the state. This is sufficient for solving many tasks, but it is often not necessary. It may be possible to design and execute successful plans without ever knowing the current state. This can lead to more robust robot systems which may also be cheaper to manufacture due to weaker sensing requirements.

Two important families of I-spaces are *nondeterministic* and *probabilistic*. A nondeterministic information state (I-state) is a set of states that are possible given the available history of sensor observations and actions applied during execution. The nondeterministic I-space is the set of all possibilities. Similarly, a probabilistic I-state is a probability density function over the state space, conditioned on the available history. The probabilistic I-space is often called the *belief space*, which represents the set of all probability density functions. Both filtering and planning over these spaces remains a topic of active research. One of most useful and classical results is the *Kalman filter* [7.97], for which the belief space reduces to Gaussians, allowing it to be completely parametrized by mean and covariance of state. Many approaches to reasoning in these I-spaces attempt to reduce its complexity, through combinatorial reasoning in the case of nondeterministic I-spaces [7.98] and through approximations,

sampling, and dimensionality reduction techniques in belief spaces [7.99–103]. For example, a sampling-based roadmap can be constructed directly in the belief space [7.104].

7.6.7 Optimal Planning

In most formulations of planning, computing an optimal solution is given little or no importance. Several factors have contributed to this trend. One of the most fundamental is that a natural criterion of optimality often does not exist. Unlike control theory, where optimality is a central goal, the main task in planning is to avoid obstacles. Imagine walking quickly through a furniture store. Moving along the shortest possible path would cause you to touch the corners of obstacles, which might not be desirable. You could try to maximize clearance, but this causes longer paths, which again might not be desirable. Another factor is that solutions produced by path planning algorithms tend to not be excessively long, especially after some quick post-processing in the case of sampling-based planners. Finally, the computational complexity of the optimal planning problem is typically worse than its feasible (not necessarily optimal) counterpart. One of the most notable exceptions is planning in field robotics (outdoor vehicles in unstructured terrain), for which cost function determines navigability at each point, leading to the well-known family of *D* algorithms* [7.105, 106].

In spite of these issues, several useful approaches and interesting ideas have emerged. As mentioned in Sect. 7.4.1, the shortest-path roadmap is an effective multiple-query approach in the case of a 2-D, polygonal C-space. Alternatively, the *continuous Dijkstra* method provides an effective single-query approach by propagating wavefronts that correspond to level sets of the optimal cost from the initial configuration [7.107, 108]. The wavefronts are propagated combinatorially, stopping only at critical events, leading to an exact, optimal solution. In the case of 3-D polyhedral C-spaces, the shortest path problem already becomes PSPACE-hard [7.15]. However, algorithms that produce approximately optimal solutions exist [7.109–111], and are useful in C-spaces of several dimensions. Dijkstra-like approaches can be adapted to include various forms of uncertainty and differential constraints, and are all derived in some way from *value iteration* methods introduced by Bellman in the 1950s. See Chaps. 7, 10, and 14 of [7.7] for more discussion. Pushing into even higher dimensions, recent sampling-based planning methods have produced asymptotically optimal versions RRTs [7.112] and PRMs [7.112, 113], which have been shown to produce paths that improve in quality as time progresses.

7.7 Advanced Issues

We cover here a series of more advanced issues, such as topics from topology and sampling theory, and how they influence the performance of motion planners. The last section is devoted to computational algebraic geometry techniques that achieve completeness in the general case. Rather than being a practical alternative, these techniques serve as an upper bound on the best asymptotic running time that could be obtained.

7.7.1 Topology of Configuration Spaces

Manifolds

One reason that the topology of a C-space is important is because it affects its representation. Another reason is that, if a path-planning algorithm can solve problems in a topological space, then that algorithm may carry over to topologically equivalent spaces.

The following definitions are important in order to describe the topology of C-space. A map $\phi : S \to T$ is called a *homeomorphism* if ϕ is a bijection and both ϕ and ϕ^{-1} are continuous. When such a map exists, S and T are said to be homeomorphic. A set S is an n-dimensional *manifold* if it is locally homeomorphic to $\mathbb{R}^n$, meaning that each point in S possesses a neighborhood that is homeomorphic to $\mathbb{R}^n$. For more details, see [7.114, 115].

In the vast majority of motion planning problems, the configuration space is a manifold. An example of a C-space that is not a manifold is the closed unit square: $[0, 1] \times [0, 1] \subset \mathbb{R}^2$, which is a manifold with boundary obtained by pasting the one-dimensional boundary on the two-dimensional open set $(0, 1) \times (0, 1)$. When a C-space is a manifold, then we can represent it with just n parameters, in which n is the dimension of the configuration space. Although an n-dimensional manifold can be represented using as few as n parameters, due to constraints it might be easier to use a representation that has higher number of parameters, e.g., the unit circle $\mathbb{S}^1$ can be represented as $\mathbb{S}^1 = \{(x, y) | x^2 + y^2 = 1\}$ by embedding $\mathbb{S}^1$ in $\mathbb{R}^2$. Similarly, the torus T^2 can be embedded in $\mathbb{R}^3$.

Representation

Embeddings into higher-dimensional spaces can facilitate many C-space operations. For example, the orientation of a rigid body in space can be represented by

a $n \times n$ matrix of real numbers. The n^2 matrix entries must satisfy a number of smooth equality constraints, making the manifold of such matrices a *submanifold* of $\mathbb{R}^{n^2}$. One advantage is that these matrices can be multiplied to get another matrix in the manifold. For example, the orientation of a rigid-body in n-dimensional space ($n = 2$ or 3) is described by the set $SO(n)$, the set of all $n \times n$ rotation matrices. The position and orientation of a rigid body is represented by the set $SE(n)$, the set of all $n \times n$ homogeneous transformation matrices. These matrix groups can be used to (1) represent rigid-body configurations, (2) change the reference frame for the representation of a configuration, and (3) displace a configuration.

There are numerous parameterizations of $SO(3)$ [7.116] but unit quaternions correctly preserve the C-space topology as $\mathbb{S}^1$ represents 2-D rotations. Quaternions were introduced in Chap. 2. There is, however, a two-to-one correspondence between unit quaternions and 3-D rotation matrices. This causes a topological issue that is similar to the equivalence of 0 and 2π for 2-D rotations. One way to account for this is to declare antipodal (opposite) points on $\mathbb{S}^3$ to be equivalent. In planning, only the upper hemisphere of $\mathbb{S}^3$ is needed, and paths that cross the equator instantly reappear on the opposite side of $\mathbb{S}^3$, heading back into the northern hemisphere. In topology, this is called a real projective space: $\mathbb{RP}^3$. Hence, the C-space of a 3-D body capable only of rotation is $\mathbb{RP}^3$. If both translation and rotation are allowed, then $SE(3)$, the set of all 4×4 homogeneous transformation matrices, yields

$$C = \mathbb{R}^3 \times \mathbb{RP}^3 \,, \tag{7.11}$$

which is six dimensional. A configuration $\boldsymbol{q} \in C$ can be expressed using quaternions with seven coordinates, (x, y, z, a, b, c, d), in which $a^2 + b^2 + c^2 + d^2 = 1$. More examples can be found in Table 7.1.

7.7.2 Sampling Theory

Since the most successful paradigm for motion planning today is the sampling-based framework, presented in Sect. 7.3, sampling theory becomes relevant to the motion planning problem.

Table 7.1 Some common robots and their C-spaces

Type of robot	C-space representation
Mobile robot translating in the plane	$\mathbb{R}^2$
Mobile robot translating and rotating in the plane	$SE(2)$ or $\mathbb{R}^2 \times S^1$
Rigid body translating in the three-space	$\mathbb{R}^3$
A spacecraft	$SE(3)$ or $\mathbb{R}^3 \times SO(3)$
An n-joint revolute arm	T^n
A planar mobile robot with an attached n-joint arm	$SE(2) \times T^n$

Metrics in Configuration/State Spaces

Virtually all sampling-based methods require some notion of distance on C. For example, the sampling-based roadmap method selects candidate vertices to connect a new configuration given a distance-defined neighborhood. Similarly, the rapidly exploring dense trees expands the tree from the nearest node of the tree to a newly sampled configuration. Usually, a *metric*, $\rho : C \times C \to \mathbb{R}$, is defined, which satisfies the standard axioms: nonnegativity, reflexivity, symmetry, and the triangle inequality.

Two difficult issues that arise in constructing a metric are: (1) the topology of C must be respected, and (2) several different quantities, such as linear and angular displacements, must be compared in some way. To illustrate the second issue, consider defining a metric ρ_z for a space constructed as $Z = X \times Y$ as

$$\begin{aligned} \rho_z(z, z') &= \rho_z(x, y, x', y') \\ &= c_1 \rho_x(x, x') + c_2 \rho_y(y, y') \,. \end{aligned} \tag{7.12}$$

Above, c_1 and c_2 are arbitrary positive constants that indicate the relative weights of the two components. For a 2-D rotation, θ_i, expressed as $a_i = \cos\theta_i$ and $b_i = \sin\theta_i$, a useful metric is

$$\rho(a_1, b_1, a_2, b_2) = \cos^{-1}(a_1 a_2 + b_1 b_2) \,. \tag{7.13}$$

The 3-D equivalent is obtained by defining

$$\rho_0(\boldsymbol{h}_1, \boldsymbol{h}_2) = \cos^{-1}(a_1 a_2 + b_1 b_2 + c_1 c_2 + d_1 d_2) \,, \tag{7.14}$$

in which each $\boldsymbol{h}_i = (a_i, b_i, c_i, d_i)$ is a unit quaternion. The metric is defined as $\rho(\boldsymbol{h}_1, \boldsymbol{h}_2) = \min(\rho_0(\boldsymbol{h}_1, \boldsymbol{h}_2), \rho_0(\boldsymbol{h}_1, -\boldsymbol{h}_2))$, by respecting the required identification of antipodal points. This computes the shortest distance in $\mathbb{R}^4$, for a path constrained to the unit sphere.

In some algorithms, defining volume on C may also be important. In general, this leads to a *measure space*, for which the volume function (called the *measure*) must satisfy axioms that resemble the probability axioms, but without normalization. For transformation groups, one must be careful to define volumes in a way that is invariant with respect to transformations. Such volumes are called *Haar measures*. Defining volumes via balls using the metric definitions (7.13) and (7.14) actually satisfy this concern.

Probabilistic Versus Deterministic Sampling

The C-space may be sampled *probabilistically* or *deterministically*. Either way, the requirement is usually that a dense sequence α of samples is obtained.

This means that, in the limit as the number of samples tends to infinity, the samples become arbitrarily close to every point in C. For probabilistic sampling, this denseness (with probability one) ensures *probabilistic completeness* of a planning algorithm. For deterministic sampling, it ensures *resolution completeness*, which means that, if a solution exists, the algorithm is guaranteed to find it; otherwise, it may run forever.

For probabilistic sampling, samples are selected randomly over C, using a uniform probability density function. To obtain uniformity in a meaningful way, the Haar measure should be used. This is straightforward in many cases; $SO(3)$ however is tricky. A uniform (with respect to Haar measure) random quaternion is selected as follows. Choose three points $u_1, u_2, u_3 \in [0, 1]$ uniformly at random, and let [7.117]

$$\boldsymbol{h} = \Big(\sqrt{1-u_1} \sin 2\pi u_2,\ \sqrt{1-u_1} \cos 2\pi u_2\ , \\ \sqrt{u_1} \sin 2\pi u_3,\ \sqrt{u_1} \cos 2\pi u_3 \Big)\ . \tag{7.15}$$

Even though random samples are uniform in some sense, they are also required to have some irregularity to satisfy statistical tests. This has motivated the development of deterministic sampling schemes that offer better performance [7.118]. Instead of being concerned with randomness, deterministic sampling techniques are designed to optimize criteria, such as *discrepancy* and *dispersion*. Discrepancy penalizes regularity in the sample, which frequently causes trouble in numerical integration. Dispersion gives the radius of the largest empty (not containing samples) ball. Thus, driving dispersion down quickly means that the whole space is explored quickly. Deterministic samples may be *irregular* neighborhood structure (appearing much like random samples), or *regular* neighborhood structure, which means that points are arranged along a grid or lattice. For more details in the context of motion planning, see [7.7].

7.7.3 Computational Algebraic Geometry Techniques

Sampling-based algorithms provide good practical performance at the expense of achieving only a weaker form of completeness. On the other hand, complete algorithms, which are the focus of this section, are able to deduce that there is no solution to a planning problem.

Complete algorithms are able to solve virtually any motion planning problem as long as C_{obs} is represented by patches of algebraic surfaces. Formally, the model must be *semi-algebraic*, which means that it is formed from unions and intersections of roots of multivariate polynomials in q, and for computability, the polynomials must have rational coefficients (otherwise roots may not have finite representations). The set of all roots to polynomials with rational coefficients is called *real algebraic numbers* and has many nice computational properties. See [7.12, 119–121] for more information on the exact representation and calculation with real algebraic numbers. For a gentle introduction to algebraic geometry, see [7.82].

To use techniques based on algebraic geometry, the first step is to convert the models into the required polynomials. Suppose that the models, the robot, $\mathcal{A}$, and the obstacles $\mathcal{O}$ are semi-algebraic (this includes polyhedral models). For any number of attached 2-D or 3-D bodies, the kinematic transformations can be expressed using polynomials. Since polynomial transformations of polynomials yield polynomials, the transformed robot model is polynomial. The algebraic surfaces that comprise C_{obs} are computed by carefully considering all contact types, which characterize all ways to pair a robot feature (faces, edges, vertices) with an obstacle feature [7.6, 7, 9, 122]. This step already produces too many model primitives to be useful in most applications.

Once the semi-algebraic representation has been obtained, powerful techniques from algebraic geometry can be exploited. One of the most widely known algorithms, *cylindrical algebraic decomposition* [7.119, 123, 124], provides the information needed to solve the motion planning problem. It was originally designed to determine whether *Tarski sentences*, which involve quantifiers and polynomials, are satisfiable, and to find an equivalent expression that does not involve quantifiers. The decomposition produces a finite set of cells over which the signs of the polynomials remain fixed. This enables a systematic approach to satisfiability and quantifier elimination. It was recognized by *Schwartz* and *Sharir* [7.121] that it also solves motion planning.

The method is conceptually simple, but there are many difficult technical details. The decomposition is called cylindrical because the cells are organized into vertical columns of cells, see Fig. 7.18 for a 2-D example. There are two kinds of critical events, shown in Fig. 7.19. At critical points, rays are extended indefinitely in both vertical directions. The decomposition differs from the vertical decomposition in Fig. 7.7 because there the rays were only extended until the next obstacle was hit. Here, columns of cells are obtained.

In n dimensions, each column represents a chain of cells. The first and last cells are n-dimensional and unbounded. The remaining cells are bounded and alternate between being $(n-1)$-dimensional and n-dimensional.

The bounded n-dimensional cells are bounded above and below by the roots of single multivariate polynomials. This makes it simple to describe the cells and their connectivity. To compute this cell decomposition, the algorithm constructs a cascading chain of projections. In the first step, C_{obs} is projected from $\mathbb{R}^n$ to $\mathbb{R}^{n-1}$. This is followed by a projection into $\mathbb{R}^{n-2}$. This repeats until $\mathbb{R}$ is obtained with a univariate polynomial that encodes the places at which all critical boundaries need to be placed. In a second phase of the algorithm, a series of liftings is performed. Each lifting takes the polynomials and cell decomposition over $\mathbb{R}^i$ and lifts them via columns of cells to $\mathbb{R}^{i+1}$. A single lifting is illustrated in Fig. 7.18b. The running time of the full algorithm depends on the particular methods used to perform the algebraic computations. The total running time required to use cylindrical algebraic decomposition for motion planning is bounded by $(md)^{O(1)^n}$, in which m is the number of polynomials to describe C_{obs} (a huge number), and d is the maximum algebraic degree. (It may seem odd for $O(\cdot)$ to appear in the middle of an expression. In this context, it means that there exists some $c \in [0, \infty)$ such that the running time is bounded by $(md)^{c^n}$. Note that another O is not necessary in the front of the whole formula.) The main point to remember is that the algorithm is doubly exponential in the dimension of C (even the number of cells is doubly exponential).

Although performing the cylindrical decomposition is sufficient for solving motion planning, it computes more information than is necessary. This motivates *Canny*'s roadmap algorithm [7.12], which produces a roadmap directly from the semi-algebraic set, rather than constructing a cell decomposition along the way. Since there are doubly exponentially many cells in the cylindrical algebraic decomposition, avoiding this construction pays off. The resulting roadmap method of *Canny* solves the motion planning problem in time that is again polynomial in the number of polynomials and polynomial in the algebraic degree, but is only singly exponential in dimension [7.12].

The basic idea is to find silhouette curves in $\mathbb{R}^2$ of C_{obs} in $\mathbb{R}^n$. The method finds zero-dimensional critical points and one-dimensional critical curves. The critical curves become roadmap edges, and the critical points are places at which the algorithm recursively finds silhouettes of $(n-1)$-dimensional slices of C_{obs}. These contribute more critical points and curves. The curves are added to the roadmap, and the algorithm recurses again on the critical points. The recursive iterations terminate at $n = 2$. Canny showed that the resulting union of critical curves preserves the connectivity of C_{obs} (and hence, C_{free}). Some of the technical issues are: the algorithm works with a stratification of C_{obs} into manifolds; there are strong general position assumptions that are hard to meet; paths are actually considered along the boundary of C_{free}; and the method does not produce a parameterized solution path. For improvements to Canny's algorithm and many other important details, see [7.119].

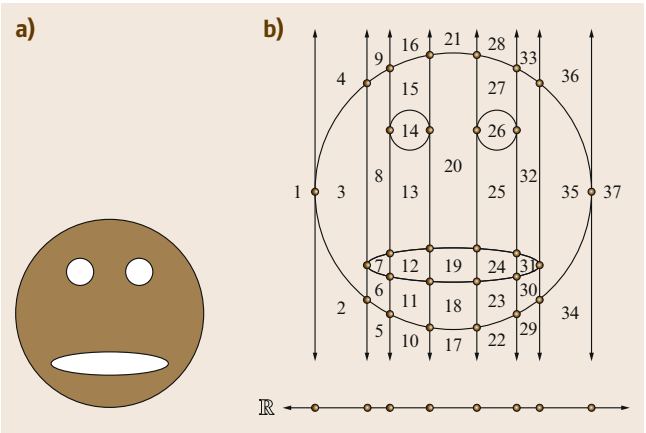


Fig. 7.18 **(a)** A face modeled with four algebraic primitives, and **(b)** a cylindrical algebraic decomposition of the face

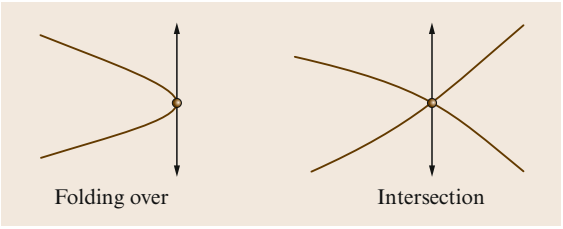


Fig. 7.19 Critical points occur either when the surface folds over in the vertical direction or when surfaces intersect

7.8 Conclusions and Further Reading

The brief survey given here hardly does justice to motion planning, which is a rich and active research field. For more details, we recommend consulting two recent textbooks [7.5, 7]. In addition, see the classic textbook of *Latombe* [7.6], the classic papers in [7.4], and the recent surveys in [7.2, 3]. Furthermore, consult the related handbook chapters that were indicated throughout this chapter.

Video-References

VIDEO 17 Powder transfer task using demonstration-guided motion planning
available from http://handbookofrobotics.org/view-chapter/07/videodetails/17

VIDEO 21 Simulation of a large crowd
available from http://handbookofrobotics.org/view-chapter/07/videodetails/21

VIDEO 22 Motion planning in multi-robot scenario
available from http://handbookofrobotics.org/view-chapter/07/videodetails/22

VIDEO 23 Alpha puzzle
available from http://handbookofrobotics.org/view-chapter/07/videodetails/23

VIDEO 24 Kinodynamic motion planning for a car-like robot
available from http://handbookofrobotics.org/view-chapter/07/videodetails/24

References

7.1 J.H. Reif: Complexity of the mover's problem and generalizations, IEEE Symp. Found. Comput. Sci. (1979) pp. 421–427

7.2 H.H. Gonzalez-Banos, D. Hsu, J.C. Latombe: Motion planning: Recent developments. In: *Automous Mobile Robots: Sensing, Control, Decision-Making and Applications*, ed. by S.S. Ge, F.L. Lewis (CRC, Boca Raton 2006)

7.3 S.R. Lindemann, S.M. LaValle: Current issues in sampling-based motion planning, 11th Int. Symp. Robotics Res. (Springer, Berlin, Heidelberg 2005) pp. 36–54

7.4 J.T. Schwartz, M. Sharir: A survey of motion planning and related geometric algorithms, Artif. Intell. J. **37**, 157–169 (1988)

7.5 H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, S. Thrun: *Principles of Robot Motion: Theory, Algorithms, and Implementations* (MIT, Cambridge 2005)

7.6 J.C. Latombe: *Robot Motion Planning* (Kluwer, Boston 1991)

7.7 S.M. LaValle: *Planning Algorithms* (Cambridge Univ. Press, Cambridge 2006)

7.8 S. Udupa: Collision detection and avoidance in computer controlled manipulators, Ph.D. Thesis (Dept. of Electical Engineering, California Institute of Technology, Pasadena 1977)

7.9 T. Lozano-Pérez: Spatial planning: A configuration space approach, IEEE Trans. Comput. C **32**(2), 108–120 (1983)

7.10 J.T. Schwartz, M. Sharir: On the piano movers' problem: III. Coordinating the motion of several independent bodies, Int. J. Robotics Res. **2**(3), 97–140 (1983)

7.11 J.T. Schwartz, M. Sharir: On the piano movers' problem: V. The case of a rod moving in three-dimensional space amidst polyhedral obstacles, Commun. Pure Appl. Math. **37**, 815–848 (1984)

7.12 J.F. Canny: *The Complexity of Robot Motion Planning* (MIT, Cambridge 1988)

7.13 D. Halperin, M. Sharir: A near-quadratic algorithm for planning the motion of a polygon in a polygonal environment, Discret. Comput. Geom. **16**, 121–134 (1996)

7.14 J.E. Hopcroft, J.T. Schwartz, M. Sharir: On the complexity of motion planning for multiple independent objects: PSPACE-hardness of the warehouseman's problem, Int. J. Robotics Res. **3**(4), 76–88 (1984)

7.15 J. Canny, J. Reif: New lower bound techniques for robot motion planning problems, IEEE Symp. Found. Comput. Sci. (1987) pp. 49–60

7.16 M.C. Lin, J.F. Canny: Efficient algorithms for incremental distance computation, IEEE Int. Conf. Robotics Autom. (1991) pp. 1008–1014

7.17 P. Jiménez, F. Thomas, C. Torras: Collision detection algorithms for motion planning. In: *Robot Motion Planning and Control*, ed. by J.P. Laumond (Springer, Berlin, Heidelberg 1998) pp. 1–53

7.18 M.C. Lin, D. Manocha: Collision and proximity queries. In: *Handbook of Discrete and Computational Geometry*, 2nd edn., ed. by J.E. Goodman, J. O'Rourke (Chapman Hall/CRC, Boca Raton 2004) pp. 787–807

7.19 L.E. Kavraki, P. Svestka, J.C. Latombe, M.H. Overmars: Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robotics Autom. **12**(4), 566–580 (1996)

7.20 N.M. Amato, O.B. Bayazit, L.K. Dale, C. Jones, D. Vallejo: OBPRM: An obstacle-based PRM for 3-D workspaces, Workshop Algorith. Found. Robotics (1998) pp. 155–168

7.21 V. Boor, M.H. Overmars, A.F. van der Stappen: The Gaussian sampling strategy for probabilistic roadmap planners, IEEE Int. Conf. Robotics Autom. (1999) pp. 1018–1023

7.22 C. Holleman, L.E. Kavraki: A framework for using the workspace medial axis in PRM planners, IEEE Int. Conf. Robotics Autom. (2000) pp. 1408–1413

7.23 J.M. Lien, S.L. Thomas, N.M. Amato: A general framework for sampling on the medial axis of the free space, IEEE Int. Conf. Robotics Autom. (2003)

7.24 S.M. LaValle, M.S. Branicky, S.R. Lindemann: On the relationship between classical grid search and probabilistic roadmaps, Int. J. Robotics Res. **23**(7/8), 673–692 (2004)

7.25 T. Siméon, J.-P. Laumond, C. Nissoux: Visibility based probabilistic roadmaps for motion planning, Adv. Robotics **14**(6), 477–493 (2000)

7.26 J. Barraquand, L. Kavraki, J.-C. Latombe, T.-Y. Li, R. Motwani, P. Raghavan: A random sampling

scheme for robot path planning, Proc. Int. Symp. Robotics Res. (1996) pp. 249–264
7.27 A. Ladd, L.E. Kavraki: Measure theoretic analysis of probabilistic path planning, IEEE Trans. Robotics Autom. **20**(2), 229–242 (2004)
7.28 R. Geraerts, M. Overmars: Sampling techniques for probabilistic roadmap planners, Int. Conf. Intell. Auton. Syst. (2004) pp. 600–609
7.29 D. Hsu, T. Jiang, J. Reif, Z. Sun: The bridge test for sampling narrow passages with probabilistic roadmap planners, IEEE Int. Conf. Robotics Autom. (2003) pp. 4420–4426
7.30 R. Bohlin, L. Kavraki: Path planning using lazy PRM, IEEE Int. Conf. Robotics Autom. (2000) pp. 521–528
7.31 B. Burns, O. Brock: Sampling-based motion planning using predictive models, IEEE Int. Conf. Robotics Autom. (2005) pp. 3120–3125
7.32 P. Isto: Constructing probabilistic roadmaps with powerful local planning and path optimization, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2002) pp. 2323–2328
7.33 P. Leven, S.A. Hutchinson: Using manipulability to bias sampling during the construction of probabilistic roadmaps, IEEE Trans. Robotics Autom. **19**(6), 1020–1026 (2003)
7.34 D. Nieuwenhuisen, M.H. Overmars: Useful cycles in probabilistic roadmap graphs, IEEE Int. Conf. Robotics Autom. (2004) pp. 446–452
7.35 S.M. LaValle, J.J. Kuffner: Rapidly-exploring random trees: progress and prospects. In: *Algorithmic and Computational Robotics: New Direction*, ed. by B.R. Donald, K.M. Lynch, D. Rus (A.K. Peters, Wellesley 2001) pp. 293–308
7.36 K.E. Bekris, B.Y. Chen, A. Ladd, E. Plaku, L.E. Kavraki: Multiple query probabilistic roadmap planning using single query primitives, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2003) pp. 656–661
7.37 M. Strandberg: Augmenting RRT-planners with local trees, IEEE Int. Conf. Robotics Autom. (2004) pp. 3258–3262
7.38 J.J. Kuffner, S.M. LaValle: *An Efficient Approach to Path Planning Using Balanced Bidirectional RRT Search*, Techn. Rep. CMU-RI-TR-05-34 Robotics Institute (Carnegie Mellon University, Pittsburgh 2005)
7.39 J. Bruce, M. Veloso: Real-time randomized path planning for robot navigation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2002) pp. 2383–2388
7.40 E. Frazzoli, M.A. Dahleh, E. Feron: Real-time motion planning for agile autonomous vehicles, AIAA J. Guid. Contr. **25**(1), 116–129 (2002)
7.41 M. Kallmann, M. Mataric: Motion planning using dynamic roadmaps, IEEE Int. Conf. Robotics Autom. (2004) pp. 4399–4404
7.42 A. Yershova, L. Jaillet, T. Simeon, S.M. LaValle: Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain, IEEE Int. Conf. Robotics Autom. (2005) pp. 3867–3872
7.43 D. Hsu, J.C. Latombe, R. Motwani: Path planning in expansive configuration spaces, Int. J. Comput. Geom. Appl. **4**, 495–512 (1999)
7.44 D. Hsu, R. Kindel, J.C. Latombe, S. Rock: Randomized kinodynamic motion planning with moving obstacles. In: *Algorithmic and Computational Robotics: New Directions*, ed. by B.R. Donald, K.M. Lynch, D. Rus (A.K. Peters, Wellesley 2001) pp. 247–264
7.45 G. Sánchez, J.-C. Latombe: A single-query bi-directional probabilistic roadmap planner with lazy collision checking, ISRR Int. Symp. Robotics Res. (2007) pp. 403–413
7.46 S. Carpin, G. Pillonetto: Robot motion planning using adaptive random walks, IEEE Int. Conf. Robotics Autom. (2003) pp. 3809–3814
7.47 A. Ladd, L.E. Kavraki: Fast exploration for robots with dynamics, Workshop Algorithm. Found. Robotics, Amsterdam (2004)
7.48 K.E. Bekris, L.E. Kavraki: Greedy but safe replanning under differential constraints, IEEE Int. Conf. Robotics Autom. (2007) pp. 704–710
7.49 C. O'Dunlaing, C.K. Yap: A retraction method for planning the motion of a disc, J. Algorithms **6**, 104–111 (1982)
7.50 D. Leven, M. Sharir: Planning a purely translational motion for a convex object in two-dimensional space using generalized Voronoi diagrams, Discret. Comput. Geom. **2**, 9–31 (1987)
7.51 M. Sharir: Algorithmic motion planning. In: *Handbook of Discrete and Computational Geometry*, 2nd edn., ed. by J.E. Goodman, J. O'Rourke (Chapman Hall/CRC, Boca Raton 2004) pp. 1037–1064
7.52 N.J. Nilsson: A mobile automaton: An application of artificial intelligence techniques, 1st Int. Conf. Artif. Intell. (1969) pp. 509–520
7.53 J. O'Rourke: Visibility. In: *Handbook of Discrete and Computational Geometry*, 2nd edn., ed. by J.E. Goodman, J. O'Rourke (Chapman Hall/CRC, Boca Raton 2004) pp. 643–663
7.54 B. Chazelle: Approximation and decomposition of shapes. In: *Algorithmic and Geometric Aspects of Robotics*, ed. by J.T. Schwartz, C.K. Yap (Lawrence Erlbaum, Hillsdale 1987) pp. 145–185
7.55 M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf: *Computational Geometry: Algorithms and Applications*, 2nd edn. (Springer, Berlin, Heidelberg 2000)
7.56 J.M. Keil: Polygon decomposition. In: *Handbook on Computational Geometry*, ed. by J.R. Sack, J. Urrutia (Elsevier, New York 2000)
7.57 J.T. Schwartz, M. Sharir: On the piano movers' problem: I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers, Commun. Pure Appl. Math. **36**, 345–398 (1983)
7.58 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robotics Res. **5**(1), 90–98 (1986)
7.59 J. Barraquand, J.-C. Latombe: Robot motion planning: A distributed representation approach, Int. J. Robotics Res. **10**(6), 628–649 (1991)
7.60 E. Rimon, D.E. Koditschek: Exact robot navigation using artificial potential fields, IEEE Trans. Robotics Autom. **8**(5), 501–518 (1992)

7.61 J.P. Laumond: Trajectories for mobile robots with kinematic and environment constraints, Int. Conf. Intell. Auton. Syst. (1986) pp. 346–354

7.62 J.P. Laumond, S. Sekhavat, F. Lamiraux: Guidelines in nonholonomic motion planning for mobile robots. In: *Robot Motion Planning and Control*, ed. by J.P. Laumond (Springer, Berlin, Heidelberg 1998) pp. 1–53

7.63 B.R. Donald, P.G. Xavier, J. Canny, J. Reif: Kinodynamic planning, Journal ACM **40**, 1048–1066 (1993)

7.64 C. O'Dunlaing: Motion planning with inertial constraints, Algorithmica **2**(4), 431–475 (1987)

7.65 J. Canny, A. Rege, J. Reif: An exact algorithm for kinodynamic planning in the plane, Discret. Comput. Geom. **6**, 461–484 (1991)

7.66 J. Go, T. Vu, J.J. Kuffner: Autonomous behaviors for interactive vehicle animations, SIGGRAPH/Eurographics Symp. Comput. Animat., Aire-la-Ville (2004) pp. 9–18

7.67 M. Pivtoraiko, A. Kelly: Generating near minimal spanning control sets for constrained motion planning in discrete state spaces, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2005) pp. 3231–3237

7.68 J. Hollerbach: *Dynamic scaling of manipulator trajectories*, Tech. Rep. 700 (MIT, Cambridge 1983)

7.69 K.G. Shin, N.D. McKay: Minimum-time control of robot manipulators with geometric path constraints, IEEE Trans. Autom. Control **30**(6), 531–541 (1985)

7.70 K.G. Shin, N.D. McKay: A dynamic programming approach to trajectory planning of robotic manipulators, IEEE Trans. Autom. Control **31**(6), 491–500 (1986)

7.71 S. Sastry: *Nonlinear Systems: Analysis, Stability, and Control* (Springer, Berlin, Heidelberg 1999)

7.72 D.J. Balkcom, M.T. Mason: Time optimal trajectories for bounded velocity differential drive vehicles, Int. J. Robotics Res. **21**(3), 199–217 (2002)

7.73 P. Souères, J.-D. Boissonnat: Optimal trajectories for nonholonomic mobile robots. In: *Robot Motion Planning and Control*, ed. by J.P. Laumond (Springer, Berlin, Heidelberg 1998) pp. 93–169

7.74 P. Svestka, M.H. Overmars: Coordinated motion planning for multiple car-like robots using probabilistic roadmaps, IEEE Int. Conf. Robotics Autom. (1995) pp. 1631–1636

7.75 S. Sekhavat, P. Svestka, J.-P. Laumond, M.H. Overmars: Multilevel path planning for nonholonomic robots using semiholonomic subsystems, Int. J. Robotics Res. **17**, 840–857 (1998)

7.76 P. Ferbach: A method of progressive constraints for nonholonomic motion planning, IEEE Int. Conf. Robotics Autom. (1996) pp. 2949–2955

7.77 S. Pancanti, L. Pallottino, D. Salvadorini, A. Bicchi: Motion planning through symbols and lattices, IEEE Int. Conf. Robotics Autom. (2004) pp. 3914–3919

7.78 J. Barraquand, J.-C. Latombe: Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles, Algorithmica **10**, 121–155 (1993)

7.79 S.M. LaValle, J.J. Kuffner: Randomized kinodynamic planning, IEEE Int. Conf. Robotics Autom. (1999) pp. 473–479

7.80 A.M. Ladd, L.E. Kavraki: Motion planning in the presence of drift underactuation and discrete system changes, Robotics Sci. Syst. I, Cambridge (2005) pp. 233–241

7.81 J.-P. Merlet: *Parallel Robots* (Kluwer, Boston 2000)

7.82 D. Cox, J. Little, D. O'Shea: *Ideals, Varieties, and Algorithms* (Springer, Berlin, Heidelberg 1992)

7.83 R.J. Milgram, J.C. Trinkle: The geometry of configuration spaces for closed chains in two and three dimensions, Homol. Homot. Appl. **6**(1), 237–267 (2004)

7.84 J. Yakey, S.M. LaValle, L.E. Kavraki: Randomized path planning for linkages with closed kinematic chains, IEEE Trans. Robotics Autom. **17**(6), 951–958 (2001)

7.85 L. Han, N.M. Amato: A kinematics-based probabilistic roadmap method for closed chain systems. In: *Algorithmic and Computational Robotics: New Directions*, ed. by B.R. Donald, K.M. Lynch, D. Rus (A.K. Peters, Wellesley 2001) pp. 233–246

7.86 J. Cortés: Motion Planning Algorithms for General Closed-Chain Mechanisms, Ph.D. Thesis (Institut National Polytechnique do Toulouse, Toulouse 2003)

7.87 R. Alami, J.-P. Laumond, T. Siméon: Two manipulation planning algorithms. In: *Algorithms for Robotic Motion and Manipulation*, ed. by J.P. Laumond, M. Overmars (A.K. Peters, Wellesley 1997)

7.88 L.E. Kavraki, M. Kolountzakis: Partitioning a planar assembly into two connected parts is NP-complete, Inform. Process. Lett. **55**(3), 159–165 (1995)

7.89 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT, Cambridge 2001)

7.90 K. Sutner, W. Maass: Motion planning among time dependent obstacles, Acta Inform. **26**, 93–122 (1988)

7.91 J.H. Reif, M. Sharir: Motion planning in the presence of moving obstacles, Journal ACM **41**, 764–790 (1994)

7.92 M.A. Erdmann, T. Lozano-Pérez: On multiple moving objects, Algorithmica **2**, 477–521 (1987)

7.93 J. van den Berg, M. Overmars: Prioritized motion planning for multiple robots, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2005) pp. 2217–2222

7.94 T. Siméon, S. Leroy, J.-P. Laumond: Path coordination for multiple mobile robots: A resolution complete algorithm, IEEE Trans. Robotics Autom. **18**(1), 42–49 (2002)

7.95 R. Ghrist, J.M. O'Kane, S.M. LaValle: Pareto optimal coordination on roadmaps, Workshop Algorithm. Found. Robotics (2004) pp. 185–200

7.96 S.M. LaValle, S.A. Hutchinson: Optimal motion planning for multiple robots having independent goals, IEEE Trans. Robotics Autom. **14**(6), 912–925 (1998)

7.97 P.R. Kumar, P. Varaiya: *Stochastic Systems* (Prentice Hall, Englewood Cliffs 1986)

7.98 S.M. LaValle: *Sensing and Filtering: A Fresh Perspective Based on Preimages and Information Spaces*, Foundations and Trends in Robotics (Now Publ., Delft 2012)

7.99 R. Alterovitz, N. Simeon, K. Goldberg: The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty, Robotics Sci. Syst. **3**, 233–241 (2007)

7.100 H. Kurniawati, D. Hsu, W.S. Lee: SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces, Robotics Sci. Syst. (2008)

7.101 J. Pineau, G. Gordon, S. Thrun: Point-based value iteration, Int. Joint Conf. Artif. Intell. (2003) pp. 1025–1032

7.102 R. Platt, R. Tedrake, T. Lozano-Perez, L.P. Kaelbling: Belief space planning assuming maximum likelihood observations, Robotics Sci. Syst. (2010)

7.103 N. Roy, G. Gordon: Exponential family PCA for belief compression in POMDPs, Adv. Neural Inform. Process. Syst. (2003)

7.104 R. He, S. Prentice, N. Roy: Planning in information space for a quadrotor helicopter in a GPS-denied environment, IEEE Int. Conf. Robotics Autom. (2008)

7.105 S. Koenig, M. Likhachev: *D** lite, AAAI Nat. Conf. Artif. Intell. (2002) pp. 476–483

7.106 A. Stentz: Optimal and efficient path planning for partially-known environments, IEEE Int. Conf. Robotics Autom. (1994) pp. 3310–3317

7.107 J. Hershberger, S. Suri: Efficient Computation of Euclidean shortest paths in the plane, IEEE Symp. Found. Comp. Sci. (1995) pp. 508–517

7.108 J.S.B. Mitchell: Shortest paths among obstacles in the plane, Int. J. Comput. Geom. Applic. **6**(3), 309 (1996)

7.109 J. Choi, J. Sellen, C.K. Yap: Precision-sensitive Euclidean shortest path in 3-space, ACM Symp. Comput. Geo. (1995) pp. 350–359

7.110 C.H. Papadimitriou: An algorithm for shortest-path planning in three dimensions, Inform. Process. Lett. **20**(5), 259 (1985)

7.111 D. Yershov, S.M. LaValle: Simplicial Dijkstra and A* algorithms for optimal feedback planning, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2011)

7.112 S. Karaman, E. Frazzoli: Sampling-based algorithms for optimal motion planning, Int. J. Robotics Res. **30**(7), 846 (2011)

7.113 J.D. Marble, K.E. Bekris: Towards small asymptotically near-optimal roadmaps, IEEE Int. Conf. Robotics Autom. (2012)

7.114 W.M. Boothby: *An Introduction to Differentiable Manifolds and Riemannian Geometry*, 2nd edn. (Academic, New York 2003)

7.115 A. Hatcher: *Algebraic Topology* (Cambridge Univ. Press, Cambridge 2002)

7.116 G.S. Chirikjian, A.B. Kyatkin: *Engineering Applications of Noncommutative Harmonic Analysis* (CRC, Boca Raton 2001)

7.117 J. Arvo: Fast random rotation matrices. In: *Graphics Gems III*, ed. by D. Kirk (Academic, New York 1992) pp. 117–120

7.118 H. Niederreiter: *Random Number Generation and Quasi-Monte-Carlo Methods* (Society for Industrial and Applied Mathematics, Philadelphia 1992)

7.119 S. Basu, R. Pollack, M.-F. Roy: *Algorithms in Real Algebraic Geometry* (Springer, Berlin, Heidelberg 2003)

7.120 B. Mishra: Computational real algebraic geometry. In: *Handbook of Discrete and Computational Geometry*, ed. by J.E. Goodman, J. O'Rourke (CRC, Boca Raton 1997) pp. 537–556

7.121 J.T. Schwartz, M. Sharir: On the piano movers' problem: II. General techniques for computing topological properties of algebraic manifolds, Commun. Pure Appl. Math. **36**, 345–398 (1983)

7.122 B.R. Donald: A search algorithm for motion planning with six degrees of freedom, Artif. Intell. J. **31**, 295–353 (1987)

7.123 D.S. Arnon: Geometric reasoning with logic and algebra, Artif. Intell. J. **37**(1-3), 37–60 (1988)

7.124 G.E. Collins: Quantifier elimination by cylindrical algebraic decomposition–twenty years of progress. In: *Quantifier Elimination and Cylindrical Algebraic Decomposition*, ed. by B.F. Caviness, J.R. Johnson (Springer, Berlin, Heidelberg 1998) pp. 8–23

Multimedia Contents

8. Motion Control

Wan Kyun Chung, Li-Chen Fu, Torsten Kröger

This chapter will focus on the motion control of robotic rigid manipulators. In other words, this chapter does not treat the motion control of mobile robots, flexible manipulators, and manipulators with elastic joints. The main challenge in the motion control problem of rigid manipulators is the complexity of their dynamics and uncertainties. The former results from nonlinearity and coupling in the robot manipulators. The latter is twofold: structured and unstructured. Structured uncertainty means imprecise knowledge of the dynamic parameters and will be touched upon in this chapter, whereas unstructured uncertainty results from joint and link flexibility, actuator dynamics, friction, sensor noise, and unknown environment dynamics, and will be treated in other chapters.

In this chapter, we begin with an introduction to motion control of robot manipulators from a fundamental viewpoint, followed by a survey and brief review of the relevant advanced materials. Specifically, the dynamic model and useful properties of robot manipulators are recalled in Sect. 8.1. The joint and operational space control approaches, two different viewpoints on control of robot manipulators, are compared in Sect. 8.2. Independent joint control and proportional–integral–derivative (PID) control, widely adopted in the field of industrial robots, are presented in Sects. 8.3 and 8.4, respectively. Tracking control, based on feedback linearization, is introduced in Sect. 8.5. The computed-torque control and its variants are described in Sect. 8.6. Adaptive control is introduced in Sect. 8.7 to solve the problem of structural uncertainty, whereas the optimality and robustness issues are covered in Sect. 8.8. To compute suitable set point signals as input values for these motion controllers, Sect. 8.9 introduces reference trajectory planning concepts. Since most controllers of robot manipulators are implemented by using microprocessors, the issues of digital implementation are discussed in Sect. 8.10. Finally, learning control, one popular approach to intelligent control, is illustrated in Sect. 8.11.

8.1 **Introduction to Motion Control** 164
8.1.1 Dynamical Model 164
8.1.2 Control Tasks 165
8.1.3 Summary 165
8.2 **Joint Space Versus Operational Space Control** 166
8.2.1 Joint Space Control 166
8.2.2 Operational Space Control 166
8.3 **Independent-Joint Control** 167
8.3.1 Controller Design Based on the Single-Joint Model 167
8.3.2 Controller Design Based on the Multijoint Model 169
8.3.3 Summary 169
8.4 **PID Control** 169
8.4.1 PD Control for Regulation 169
8.4.2 PID Control for Regulation 170
8.4.3 PID Gain Tuning 170
8.4.4 Automatic Tuning 171
8.5 **Tracking Control** 172
8.5.1 Inverse Dynamics Control 172
8.5.2 Feedback Linearization 172
8.5.3 Passivity-Based Control 173
8.5.4 Summary 174
8.6 **Computed-Torque Control** 174
8.6.1 Computed-Torque Control 174
8.6.2 Computed-Torque-Like Control 175
8.6.3 Summary 177
8.7 **Adaptive Control** 177
8.7.1 Adaptive Computed-Torque Control 177
8.7.2 Adaptive Inertia-Related Control 178
8.7.3 Adaptive Control Based on Passivity 179

8.7.4 Adaptive Control with Desired Compensation 180
8.7.5 Summary 180
8.8 **Optimal and Robust Control** 181
8.8.1 Quadratic Optimal Control 181
8.8.2 Nonlinear $\mathcal{H}_\infty$ Control 181
8.8.3 Passivity-Based Design of Nonlinear $\mathcal{H}_\infty$ Control 182
8.8.4 A Solution to Inverse Nonlinear $\mathcal{H}_\infty$ Control 183
8.9 **Trajectory Generation and Planning** 183
8.9.1 Geometric Paths and Trajectories 183
8.9.2 Joint Space and Operational Space Trajectories 184
8.9.3 Trajectory Representations 184
8.9.4 Trajectory Planning Algorithms ... 186
8.10 **Digital Implementation** 187
8.10.1 *Z*-Transform for Motion Control Implementation 188
8.10.2 Digital Control for Coding 189
8.11 **Learning Control** 190
8.11.1 Pure P-Type Learning Control 190
8.11.2 P-Type Learning Control with a Forgetting Factor 190
8.11.3 Summary 191
Video-References 191
References 191

8.1 Introduction to Motion Control

The dynamical model of robot manipulators will be recalled in this section. Furthermore, important properties of this dynamical model, which is useful in controller design, will then be addressed. Finally, different control tasks of the robot manipulators will be defined.

8.1.1 Dynamical Model

For motion control, the dynamical model of rigid robot manipulators is conveniently described by Lagrange dynamics. Let the robot manipulator have n links and let the $(n\times 1)$-vector $\boldsymbol{q}$ of joint variables be $\boldsymbol{q} = [q_1, \ldots, q_n]^{\mathrm{T}}$. The dynamic model of the robot manipulator is then described by Lagrange's equation [8.1–6]

$$\mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) = \boldsymbol{\tau} \ , \tag{8.1}$$

where $\mathbf{H}(\boldsymbol{q})$ is the $(n\times n)$ inertia matrix, $\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ is the $(n\times 1)$-vector of Coriolis and centrifugal forces, $\boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q})$ is the $(n\times 1)$-vector of gravity force, and $\boldsymbol{\tau}$ is the $(n\times 1)$-vector of joint control inputs to be designed. Friction and disturbance input have been neglected here.

Remark 8.1
Other contributions to the dynamic description of the robot manipulators may include the dynamics of the actuators, joint and link flexibility, friction, noise, and disturbances. Without loss of generality, the case of the rigid robot manipulators is stressed here.

The control schemes that we will introduce in this chapter are based on some important properties of the dynamical model of robot manipulators. Before giving a detailed introduction to these different schemes, let us first give a list of those properties.

Property 8.1
The inertia matrix is a symmetric positive-definite matrix, which can be expressed

$$\lambda_{\mathrm{h}}\mathbf{I}_n \le \mathbf{H}(\boldsymbol{q}) \le \lambda_{\mathrm{H}}\mathbf{I}_n \ , \tag{8.2}$$

where λ_{h} and λ_{H} denote positive constants.

Property 8.2
The matrix $\mathbf{N}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \dot{\mathbf{H}}(\boldsymbol{q}) - 2\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is skew-symmetric for a particular choice of $\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ (which is always possible), i. e.,

$$\boldsymbol{z}^{\mathrm{T}}\mathbf{N}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{z} = 0 \tag{8.3}$$

for any $(n\times 1)$-vector $\boldsymbol{z}$.

Property 8.3
The $(n\times n)$-matrix $\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ satisfies

$$\| \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \| \le c_{\mathrm{o}} \| \dot{\boldsymbol{q}} \| \tag{8.4}$$

for some bounded constant c_{o}.

Property 8.4
The gravity force/torque vector satisfies

$$\| \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) \| \le g_{\mathrm{o}} \tag{8.5}$$

for some bounded constant g_{o}.

Property 8.5
The equation of motion is linear in the inertia parameters. In other words, there is a $(r\times 1)$ constant

vector $\boldsymbol{a}$ and an $(n \times r)$ regressor matrix $\mathbf{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ such that

$$\mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) = \mathbf{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\boldsymbol{a} \,. \tag{8.6}$$

The vector $\boldsymbol{a}$ is comprised of link masses, moments of inertia, and the link, in various combinations.

Property 8.6

The mapping $\boldsymbol{\tau} \rightarrow \dot{\boldsymbol{q}}$ is passive; i. e., there exists $\alpha \geq 0$ such that

$$\int_0^t \dot{\boldsymbol{q}}^{\mathrm{T}}(\beta)\boldsymbol{\tau}(\beta)\,\mathrm{d}\beta \geq -\alpha \,, \quad \forall t < \infty \,. \tag{8.7}$$

Remarks 8.1

- Properties 8.3 and 8.4 are very useful since they allow us to establish upper bounds on the nonlinear terms in the dynamical model. As we will see further, several control schemes require knowledge of such upper bounds.
- In Property 8.5, the parameter vector $\boldsymbol{a}$ is comprised of several variables in various combinations. The dimensionality of the parameter space is not unique, and the search over the parameter space is an important problem.
- In this section, we assume that *the robot manipulator is fully actuated* and this indicates that there is an independent control input for each degree of freedom (DOF). In contrast, the robot manipulators with joint or link flexibility are no longer fully actuated and the control problems are more difficult in general.

8.1.2 Control Tasks

It is instructive for comparative purposes to classify control objectives into the following two classes:

- *Trajectory tracking* is aimed at following a time-varying joint reference trajectory specified within the manipulator workspace. In general, this desired trajectory is assumed to comply with the actuators' capacity. In other words, the joint velocity and acceleration associated with the desired trajectory should not violate, respectively, the velocity and acceleration limit of the manipulator. In practice, the capacity of actuators is set by torque limits, which result in bounds on the acceleration that are complex and state dependent.
- *Regulation* sometimes is also called point-to-point control. A fixed configuration in the joint space is specified; the objective is to bring to and keep the joint variable at the desired position in spite of torque disturbances and independently of the initial conditions. The behavior of transients and overshooting, are in general, not guaranteed.

The selection of the controller may depend on the type of task to be performed. For example, tasks only requiring the manipulator to move from one position to another without requiring significant precision during the motion between these two points can be solved by regulators, whereas such as welding, painting, and so on, require tracking controllers.

Remarks 8.2

- The regulation problem may be seen as a special case of the tracking problem (for which the desired joint velocity and acceleration are zero).
- The task specification above is given in the joint space and results in joint space control, which is the main content of this chapter. Sometimes, the task specification of the robot manipulators in terms of the desired trajectory of the end-effector (e.g., control with eye-in-hand) is carried out in the task space and gives rise to the operational space control, which will be introduced in Sect. 8.2.

8.1.3 Summary

In this section, we introduced the dynamical model of the robot manipulators and important properties of this dynamical model. Finally, we defined different control tasks of the robot manipulators.

8.2 Joint Space Versus Operational Space Control

In a motion control problem, the manipulator moves to a position to pick up an object, transports that object to another location, and deposits it. Such a task is an integral part of any higher-level manipulation tasks such as painting or spot-welding.

Tasks are usually specified in the task space in terms of a desired trajectory of the end-effector, while control actions are performed in the joint space to achieve the desired goals. This fact naturally leads to two kinds of general control methods, namely *joint space control* and *operational space control (task space control)* schemes.

8.2.1 Joint Space Control

The main goal of the joint space control is to design a feedback controller such that the joint coordinates $\boldsymbol{q}(t) \in R^n$ track the desired motion $\boldsymbol{q}_{\mathrm{d}}(t)$ as closely as possible. To this end, consider the equations of motion (8.1) of an n-DOF manipulator expressed in the joint space [8.2, 4]. In this case, the control of robot manipulators is naturally achieved in the joint space, since the control inputs are the joint torques. Nevertheless, the user specifies a motion in terms of end-effector coordinates, and thus it is necessary to understand the following strategy.

Figure 8.1 shows the basic outline of the joint space control methods. Firstly, the desired motion, which is described in terms of end-effector coordinates, is converted to a corresponding joint trajectory using the inverse kinematics of the manipulator. Then the feedback controller determines the joint torque necessary to move the manipulator along the desired trajectory specified in joint coordinates starting from measurements of the current joint states [8.1, 4, 7, 8].

Since it is always assumed that the desired task is given in terms of the time sequence of the joint motion, joint space control schemes are quite adequate in situations where manipulator tasks can be accurately preplanned and little or no online trajectory adjustments are necessary [8.1, 4, 7, 9]. Typically, inverse kinematics is performed for some intermediate task points, and the joint trajectory is interpolated using the intermediate joint solutions. Although the command trajectory consists of straight-line motions in end-effector coordinates between interpolation points, the resulting joint motion consists of curvilinear segments that match the desired end-effector trajectory at the interpolation points.

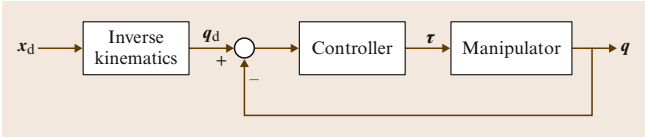


Fig. 8.1 Generic concept of joint space control

In fact, the joint space control includes simple proportional–derivative (PD) control, PID control, inverse dynamic control, Lyapunov-based control, and passivity-based control, as explained in the following sections.

8.2.2 Operational Space Control

In more complicated and less certain environments, end-effector motion may be subject to online modifications in order to accommodate unexpected events or to respond to sensor inputs. There are a variety of tasks in manufacturing where these type of control problem arise. In particular, it is essential when controlling the interaction between the manipulator and environment is of concern.

Since the desired task is often specified in the operational space and requires precise control of the end-effector motion, joint space control schemes are not suitable in these situations. This motivated a different approach, which can develop control schemes directly based on the dynamics expressed in the operational space [8.10, 11].

Let us suppose that the Jacobian matrix, denoted by $\mathbf{J}(\boldsymbol{q}) \in R^{n\times n}$, transforms the joint velocity ($\dot{\boldsymbol{q}} \in R^n$) to the task velocity ($\dot{\boldsymbol{x}} \in R^n$) according to

$$\dot{\boldsymbol{x}} = \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \,. \tag{8.8}$$

Furthermore, assume that it is invertible. Then, the operational space dynamics is expressed as follows

$$\boldsymbol{f}_{\mathrm{c}} = \boldsymbol{\Lambda}(\boldsymbol{q})\ddot{\boldsymbol{x}} + \boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{\eta}(\boldsymbol{q}) \,, \tag{8.9}$$

where $\boldsymbol{f}_{\mathrm{c}} \in R^n$ denotes the command forces in the operational space; the *pseudo-inertia matrix* is defined by

$$\boldsymbol{\Lambda}(\boldsymbol{q}) = \mathbf{J}^{-\mathrm{T}}(\boldsymbol{q})\mathbf{H}(\boldsymbol{q})\mathbf{J}^{-1}(\boldsymbol{q}) \,, \tag{8.10}$$

and $\boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and $\boldsymbol{\eta}(\boldsymbol{q})$ are given by

$$\begin{aligned}\boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \mathbf{J}^{-\mathrm{T}}(\boldsymbol{q})\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\mathbf{J}^{-1}(\boldsymbol{q}) \\ &\quad - \boldsymbol{\Lambda}(\boldsymbol{q})\mathbf{J}(\boldsymbol{q})\mathbf{J}^{-1}(\boldsymbol{q}) \,, \\ \boldsymbol{\eta}(\boldsymbol{q}) &= \mathbf{J}^{-\mathrm{T}}(\boldsymbol{q})\boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) \,.\end{aligned}$$

The task space variables are usually reconstructed from the joint space variables, via the kinematic mappings. In fact, it is quite rare to have sensors to directly measure end-effector positions and velocities. Also, it

is worth remarking that an analytical Jacobian is utilized since the control schemes operate directly on task space quantities, i. e., the end-effector position and orientation.

The main goal of the operational space control is to design a feedback controller that allows execution of an end-effector motion $\boldsymbol{x}(t) \in R^n$ that tracks the desired end-effector motion $\boldsymbol{x}_d(t)$ as closely as possible. To this end, consider the equations of motion (8.9) of the manipulator expressed in the operational space. For this case, Fig. 8.2 shows a schematic diagram of the operational space control methods. There are several advantages to such an approach because operational space controllers employ a feedback loop that directly minimizes task errors. Inverse kinematics need not be calculated explicitly, since the control algorithm embeds the velocity-level forward kinematics (8.8), as shown in the figure. Now, motion between points can be a straight-line segment in the task space.

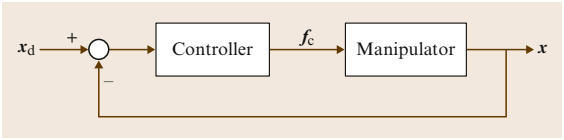


Fig. 8.2 Basic concept of operational space control

8.3 Independent-Joint Control

By independent-joint control (i. e., decentralized control), we mean that the control inputs of each joint only depends on the measurement of the corresponding joint displacement and velocity. Due to its simple structure, this kind of control schemes offers many advantages. For example, by using independent-joint control, communication among different joints is saved. Moreover, since the computational load of controllers may be reduced, only low-cost hardware is required in actual implementations. Finally, independent-joint control has the feature of scalability, since the controllers on all the joints have the same formulation. In this section, two kinds of design of independent-joint control will be introduced: one focused on the dynamical model of each joint (i. e., based on the single-joint model) and the other based on the analysis of the overall dynamical model (i. e., the multijoint model) of robot manipulators.

8.3.1 Controller Design Based on the Single-Joint Model

The simplest independent-joint control strategy is to control each joint axis as a single-input single-output (SISO) system. Coupling effects among joints due to varying configuration during motion are treated as disturbance inputs. Without loss of generality, the actuator is taken as a rotary electric direct-current (DC) motor. Hence, the block diagram of the control scheme of joint i can be represented in the domain of the complex variables as shown in Fig. 8.3. In this scheme, θ is the angular variable of the motor, J is the effective inertia viewed from the motor side, R_a is the armature resistance (auto-inductance being neglected), and k_t and k_v are, respectively, the torque and motor constants. Furthermore, G_v denotes the voltage gain of the power amplifier so that the reference input is the input voltage V_c of the amplifier instead of the armature voltage V_a. It has also been assumed that $F_m \ll k_v k_t / R_a$, i. e., the mechanical (viscous) friction coefficient has been neglected with respect to the electrical coefficient. Now the input–output transfer function of the motor can be written as

$$M(s) = \frac{k_m}{s(1 + sT_m)} , \tag{8.11}$$

where $k_m = G_v / k_v$ and $T_m = R_a J / k_v k_t$ are, respectively, the voltage-to-velocity gain and the time constant of the motor.

To guide selection of the control structure, start by noticing that an effective rejection of the disturbance d on the output θ is ensured by

1. A large value of the amplifier before the point of intervention of the disturbance
2. The presence of an integral action in the controller so as to cancel the effect of the gravitational component on the output at the steady state (i. e., constant θ).

In this case, as shown in Fig. 8.4, the types of control action with position and velocity feedback are

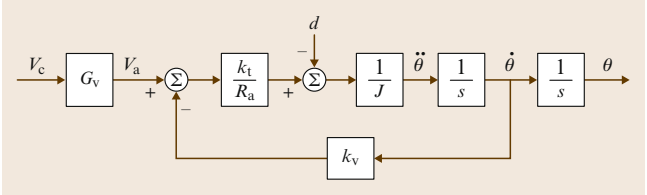


Fig. 8.3 Block scheme of a joint-driven system (after [8.4])

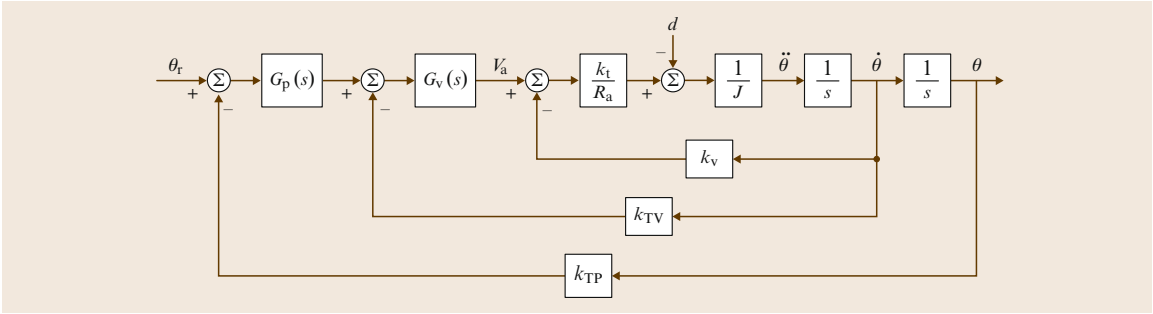


Fig. 8.4 Block scheme of position and velocity feedback (after [8.4])

characterized by [8.4]

$$G_p(s) = K_P\,, \quad G_v(s) = K_V \frac{1 + sT_V}{s}\,, \tag{8.12}$$

where $G_p(s)$ and $G_v(s)$ correspond to the position and velocity control actions, respectively. It is worth noting that the inner control action $G_v(s)$ is in a form of propositional–integral (PI) control to yield zero error in the steady state under a constant disturbance d. Furthermore, k_{TP} and k_{TV} are both transducer constants, and the amplifier gain K_V has been embedded in the gain of the inner controller. From the scheme of Fig. 8.4, the transfer function of the forward path is

$$P(s) = \frac{k_m K_P K_V (1 + sT_V)}{s(1 + sT_m)}\,, \tag{8.13}$$

while that of the return path is

$$H(s) = k_{TP}\left(1 + s\frac{k_{TV}}{K_P k_{TP}}\right)\,. \tag{8.14}$$

The zero of the controller at $s = -1/T_V$ can be chosen so as to cancel the effects of the real pole of the motor at $s = -1/T_m$. Then, by setting $T_V = T_m$, the poles of the closed-loop system move on the root locus as a function of the loop gain, $k_m K_V k_{TV}$. By increasing the position feedback gain K_P, it is possible to confine the closed-loop poles to a region of the complex plane with large absolute real part. Then, the actual location can be established by a suitable choice of K_V.

The closed-loop input–output transfer function is

$$\frac{\Theta(s)}{\Theta_r(s)} = \frac{\dfrac{1}{k_{TP}}}{1 + \dfrac{sk_{TP}}{K_P k_{TP}} + \dfrac{s^2}{k_m K_P k_{TP} K_V}}\,, \tag{8.15}$$

which can be compared with the typical transfer function of a second-order system

$$W(s) = \frac{\dfrac{1}{k_{TP}}}{1 + \dfrac{2\zeta s}{\omega_n} + \dfrac{s^2}{\omega_n^2}}\,. \tag{8.16}$$

It can be recognized that, with a suitable choice of the gains, it is possible to obtain any value of natural frequency ω_n and damping ratio ζ. Hence, if ω_n and ζ are given as design specifications, the following relations can be found

$$K_V k_{TV} = \frac{2\zeta\omega_n}{k_m} \quad \text{and} \quad K_P k_{TP} K_V = \frac{\omega_n^2}{k_m}\,. \tag{8.17}$$

For given transducer constants k_{TP} and k_{TV}, K_V and K_P will be chosen to satisfy the two equations above, respectively. On the other hand, the closed-loop disturbance/output function is

$$\frac{\Theta(s)}{D(s)} = \frac{\dfrac{sR_a}{k_t K_P K_{TP} K_V (1 + sT_m)}}{1 + \dfrac{sk_{TV}}{K_P k_{TP}} + \dfrac{s^2}{k_m K_P k_{TP} K_V}}\,, \tag{8.18}$$

which shows that the disturbance rejection factor is $X_R(s) = K_P k_{TP} K_V$ and is fixed, provided that K_V and K_P have been chosen via the approach above. Concerning the disturbance dynamics, the zero at the origin introduced by the PI, a real pole at $s = -1/T_m$, and the pair of complex poles with real part $-\zeta\omega_n$ should be kept in mind. In this case, an estimate T_R of the output recovery time needed by the control system to recover from the effect of a disturbance on the joint position can be evaluated by analyzing models of the transfer function above. Such an estimate can reasonably be expressed as $T_R = \max\{T_m, 1/\zeta\omega\}$.

8.3.2 Controller Design Based on the Multijoint Model

In recent years, independent-joint control schemes based on the complete dynamic model of the robot manipulators (i. e., a multijoint model) have been proposed. For example, following the approach of computed-torque-like control, [8.12] dealt with the regulation task for horizontal motions, and [8.13] and [8.14] handled the tracking task for arbitrary smooth trajectories. Since the overall dynamic model is considered, the effects of coupling among joints are handled. These schemes will be introduced in detail in Sect. 8.6.

8.3.3 Summary

In this section, we have presented two independent-joint control schemes: one is based on the single-joint model and the other is based on the multijoint model. The former focuses on the dynamics of a single joint and regards the interaction among joints as a disturbance. This control scheme is simple but may not be suitable for high-speed tracking. Hence, we introduce the latter, which considers the overall dynamical model of robot manipulators such that the interaction among joints can be handled.

Further Reading

There are different types of feedback applied in the independent-joint control based on the single-joint model (such as pure position feedback or position, velocity, and acceleration feedback). A complete discussion is given in [8.4]. When the joint control servos are required to track reference trajectories with high speeds and accelerations, the tracking capabilities of the above schemes are inevitably degraded. A possible remedy is to adopt decentralized feedforward compensation to reduce tracking errors [8.4, 5].

8.4 PID Control

Traditionally, control design in robot manipulators can be understood as the simple fact of tuning of a PD or PID compensator at the level of each motor driving the manipulator joints [8.1]. Fundamentally, a PD controller is a position and velocity feedback that has good closed-loop properties when applied to a double integrator system.

The PID control has a long history since *Ziegler* and *Nichols*' PID tuning rules were published in 1942 [8.15]. Actually, the strong point of PID control lies in its *simplicity* and clear physical meaning. Simple control is preferable to complex control, at least in industry, if the performance enhancement obtained by using complex control is not significant enough. The physical meanings of PID control [8.16] are as follows:

- *P-control* means the present effort making a present state into desired state
- *I-control* means the accumulated effort using the experience information of previous states
- *D-control* means the predictive effort reflecting the information about trends in future states.

8.4.1 PD Control for Regulation

A simple design method for manipulator control is to utilize a linear control scheme based on the linearization of the system about an operating point. An example of this method is a PD control with a gravity compensation scheme [8.17, 18]. Gravity compensation acts as a bias correction, compensating only for the amount of forces that create overshooting and an asymmetric transient behavior. Formally, it has the following form

$$\boldsymbol{\tau} = \mathbf{K}_{\mathrm{P}}(\boldsymbol{q}_{\mathrm{d}} - \boldsymbol{q}) - \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) \,, \tag{8.19}$$

where $\mathbf{K}_{\mathrm{P}}$ and $\mathbf{K}_{\mathrm{V}} \in R^{n\times n}$ are positive-definite gain matrices. This controller is very useful for set-point regulation, i. e., $\boldsymbol{q}_{\mathrm{d}} =$ constant [8.7, 18]. When this controller is applied to (8.1), the closed-loop equation becomes

$$\mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{q}} - \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}} = \mathbf{0} \,, \tag{8.20}$$

where $\boldsymbol{e}_{\mathrm{q}} = \boldsymbol{q}_{\mathrm{d}} - \boldsymbol{q}$, and the equilibrium point is $\boldsymbol{y} = [\boldsymbol{e}_{\mathrm{q}}^{\mathrm{T}}, \dot{\boldsymbol{q}}^{\mathrm{T}}]^{\mathrm{T}} = \mathbf{0}$. Now, the stability achieved by PD control with gravity compensation can be analyzed according to the closed-loop dynamic (8.20). Consider the positive-definite function

$$V = \frac{1}{2}\dot{\boldsymbol{q}}^{\mathrm{T}}\mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{q}} + \frac{1}{2}\boldsymbol{e}_{\mathrm{q}}^{\mathrm{T}}\mathbf{K}_{\mathrm{V}}\boldsymbol{e}_{\mathrm{q}} \,.$$

Then, the derivative of function becomes negative semidefinite for any value of $\dot{\boldsymbol{q}}$ by using Property 8.2 in Sect. 8.1, i. e.,

$$\dot{V} = -\dot{\boldsymbol{q}}^{\mathrm{T}}\mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{q}} \le -\lambda_{\min}(\mathbf{K}_{\mathrm{V}})\|\dot{\boldsymbol{q}}\|^2 \,, \tag{8.21}$$

where $\lambda_{\min}(\mathbf{K}_{\mathrm{V}})$ means the smallest eigenvalue of $\mathbf{K}_{\mathrm{V}}$. By invoking the Lyapunov stability theory and LaSalle's theorem [8.1], it can be shown that the regulation error will converge asymptotically to zero, while

their high-order derivatives remain bounded. This controller requires knowledge of the gravity components (structure and parameters), though it is simple.

Now, consider simple PD control without gravity compensation

$$\boldsymbol{\tau} = \mathbf{K}_\mathrm{P}(\boldsymbol{q}_\mathrm{d} - \boldsymbol{q}) - \mathbf{K}_\mathrm{V}\dot{\boldsymbol{q}} \,, \tag{8.22}$$

then the closed-loop dynamic equation becomes

$$\mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_\mathrm{g}(\boldsymbol{q}) + \mathbf{K}_\mathrm{V}\dot{\boldsymbol{q}} - \mathbf{K}_\mathrm{P}\boldsymbol{e}_\mathrm{q} = \mathbf{0} \,. \tag{8.23}$$

Consider the positive definite function

$$V = \frac{1}{2}\dot{\boldsymbol{q}}^\mathrm{T}\mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{q}} + \frac{1}{2}\boldsymbol{e}_\mathrm{q}^\mathrm{T}\mathbf{K}_\mathrm{V}\boldsymbol{e}_\mathrm{q} + U(\boldsymbol{q}) + U_0 \,,$$

where $U(\boldsymbol{q})$ denotes the potential energy with the relation of $\partial U(\boldsymbol{q})/\partial \boldsymbol{q} = \boldsymbol{\tau}_\mathrm{g}(\boldsymbol{q})$ and U_0 is a suitable constant. Taking time derivative of V along the closed-loop dynamics (8.23) gives the same result (8.21) with previous one using gravity compensation. In this case, the control system must be stable in the sense of Lyapunov, but it can not conclude that the regulation error will converge to zero by LaSalle's theorem [8.1]. Actually, the system precision (the size of the regulation error vector) will depend on the size of the gain matrix $\mathbf{K}_\mathrm{P}$ in the following form

$$\|\boldsymbol{e}_\mathrm{q}\| \le \|\mathbf{K}_\mathrm{P}^{-1}\| g_0 \,, \tag{8.24}$$

where g_0 is that in Property 8.4 in Sect. 8.1. Hence, the regulation error can be arbitrarily reduced by increasing $\mathbf{K}_\mathrm{P}$; nevertheless, measurement noise and other unmodeled dynamics, such as actuator friction, will limit the use of high gains in practice.

8.4.2 PID Control for Regulation

An integral action may be added to the previous PD control in order to deal with gravity forces, which to some extent can be considered as a constant disturbance (from the local point of view). The PID regulation controller can be written in the following general form

$$\boldsymbol{\tau} = \mathbf{K}_\mathrm{P}(\boldsymbol{q}_\mathrm{d} - \boldsymbol{q}) + \mathbf{K}_\mathrm{I}\int f(\boldsymbol{q}_\mathrm{d} - \boldsymbol{q})\mathrm{d}t - \mathbf{K}_\mathrm{V}\dot{\boldsymbol{q}} \,,$$

where $\mathbf{K}_\mathrm{I} \in R^{n\times n}$ is a positive-definite gain matrix, and:

- If $f(\boldsymbol{q}_\mathrm{d} - \boldsymbol{q}) = \boldsymbol{q}_\mathrm{d} - \boldsymbol{q}$, we have PID control.
- If $\mathbf{K}_\mathrm{I}\int(-\dot{\boldsymbol{q}})\mathrm{d}t$ is added, we have PI^{2}D control.
- If $f(\cdot) = \tanh(\cdot)$, we have PD + nonlinear integral control.

Global asymptotic stability (GAS) by PID control was proved in [8.12] for robotic motion control system including external disturbances, such as Coulomb friction. Also, *Tomei* proved the GAS of PD control in [8.19] by using an adaptation for gravity term. On the other hand, *Ortega* et al. showed in [8.20] that PI^{2}D control could yield semiglobal asymptotic stability (SGAS) in the presence of gravity and bounded external disturbances. Also, *Angeli* proved in [8.21] that PD control could achieve the input-output-to-state stability (IOSS) for robotic systems. Also, *Ramirez* et al. proved the SGAS (with some conditions) for PID gains in [8.22]. Also, *Kelly* proved in [8.23] that PD plus nonlinear integral control could achieve GAS under gravity.

Actually, a large integral action in PID control can cause instability of the motion control system. In order to avoid this, the integral gain should be bounded by an upper limit of the following form [8.1]

$$\frac{k_\mathrm{P}k_\mathrm{V}}{\lambda_\mathrm{H}^2} > k_\mathrm{I} \,,$$

where λ_H is that in Property 8.1 in Sect. 8.1, $\mathbf{K}_\mathrm{P} = k_\mathrm{P}\mathbf{I}$, $\mathbf{K}_\mathrm{I} = k_\mathrm{I}\mathbf{I}$, and $\mathbf{K}_\mathrm{V} = k_\mathrm{V}\mathbf{I}$. This relation gives the guidelines for gain selection implicitly. Also, PID control has generated a great variety of PID control plus something, e.g., PID plus friction compensator, PID plus gravity compensator, PID plus disturbance observer.

8.4.3 PID Gain Tuning

The PID control can be utilized for trajectory tracking as well as set-point regulation. True tracking control will be treated after Sect. 8.5. In this section, the simple but useful PID gain tuning method will be introduced for practical use. The general PID controller can be written in the following general form

$$\boldsymbol{\tau} = \mathbf{K}_\mathrm{V}\dot{\boldsymbol{e}}_\mathrm{q} + \mathbf{K}_\mathrm{P}\boldsymbol{e}_\mathrm{q} + \mathbf{K}_\mathrm{I}\int \boldsymbol{e}_\mathrm{q}\mathrm{d}t \,,$$

or, in another form,

$$\boldsymbol{\tau} = \left(\mathbf{K} + \frac{1}{\gamma^2}\mathbf{I}\right)\left(\dot{\boldsymbol{e}}_\mathrm{q} + \mathbf{K}_\mathrm{P}\boldsymbol{e}_\mathrm{q} + \mathbf{K}_\mathrm{I}\int \boldsymbol{e}_\mathrm{q}\mathrm{d}t\right) \,. \tag{8.25}$$

In a fundamental stability analysis of tracking control systems, *Qu* and *Dorsey* proved in [8.24] that PD control could satisfy uniform ultimate boundedness (UUB). Also, *Berghuis* and *Nijmeijer* proposed output feedback PD control, which satisfies semiglobal uniform ultimate boundedness (SGUUB) in [8.25] under gravity and a bounded disturbance. Recently, *Choi* et al. suggested inverse optimal PID control [8.26], assuring extended disturbance input-to-state stability (ISS).

Actually, if a PID controller (8.25) is repeatedly applied to the same set point or desired trajectory, then the maximum error will be proportional to the gains in the following form

$$\max_{0\le t\le t_f} \|\boldsymbol{e}_q(t)\| \propto \frac{\gamma^2}{\sqrt{2k\gamma^2+1}}\,, \tag{8.26}$$

where t_f denotes the final execution time of a given task and $\mathbf{K} = k\mathbf{I}$. This relation can be utilized to tune the gain of a PID controller and is referred to as the compound tuning rule [8.16]. The compound tuning rule implicitly includes simple tuning rules as follows (γ tuning method):

- Square tuning: $\max \|\boldsymbol{e}_q\| \propto \gamma^2$, for a small k
- Linear tuning: $\max \|\boldsymbol{e}_q\| \propto \gamma$, for a large k.

For example, suppose we select positive constant diagonal matrices $\mathbf{K}_P = k_P\mathbf{I}, \mathbf{K}_I = k_I\mathbf{I}$, while satisfying $k_P^2 > 2k_I$. For small k, the maximum error will be reduced by $\frac{1}{4}$ according to the square tuning rule, if we reduce the value γ by $\frac{1}{2}$. For large k, the maximum error will be proportionally reduced as γ according to the linear tuning rule. This means that we can tune the PID controller using only one parameter γ when the other gain parameters are fixed [8.16] (VIDEO 25). Although these rules are very useful in tuning the control performance, they can be utilized only for repetitive experiments for the same set point or desired trajectory because the tuning rules consist of proportional relations.

8.4.4 Automatic Tuning

For simplicity, define the composite error to be

$$\boldsymbol{s}(t) = \dot{\boldsymbol{e}}_q + \mathbf{K}_P\boldsymbol{e}_q + \mathbf{K}_I\int \boldsymbol{e}_q dt\,.$$

Now simple auto-tuning PID control is suggested by choosing one tuning parameter $\mathbf{K}$ as shown in the following control form

$$\boldsymbol{\tau} = \left(\widehat{K}(t) + \frac{1}{\gamma^2}\mathbf{I}\right)\boldsymbol{s}(t)\,,$$

and its automatic tuning rule as follow

$$\frac{d\widehat{K}_i}{dt} = \Gamma_i s_i^2(t), \quad \text{for } i = 1,\cdots,n\,, \tag{8.27}$$

where Γ_i implies an update gain parameter for i-th control joint.

For practical use of the PID control, a target performance denoted by Ω is specified in advance

$$\sup_{0\le t\le t_f} |\boldsymbol{s}(t)| = \Omega\,,$$

in order to maintain the composite error within the region Ω. Moreover, since the auto-tuning rule has the decentralized type (8.27), we suggest the decentralized criterion for auto-tuning as follows

$$|s_i| > \frac{\Omega}{\sqrt{2n}}\,, \tag{8.28}$$

where n is the number of the joint coordinates. As soon as the composite error arrives at the tuning region (8.28), the auto-tuning rule is implemented to assist the achievement of target performance. On the contrary, if the composite error stays in non-tuning region, namely, $|s_i| \le \Omega/\sqrt{2n}$, then the auto-tuning process stops. For this case, we expect that the gain $\widehat{K}$ updated by an auto-tuning rule (8.27) would be larger than the matrix $\mathbf{K}_\Omega$ able to achieve the target performance Ω. As a matter of fact, the auto-tuning rule plays a nonlinear damping role in the auto-tuning region.

Matlab Example (Multimedia)

Simple automatic tuning example for one-link manipulator control system is shown in the multimedia source to help readers' understanding.

Further Reading

The PID-type controllers were designed to solve the regulation control problem. They have the advantage of requiring knowledge of neither the model structure nor the model parameters. Also, the stability achieved by PID-type controllers was presented in this section. A range of books and papers [8.1, 15, 16, 22, 27, 28] are available to the robotics audience, detailing the individual tuning methods used in PID control and their concrete proofs.

8.5 Tracking Control

While independent PID controls are adequate in most set-point regulation problems, there are many tasks that require effective trajectory tracking capabilities such as plasma-welding, laser-cutting or high-speed operations in the presence of obstacles. In this case, employing local schemes requires moving slowly through a number of intermediate set points, thus considerably delaying the completion of the task. Therefore, to improve the trajectory tracking performance, controllers should take account of the manipulator dynamic model via a computed-torque-like technique.

The tracking control problem in the joint or task space consists of following a given time-varying trajectory $\boldsymbol{q}_\mathrm{d}(t)$ or $\boldsymbol{x}_\mathrm{d}(t)$ and its successive derivatives $\dot{\boldsymbol{q}}_\mathrm{d}(t)$ or $\dot{\boldsymbol{x}}_\mathrm{d}(t)$ and $\ddot{\boldsymbol{q}}_\mathrm{d}(t)$ or $\ddot{\boldsymbol{x}}_\mathrm{d}(t)$, which describe the desired velocity and acceleration, respectively. To obtain successful performance, significant effort has been devoted to the development of model-based control strategies [8.1, 2, 7]. Among the control approaches reported in the literature, typical methods include the inverse dynamics control, the feedback linearization technique, and the passivity-based control method.

8.5.1 Inverse Dynamics Control

Though the inverse dynamics control has a theoretical background, such as the theory of feedback linearization discussed later, its starting point is mechanical engineering intuition based on cancelling nonlinear terms and decoupling the dynamics of each link. Inverse dynamics control in joint space has the form

$$\boldsymbol{\tau} = \mathbf{H}(\boldsymbol{q})\boldsymbol{v} + \mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_\mathrm{g}(\boldsymbol{q}) \,, \tag{8.29}$$

which, applied to (8.1), yields a set of n decoupled linear systems, e.g., $\ddot{\boldsymbol{q}} = \boldsymbol{v}$, where $\boldsymbol{v}$ is an auxiliary control input to be designed. Typical choices for $\boldsymbol{v}$ are

$$\boldsymbol{v} = \ddot{\boldsymbol{q}}_\mathrm{d} + \mathbf{K}_\mathrm{V}(\dot{\boldsymbol{q}}_\mathrm{d} - \dot{\boldsymbol{q}}) + \mathbf{K}_\mathrm{P}(\boldsymbol{q}_\mathrm{d} - \boldsymbol{q}) \,, \tag{8.30}$$

or with an integral component

$$\boldsymbol{v} = \ddot{\boldsymbol{q}}_\mathrm{d} + \mathbf{K}_\mathrm{V}(\dot{\boldsymbol{q}}_\mathrm{d} - \dot{\boldsymbol{q}}) + \mathbf{K}_\mathrm{P}(\boldsymbol{q}_\mathrm{d} - \boldsymbol{q}) + \mathbf{K}_\mathrm{I}\int(\boldsymbol{q}_\mathrm{d} - \boldsymbol{q})\mathrm{d}t \,, \tag{8.31}$$

leading to the error dynamics equation

$$\ddot{\boldsymbol{e}}_\mathrm{q} + \mathbf{K}_\mathrm{V}\dot{\boldsymbol{e}}_\mathrm{q} + \mathbf{K}_\mathrm{P}\boldsymbol{e}_\mathrm{q} = \mathbf{0}$$

for an auxiliary control input (8.30), and

$$\boldsymbol{e}_\mathrm{q}^{(3)} + \mathbf{K}_\mathrm{V}\ddot{\boldsymbol{e}}_\mathrm{q} + \mathbf{K}_\mathrm{P}\dot{\boldsymbol{e}}_\mathrm{q} + \mathbf{K}_\mathrm{I}\boldsymbol{e}_\mathrm{q} = \mathbf{0} \,,$$

if an auxiliary control input (8.31) is used. Both error dynamics are exponentially stable by a suitable choice of the gain matrices $\mathbf{K}_\mathrm{V}$ and $\mathbf{K}_\mathrm{P}$ (and $\mathbf{K}_\mathrm{I}$).

Alternatively, inverse dynamics control can be described in the operational space. Consider the operational space dynamics (8.9). If the following inverse dynamics control is used in the operational space,

$$\boldsymbol{f}_\mathrm{c} = \boldsymbol{\Lambda}(\boldsymbol{q})(\ddot{\boldsymbol{x}}_\mathrm{d} + \mathbf{K}_\mathrm{V}\dot{\boldsymbol{e}}_x + \mathbf{K}_\mathrm{P}\boldsymbol{e}_x) + \boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{\eta}(\boldsymbol{q}) \,,$$

where $\boldsymbol{e}_x = \boldsymbol{x}_\mathrm{d} - \boldsymbol{x}$, the resulting error dynamics is

$$\ddot{\boldsymbol{e}}_x + \mathbf{K}_\mathrm{V}\dot{\boldsymbol{e}}_x + \mathbf{K}_\mathrm{P}\boldsymbol{e}_x = \mathbf{0} \,, \tag{8.32}$$

and it is also exponentially stable. One apparent advantage of using this controller is that $\mathbf{K}_\mathrm{P}$ and $\mathbf{K}_\mathrm{V}$ can be selected with a clear physical meaning in operational space. However, as can be seen in (8.10), $\boldsymbol{\Lambda}(\boldsymbol{q})$ becomes very large when the robot approaches singular configurations [8.8]. This means that large forces in some direction are needed to move the arm.

8.5.2 Feedback Linearization

This approach generalizes the concept of inverse dynamics of rigid manipulators. The basic idea of feedback linearization is to construct a transformation as a so-called *inner-loop control*, which exactly linearizes the nonlinear system after a suitable state space change of coordinates. One can then design a second stage or *outer-loop control* in the new coordinates to satisfy the traditional control design specifications such as tracking, disturbance rejection, etc. [8.5, 29]. The full power of the feedback linearization scheme for manipulator control becomes apparent if one includes in the dynamic description of the manipulator the transmission dynamics, such as the elasticity resulting from shaft windup, gear elasticity, etc. [8.5].

In recent years, an impressive volume of literature has emerged in the area of differential-geometric methods for nonlinear systems. Most of the results in this area are intended to give abstract coordinate-free descriptions of various geometric properties of nonlinear systems and as such are difficult for the nonmathematician to follow. It is our intention here to give only the basic idea of the feedback linearization scheme and to introduce a simple version of this technique that finds an immediate application to the manipulator control problem. The reader is referred to [8.30] for a comprehensive treatment of the feedback linearization technique using differential-geometric methods.

Let us now develop a simple approach to the determination of linear state-space representations of the

manipulator dynamics (8.1) by considering a general sort of output $\boldsymbol{\xi} \in R^p$

$$\boldsymbol{\xi} = \boldsymbol{h}(\boldsymbol{q}) + \boldsymbol{r}(t) \,, \tag{8.33}$$

where $\boldsymbol{h}(\boldsymbol{q})$ is a general predetermined function of the joint coordinate $\boldsymbol{q} \in R^n$ and $\boldsymbol{r}(t)$ is a general predetermined time function. The control objective will be to select the joint torque inputs $\boldsymbol{\tau}$ in order to make the output $\boldsymbol{\xi}(t)$ go to zero.

The choice of $\boldsymbol{h}(\boldsymbol{q})$ and $\boldsymbol{r}(t)$ is based on the control purpose. For example, if $\boldsymbol{h}(\boldsymbol{q}) = -\boldsymbol{q}$ and $\boldsymbol{r}(t) = \boldsymbol{q}_\mathrm{d}(t)$, the desired joint space trajectory we would like the manipulator to follow, then $\boldsymbol{\xi}(t) = \boldsymbol{q}_\mathrm{d}(t) - \boldsymbol{q}(t) \equiv \boldsymbol{e}_\mathrm{q}(t)$ is the joint space tracking error. Forcing $\boldsymbol{\xi}(t)$ to zero in this case would cause the joint variables $\boldsymbol{q}(t)$ to track their desired values $\boldsymbol{q}_\mathrm{d}(t)$, resulting in a manipulator trajectory-following problem. As another example, $\boldsymbol{\xi}(t)$ could represent the operational space tracking error $\boldsymbol{\xi}(t) = \boldsymbol{x}_\mathrm{d}(t) - \boldsymbol{x}(t) \equiv \boldsymbol{e}_x(t)$. Then, controlling $\boldsymbol{\xi}(t)$ to zero would result in trajectory following directly in operational space where the desired motion is usually specified.

To determine a linear state-variable model for manipulator controller design, let us simply differentiate the output $\boldsymbol{\xi}(t)$ twice to obtain

$$\dot{\boldsymbol{\xi}} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \dot{\boldsymbol{r}} = \mathbf{T}\dot{\boldsymbol{q}} + \dot{\boldsymbol{r}} \,, \tag{8.34}$$

$$\ddot{\boldsymbol{\xi}} = \mathbf{T}\ddot{\boldsymbol{q}} + \dot{\mathbf{T}}\dot{\boldsymbol{q}} + \ddot{\boldsymbol{r}} \,, \tag{8.35}$$

where we defined a $(p \times n)$ transformation matrix of the form

$$\mathbf{T}(\boldsymbol{q}) = \frac{\partial \boldsymbol{h}(\boldsymbol{q})}{\partial \boldsymbol{q}} = \left(\frac{\partial \boldsymbol{h}}{\partial q_1} \quad \frac{\partial \boldsymbol{h}}{\partial q_2} \quad \cdots \quad \frac{\partial \boldsymbol{h}}{\partial q_n} \right) . \tag{8.36}$$

Given the output $\boldsymbol{h}(\boldsymbol{q})$, it is straightforward to compute the transformation $\mathbf{T}(\boldsymbol{q})$ associated with $\boldsymbol{h}(\boldsymbol{q})$. In the special case where $\dot{\boldsymbol{\xi}}$ represents the operational space velocity error, then $\mathbf{T}(\boldsymbol{q})$ denotes the Jacobian matrix $\mathbf{J}(\boldsymbol{q})$.

According to (8.1),

$$\ddot{\boldsymbol{q}} = \mathbf{H}^{-1}(\boldsymbol{q})[\boldsymbol{\tau} - \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})] \,, \tag{8.37}$$

with the nonlinear terms represented by

$$\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_\mathrm{g}(\boldsymbol{q}) \,. \tag{8.38}$$

Then (8.35) yields

$$\ddot{\boldsymbol{\xi}} = \ddot{\boldsymbol{r}} + \dot{\mathbf{T}}\dot{\boldsymbol{q}} + \mathbf{T}(\boldsymbol{q})\mathbf{H}^{-1}(\boldsymbol{q})[\boldsymbol{\tau} - \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})] \,. \tag{8.39}$$

Define the *control input* function

$$\boldsymbol{u} = \ddot{\boldsymbol{r}} + \dot{\mathbf{T}}\dot{\boldsymbol{q}} + \mathbf{T}(\boldsymbol{q})\mathbf{H}^{-1}(\boldsymbol{q})[\boldsymbol{\tau} - \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})] \,. \tag{8.40}$$

Now we may define a state $\boldsymbol{y}(t) \in R^{2p}$ by $\boldsymbol{y} = (\boldsymbol{\xi}\, \dot{\boldsymbol{\xi}})$ and write the manipulator dynamics as

$$\dot{\boldsymbol{y}} = \begin{pmatrix} \mathbf{0} & \mathbf{I}_\mathrm{p} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \boldsymbol{y} + \begin{pmatrix} \mathbf{0} \\ \mathbf{I}_\mathrm{p} \end{pmatrix} \boldsymbol{u} \,. \tag{8.41}$$

This is a linear state-space system of the form

$$\dot{\boldsymbol{y}} = \mathbf{A}\boldsymbol{y} + \mathbf{B}\boldsymbol{u} \,, \tag{8.42}$$

driven by the control input $\boldsymbol{u}$. Due to the special form of $\mathbf{A}$ and $\mathbf{B}$, this system is called the *Brunovsky canonical form* and it is always controllable from $\boldsymbol{u}(t)$.

Since (8.40) is said to be a *linearizing transformation* for the manipulator dynamic equation, one may invert this transformation to obtain the joint torque

$$\boldsymbol{\tau} = \mathbf{H}(\boldsymbol{q})\mathbf{T}^{+}(\boldsymbol{q})(\boldsymbol{u} - \ddot{\boldsymbol{r}} - \dot{\mathbf{T}}\dot{\boldsymbol{q}}) + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \,, \tag{8.43}$$

where $\mathbf{T}^{+}$ denotes the *Moore–Penrose inverse* of the transformation matrix $\mathbf{T}(\boldsymbol{q})$.

In the special case $\boldsymbol{\xi} = \boldsymbol{e}_\mathrm{q}(t)$, and if we select $\boldsymbol{u}(t)$ so that (8.41) is stable by the PD feedback $\boldsymbol{u} = -\mathbf{K}_\mathrm{P}\boldsymbol{\xi} - \mathbf{K}_\mathrm{V}\dot{\boldsymbol{\xi}}$, then $\mathbf{T} = -\mathbf{I}_n$ and the control input torque $\boldsymbol{\tau}(t)$ defined by (8.43) makes the manipulator move in such a way that $\boldsymbol{y}(t)$ goes to zero. In this case, the feedback linearizing control and the inverse dynamics control become the same.

8.5.3 Passivity-Based Control

This method explicitly uses the passivity properties of the Lagrangian system [8.31, 32]. In comparison with the inverse dynamics method, passivity-based controllers are expected to have better robust properties because they do not rely on the exact cancellation of the manipulator nonlinearities. The passivity-based control input is given by

$$\begin{aligned} \dot{\boldsymbol{q}}_\mathrm{r} &= \dot{\boldsymbol{q}}_\mathrm{d} + \alpha \boldsymbol{e}_\mathrm{q} \,, \quad \alpha > 0 \,, \\ \boldsymbol{\tau} &= \mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}}_\mathrm{r} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}_\mathrm{r} + \boldsymbol{\tau}_\mathrm{g}(\boldsymbol{q}) + \mathbf{K}_\mathrm{V}\dot{\boldsymbol{e}}_\mathrm{q} + \mathbf{K}_\mathrm{P}\boldsymbol{e}_\mathrm{q} \,. \end{aligned} \tag{8.44}$$

With (8.44), we obtain the following closed-loop system

$$\mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{s}}_\mathrm{q} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{s}_\mathrm{q} + \mathbf{K}_\mathrm{V}\dot{\boldsymbol{e}}_\mathrm{q} + \mathbf{K}_\mathrm{P}\boldsymbol{e}_\mathrm{q} = \mathbf{0} \,, \tag{8.45}$$

where $\boldsymbol{s}_\mathrm{q} = \dot{\boldsymbol{e}}_\mathrm{q} + \alpha \boldsymbol{e}_\mathrm{q}$. Let us choose a Lyapunov function $V(\boldsymbol{y}, t)$ as follows

$$V = \frac{1}{2}\boldsymbol{y}^\mathrm{T} \begin{pmatrix} \alpha\mathbf{K}_\mathrm{V} + \mathbf{K}_\mathrm{P} + \alpha^2\mathbf{H} & \alpha\mathbf{H} \\ \alpha\mathbf{H} & \mathbf{H} \end{pmatrix} \boldsymbol{y} = \frac{1}{2}\boldsymbol{y}^\mathrm{T}\mathbf{P}\boldsymbol{y} \,. \tag{8.46}$$

Since the above equation is positive definite, it has a unique equilibrium at the origin, i. e., $\boldsymbol{y} = (\boldsymbol{e}_{\mathrm{q}}^{\mathrm{T}}, \dot{\boldsymbol{e}}_{\mathrm{q}}^{\mathrm{T}})^{\mathrm{T}} = \boldsymbol{0}$. Moreover, V can be bounded by

$$\sigma_{\mathrm{m}}\|\boldsymbol{y}\|^2 \leq \boldsymbol{y}^{\mathrm{T}}\mathbf{P}\boldsymbol{y} \leq \sigma_{\mathrm{M}}\|\boldsymbol{y}\|^2 , \quad \sigma_{\mathrm{M}} \geq \sigma_{\mathrm{m}} > 0 . \tag{8.47}$$

The time derivative of V gives

$$\dot{V} = -\dot{\boldsymbol{e}}_{\mathrm{q}}^{\mathrm{T}}\mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} - \alpha \boldsymbol{e}_{\mathrm{q}}\mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}} = -\boldsymbol{y}^{\mathrm{T}}\mathbf{Q}\boldsymbol{y} < 0 , \tag{8.48}$$

where $\mathbf{Q} = \mathrm{diag}[\alpha\mathbf{K}_{\mathrm{P}}, \mathbf{K}_{\mathrm{V}}]$. Since $\mathbf{Q}$ is positive definite and quadratic in $\boldsymbol{y}$, it can be also bounded by

$$\kappa_{\mathrm{m}}\|\boldsymbol{y}\|^2 \leq \boldsymbol{y}^{\mathrm{T}}\mathbf{Q}\boldsymbol{y} \leq \kappa_{\mathrm{M}}\|\boldsymbol{y}\|^2 , \quad \kappa_{\mathrm{M}} \geq \kappa_{\mathrm{m}} > 0 . \tag{8.49}$$

Then, from the bound of the Lyapunov function V, we get

$$\dot{V} \leq -\kappa_{\mathrm{m}}\|\boldsymbol{y}\|^2 = -2\eta V , \quad \eta = \frac{\kappa_{\mathrm{m}}}{\sigma_{\mathrm{M}}} , \tag{8.50}$$

which finally yields

$$V(t) \leq V(0)e^{-2\eta t} . \tag{8.51}$$

It has been shown that the value of α affects the tracking result dramatically [8.33]. The manipulator tends to vibrate for small values of α. Larger values of α allow better tracking performance and protect $\boldsymbol{s}_{\mathrm{q}}$ from being spoiled by the velocity measurement noise when the position error is small. In [8.34], it was suggested that

$$\mathbf{K}_{\mathrm{P}} = \alpha\mathbf{K}_{\mathrm{V}} \tag{8.52}$$

be used for quadratic optimization.

8.5.4 Summary

In this section, we have reviewed some of the model-based motion control methods proposed to date. Under some control approaches, the closed-loop system has either asymptotic stability or globally exponential stability. However, such ideal performance cannot be obtained in practical implementation because factors such as sampling rate, measurement noise, disturbances, and unmodeled dynamics will limit the achievable gain and the performance of the control algorithms [8.33, 35, 36].

8.6 Computed-Torque Control

Through the years many kinds of robot control schemes have been proposed. Most of these can be considered as special cases of the class of computed-torque control (Fig. 8.5) which is the technique of applying feedback linearization to nonlinear systems in general [8.37, 38]. In the section, computed-torque control will be first introduced, and its variant, so-called computed-torque-like control, will be introduced later.

8.6.1 Computed-Torque Control

Consider the control input (8.29)

$$\boldsymbol{\tau} = \mathbf{H}(\boldsymbol{q})\boldsymbol{v} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) ,$$

which is also known as computed-torque control; it consists of an inner nonlinear compensation loop and an outer loop with an exogenous control signal v. Substituting this control law into the dynamical model of the robot manipulator, it follows that

$$\ddot{\boldsymbol{q}} = \boldsymbol{v} . \tag{8.53}$$

It is important to note that this control input converts a complicated nonlinear controller design problem into a simple design problem for a linear system consisting of n decoupled subsystems. One approach to the outer-loop control $\boldsymbol{v}$ is propositional–derivative (PD) feedback, as in (8.30)

$$\boldsymbol{v} = \ddot{\boldsymbol{q}}_{\mathrm{d}} + \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}} ,$$

in which case the overall control input becomes

$$\boldsymbol{\tau} = \mathbf{H}(\boldsymbol{q})(\ddot{\boldsymbol{q}}_{\mathrm{d}} + \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}e_{\mathrm{q}}) + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) ,$$

and the resulting linear error dynamics are

$$\ddot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}} = \boldsymbol{0} . \tag{8.54}$$

According to linear system theory, convergence of the tracking error to zero is guaranteed [8.29, 39].

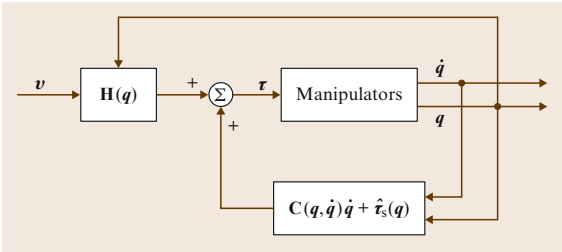


Fig. 8.5 Computed-torque control

Remark 8.2

One usually lets $\mathbf{K}_{\mathrm{V}}$ and $\mathbf{K}_{\mathrm{P}}$ be $(n \times n)$ diagonal positive-definite gain matrices (i. e., $\mathbf{K}_{\mathrm{V}} = \mathrm{diag}(k_{\mathrm{V},1},$

$\cdots, k_{V,n}) > \mathbf{0}$, $\mathbf{K}_P = \mathrm{diag}(k_{P,1}, \cdots, k_{P,n}) > \mathbf{0}$) to guarantee the stability of the error system. However, the format of the foregoing control never leads to *independent joint control* because the outer-loop multiplier $\mathbf{H}(\boldsymbol{q})$ and the full nonlinear compensation term $\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_g(\boldsymbol{q})$ in the inner loop scramble all joint signals among different control channels.

8.6.2 Computed-Torque-Like Control

It is worth noting that the implementation of computed-torque control requires that parameters of the dynamical model are accurately known and the control input is computed in real time. In order to avoid those problems, several variations of this control scheme have been proposed, for example, computed-torque-like control. An entire class of computed-torque-like controllers can be obtained by modifying the computed-torque control as

$$\boldsymbol{\tau} = \hat{\mathbf{H}}(\boldsymbol{q})\boldsymbol{v} + \hat{\mathbf{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \hat{\boldsymbol{\tau}}(\boldsymbol{q}) \,, \tag{8.55}$$

where $\hat{}$ represents the computed or nominal value and indicates that the theoretically exact feedback linearization cannot be achieved in practice due to the uncertainty in the systems. The overall control scheme is depicted in Fig. 8.6.

Computed-Torque-Like Control with Variable-Structure Compensation

Since there is parametric uncertainty, compensation is required in the outer-loop design to achieve trajectory tracking. The following shows a computed-torque-like control with variable-structure compensation

$$\boldsymbol{v} = \ddot{\boldsymbol{q}}_d + \mathbf{K}_V \dot{\boldsymbol{e}}_q + \mathbf{K}_P \boldsymbol{e}_q + \Delta \boldsymbol{v} \,, \tag{8.56}$$

where the variable-structure compensation $\Delta \boldsymbol{v}$ is devised as

$$\Delta \boldsymbol{v} = \begin{cases} -\rho(\boldsymbol{x}, t)\dfrac{\mathbf{B}^T\mathbf{P}\boldsymbol{x}}{\| \mathbf{B}^T\mathbf{P}\boldsymbol{x} \|} & \text{if} \quad \| \mathbf{B}^T\mathbf{P}\boldsymbol{x} \| \neq 0 \,, \\ 0 & \text{if} \quad \| \mathbf{B}^T\mathbf{P}\boldsymbol{x} \| = 0 \,, \end{cases} \tag{8.57}$$

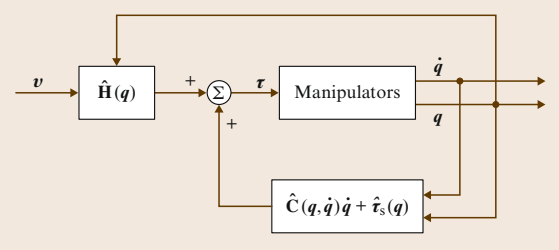


Fig. 8.6 Computed-torque-like control

where $\boldsymbol{x} = (\boldsymbol{e}_q^T, \dot{\boldsymbol{e}}_q^T)^T$, $\mathbf{B} = (\mathbf{0}, \mathbf{I}_n)^T$, $\mathbf{P}$ is a $(2n \times 2n)$ symmetric positive-definite matrix satisfying

$$\mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} = -\mathbf{Q} \,, \tag{8.58}$$

with $\mathbf{A}$ being defined as

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{I}_n \\ -\mathbf{K}_P & -\mathbf{K}_V \end{pmatrix} \,, \tag{8.59}$$

$\mathbf{Q}$ being any appropriate $(2n \times 2n)$ symmetric positive-definite matrix,

$$\rho(\boldsymbol{x}, t) = \frac{1}{1-\alpha}[\alpha\beta + \| \mathbf{K} \| \| \boldsymbol{x} \| + \bar{\mathbf{H}}\phi(\boldsymbol{x}, t)] \,, \tag{8.60}$$

where α and β are positive constants such that $\|\mathbf{H}^{-1}(\boldsymbol{q})\hat{\mathbf{H}}(\boldsymbol{q}) - \mathbf{I}_n\| \leq \alpha < 1$ for all $\boldsymbol{q} \in R^n$ and $\sup_{t \in [0,\infty)} \| \ddot{\boldsymbol{q}}_d(t) \| < \beta$, respectively, $\mathbf{K}$ is the $(n \times 2n)$ matrix defined as $\mathbf{K} = [\mathbf{K}_P \mathbf{K}_V]$, $\bar{\lambda}_H$ being a positive constant such that $\| \mathbf{H}^{-1}(\boldsymbol{q}) \| \leq \bar{\lambda}_H$ for all $\boldsymbol{q} \in R^n$, and the function ϕ being defined as

$$\begin{aligned} &\|[\hat{\mathbf{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})]\dot{\boldsymbol{q}} + [\hat{\boldsymbol{\tau}}_g(\boldsymbol{q}) - \boldsymbol{\tau}_g(\boldsymbol{q})]\| \\ &\quad \leq \phi(\boldsymbol{x}, t) \,. \end{aligned} \tag{8.61}$$

Convergence of the tracking error to zero can be shown using the Lyapunov function

$$V = \boldsymbol{x}^T\mathbf{P}\boldsymbol{x} \,, \tag{8.62}$$

following the stability analysis in [8.5, 40].

Remarks 8.3

- By Property 8.1 in Sect. 8.1, there exist positive constants $\bar{\lambda}_H$ and $\bar{\lambda}_h$ such that $\bar{\lambda}_h \leq \| \mathbf{H}^{-1}(\boldsymbol{q}) \| \leq \bar{\lambda}_H$ for all $\boldsymbol{q} \in R^n$. If we choose

$$\hat{\mathbf{H}} = \frac{1}{c}\mathbf{I}_n \,, \quad \text{where} \quad c = \frac{\bar{\lambda}_H + \bar{\lambda}_h}{2} \,, \tag{8.63}$$

it can be shown that

$$\| \mathbf{H}^{-1}(\boldsymbol{q})\hat{\mathbf{H}}(\boldsymbol{q}) - \mathbf{I}_n \| \leq \frac{\bar{\lambda}_H - \bar{\lambda}_h}{\bar{\lambda}_H + \bar{\lambda}_h} \equiv \alpha < 1 \,, \tag{8.64}$$

which indicates that there is always at least one choice of $\hat{\mathbf{H}}$ for some $\alpha < 1$.

- Due to the discontinuity in $\Delta \boldsymbol{v}$, chattering phenomenon may occur when the control scheme is applied. It is worth noting that chattering is often undesirable since the high-frequency component in

the control can excite unmodeled dynamic effect (such as joint flexibility) [8.6, 29, 38]. In order to avoid chattering, the variable-structure compensation can be modified to become smooth, i. e.,

$$\Delta \boldsymbol{v} = \begin{cases} -\rho(\boldsymbol{x},t)\dfrac{\mathbf{B}^{\mathrm{T}}\mathbf{P}\boldsymbol{x}}{\| \mathbf{B}^{\mathrm{T}}\mathbf{P}\boldsymbol{x} \|} & \text{if } \| \mathbf{B}^{\mathrm{T}}\mathbf{P}\boldsymbol{x} \| > \varepsilon \,, \\ -\dfrac{\rho(\boldsymbol{x},t)}{\varepsilon}\mathbf{B}^{\mathrm{T}}\mathbf{P}\boldsymbol{x} & \text{if } \| \mathbf{B}^{\mathrm{T}}\mathbf{P}\boldsymbol{x} \| \leq \varepsilon \,, \end{cases} \tag{8.65}$$

where ε is a positive constant and is used as the boundary layer. Following this modification, convergence of tracking errors to a residual set can be ensured, and the size of this residual set can be made smaller by use of a smaller value of ε.

Computed-Torque-Like Control with Independent-Joint Compensation

Obviously, the previous compensation scheme is centralized, which implies that the online computation load is heavy and high-cost hardware is required for practical implementation. In order to solve this problem, a scheme with independent-joint compensation is introduced below. In this scheme, a computed-torque-like control is designed with estimates as

$$\hat{\mathbf{H}}(\boldsymbol{q}) = \mathbf{I} \,, \quad \hat{\mathbf{C}}(\boldsymbol{q},\dot{\boldsymbol{q}}) = 0 \,, \quad \hat{\boldsymbol{\tau}}(\boldsymbol{q}) = \mathbf{0} \,. \tag{8.66}$$

Then, we use the outer-loop $\boldsymbol{v}$ as

$$\boldsymbol{v} = \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}} + \Delta \boldsymbol{v} \,, \tag{8.67}$$

where the positive constants $\mathbf{K}_{\mathrm{P}}$ and $\mathbf{K}_{\mathrm{V}}$ are selected to be sufficiently large, and the i-th component $\Delta \boldsymbol{v}_i$ of $\Delta \boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n)^{\mathrm{T}}$ is

$$\Delta \boldsymbol{v}_i = \begin{cases} -\left[\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{w}(\boldsymbol{q}_{\mathrm{d}},\dot{\boldsymbol{q}}_{\mathrm{d}})\right]^2 \dfrac{s_i}{\varepsilon_i} \\ \quad \text{if} \quad |\, s_i \,| \leq \dfrac{\varepsilon_i}{\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{w}(\boldsymbol{q}_{\mathrm{d}},\dot{\boldsymbol{q}}_{\mathrm{d}})} \,, \\ -\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{w}(\boldsymbol{q}_{\mathrm{d}},\dot{\boldsymbol{q}}_{\mathrm{d}})\dfrac{s_i}{|s_i|} \\ \quad \text{if} \quad |\, s_i \,| > \dfrac{\varepsilon_i}{\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{w}(\boldsymbol{q}_{\mathrm{d}},\dot{\boldsymbol{q}}_{\mathrm{d}})} \,. \end{cases} \tag{8.68}$$

In this compensation, $s_i = \dot{e}_{\mathrm{q},i} + \lambda_i e_{\mathrm{q},i}$, $i \in \{1, \ldots, n\}$, and $\lambda_i, i \in \{1, \ldots, n\}$ are positive constants. Furthermore, following the properties of robot manipulators, we have

$$\begin{aligned} &\| \mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}}_{\mathrm{d}} + \mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}_{\mathrm{d}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) \| \\ &\quad \leq \beta_1 + \beta_2 \| \boldsymbol{q} \| + \| \dot{\boldsymbol{q}} \| = \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{w}(\boldsymbol{q},\dot{\boldsymbol{q}}) \end{aligned}$$

for some suitable positive constants β_1, β_2, and β_3, where $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3)^{\mathrm{T}}$ and

$$\boldsymbol{w}(\boldsymbol{q},\dot{\boldsymbol{q}}) = [1, \| \boldsymbol{q} \|, \| \dot{\boldsymbol{q}} \|]^{\mathrm{T}} \,. \tag{8.69}$$

Finally, ε_i, $i \in \{1, \ldots, n\}$, is the variable length of the boundary layer, satisfying

$$\dot{\varepsilon}_i = -g_i\, \varepsilon_i \,, \quad \varepsilon(0) > 0 \,, \quad g_i > 0 \,. \tag{8.70}$$

It is worth pointing out that the term $\boldsymbol{w}$ in this control scheme is devised as the desired compensation rather than feedback. Furthermore, this scheme is in a form of independent-joint control and hence has the advantages introduced before. The convergence of the tracking error to zero can be shown using the Lyapunov function

$$V = \frac{1}{2}\left(\boldsymbol{e}_{\mathrm{q}}^{\mathrm{T}}\dot{\boldsymbol{e}}_{\mathrm{q}}^{\mathrm{T}}\right)\begin{pmatrix} \lambda\mathbf{K}_{\mathrm{P}} & \mathbf{H} \\ \mathbf{H} & \lambda\mathbf{H} \end{pmatrix}\begin{pmatrix} \boldsymbol{e}_{\mathrm{q}} \\ \dot{\boldsymbol{e}}_{\mathrm{q}} \end{pmatrix} + \sum_{i=1}^{n} g_i^{-1}\varepsilon_i \,, \tag{8.71}$$

whose time derivative along the trajectory of the closed-loop systems follows

$$\dot{V} = -\alpha \left| \begin{pmatrix} \boldsymbol{e}_{\mathrm{q}} \\ \dot{\boldsymbol{e}}_{\mathrm{q}} \end{pmatrix} \right|^2 \,, \tag{8.72}$$

with α being some positive constant, if sufficiently large $\mathbf{K}_{\mathrm{P}}$ and γ are used. The detailed analysis of stability, which requires Properties 8.3 and 8.4, can be found in [8.13].

Remark 8.3

Similar to computed-torque-like control with variable-structure compensation, we can consider the nonzero boundary layer as

$$\dot{\varepsilon}_i = -g_i\varepsilon_i \,, \quad \varepsilon(0) > 0 \,, \quad g_i \,,\, \alpha_i > 0 \,. \tag{8.73}$$

Following this modification, tracking errors converge to a residual set. The size of this residual set can be made smaller by the use of a smaller value of ε (i. e., smaller α_i).

For the task of point-to-point control, one PD controller with gravity compensation is designed with the estimates

$$\hat{\mathbf{H}}(\boldsymbol{q}) = \mathbf{I} \,, \quad \hat{\mathbf{C}}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \mathbf{0} \,, \quad \hat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q}) = \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) \,, \tag{8.74}$$

with $\boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q})$ being the *gravity term* of the dynamical model of the robot manipulators. Then, we use the outer-loop $\boldsymbol{v}$ as

$$\boldsymbol{v} = \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}} \,, \tag{8.75}$$

such that the control input becomes

$$\boldsymbol{\tau} = \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) \,. \tag{8.76}$$

This scheme is much simpler to implement than the exact computed-torque controller. The convergence of the tracking error to zero can be shown using the Lyapunov function

$$V = \frac{1}{2}\dot{\boldsymbol{e}}_{\mathrm{q}}^{\mathrm{T}}\mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{e}}_{\mathrm{q}} + \frac{1}{2}\boldsymbol{e}_{\mathrm{q}}^{\mathrm{T}}\mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}}\,, \tag{8.77}$$

whose time derivative along the solution trajectories of the closed-loop system is

$$\dot{V} = -\dot{\boldsymbol{e}}_{\mathrm{q}}^{\mathrm{T}}\mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}}\,. \tag{8.78}$$

The detailed analysis of stability is given in [8.12]. It is necessary to note that this result is for the case of regulation, rather than for the case of tracking, since the former theoretical base, which relies on LaSalle's lemma, requires the system be autonomous (time invariant) [8.38, 41, 42].

Remark 8.4

If we neglect gravity in the dynamical model of the robot manipulators, then the gravity estimation can be omitted here, i. e., $\hat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q}) = \mathbf{0}$, so that the control law becomes

$$\boldsymbol{\tau} = \boldsymbol{v} = \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}}\,, \tag{8.79}$$

which leads to pure PD control. The gain matrices, $\mathbf{K}_{\mathrm{P}}$ and $\mathbf{K}_{\mathrm{V}}$, can be selected to be diagonal so that this PD control is in a form of independent-joint control developed based on the multijoint dynamic model.

8.6.3 Summary

In this section, we have presented two control schemes: the computed-torque and computed-torque-like control. The former transforms a multi-input multi-output (MIMO) nonlinear robotic system into a very simple decoupled linear closed-loop system whose control design is a well-established problem. Since the practical implementation of the former control requires preknowledge of all the manipulator parameters and its payload, which may not be realistic, the latter was introduced to relax the foregoing requirement and still to achieve the objective of tracking subject to system's uncertainty.

Further Reading

The PD control with different feedforward compensation for tracking control is investigated in [8.43]. An adaptive control scheme based on PD control is presented in [8.19].

8.7 Adaptive Control

An adaptive controller differs from an ordinary controller in that the controller parameters are time varying, and there is a mechanism for adjusting these parameters online based on some signals in the closed-loop systems. By the use of such control scheme, the control goal can be achieved even if there is parametric uncertainty in the plant. In this section, we will introduce several adaptive control schemes to deal with the case of imperfect knowledge of dynamical parameters of the robot manipulators. The control performance of those adaptive control schemes, including adaptive computed-torque control, adaptive inertia-related control, adaptive control based on passivity, and adaptive control with desired compensation, are basically derived from Property 8.5. Finally, the condition of persistent excitation, which is important in parameter convergence, will be addressed.

8.7.1 Adaptive Computed-Torque Control

The computed-torque control scheme is appealing, since it allows the designer to transform a MIMO highly coupled nonlinear system into a very simple decoupled linear system, whose control design is a well-established problem. However, this method of feedback linearization relies on perfect knowledge of the system parameters, and failure to have this will cause erroneous parametric estimates, resulting in a mismatch term in the closed-loop model of the error system. That term can be interpreted as a nonlinear perturbation acting at the input of the closed-loop system. In order to solve the problem due to parametric uncertainty, we instead consider the inverse dynamics method with parameter estimates of

$$\boldsymbol{\tau} = \hat{\mathbf{H}}(\boldsymbol{q})(\ddot{\boldsymbol{q}}_{\mathrm{d}} + \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}}) + \hat{\mathbf{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \hat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q})\,, \tag{8.80}$$

where $\hat{\mathbf{H}}$, $\hat{\mathbf{C}}$, $\hat{\boldsymbol{\tau}}_{\mathrm{g}}$ have the same functional form as $\mathbf{H}$, $\mathbf{C}$, $\boldsymbol{\tau}_{\mathrm{g}}$. From Property 8.5 of the dynamics model, we have

$$\hat{\mathbf{H}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \hat{\mathbf{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \hat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q}) = \mathbf{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})\hat{\boldsymbol{a}}\,, \tag{8.81}$$

where $\mathbf{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})$, called the regressor, is a known $(n\times r)$ function matrix and $\boldsymbol{a}$ is the $(r\times 1)$ vector that summarizes all the estimated parameters. Substituting this

control input $\boldsymbol{\tau}$ into the manipulator dynamics gives the following closed-loop error model

$$\hat{\mathbf{H}}(\boldsymbol{q})(\ddot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}}) = \mathbf{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\tilde{\boldsymbol{a}} \,, \tag{8.82}$$

where $\tilde{\boldsymbol{a}} = \hat{\boldsymbol{a}} - \boldsymbol{a}$. In order to acquire an appropriate adaptive law, we first assume that the acceleration term $\ddot{\boldsymbol{q}}$ is measurable, and that the estimated inertia matrix $\hat{\mathbf{H}}(\boldsymbol{q})$ is never singular. Now, for convenience, the error equation is rewritten as

$$\dot{\boldsymbol{x}} = \mathbf{A}\boldsymbol{x} + \mathbf{B}\hat{\mathbf{H}}^{-1}(\boldsymbol{q})\mathbf{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\tilde{\boldsymbol{a}} \,, \tag{8.83}$$

with $\boldsymbol{x} = \left(\boldsymbol{e}_{\mathrm{q}}^{\mathrm{T}}, \dot{\boldsymbol{e}}_{\mathrm{q}}^{\mathrm{T}}\right)^{\mathrm{T}}$,

$$\mathbf{A} = \begin{pmatrix} \mathbf{0}_n & \mathbf{I}_n \\ -\mathbf{K}_{\mathrm{P}} & -\mathbf{K}_{\mathrm{V}} \end{pmatrix} \,, \quad \mathbf{B} = \begin{pmatrix} \mathbf{0}_n \\ \mathbf{I}_n \end{pmatrix} \,. \tag{8.84}$$

The adaptive law is considered as

$$\dot{\hat{\boldsymbol{a}}} = -\boldsymbol{\Gamma}^{-1}\mathbf{Y}^{\mathrm{T}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\hat{\mathbf{H}}^{-1}(\boldsymbol{q})\mathbf{B}^{\mathrm{T}}\mathbf{P}\boldsymbol{x} \,, \tag{8.85}$$

where $\boldsymbol{\Gamma}$ is an $(r \times r)$ positive-definite constant matrix, and $\mathbf{P}$ is a $(2n \times 2n)$ symmetric positive-definite constant matrix satisfying

$$\mathbf{P}\mathbf{A} + \mathbf{A}^{\mathrm{T}}\mathbf{P} = -\mathbf{Q} \,, \tag{8.86}$$

with $\mathbf{Q}$ being a symmetric positive-definite constant matrix with coherent dimension. In this adaptive law, we made two assumptions:

- The joint acceleration $\ddot{\boldsymbol{q}}$ is measurable, and
- The bounded range of the unknown parameter is available.

The first assumption is to ensure that the regressor $\mathbf{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ is known a priori, whereas the second assumption is to allow one to keep the estimate $\hat{\mathbf{H}}(\boldsymbol{q})$ nonsingular by restricting the estimated parameter $\hat{\boldsymbol{a}}$ to lie within a range about the true parameter value.

Convergence of the tracking error and maintaining boundedness of all internal signals can actually be guaranteed by Lyapunov stability theory with the Lyapunov function

$$\dot{V} = -\boldsymbol{x}^{\mathrm{T}}\mathbf{Q}\boldsymbol{x} \,. \tag{8.87}$$

Detailed stability analysis is given in [8.2].

Remark 8.5

For practical and theoretical reasons, the first assumption above is hardly acceptable. In most cases, it is not easy to obtain an accurate measure of acceleration; the robustness of the above adaptive control scheme with respect to such a disturbance has to be established. Moreover, from a pure theoretical viewpoint, measuring $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ means that not only do we need the whole system state vector, but we also need its derivative.

8.7.2 Adaptive Inertia-Related Control

Another adaptive control scheme is now introduced. This proposed scheme does not require the measurement of the manipulator's acceleration nor does it require inversion of the estimated inertia matrix. Hence, the drawbacks of the adaptive computed-torque control scheme are avoided. Let us consider the control input

$$\boldsymbol{\tau} = \hat{\mathbf{H}}(\boldsymbol{q})\dot{\boldsymbol{v}} + \hat{\mathbf{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{v} + \hat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q}) + \mathbf{K}_{\mathrm{D}}\boldsymbol{s} \,, \tag{8.88}$$

where the auxiliary signals $\boldsymbol{v}$ and $\boldsymbol{s}$ are defined as $\boldsymbol{v} = \dot{\boldsymbol{q}}_{\mathrm{d}} + \boldsymbol{\Lambda}\boldsymbol{e}_{\mathrm{q}}$ and $\boldsymbol{s} = \boldsymbol{v} - \dot{\boldsymbol{q}} = \dot{\boldsymbol{e}}_{\mathrm{q}} + \boldsymbol{\Lambda}\boldsymbol{e}_{\mathrm{q}}$, with $\boldsymbol{\Lambda}$ being an $(n \times n)$ positive-definite matrix. Following Property 8.5 of the dynamic model, we have

$$\mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{v}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{v} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) = \bar{\mathbf{Y}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{v}, \dot{\boldsymbol{v}})\boldsymbol{a} \,, \tag{8.89}$$

where $\bar{\mathbf{Y}}(\cdot, \cdot, \cdot, \cdot)$ is an $(n \times r)$ matrix of known time functions. The formulation above is the same type of the parameter separation that was used in the formulation of the adaptive computed-torque control. Note that $\bar{\mathbf{Y}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{v}, \dot{\boldsymbol{v}})$ is independent of the joint acceleration. Similar to the formulation above, we also have

$$\hat{\mathbf{H}}(\boldsymbol{q})\dot{\boldsymbol{v}} + \hat{\mathbf{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{v} + \hat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q}) = \bar{\mathbf{Y}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{v}, \dot{\boldsymbol{v}})\hat{\boldsymbol{a}} \,. \tag{8.90}$$

Substituting the control input into the equation of motion, it follows that

$$\begin{aligned} &\mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{p}) \\ &\quad = \hat{\mathbf{H}}(\boldsymbol{q})\dot{\boldsymbol{v}} + \hat{\mathbf{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{v} + \hat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q}) + \mathbf{K}_{\mathrm{D}}\boldsymbol{s} \,. \end{aligned}$$

Since $\ddot{\boldsymbol{q}} = \dot{\boldsymbol{v}} - \dot{\boldsymbol{s}}, \dot{\boldsymbol{q}} = \boldsymbol{v} - \boldsymbol{s}$, the previous result can be rewritten as

$$\mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{s}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{s} + \mathbf{K}_{\mathrm{D}}\boldsymbol{s} = \bar{\mathbf{Y}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{v}, \dot{\boldsymbol{v}})\tilde{\boldsymbol{a}} \,, \tag{8.91}$$

where $\tilde{\boldsymbol{a}} = \boldsymbol{a} - \hat{\boldsymbol{a}}$. The adaptive law is considered as

$$\dot{\hat{\boldsymbol{a}}} = \boldsymbol{\Gamma}\bar{\mathbf{Y}}^{\mathrm{T}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{v}, \dot{\boldsymbol{v}})\boldsymbol{s} \,. \tag{8.92}$$

The convergence of the tracking error to zero with boundedness on all internal signals can be shown

through Lyapunov stability theory using the following Lyapunov-like function

$$V = \frac{1}{2}\boldsymbol{s}^{\mathrm{T}}\mathbf{H}(\boldsymbol{q})\boldsymbol{s} + \frac{1}{2}\tilde{\boldsymbol{a}}^{\mathrm{T}}\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{a}}\,, \tag{8.93}$$

whose time derivative along the trajectories of the closed-loop systems can be found to be

$$\dot{V} = -\boldsymbol{s}^{\mathrm{T}}\mathbf{K}_{\mathrm{D}}\boldsymbol{s}\,. \tag{8.94}$$

The detailed stability analysis is given in [8.32].

Remark 8.6

- The restrictions for adaptive computed-torque control formerly seen have been removed here.
- The term $\mathbf{K}_{\mathrm{D}}\boldsymbol{s}$ introduces a PD-type linear stabilizing control action to the error system model.
- The estimated parameters converge to the true parameters provided the reference trajectory satisfies the condition of persistency of excitation,

$$\alpha_1\mathbf{I}_r \le \int_{t_0}^{t_0+t} \mathbf{Y}^{\mathrm{T}}(\boldsymbol{q}_{\mathrm{d}}, \dot{\boldsymbol{q}}_{\mathrm{d}}, \boldsymbol{v}, \dot{\boldsymbol{v}})\mathbf{Y}(\boldsymbol{q}_{\mathrm{d}}, \dot{\boldsymbol{q}}_{\mathrm{d}}, \boldsymbol{v}, \dot{\boldsymbol{v}})\,\mathrm{d}t \le \alpha_2\mathbf{I}_r\,,$$

for all t_0, where α_1, α_2, and t are all positive constants.

8.7.3 Adaptive Control Based on Passivity

Taking a physics viewpoint of control, we see that the concept of passivity has become popular for the development of adaptive control schemes. Here, we illustrate how the concept of passivity can be used to design a class of adaptive control laws for robot manipulators. First, we define an auxiliary filtered tracking error signal $\boldsymbol{r}$ as

$$\boldsymbol{r} = \mathbf{F}^{-1}(s)\boldsymbol{e}_{\mathrm{q}}\,, \tag{8.95}$$

where

$$\mathbf{F}^{-1}(s) = \left[s\mathbf{I}_n + \frac{1}{s}\mathbf{K}(s)\right]\,, \tag{8.96}$$

and s is the Laplace transform variable. The $(n\times n)$ matrix $\mathbf{K}(s)$ is chosen such that $\mathbf{F}(s)$ is a strictly proper, stable transfer function matrix. As in the preceding schemes, the adaptive control strategies has close ties to the ability to separate the known functions from the unknown constant parameters. We use the expression given above to define

$$\mathbf{Z}\boldsymbol{\varphi} = \mathbf{H}(\boldsymbol{q})[\ddot{\boldsymbol{q}}_{\mathrm{d}} + \mathbf{K}(s)\boldsymbol{e}_{\mathrm{q}}] + \boldsymbol{V}(\boldsymbol{q}, \dot{\boldsymbol{q}})\left[\dot{\boldsymbol{q}}_n + \frac{1}{s}\mathbf{K}(s)\boldsymbol{e}_{\mathrm{q}}\right] + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q})\,,$$

where $\mathbf{Z}$ is a known $(n\times r)$ regression matrix and $\boldsymbol{\varphi}$ is a vector of unknown system parameters in the adaptive context. It is important to note that the above can be arranged such that $\mathbf{Z}$ and $\boldsymbol{r}$ do not depend on the measurement of the joint acceleration $\ddot{\boldsymbol{q}}$. The adaptive control scheme given here is called the passivity approach because the mapping of $-\boldsymbol{r} \to \mathbf{Z}\tilde{\boldsymbol{\varphi}}$ is constructed to be a passive mapping. That is, we develop an adaptive law such that

$$\int_0^t -\boldsymbol{r}^{\mathrm{T}}(\sigma)\mathbf{Z}(\sigma)\tilde{\boldsymbol{\varphi}}(\sigma)\mathrm{d}\sigma \ge -\beta \tag{8.97}$$

is satisfied for all time and for some positive scalar constant β. For this class of adaptive controllers, the control input is given by

$$\boldsymbol{\tau} = \mathbf{Z}\hat{\boldsymbol{\varphi}} + \mathbf{K}_{\mathrm{D}}\boldsymbol{r}\,, \tag{8.98}$$

Detailed analysis of stability is given in [8.44].

Remarks 8.4

- If $\mathbf{K}(s)$ is selected such that $\mathbf{H}(s)$ has a relative degree of one, $\mathbf{Z}$ and $\boldsymbol{r}$ will not depend on $\ddot{\boldsymbol{q}}$.
- Many types of control schemes can be generated from the adaptive passive control approach by selected different transfer function matrices $\mathbf{K}(s)$ in the definition of $\boldsymbol{r}$.
- Note that, by defining $\mathbf{K}(s) = s\boldsymbol{\Lambda}$ such that $\mathbf{F}(s) = (s\mathbf{I}_n + \boldsymbol{\Lambda})^{-1}$, we have the control input

$$\boldsymbol{\tau} = \mathbf{Z}\hat{\boldsymbol{\varphi}} - \mathbf{K}_{\mathrm{D}}\boldsymbol{r}\,,$$

with

$$\mathbf{Z}\hat{\boldsymbol{\varphi}} = \hat{\mathbf{H}}(\boldsymbol{q})(\ddot{\boldsymbol{q}}_{\mathrm{d}} + \boldsymbol{\Lambda}\dot{\boldsymbol{e}}_{\mathrm{q}}) + \hat{\mathbf{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})(\dot{\boldsymbol{q}}_{\mathrm{d}} + \boldsymbol{\Lambda}\boldsymbol{e}_{\mathrm{q}}) + \hat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q})\,.$$

The adaptive law may be chosen as

$$\dot{\hat{\boldsymbol{\varphi}}} = \boldsymbol{\Gamma}\mathbf{Z}^{\mathrm{T}}(\dot{\boldsymbol{e}}_{\mathrm{q}} + \boldsymbol{\Lambda}\boldsymbol{e}_{\mathrm{q}})$$

to satisfy the condition of passive mapping. This indicates that adaptive inertia-related control can be viewed as a special case of adaptive passive control.

8.7.4 Adaptive Control with Desired Compensation

In order to implement the adaptive control scheme, one needs to calculate the elements of $\mathbf{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ in real time. However, this procedure may be excessively time consuming since it involves computations with highly nonlinear functions of joint position and velocity. Consequently, the real-time implementation of such a scheme is rather difficult. To overcome this difficulty, the adaptive control with desired compensation was proposed and is discussed here. In other words, the variables $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, and $\ddot{\boldsymbol{q}}$ are replaced with the desired ones, namely, $\boldsymbol{q}_\mathrm{d}$, $\dot{\boldsymbol{q}}_\mathrm{d}$, and $\ddot{\boldsymbol{q}}_\mathrm{d}$. Since the desired quantities are known in advance, all their corresponding calculations can be performed offline, which renders real-time implementation more plausible. Let us consider the control input

$$\boldsymbol{\tau} = \mathbf{Y}(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ddot{\boldsymbol{q}}_\mathrm{d})\hat{\boldsymbol{a}} + k_\mathrm{a}\boldsymbol{s} + k_\mathrm{p}\boldsymbol{e}_\mathrm{q} + k_n\|\boldsymbol{e}_\mathrm{q}\|^2\boldsymbol{s}\,, \quad (8.99)$$

where the positive constants k_a, k_p, and k_n are sufficiently large, and the auxiliary signal $\boldsymbol{s}$ is defined as $\boldsymbol{s} = \dot{\boldsymbol{e}}_\mathrm{q} + \boldsymbol{e}_\mathrm{q}$. The adaptive law is considered as

$$\dot{\hat{\boldsymbol{a}}} = -\boldsymbol{\Gamma}\,\mathbf{Y}^\mathrm{T}(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ddot{\boldsymbol{q}}_\mathrm{d})\boldsymbol{s}\,. \quad (8.100)$$

It is worth noting that the desired compensation is adopted in both the control and adaptive laws such that the computational load can be drastically reduced. For the sake of this analysis, we note that

$$\begin{aligned}&\|\mathbf{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\boldsymbol{a} - \mathbf{Y}(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ddot{\boldsymbol{q}}_\mathrm{d})\hat{\boldsymbol{a}}\| \\ &\quad \le \zeta_1\|\boldsymbol{e}_\mathrm{q}\| + \zeta_2\|\boldsymbol{e}_\mathrm{q}\|^2 + \zeta_3\|\boldsymbol{s}\| + \zeta_4\|\boldsymbol{s}\|\|\boldsymbol{e}_\mathrm{q}\|\,,\end{aligned}$$

where ζ_1, ζ_2, ζ_3, and ζ_4 are positive constants. In order to achieve trajectory tracking, it is required that

$$\begin{aligned}k_\mathrm{a} &> \zeta_2 + \zeta_4\,,\\ k_\mathrm{p} &> \frac{\zeta_1}{2} + \frac{\zeta_2}{4}\,,\\ k_\mathrm{v} &> \frac{\zeta_1}{2} + \zeta_3 + \frac{\zeta_2}{4}\,,\end{aligned}$$

(i. e., the gains k_a, k_p, and k_v must be sufficiently large). The convergence of the tracking error to zero with boundedness on all internal signals can be proved through application of Lyapunov stability theory with the following Lyapunov-like function

$$V = \frac{1}{2}\boldsymbol{s}^\mathrm{T}\mathbf{H}(\boldsymbol{q})\boldsymbol{s} + \frac{1}{2}k_\mathrm{p}\boldsymbol{e}_\mathrm{q}^\mathrm{T}\boldsymbol{e}_\mathrm{q} + \frac{1}{2}\tilde{\boldsymbol{a}}^\mathrm{T}\boldsymbol{\Gamma}^{-1}\tilde{\boldsymbol{a}}\,, \quad (8.101)$$

whose time derivative along the trajectories of the closed-loop system can be derived as

$$\dot{V} \le -\boldsymbol{x}^\mathrm{T}\mathbf{Q}\boldsymbol{x}\,, \quad (8.102)$$

where

$$\boldsymbol{x} = \begin{pmatrix}\|\boldsymbol{e}_\mathrm{q}\| \\ \|\boldsymbol{s}\|\end{pmatrix}\,,$$

$$\mathbf{Q} = \begin{pmatrix}\dfrac{k_\mathrm{p} - \zeta_2}{4} & \dfrac{-\zeta_1}{2} \\ \dfrac{-\zeta_1}{2} & \dfrac{k_\mathrm{v} - \zeta_3 - \zeta_4}{4}\end{pmatrix}\,.$$

Detailed stability analysis can be found in [8.45].

8.7.5 Summary

Since the computed-torque control suffers from parametric uncertainty, a variety of adaptive control schemes have been proposed. Firstly, we have presented an adaptive control scheme based on computed-torque control. Then, in order to overcome the mentioned drawbacks such as the measurability of the joint acceleration and the invertibility of the estimated inertia matrix, we presented an alternative adaptive control scheme that is free of these drawbacks. Recently, to incorporate a physics viewpoint into control, adaptive passivity-based control has become popular, and hence is introduced and discussed here. Finally, to reduce the computational load of the adaptive schemes, we presented an adaptive control with desired compensation.

Further Reading

A computationally very fast scheme dealing with adaptive control of rigid manipulators was presented in [8.46]. The stability analysis was completed by assuming that the joint dynamics are decoupled, i. e., that each joint is considered as an independent second-order linear system. A decentralized high-order adaptive variable-structure control is discussed in [8.47], the proposed scheme makes both position and velocity tracking errors of robot manipulators globally converge to zero asymptotically while allowing all signals in closed-loop systems to be bounded without the information of manipulator parameters. Other pioneering works in the field can be found, for example, in [8.48, 49]; although none of the fundamental dynamic model properties are used, the complete dynamics are taken into account, but the control input is discontinuous and may lead to chattering. Positive definiteness of the inertia matrix is explicitly used in [8.50], although it was assumed that some time-varying quantities remain constant during the adaptation. It is interesting to note that all of these schemes were based on the concept of model reference adaptive control (MRAC) developed in [8.51] for linear systems. Therefore, they are conceptually very different from the truly nonlinear schemes presented in this section.

A passive-based modified version of the least-squares estimation scheme has been proposed in [8.52] and [8.53], which guaranteed closed-loop stability of the scheme. Other schemes can be found in [8.54], where no use is made of the skew-symmetry property, and in [8.55], where the recursive Newton–Euler formulations is used instead of the Lagrange formulation to derive the manipulator dynamics, and thus computation is simplified to facilitate practical implementation.

Even though adaptive control provides a solution to the problem of parametric uncertainty, the robustness of adaptive controllers remains a topic of great interest in the field. Indeed, measurement noise or unmodeled dynamics (e.g., flexibility) may result in unbounded closed-loop signals. In particular, the estimated parameters may diverge; this is a well-known phenomenon in adaptive control and is called parameter drift. Solutions inspired from the adaptive control of linear systems have been studied [8.56, 57], where a modified estimation ensures boundedness of the estimates. In [8.58], the controller in [8.32] is modified to enhance robustness.

8.8 Optimal and Robust Control

Given a nonlinear system, such as robotic manipulators, one can develop many stabilizing controls [8.29, 41]. In other words, the stability of the control system cannot determine a unique controller. It is natural that one seeks an optimal controller among the many stable ones. However, the design of an optimal controller is possible provided that a rather exact information on the target system is available, such as an exact system model [8.34, 59]. In the presence of discrepancy between the real system and its mathematical model, a designed optimal controller is no longer optimal, and may even end up being instable in the actual system. Generally speaking, the optimal control design framework is not the best one to deal with system uncertainty. To handle system uncertainty from the control design stage, a robust control design framework is necessary [8.60]. One of main objectives of robust control is to keep the controlled system stable even in presence of uncertainties in the mathematical model, unmodeled dynamics, and the like.

Let us consider an affine nonlinear system described by nonlinear time-varying differential equation in the state $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)^{\mathrm{T}} \in R^n$

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}, t) + \mathbf{G}(\boldsymbol{x}, t)\boldsymbol{u} + \mathbf{P}(\boldsymbol{x}, t)\boldsymbol{w} \,, \tag{8.103}$$

where $\boldsymbol{u} \in R^m$ is the control input, and $\boldsymbol{w} \in R^w$ is the disturbance. Without disturbances or unmodeled dynamics, the system simplifies to

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}, t) + \mathbf{G}(\boldsymbol{x}, t)\boldsymbol{u} \,. \tag{8.104}$$

Actually, there are many kinds of methods describing the nonlinear system according to the objective of control [8.1, 16, 21, 23, 34, 54].

8.8.1 Quadratic Optimal Control

Every optimal controller is based on its own cost function [8.61, 62]. One can define a cost function as [8.63, 64]

$$z = \mathbf{H}(\boldsymbol{x}, t)\boldsymbol{x} + \mathbf{K}(\boldsymbol{x}, t)\boldsymbol{u} \,,$$

such that $\mathbf{H}^{\mathrm{T}}(\boldsymbol{x}, t)\mathbf{K}(\boldsymbol{x}, t) = \mathbf{0}$, $\mathbf{K}^{\mathrm{T}}(\boldsymbol{x}, t)\mathbf{K}(\boldsymbol{x}, t) = \mathbf{R}(\boldsymbol{x}, t) > \mathbf{0}$, and $\mathbf{H}^{\mathrm{T}}(\boldsymbol{x}, t)\mathbf{H}(\boldsymbol{x}, t) = \mathbf{Q}(\boldsymbol{x}, t) > \mathbf{0}$. Then, we have

$$\frac{1}{2}z^{\mathrm{T}}z = \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\mathbf{Q}(\boldsymbol{x}, t)\boldsymbol{x} + \frac{1}{2}\boldsymbol{u}^{\mathrm{T}}\mathbf{R}(\boldsymbol{x}, t)\boldsymbol{u} \,.$$

The quadratic optimal control for the system (8.104) is found by solving, for a first order differentiable positive-definite function $V(\boldsymbol{x}, t)$, the Hamilton–Jacobi–Bellman (HJB) equation [8.34, 59]

$$\begin{aligned} 0 = \mathrm{HJB}(\boldsymbol{x}, t; V) &= V_t(\boldsymbol{x}, t) + \boldsymbol{V}_x(\boldsymbol{x}, t)\boldsymbol{f}(\boldsymbol{x}, t) \\ &\quad - \frac{1}{2}\boldsymbol{V}_x(\boldsymbol{x}, t)\mathbf{G}(\boldsymbol{x}, t)\mathbf{R}^{-1}(\boldsymbol{x}, t)\mathbf{G}^{\mathrm{T}}(\boldsymbol{x}, t)\boldsymbol{V}_x^{\mathrm{T}}(\boldsymbol{x}, t) \\ &\quad + \frac{1}{2}\mathbf{Q}(\boldsymbol{x}, t) \,, \end{aligned}$$

where $V_t = \frac{\partial V}{\partial t}$ and $\boldsymbol{V}_x = \frac{\partial V}{\partial \boldsymbol{x}^{\mathrm{T}}}$. Then the quadratic optimal control is defined by

$$\boldsymbol{u} = -\mathbf{R}^{-1}(\boldsymbol{x}, t)\mathbf{G}^{\mathrm{T}}(\boldsymbol{x}, t)\boldsymbol{V}_x^{\mathrm{T}}(\boldsymbol{x}, t) \,. \tag{8.105}$$

Note that the HJB equation is a nonlinear second-order partial differential equation in $V(\boldsymbol{x}, t)$.

Unlike the aforementioned optimal control problem, the so-called inverse quadratic optimal control problem is to find a set of $\mathbf{Q}(\boldsymbol{x}, t)$ and $\mathbf{R}(\boldsymbol{x}, t)$ for which the HJB equation has a solution $V(\boldsymbol{x}, t)$. Then the inverse quadratic optimal control is defined by (8.105).

8.8.2 Nonlinear $\mathcal{H}_\infty$ Control

When disturbances are not negligible, one can deal with their effect such that

$$\int_0^t z^{\mathrm{T}}(\boldsymbol{x}, \tau)z(\boldsymbol{x}, \tau)\,\mathrm{d}\tau \le \gamma^2 \int_0^t \boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}\,\mathrm{d}\tau \,, \tag{8.106}$$

where $\gamma > 0$ specifies the L_2-gain of the closed-loop system from the disturbance input $\boldsymbol{w}$ to the cost variable $\boldsymbol{z}$. This is called the L_2-gain attenuation requirement [8.63–65]. One systematic way to design an optimal and robust control is given by the nonlinear $\mathcal{H}_\infty$ optimal control. Let $\gamma > 0$ be given, by solving the following equation

$$\begin{aligned}
&\mathrm{HJI}_\gamma(\boldsymbol{x}, t; V) \\
&\quad = V_t(\boldsymbol{x}, t) + V_x(\boldsymbol{x}, t)\boldsymbol{f}(\boldsymbol{x}, t) \\
&\qquad - \frac{1}{2} V_x(\boldsymbol{x}, t)\{\mathbf{G}(\boldsymbol{x}, t)\mathbf{R}^{-1}(\boldsymbol{x}, t)\mathbf{G}^{\mathrm{T}}(\boldsymbol{x}, t) \\
&\qquad - \gamma^{-2}\mathbf{P}(\boldsymbol{x}, t)\mathbf{P}^{\mathrm{T}}(\boldsymbol{x}, t)\} V_x^{\mathrm{T}}(\boldsymbol{x}, t) \\
&\qquad + \frac{1}{2}\mathbf{Q}(\boldsymbol{x}, t) \le 0 \,, \qquad (8.107)
\end{aligned}$$

then the control is defined by

$$\boldsymbol{u} = -\mathbf{R}^{-1}(\boldsymbol{x}, t)\mathbf{G}^{\mathrm{T}}(\boldsymbol{x}, t) V_x^{\mathrm{T}}(\boldsymbol{x}, t) \,. \qquad (8.108)$$

The partial differential inequality (8.107) is called the Hamilton–Jacobi–Isaac (HJI) inequality. Then one can define the inverse nonlinear $\mathcal{H}_\infty$ optimal control problem that finds a set of $\mathbf{Q}(\boldsymbol{x}, t)$ and $\mathbf{R}(\boldsymbol{x}, t)$ such that the L_2-gain requirement is achieved for a prescribed L_2-gain γ [8.66].

Two things deserve further comment. The first is that the L_2-gain requirement is only valid for disturbance signals $\boldsymbol{w}$ whose L_2-norm is bounded. The second is that $\mathcal{H}_\infty$ optimal control is not uniquely defined. Hence, one can choose a quadratic optimal among many $\mathcal{H}_\infty$ optimal controllers. Precisely speaking, the control (8.108) should be called $\mathcal{H}_\infty$ *suboptimal* control, since the desired L_2-gain is prescribed a priori. A true $\mathcal{H}_\infty$ optimal control is to find a minimal value of γ, such that the L_2-gain requirement is achieved.

8.8.3 Passivity-Based Design of Nonlinear $\mathcal{H}_\infty$ Control

There are many methodologies for the design of optimal and/or robust controls. Among these, passivity-based controls can take full advantage of the properties described above [8.31]. They consist of two parts: one coming from the reference motion compensation while preserving the passivity of the system, and the other to achieve stability, robustness, and/or optimality [8.66, 67].

Let us suppose that the dynamic parameters are identified as $\widehat{\mathbf{H}}(\boldsymbol{q})$, $\widehat{\mathbf{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, and $\widehat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q})$, whose counterparts are $\mathbf{H}(\boldsymbol{q})$, $\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, and $\boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q})$, respectively. Then, passivity-based control generates the following tracking control laws

$$\boldsymbol{\tau} = \widehat{\mathbf{H}}(\boldsymbol{q})\ddot{\boldsymbol{q}}_{\mathrm{ref}} + \widehat{\mathbf{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}_{\mathrm{ref}} + \widehat{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q}) - \boldsymbol{u} \,, \qquad (8.109)$$

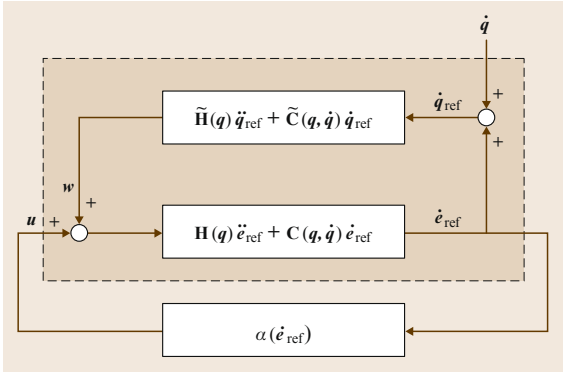


Fig. 8.7 The closed-loop system according to (8.111)

where $\ddot{\boldsymbol{q}}_{\mathrm{ref}}$ is the reference acceleration defined by

$$\ddot{\boldsymbol{q}}_{\mathrm{ref}} = \ddot{\boldsymbol{q}}_{\mathrm{d}} + \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}} \,, \qquad (8.110)$$

where $\mathbf{K}_{\mathrm{V}} = \mathrm{diag}\{k_{\mathrm{V},i}\} > 0$ and $\mathbf{K}_{\mathrm{P}} = \mathrm{diag}\{k_{\mathrm{P},i}\} > 0$. Two parameters are involved in generating the reference acceleration. Sometimes the following alternative method can be adopted

$$\ddot{\boldsymbol{q}}_{\mathrm{ref}} = \ddot{\boldsymbol{q}}_{\mathrm{d}} + \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} \,.$$

This reduces the order of the closed-loop system because the state $\boldsymbol{x} = (\boldsymbol{e}_{\mathrm{q}}^{\mathrm{T}}, \dot{\boldsymbol{e}}_{\mathrm{q}}^{\mathrm{T}})^{\mathrm{T}}$ is sufficient for the system description, while the definition of (8.110) requires the state $\boldsymbol{x} = (\int \boldsymbol{e}_{\mathrm{q}}^{\mathrm{T}}, \boldsymbol{e}_{\mathrm{q}}^{\mathrm{T}}, \dot{\boldsymbol{e}}_{\mathrm{q}}^{\mathrm{T}})^{\mathrm{T}}$.

In Fig. 8.7, the closed-loop dynamics under the control is given by

$$\mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{e}}_{\mathrm{ref}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{e}}_{\mathrm{ref}} = \boldsymbol{u} + \boldsymbol{w} \,, \qquad (8.111)$$

where

$$\begin{aligned}
\ddot{\boldsymbol{e}}_{\mathrm{ref}} &= \ddot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{V}}\dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\boldsymbol{e}_{\mathrm{q}} \,, \\
\dot{\boldsymbol{e}}_{\mathrm{ref}} &= \dot{\boldsymbol{e}}_{\mathrm{q}} + \mathbf{K}_{\mathrm{V}}\boldsymbol{e}_{\mathrm{q}} + \mathbf{K}_{\mathrm{P}}\int \boldsymbol{e}_{\mathrm{q}} \,.
\end{aligned}$$

If $\boldsymbol{d}(t) = 0$ and $\widehat{\mathbf{H}} = \mathbf{H}$, $\widehat{\mathbf{C}} = \mathbf{C}$, $\widehat{\boldsymbol{\tau}}_{\mathrm{g}} = \boldsymbol{\tau}_{\mathrm{g}}$, then $\boldsymbol{w} = 0$. Otherwise, the disturbance is defined as

$$\boldsymbol{w} = \widetilde{\mathbf{H}}(\boldsymbol{q})\ddot{\boldsymbol{q}}_{\mathrm{ref}} + \widetilde{\mathbf{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}_{\mathrm{ref}} + \widetilde{\boldsymbol{\tau}}_{\mathrm{g}}(\boldsymbol{q}) + \boldsymbol{d}(t) \,, \qquad (8.112)$$

where $\widetilde{\mathbf{H}} = \mathbf{H} - \widehat{\mathbf{H}}$, $\widetilde{\mathbf{C}} = \mathbf{C} - \widehat{\mathbf{C}}$, and $\widetilde{\boldsymbol{\tau}}_{\mathrm{g}} = \widehat{\boldsymbol{\tau}}_{\mathrm{g}} - \boldsymbol{\tau}_{\mathrm{g}}$. It is of particular interest that the system (8.111) defines a passive mapping between $\boldsymbol{u} + \boldsymbol{w}$ and $\dot{\boldsymbol{e}}_{\mathrm{ref}}$.

According to the manner in which the auxiliary control input $\boldsymbol{u}$ is specified, passivity-based control can achieve stability, robustness, and/or optimality (Fig. 8.7).

8.8.4 A Solution to Inverse Nonlinear $\mathcal{H}_\infty$ Control

Let us define the auxiliary control input by the reference-error feedback

$$\boldsymbol{u} = -\alpha \mathbf{R}^{-1}(\boldsymbol{x}, t)\dot{\boldsymbol{e}}_{\mathrm{ref}} , \tag{8.113}$$

where $\alpha > 1$ is arbitrary. Then, the control provides the inverse nonlinear $\mathcal{H}_\infty$ optimality.

Theorem 8.1 Inverse Nonlinear $\mathcal{H}_\infty$ Optimality [8.66]
Let the reference acceleration generation gain matrices $\mathbf{K}_\mathrm{V}$ and $\mathbf{K}_\mathrm{P}$ satisfy

$$\mathbf{K}_\mathrm{V}^2 > 2\mathbf{K}_\mathrm{P} . \tag{8.114}$$

Then for a given $\gamma > 0$, the reference error feedback

$$\boldsymbol{u} = -\mathbf{K}\dot{\boldsymbol{e}}_{\mathrm{ref}} = -\mathbf{K}\left(\dot{\boldsymbol{e}}_\mathrm{q} + \mathbf{K}_\mathrm{V}\boldsymbol{e}_\mathrm{q} + \mathbf{K}_\mathrm{P}\int \boldsymbol{e}_\mathrm{q}\right) \tag{8.115}$$

satisfies the L_2-gain attenuation requirement for

$$\mathbf{Q} = \begin{pmatrix} \mathbf{K}_\mathrm{P}^2\mathbf{K}_\gamma & 0 & 0 \\ 0 & (\mathbf{K}_\mathrm{V}^2 - 2\mathbf{K}_\mathrm{P})\mathbf{K}_\gamma & 0 \\ 0 & 0 & \mathbf{K}_\gamma \end{pmatrix} , \tag{8.116}$$

$$\mathbf{R} = \mathbf{K}^{-1} , \tag{8.117}$$

provided that

$$\mathbf{K}_\gamma = \mathbf{K} - \frac{1}{\gamma^2}\mathbf{I} > 0 . \tag{8.118}$$

Given γ, one can set $\mathbf{K} = \alpha\frac{1}{\gamma^2}\mathbf{I}$ for $\alpha > 1$. This yields $\mathbf{K}_\gamma = (\alpha - 1)\frac{1}{\gamma^2}\mathbf{I}$.

When the inertia matrix is identified as a diagonal constant matrix such as $\widehat{\mathbf{H}} = \mathrm{diag}\{\widehat{m}_i\}$, one should set $\widehat{\mathbf{C}} = \mathbf{0}$. In addition, one can set $\widehat{\boldsymbol{\tau}}_\mathrm{g} = \mathbf{0}$. Then this results in a decoupled PID control of the form

$$\begin{aligned}\tau_i = {} & \widehat{m}_i\left(\ddot{q}_{\mathrm{d},i} + k_{\mathrm{V},i}\dot{e}_{\mathrm{q},i} + k_{\mathrm{P},i}e_{\mathrm{q},i}\right) \\ & + \alpha\frac{1}{\gamma^2}\left(\dot{e}_{\mathrm{q},i} + k_{\mathrm{V},i}e_{\mathrm{q},i} + k_{\mathrm{P},i}\int e_{\mathrm{q},i}\right)\end{aligned}$$

for $\alpha > 1$, which can be rewritten as

$$\begin{aligned}\tau_i = {} & \widehat{m}_i\ddot{q}_{\mathrm{d},i} + \left(\widehat{m}_i k_{\mathrm{V},i} + \alpha\frac{1}{\gamma^2}\right)\dot{e}_{\mathrm{q},i} \\ & + \left(\widehat{m}_i k_{\mathrm{P},i} + \alpha\frac{k_{\mathrm{V},i}}{\gamma^2}\right)e_{\mathrm{q},i} + \alpha\frac{k_{\mathrm{P},i}}{\gamma^2}\int e_{\mathrm{q},i} .\end{aligned} \tag{8.119}$$

This leads to a PID control with the desired acceleration feedforward [8.68] given by

$$\tau_i = \widehat{m}_i\ddot{q}_{\mathrm{d},i} + k^*_{\mathrm{V},i}\dot{e}_{\mathrm{q},i} + k^*_{\mathrm{P},i}e_{\mathrm{q},i} + k^*_{\mathrm{I},i}\int e_{\mathrm{q},i} , \tag{8.120a}$$

where

$$k^*_{\mathrm{V},i} = \widehat{m}_i k_{\mathrm{V},i} + \alpha\frac{1}{\gamma^2} , \tag{8.120b}$$

$$k^*_{\mathrm{P},i} = \widehat{m}_i k_{\mathrm{P},i} + \alpha\frac{k_{\mathrm{V},i}}{\gamma^2} , \tag{8.120c}$$

$$k^*_{\mathrm{I},i} = \alpha\frac{k_{\mathrm{P},i}}{\gamma^2} . \tag{8.120d}$$

8.9 Trajectory Generation and Planning

This section deals with the problem of *reference trajectory generation*, that is, the computation of desired position, velocity, acceleration and/or force/torque signals that are used as input values for the robot motion controllers introduced in Sects 8.3–8.8.

8.9.1 Geometric Paths and Trajectories

Path Planning

A path is a geometric representation of a plan to move from a start to a target pose. The task of planning is to find a collision-free path among a collection of static and dynamic obstacles. Path planning can also include the consideration of dynamic constraints such as workspace boundaries, maximum velocities, maximum accelerations, and maximum jerks. We distinguish between *online* and *offline* path planning algorithms. Offline planned paths are static and calculated prior to execution. Online methods require algorithms that meet real-time constraints (i. e., algorithms that do not exceed a determinable worst-case computation time) to enable path (re-)calculations and/or adaptations during the robot motions in order to react to and interact with dynamic environments. This means that a robot

moves along a path that has not necessarily been computed completely, and which may change during the movement. Details about path planning concepts are described in Chap. 7, in Parts D and E, and specifically in Chap. 47.

Trajectory Planning

A trajectory is more than a path: It also includes velocities, accelerations, and/or jerks along a path (VIDEO 760). A common method is computing trajectories for a priori specified paths, which fulfill a certain criterion (e.g., minimum execution time). We distinguish between *online* and *offline* trajectory planning methods. An offline calculated trajectory cannot be influenced during its execution, while online trajectory planning methods can (re-)calculate and/or adapt robot motions behavior during the movement. The reasons for this (re-)calculation and/or adaptation can vary: improvement of accuracy, better utilization of currently available dynamics, reaction to and interaction with a dynamic environment, or reaction to (sensor) events. Besides the distinction between online and offline methods, we can further distinguish between (1) one-dimensional (1-D) and multi-dimensional trajectories and (2) single-point and multi-point trajectories. Multi-point trajectories typically relate to a path.

8.9.2 Joint Space and Operational Space Trajectories

Depending on the control state space, trajectory generators provide set points for tracking controllers in joint space or in operational space. In either space, a trajectory can be represented in several different ways: cubic splines, quintic splines, higher-order splines, harmonics (sinusoidal functions), exponential functions, Fourier series, and more.

Joint Space Trajectories

Consider the torque control input (8.29)

$$\boldsymbol{\tau} = \mathbf{H}(\boldsymbol{q})\boldsymbol{v} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q}) \,,$$

and the PD controller (8.30)

$$\boldsymbol{v} = \ddot{\boldsymbol{q}}_{\mathrm{d}} + \mathbf{K}_{\mathrm{V}}(\dot{\boldsymbol{q}}_{\mathrm{d}} - \dot{\boldsymbol{q}}) + \mathbf{K}_{\mathrm{P}}(\boldsymbol{q}_{\mathrm{d}} - \boldsymbol{q}) \,, \tag{8.121}$$

or PID controller (8.31), respectively, the task of a trajectory generator in joint space coordinates is computing the signals $\boldsymbol{q}_{\mathrm{d}}(t)$, $\dot{\boldsymbol{q}}_{\mathrm{d}}(t)$, and $\ddot{\boldsymbol{q}}_{\mathrm{d}}(t)$. These three signals contain the reference trajectory and are used as input values for the tracking controller.

During nominal operation the joint torques required to execute a trajectory should not exceed joint force/torque limits $\boldsymbol{\tau}_{\min}(t)$ and $\boldsymbol{\tau}_{\max}(t)$,

$$\boldsymbol{\tau}_{\min}(t) \leq \boldsymbol{\tau}(t) \leq \boldsymbol{\tau}_{\max}(t) \;\forall\; t \in \mathbb{R}. \tag{8.122}$$

Operational Space Trajectories

Similarly to (8.29), we can also consider trajectories represented by $\boldsymbol{x}_{\mathrm{d}}$, $\dot{\boldsymbol{x}}_{\mathrm{d}}$, and $\ddot{\boldsymbol{x}}_{\mathrm{d}}$ for an operational space controller (8.9)

$$\boldsymbol{f}_{\mathrm{c}} = \boldsymbol{\Lambda}(\boldsymbol{q})\boldsymbol{\mu} + \boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{\eta}(\boldsymbol{q}) \,,$$

with a PD control law of

$$\boldsymbol{\mu} = \ddot{\boldsymbol{x}}_{\mathrm{d}} + \mathbf{K}_{\mathrm{V}}(\dot{\boldsymbol{x}}_{\mathrm{d}} - \dot{\boldsymbol{x}}) + \mathbf{K}_{\mathrm{P}}(\boldsymbol{x}_{\mathrm{d}} - \boldsymbol{x}) \,. \tag{8.123}$$

With the transformation into joint space using the inverse of (8.8), the operational space trajectory generator must assure that the limits given in (8.122) are not violated. It is the responsibility of the path planner (Chap. 7) that all points along the trajectory are in the robot workspace and that start and goal poses can be reached in the same joint configuration. It is the responsibility of the trajectory planner that joint torque and velocity constraints are not violated even in the presence of kinematic singularities.

8.9.3 Trajectory Representations

Mathematical Representations

Functions for $\ddot{\boldsymbol{q}}_{\mathrm{d}}(t)$ (8.121) and $\ddot{\boldsymbol{x}}_{\mathrm{d}}$ (8.123) can be represented in several ways that are described here.

Polynomial Trajectories. One of the simplest ways to represent a robot trajectory is a polynomial function of degree m for each joint $i \in \{1, \ldots, n\}$

$$q_i(t) = a_{i,0} + a_{i,1}t + a_{i,2}t^2 + \ldots + a_{i,m}t^m \,, \tag{8.124}$$

so that $\boldsymbol{q}(t)$ can be composed (or $\boldsymbol{x}(t)$ in operational space, respectively). In the simplest case, cubic polynomials are used, which, however, leads to non-steady acceleration signals with infinite jerks. Quintic and higher-order polynomials allow for steady acceleration signals as well as arbitrary position, velocity, and acceleration vectors and the beginning and at the end of the trajectory. To determine the coefficients $a_{i,j}$ $\forall$ $(i, j) \in \{1, \ldots, n\} \times \{0, \ldots, m\}$ of (8.124), the execution time t_{trgt}, at which the target state will be reached needs to be known. The left part of Fig. 8.8a shows a quintic trajectory for three DOFs starting at $t_0 = 0\,s$ with an execution time of $t_{\mathrm{trgt}} = 2.5\,s$. To connect a trajectory segment to preceding and succeeding segments, the fol-

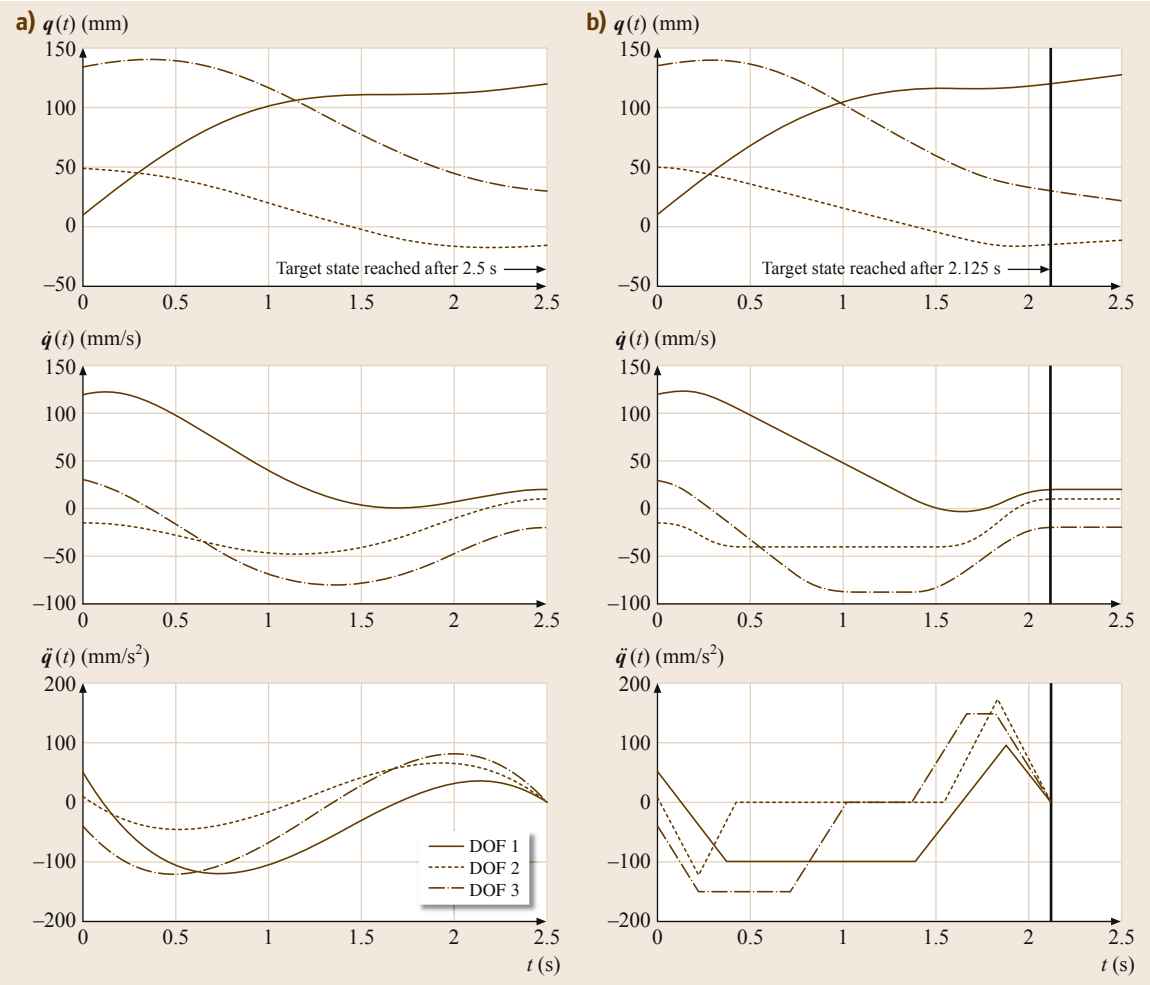


Fig.8.8a,b Two sample trajectories for a three-DOF robot ($n = 3$). The trajectory in **(a)** is represented by quintic splines, and the trajectory in **(b)** by piecewise polynomials

lowing six constraints need to be satisfied for all n joints

$$
\begin{aligned}
q_i(t_0) &= q_{i,\mathrm{start}} = a_{i,0} \\
q_i(t_{\mathrm{trgt}}) &= q_{i,\mathrm{trgt}} = a_{i,0} + a_{i,1}t_{\mathrm{trgt}} + a_{i,2}t_{\mathrm{trgt}}^2 \\
&\quad + a_{i,3}t_{\mathrm{trgt}}^3 + a_{i,4}t_{\mathrm{trgt}}^4 + a_{i,5}t_{\mathrm{trgt}}^5 \\
\dot{q}_i(t_0) &= \dot{q}_{i,0} = a_{i,1} \\
\dot{q}_i(t_{\mathrm{trgt}}) &= \dot{q}_{i,\mathrm{trgt}} = a_{i,1} + 2a_{i,2}t_{\mathrm{trgt}} + 3a_{i,3}t_{\mathrm{trgt}}^2 \\
&\quad + 4a_{i,4}t_{\mathrm{trgt}}^3 + 5a_{i,5}t_{\mathrm{trgt}}^4 \\
\ddot{q}_i(t_0) &= \ddot{q}_{i,0} = 2a_{i,2} \\
\ddot{q}_i(t_{\mathrm{trgt}}) &= \ddot{q}_{i,\mathrm{trgt}} = 2a_{i,2} + 6a_{i,3}t_{\mathrm{trgt}} + 12a_{i,4}t_{\mathrm{trgt}}^2 \\
&\quad + 20a_{i,5}t_{\mathrm{trgt}}^3 \,.
\end{aligned}
$$

A unique closed-form solution can be computed, so that all polynomial coefficients $a_{i,j}\ \forall\ (i,j) \in \{1, \ldots, n\} \times \{0, \ldots, 5\}$ can be determined for one trajectory segment.

Piecewise Polynomials. Polynomials of different degrees can be concatenated to represent a trajectory between an initial state $(\boldsymbol{q}_0, \dot{\boldsymbol{q}}_0, \ddot{\boldsymbol{q}}_0)$ and a target state $(\boldsymbol{q}_{\mathrm{trgt}}, \dot{\boldsymbol{q}}_{\mathrm{trgt}}, \ddot{\boldsymbol{q}}_{\mathrm{trgt}})$. For instance, the classical double-S velocity profile [8.69] with trapezoidal acceleration and deceleration profiles and a cruise-velocity segment in the middle consists of seven polynomials of degrees 3-2-3-1-3-2-3 (m in (8.124)). The right part of Fig. 8.8 shows a trajectory represented by piecewise polynomials. To compute time-optimal trajectories under purely kinematic constraints (e.g., $\dot{\boldsymbol{q}}_{\max}, \ddot{\boldsymbol{q}}_{\max}, \dddot{\boldsymbol{q}}_{\max}$, etc.), piecewise polynomials are used, because they allow always using one signal at its kinematic limit (Fig. 8.8b and [8.70]).

Trigonometric Trajectories. Similarly to (8.124), trigonometric functions can be used to represent harmonic, cycloidal, and elliptic trajectories [8.71, 72]. A simple example for a harmonic trajectory for one joint i is

$$q_i(t) = \frac{q_{i,\mathrm{trgt}} - q_{i,0}}{2}\left(1 - \cos\frac{\pi\,(t - t_0)}{t_{\mathrm{trgt}} - t_0}\right) + q_{i,0}\,. \tag{8.125}$$

While any order of derivatives of trigonometric functions is continuous, they might be discontinuous at t_0 and t_{trgt}.

Other Representations. *Exponential Trajectories* and *Fourier Series Expansions* [8.72] are particularly suited to minimize natural vibrations on robot mechanisms introduced by reference trajectories.

Trajectories and Paths

To draw the connection to Chap. 7 and Parts D and E, trajectories and paths are often tightly coupled.

Trajectories Along Predefined Paths. A path in joint space can be described by a function $\boldsymbol{q}(s(t))$ with $s \in [t_0, t_{\mathrm{trgt}}]$, where the start configuration of the path is $\boldsymbol{q}(t_0)$ and the target configuration is $\boldsymbol{q}(t_{\mathrm{trgt}})$. To move a robot along the path, an appropriate function $s(t)$ needs to be computed that does not violate any of the kinematic and dynamic constraints [8.73–75]. If a path is given in operational space, $\boldsymbol{x}(s(t))$ can be mapped to $\boldsymbol{q}(t)$ (Sect. 8.2).

Multi-Dimensional Trajectories. Instead of using a one-dimensional function $s(t)$ to parameterize a path segment, trajectories can also be described by individual functions for each DOF i to represent $\boldsymbol{q}(s(t))$ or $\boldsymbol{x}(s(t))$, respectively. To connect two arbitrary states, the signals for each individual degree of freedom need to be time-synchronized [8.70], so that all DOFs reach their target state of motion at the very same instant. Those trajectories may also be phase-synchronized [8.76], so that the trajectories of all DOFs are derived from a one master DOF and only scaled by a factor to achieve homothety [8.77]. The two trajectories in Fig. 8.8 are time-synchronized but not phase-synchronized.

Multi-Point Trajectories. If instead of an entirely defined geometric path or a motion to a single way point, an entire series of geometric way points is given, the trajectory that connects all way points in a given state space needs to be computed. Trajectory segments between two way points can be represented with any of the above mentioned representations as long as the position signal and its derivatives up an appropriate order are continuous (at least C^1 continuous). Splines, B-splines, or Bezier splines are used to generate either a reference trajectory or a geometric path, which then can be parameterized with a function $s(t)$.

8.9.4 Trajectory Planning Algorithms

The following part provides an overview of online and offline trajectory planning concepts.

Constraints

Constraints for trajectory planners can manifold:

- *Kinematic*: maximum velocities, accelerations, jerks, etc. and workspace space limits
- *Dynamic*: maximum joint or actuator forces and/or torques
- *Geometric*: no collisions with static and dynamic objects in the workspace
- *Temporal*: reaching a state within a given time interval or at a given time.

These and other constraints may have to be taken into account at the same time. Depending on the robot and the task, additional optimization criteria may be considered (e.g., time-optimality, minimum-jerk, maximum distance to workspace boundaries, minimum energy).

Offline Trajectory Planning

Kahn and *Roth* [8.78] showed results using optimal, linear control theory to achieve a near-time-optimal solution for linearized manipulators. The resulting trajectories are jerk-limited and lead to smaller trajectory-following errors and to less excitation of structural natural frequencies in the system.

The work of *Brady* [8.79] introduced several techniques of trajectory planning in joint space and *Paul* [8.80] and *Taylor* [8.81] published works about the planning of trajectories in Cartesian space in parallel to Brady. *Lin* et al. [8.82] published another purely kinematic approach as did *Castain* and *Paul* [8.69].

Hollerbach [8.83] first introduced the consideration of the nonlinear inverse robot dynamics for the generation of manipulator trajectories.

During the middle of the 1980s, three groups developed techniques for time-optimal trajectory planning for arbitrarily specified paths: *Bobrow* [8.73], *Shin* and *McKay* [8.74], and *Pfeiffer* and *Johanni* [8.75]. Trajectories are composed of three curves: the maximum acceleration curve, the maximum velocity curve, and

the maximum deceleration curve. The proposed algorithms find the intersection points of these three curves.

These algorithms have become the fundament for many follow-up works: *Kyriakopoulos* and *Sridis* [8.84] added minimum-jerk criteria; *Slotine* and *Yang* abandoned the computationally expensive calculation of the maximum velocity curve [8.85]; *Shiller* and *Lu* added methods for handling dynamic singularities [8.86]; *Fiorini* and *Shiller* extended the algorithm for known dynamic environments with moving obstacles [8.87].

Online Trajectory Planning

An online modification of a planned trajectory may have several reasons: (i) The trajectory becomes adapted in order to improve the accuracy with a path specified beforehand; (ii) The robotic system reacts on sensor signals and/or events that cannot be predicted beforehand, because the robot acts in a (partly) unknown and dynamic environment.

Improving Path Accuracy. All previously described off-line trajectory planning methods assume a dynamic model that describes the behavior of the real robot exactly. In practice, this is often not the case, and some robot parameters are only estimated, some dynamic effects remain unmodeled, and system parameters may change during operation. If this is the case, the resulting robot motion is not time-optimal anymore and/or the maximum actuator forces and/or torques are exceeded, which leads to an undesired difference between the specified and the executed path.

Dahl and *Nielsen* [8.88] extended [8.73–75] by adapting the acceleration along the path, so that the underlying trajectory-following controller becomes adapted depending on the current state of motion. The approaches of *Cao* et al. [8.89, 90] use cubic splines to generate smooth paths in joint space with time-optimal trajectories. *Constantinescu* and *Croft* [8.91] suggest a further improvement to the approach of [8.86] with objective to limit the derivative of actuator forces/torques. *Macfarlane* and *Croft* extended this approach further by adding jerk limitations to quintic splines (Fig. 8.8) [8.92].

Sensor-Based Trajectory Adaptation

The last paragraph presented an overview of online trajectory generation methods for improving the path accuracy, while this one focuses on the online consideration of sensor signals, for instance, for the purpose of collision avoidance (VIDEO 757, VIDEO 758) or switching between controllers or control gains (VIDEO 759, VIDEO 761).

In 1988, *Andersson* presented an online trajectory planning for a Ping-Pong-playing PUMA 260 manipulator that computes parameterized quintic polynomials [8.93, 94]. Based on [8.73–75], *Lloyd* and *Hayward* proposed a technique to transition between two different trajectory segments [8.95] using a transition window [8.81]. *Ahn* et al. introduced a method to connect two arbitrary motion states online, which does not take into account kinematic or dynamic constraints [8.96]. *Broquère* et al., *Haschke* et al., and *Kröger* extended this approach for multi-dimensional trajectories, so that kinematic constraints are taken into account [8.70, 97, 98].

Further Reading

Overviews of the domain of robot reference trajectory generation can found in the textbooks of *Biagiotti* and *Melchiorri* [8.72], *Craig* [8.99], *Fu* et al. [8.100], and *Spong* et al. [8.101].

8.10 Digital Implementation

Most controllers introduced in the previous sections are digitally implemented on microprocessors. In this section basic but essential practical issues related to their computer implementation are discussed. When the controller is implemented on a computer control system, the analog inputs are read and the outputs are set with a certain sampling period. This is a drawback compared to analog implementations, since sampling introduces time delays into the control loop. Figure 8.9 shows the overall block diagram of control system with a boxed digital implementation part. When a digital computer is used to implement a control law, it is convenient to divide coding sequence in the interrupt routine into four process routines, as shown in Fig. 8.10. Reading the input signal from sensors and writing the control signal to digital-to-analog (D/A) converters synchronized at the correct frequency is very important. Therefore, these processes are located in the first routine. After saving counter values and extracting D/A values, which are already calculated one step before, the next routine produces reference values. Control routines with filters follow and produce scalar or vector control outputs. Finally, the user interface for checking parameter values is made and will be used for tuning and debugging.

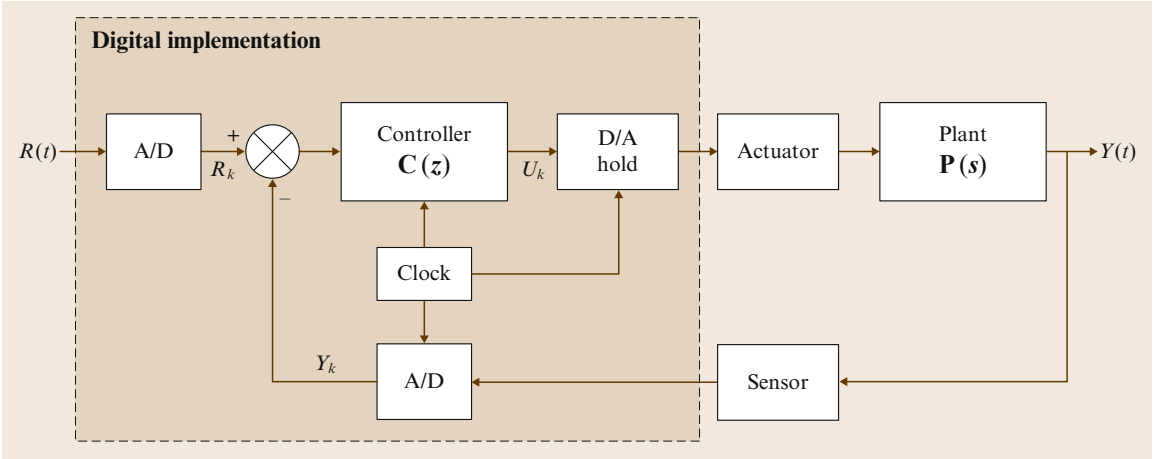


Fig. 8.9 Digital implementation of system control

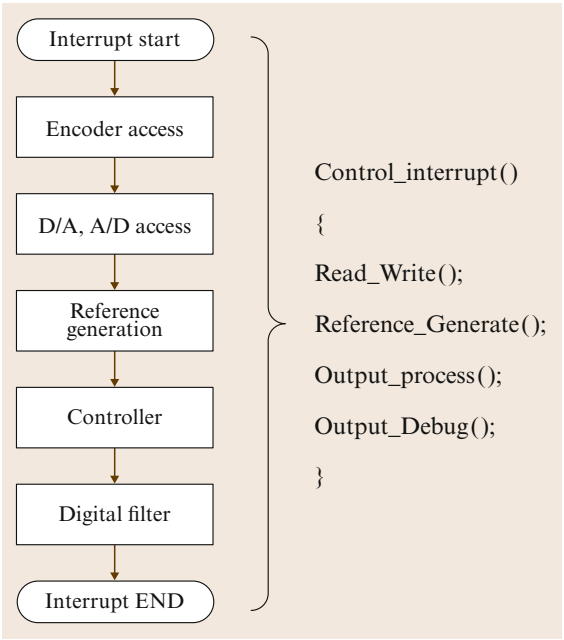


Fig. 8.10 The sequence in the interrupt routine for digital control

8.10.1 Z-Transform for Motion Control Implementation

Continuous-time systems are transformed into discrete-time systems by using the Z-transform. A discrete-time system is used to obtain a mathematical model that gives the behavior of a physical process at the sampling points, although the physical process is still a continuous-time system. A Laplace transform is used for the analysis of control system in the s-domain. In most cases, the design of controllers and filters are done using tools in the s-domain. In order to realize those results in program code, understanding the Z-transform is essential. All controllers and filters designed in the s-domain can be easily translated to a program code through a Z-transform because it has the form of digitized difference sequences.

A PID controller is used as an example. In transfer function form, this controller has the basic structure

$$\frac{Y(s)}{E(s)} = K_\mathrm{P} + \frac{K_\mathrm{I}}{s} + sK_\mathrm{V} \,. \tag{8.126}$$

There are several methods for transformation from the frequency domain to the discrete domain. For stability conservation, backward Euler and Tustin algorithms are often used. Though the Tustin algorithm is known as the more exact one, the backward Euler algorithm is utilized in the following procedure.

After substituting the backward Euler equation into (8.126),

$$s \cong \frac{1 - z^{-1}}{T} \,,$$

the following discrete form is produced

$$\frac{Y(z)}{E(z)} = \frac{\alpha + \beta z^{-1} + \gamma z^{-2}}{T(1 - z^{-1})} \,, \tag{8.127}$$

where

$$\alpha = K_\mathrm{I} T^2 + K_\mathrm{P} T + K_\mathrm{V} \,,$$
$$\beta = -K_\mathrm{P} T - 2K_\mathrm{V} \,,$$
$$\gamma = K_\mathrm{V} \,.$$

Sometimes a differentiator s in the PID controller makes the implementation infeasible when the mea-

surement noise is severe. One can remedy the controller (8.126) by attaching a lowpass filter with filter time constant σ

$$\frac{Y(s)}{E(s)} = K_\mathrm{P} + \frac{K_\mathrm{I}}{s} + \frac{s}{\sigma s + 1} K_\mathrm{V} \,. \tag{8.128}$$

Again, substituting the backward Euler equation into (8.128) produces

$$\frac{Y(z)}{E(z)} = \frac{\alpha + \beta z^{-1} + \gamma z^{-2}}{1 - \delta z^{-1} - \psi z^{-2}} \,, \tag{8.129}$$

where

$$\begin{aligned}
\alpha &= K_\mathrm{P} + K_\mathrm{I} T + \frac{K_\mathrm{V}}{\sigma + T} \,, \\
\beta &= -\frac{(2\sigma + T)K_\mathrm{P} + \sigma T K_\mathrm{I} + 2K_\mathrm{V}}{\sigma + T} \,, \\
\gamma &= \frac{\sigma K_\mathrm{P} + K_\mathrm{V}}{\sigma + T} \,, \\
\delta &= \frac{2\sigma + T}{\sigma + T} \,, \\
\psi &= -\frac{\sigma}{\sigma + T} \,,
\end{aligned}$$

in which the filter time constant σ is determined from a cutoff frequency f_c[Hz] for removing noises such as $\sigma = \frac{1}{2\pi f_\mathrm{c}}$.

8.10.2 Digital Control for Coding

Inverse Z-transform produces a difference equation for digital control. Furthermore the difference equation can be directly converted into the control program code. Since the inverse Z-transform of $Y(z)$ is y_k and z^{-1} implies the previous sample time, $z^{-1}Y(z) = y_{k-1}$ and $z^{-2}Y(z) = y_{k-2}$.

Now, the PID controller expressed by (8.127) is rearranged using the difference equation

$$\begin{aligned}
T(y_k - y_{k-1}) &= \alpha e_k + \beta e_{k-1} + \gamma e_{k-2} \,, \\
y_k - y_{k-1} &= \frac{1}{T}(\alpha e_k + \beta e_{k-1} + \gamma e_{k-2}) \,.
\end{aligned} \tag{8.130}$$

For pratical use, the PID controller can be directly coded in the program as follows

$$y_k = K_{\mathrm{P,c}} e_k + K_{\mathrm{V,c}} e_k^v + K_{\mathrm{I,c}} e_k^i \,,$$

where

$$\begin{aligned}
e_k &= p_{k,\mathrm{desired}} - p_k \,, \\
e_k^v &= v_{k,\mathrm{desired}} - v_k \\
&= (p_{k,\mathrm{desired}} - p_{k-1,\mathrm{desired}}) - (p_k - p_{k-1}) \\
&= e_k - e_{k-1} \,, \\
e_k^i &= e_{k-1}^i + e_k = \sum_{j=0}^{k} e_k \,,
\end{aligned}$$

in which p_k is the present position, v_k is the present velocity, desired means reference to be followed, and c means coded form for digitial control. Now let us obtain the difference between the present control output and the previous one

$$\begin{aligned}
&y_k - y_{k-1} \\
&\quad = [K_{\mathrm{P,c}} e_k - K_{\mathrm{P,c}} e_{k-1}] \\
&\qquad + [K_{\mathrm{V,c}}(e_k - e_{k-1}) - K_{\mathrm{V,c}}(e_{k-1} - e_{k-2})] \\
&\qquad + [K_{\mathrm{I,c}} e_k^i - K_{\mathrm{I,c}} e_{k-1}^i] \\
&\quad = (K_{\mathrm{P,c}} + K_{\mathrm{V,c}} + K_{\mathrm{I,c}}) e_k \\
&\qquad - (K_{\mathrm{P,c}} + 2K_{\mathrm{V,c}}) e_{k-1} + K_{\mathrm{V,c}} e_{k-2} \,.
\end{aligned} \tag{8.131}$$

Comparing the parameters in (8.130) and (8.131), one obtains

$$\begin{aligned}
\frac{\alpha}{T} &= K_\mathrm{P} + \frac{K_\mathrm{V}}{T} + K_\mathrm{I} T = (K_{\mathrm{P,c}} + K_{\mathrm{V,c}} + K_{\mathrm{I,c}}) \,, \\
\frac{\beta}{T} &= -K_\mathrm{P} - \frac{2K_\mathrm{V}}{T} = -(K_{\mathrm{P,c}} + 2K_{\mathrm{V,c}}) \,, \\
\frac{\gamma}{T} &= \frac{K_\mathrm{V}}{T} = K_{\mathrm{V,c}} \,,
\end{aligned}$$

which shows that there is a relation between the designed and coded forms of the gains

$$\begin{aligned}
K_{\mathrm{P,c}} &= K_\mathrm{P} \,, \\
K_{\mathrm{V,c}} &= \frac{K_\mathrm{V}}{T} \,, \\
K_{\mathrm{I,c}} &= K_\mathrm{I} T \,.
\end{aligned} \tag{8.132}$$

As the sampling frequency is increased in the same system, the coded K_V gain should be increased and the coded K_I gain should be decreased. Using this method, the designed controller can be coded in a digital signal processor (DSP) or microprocessor. However, sufficient analysis and simulation for control algorithms should be performed beforehand to obtain successful control system performance.

In addition, the PID controller with lowpass filter (8.129) can be implemented as

$$\begin{aligned}
y_k - \delta y_{k-1} - \psi y_{k-2} &= \alpha e_k + \beta e_{k-1} + \gamma e_{k-2} \,, \\
y_k &= \delta y_{k-1} + \psi y_{k-2} + \alpha e_k + \beta e_{k-1} + \gamma e_{k-2} \,.
\end{aligned} \tag{8.133}$$

Using the same procedures, one can arrive at the similar control program code for digital control.

PID Control Experiment (Multimedia)
According as the gains change, the performance variations of PID controller implemented in the digital control system are shown in the multimedia source to help readers' understanding VIDEO 25.

8.11 Learning Control

Since many robotic applications, such as pick-and-place operations, paintings, and circuit-board assembly, involve repetitive motions, it is natural to consider the use of data gathered in previous cycles to try to improve the performance of the manipulator in subsequent cycles. This is the basic idea of repetitive control or learning control. Consider the robot model given in Sect. 8.1 and suppose that one is given a desired joint trajectory $\boldsymbol{q}_{\mathrm{d}}(t)$ on a finite time interval $0 \le t \le T$. The reference trajectory $\boldsymbol{q}_{\mathrm{d}}$ is used in repeated trails of the manipulator, assuming either that the trajectory is periodic, $\boldsymbol{q}_{\mathrm{d}}(T) = \boldsymbol{q}_{\mathrm{d}}(0)$, (repetitive control) or that the robot is reinitialized to lie on the desired trajectory at the beginning of each trail (learning control). Hereafter, we use the term learning control to mean either repetitive or learning control.

8.11.1 Pure P-Type Learning Control

Let $\boldsymbol{\tau}_k$ be the input torque during the k-th cycle, which produces an output $\boldsymbol{q}_k(t), 0 \le t \le T_{\mathrm{bnd}}$. Now, let us consider the following set of assumptions:

- Assumption 1: Every trial ends at a fixed time of duration $T_{\mathrm{bnd}} > 0$.
- Assumption 2: Repetition of the initial setting is satisfied.
- Assumption 3: Invariance of the system dynamics is ensured throughout these repeated trails.
- Assumption 4: Every output $\boldsymbol{q}_k$ can be measured and thus the error signal $\Delta\boldsymbol{q}_k = \boldsymbol{q}_k - \boldsymbol{q}_{\mathrm{d}}$ can be utilized in the construction of the next input $\boldsymbol{\tau}_{k+1}$.
- Assumption 5: The dynamics of the robot manipulators is invertible.

The learning control problem is to determine a recursive learning law $\boldsymbol{L}$

$$\boldsymbol{\tau}_{k+1} = \boldsymbol{L}[\boldsymbol{\tau}_k(t), \Delta\boldsymbol{q}_k(t)] \,, \quad 0 \le t \le T_{\mathrm{bnd}} \,, \tag{8.134}$$

where $\Delta\boldsymbol{q}_k(t) = \boldsymbol{q}_k(t) - \boldsymbol{q}_{\mathrm{d}}(t)$, such that $\|\Delta\boldsymbol{q}_k\| \to 0$ as $k \to \infty$ in some suitably defined function norm, $\|\cdot\|$. The initial control input can be any control input that produces a stable output, such as PD control. Such learning control schemes are attractive because accurate models of the dynamics need not be known a priori.

Several approaches have been used to generate a suitable learning law $\boldsymbol{L}$ and to prove convergence of the output error. A pure P-type learning law is one of the form

$$\boldsymbol{\tau}_{k+1}(t) = \boldsymbol{\tau}_k(t) - \boldsymbol{\Phi}\Delta\boldsymbol{q}_k(t) \,, \tag{8.135}$$

and is given this name because the correction term for the input torque at each iteration is proportional to the error $\Delta\boldsymbol{q}_k$. Now let $\boldsymbol{\tau}_d$ be defined by the computed-torque control, i. e.,

$$\begin{aligned}\boldsymbol{\tau}_d(t) = {} & \mathbf{H}[\boldsymbol{q}_{\mathrm{d}}(t)]\ddot{\boldsymbol{q}}_{\mathrm{d}}(t) + \mathbf{C}[\boldsymbol{q}_{\mathrm{d}}(t), \dot{\boldsymbol{q}}_{\mathrm{d}}(t)]\dot{\boldsymbol{q}}_{\mathrm{d}}(t) \\ & + \boldsymbol{\tau}_{\mathrm{g}}[\boldsymbol{q}_{\mathrm{d}}(t)] \,.\end{aligned} \tag{8.136}$$

One should recall that the function $\boldsymbol{\tau}_k$ actually does not need to be computed; it is sufficient to know that it exists. Considering the P-type learning control law, we have

$$\Delta\boldsymbol{\tau}_{k+1}(t) = \Delta\boldsymbol{\tau}_k(t) - \boldsymbol{\Phi}\Delta\boldsymbol{q}_k(t) \,, \tag{8.137}$$

where $\Delta\boldsymbol{\tau}_k(t) = \boldsymbol{\tau}_k(t) - \boldsymbol{\tau}_d(t)$, so that

$$\|\Delta\boldsymbol{\tau}_{k+1}(t)\|^2 \le \|\Delta\boldsymbol{\tau}_k(t)\|^2 - \beta\|\boldsymbol{\Phi}\Delta\boldsymbol{q}_k(t)\|^2 \tag{8.138}$$

provided there exist positive constant λ and β such that

$$\int_0^{T_{\mathrm{bnd}}} \mathrm{e}^{-\lambda t}\Delta\boldsymbol{q}_k{}^{\mathrm{T}}\Delta\boldsymbol{\tau}_k(t)\mathrm{d}t \ge \frac{1+\beta}{2}\|\boldsymbol{\Phi}\Delta\boldsymbol{q}_k(t)\|^2 \tag{8.139}$$

for all k. It then follows from the inequality above that $\Delta q_k \to 0$ in the norm sense as $k \to \infty$. Detailed stability analysis of this control scheme is given in [8.102, 103].

8.11.2 P-Type Learning Control with a Forgetting Factor

Although pure P-type learning control achieves the desired goal, several strict assumptions may be not valid in actual implementations, for example, there may be an initial setting error. Furthermore, there may be small but

nonrepeatable fluctuations of dynamics. Finally, there may exit a (bounded) measurement noise $\boldsymbol{\xi}_k$ such that

$$\Delta \boldsymbol{q}_k(t) + \boldsymbol{\xi}_k(t) = [\boldsymbol{q}_k(t) + \boldsymbol{\xi}_k(t)] - \boldsymbol{q}_\mathrm{d}(t) \,. \qquad (8.140)$$

Thus the learning control scheme may fail. In order to enhance the robustness of P-type learning control, a forgetting factor is introduced in the form

$$\boldsymbol{\tau}_{k+1}(t) = (1-\alpha)\boldsymbol{\tau}_k(t) + \alpha\boldsymbol{\tau}_0(t) - \boldsymbol{\Phi}[\Delta \boldsymbol{q}_k(t) + \boldsymbol{\xi}_k(t)] \,. \qquad (8.141)$$

The original idea of using a forgetting factor in learning control originated with [8.104].

It has been rigorously proven that P-type learning control with a forgetting factor guarantees convergence to a neighborhood of the desired one of size $O(\alpha)$. Moreover, if the content of a long-term memory is refreshed after every k trials, where k is of $O(1/\alpha)$, then the trajectories converge to an ε-neighborhood of the desired control goal. The size of ε is dependent on the magnitude of the initial setting error, the nonrepeatable fluctuations of the dynamics, and the measurement noise. For a detailed stability investigation, please refer to [8.105, 106].

8.11.3 Summary

By applying learning control, the performance of repetitive tasks (such as painting or pick-and-place operation) is improved by utilizing data gathered in the previous cycles. In this section, two learning control schemes were introduced. First, pure P-type learning control and its robustness problem were described. Then P-type learning control with a forgetting factor was presented, enhancing the robustness of learning control.

Further Reading

Rigorous and refined exploration of learning control is first discussed independently in [8.2, 12].

Video-References

VIDEO 25 Gain change of the PID controller
available from http://handbookofrobotics.org/view-chapter/08/videodetails/25

VIDEO 757 Safe human-robot cooperation
available from http://handbookofrobotics.org/view-chapter/08/videodetails/757

VIDEO 758 Virtual whiskers – Highly responsive robot collision avoidance
available from http://handbookofrobotics.org/view-chapter/08/videodetails/758

VIDEO 759 JediBot – Experiments in human-robot sword-fighting
available from http://handbookofrobotics.org/view-chapter/08/videodetails/759

VIDEO 760 Different jerk limits of robot arm trajectories
available from http://handbookofrobotics.org/view-chapter/08/videodetails/760

VIDEO 761 Sensor-based online trajectory generation
available from http://handbookofrobotics.org/view-chapter/08/videodetails/761

References

8.1 C. Canudas de Wit, B. Siciliano, G. Bastin: *Theory of Robot Control* (Springer, London 1996)

8.2 J.J. Craig: Adaptive Control of Mechanical Manipulators, Ph.D. Thesis (UMI Dissertation Information Service, Ann Arbor 1986)

8.3 R.J. Schilling: *Fundametals of Robotics: Analysis and Control* (Prentice Hall, Englewood Cliffs 1989)

8.4 L. Sciavicco, B. Siciliano: *Modeling and Control of Robot Manipulator* (McGraw-Hill, New York 1996)

8.5 M.W. Spong, M. Vidyasagar: *Robot Dynamics and Control* (Wiley, New York 1989)

8.6 M.W. Spong, F.L. Lewis, C.T. Abdallah (Eds.): *Robot Control* (IEEE, New York 1989)

8.7 C.H. An, C.G. Atkeson, J.M. Hollerbach: *Model-Based Control of a Robot Manipulator* (MIT Press, Cambridge, 1988)

8.8 R.M. Murray, Z. Xi, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)

8.9 T. Yoshikawa: *Foundations of Robotics* (MIT Press, Cambridge 1990)

8.10 O. Khatib: A unified approach for motion and force control of robot manipulators: The operational space formulation, IEEE J. Robotics Autom. **3**(1), 43–53 (1987)

8.11 J.Y.S. Luh, M.W. Walker, R.P.C. Paul: Resolved-acceleration control of mechanical manipulator, IEEE Trans. Autom. Control **25**(3), 468–474 (1980)

8.12 S. Arimoto, F. Miyazaki: Stability and robustness of PID feedback control for robot manipulators of sensory capability. In: *Robotics Research*, ed.

by M. Brady, R. Paul (MIT Press, Cambridge 1984) pp. 783–799
8.13 L.C. Fu: Robust adaptive decentralized control of robot manipulators, IEEE Trans. Autom. Control **37**(1), 106–110 (1992)
8.14 H. Seraji: Decentralized adaptive control of manipulators: Theory, simulation, and experimentation, IEEE Trans. Robotics Autom. **5**(2), 183–201 (1989)
8.15 J.G. Ziegler, N.B. Nichols: Optimum settings for automatic controllers, Trans. ASME **64**, 759–768 (1942)
8.16 Y. Choi, W.K. Chung: *PID Trajectory Tracking Control for Mechanical Systems*, Lecture Notes in Control and Information Sciences, Vol. 289 (Springer, New York 2004)
8.17 R. Kelly: PD control with desired gravity compensation of robot manipulators: A review, Int. J. Robotics Res. **16**(5), 660–672 (1997)
8.18 M. Takegaki, S. Arimoto: A new feedback method for dynamic control of manipulators, Trans. ASME J. Dyn. Syst. Meas, Control **103**, 119–125 (1981)
8.19 P. Tomei: Adaptive PD controller for robot manipulators, IEEE Trans. Robotics Autom. **7**(4), 565–570 (1991)
8.20 R. Ortega, A. Loria, R. Kelly: A semi-globally stable output feedback PI^2 D regulator for robot manipulators, IEEE Trans. Autom. Control **40**(8), 1432–1436 (1995)
8.21 D. Angeli: Input-to-State stability of PD-controlled robotic systems, Automatica **35**, 1285–1290 (1999)
8.22 J.A. Ramirez, I. Cervantes, R. Kelly: PID regulation of robot manipulators: Stability and performance, Syst. Control Lett. **41**, 73–83 (2000)
8.23 R. Kelly: Global positioning of robot manipulators via PD control plus a class of nonlinear integral actions, IEEE Trans. Autom. Control **43**(7), 934–937 (1998)
8.24 Z. Qu, J. Dorsey: Robust tracking control of robots by a linear feedback law, IEEE Trans. Autom. Control **36**(9), 1081–1084 (1991)
8.25 H. Berghuis, H. Nijmeijer: Robust control of robots via linear estimated state feedback, IEEE Trans. Autom. Control **39**(10), 2159–2162 (1994)
8.26 Y. Choi, W.K. Chung, I.H. Suh: Performance and $\mathcal{H}_\infty$ optimality of PID trajectory tracking controller for Lagrangian systems, IEEE Trans. Robotics Autom. **17**(6), 857–869 (2001)
8.27 K. Aström, T. Hagglund: *PID Controllers: Theory, Design, and Tuning* (Instrument Society of America, Research Triangle Park 1995)
8.28 C.C. Yu: *Autotuning of PID Controllers: Relay Feedback Approach* (Springer, London 1999)
8.29 F.L. Lewis, C.T. Abdallah, D.M. Dawson: *Control of Robot Manipulators* (Macmillan, New York 1993)
8.30 A. Isidori: *Nonlinear Control Systems: An Introduction*, Lecture Notes in Control and Information Sciences, Vol. 72 (Springer, New York 1985)
8.31 H. Berghuis, H. Nijmeijer: A passivity approach to controller–observer design for robots, IEEE Trans. Robotics Autom. **9**, 740–754 (1993)
8.32 J.J. Slotine, W. Li: On the adaptive control of robot manipulators, Int. J. Robotics Res. **6**(3), 49–59 (1987)
8.33 G. Liu, A.A. Goldenberg: Comparative study of robust saturation–based control of robot manipulators: analysis and experiments, Int. J. Robotics Res. **15**(5), 473–491 (1996)
8.34 D.M. Dawson, M. Grabbe, F.L. Lewis: Optimal control of a modified computed–torque controller for a robot manipulator, Int. J. Robotics Autom. **6**(3), 161–165 (1991)
8.35 D.M. Dawson, Z. Qu, J. Duffie: Robust tracking control for robot manipulators: Theory, simulation and implementation, Robotica **11**, 201–208 (1993)
8.36 A. Jaritz, M.W. Spong: An experimental comparison of robust control algorithms on a direct drive manipulator, IEEE Trans. Control Syst. Technol. **4**(6), 627–640 (1996)
8.37 A. Isidori: *Nonlinear Control Systems*, 3rd edn. (Springer, New York 1995)
8.38 J.J. Slotine, W. Li: *Applied Nonlinear Control* (Prentice Hall, Englewood Cliffs 1991)
8.39 W.J. Rugh: *Linear System Theory*, 2nd edn. (Prentice Hall, Upper Saddle River 1996)
8.40 M.W. Spong, M. Vidyasagar: Robust microprocessor control of robot manipulators, Automatica **23**(3), 373–379 (1987)
8.41 H.K. Khalil: *Nonlinear Systems*, 3rd edn. (Prentice Hall, Upper Saddle River 2002)
8.42 M. Vidysagar: *Nonlinear Systems Analysis*, 2nd edn. (Prentice Hall, Englewood Ciffs 1993)
8.43 J.T. Wen: A unified perspective on robot control: The energy Lyapunov function approach, Int. J. Adapt. Control Signal Process. **4**, 487–500 (1990)
8.44 R. Ortega, M.W. Spong: Adaptive motion control of rigid robots: A tutorial, Automatica **25**(6), 877–888 (1989)
8.45 N. Sadegh, R. Horowitz: Stability and robustness analysis of a class of adaptive contollers for robotic manipulators, Int. J. Robotics Res. **9**(3), 74–92 (1990)
8.46 S. Dubowsky, D.T. DesForges: The application of model-reference adaptive control to robotic manipulators, ASME J. Dyn. Syst. Meas. Control **37**(1), 106–110 (1992)
8.47 S.H. Hsu, L.C. Fu: A fully adaptive decentralized control of robot manipulators, Automatica **42**, 1761–1767 (2008)
8.48 A. Balestrino, G. de Maria, L. Sciavicco: An adaptive model following control for robotic manipulators, ASME J. Dyn. Syst. Meas. Control **105**, 143–151 (1983)
8.49 S. Nicosia, P. Tomei: Model reference adaptive control algorithms for industrial robots, Automatica **20**, 635–644 (1984)
8.50 R. Horowitz, M. Tomizuka: An adaptive control scheme for mechanical manipulators-Compensation of nonlinearity and decoupling control, ASME J. Dyn. Syst. Meas. Control **108**, 127–135 (1986)
8.51 I.D. Laudau: *Adaptive Control: The Model Reference Approach* (Dekker, New York 1979)
8.52 R. Lozano, C. Canudas de Wit: Passivity based adaptive control for mechanical manipulators using LS type estimation, IEEE Trans. Autom. Control **35**(12), 1363–1365 (1990)

8.53 B. Brogliato, I.D. Laudau, R. Lozano: Passive least squares type estimation algorithm for direct adaptive control, Int. J. Adapt. Control Signal Process. **6**, 35–44 (1992)

8.54 R. Johansson: Adaptive control of robot manipulator motion, IEEE Trans. Robotics Autom. **6**(4), 483–490 (1990)

8.55 M.W. Walker: Adaptive control of manipulators containing closed kinematic loops, IEEE Trans. Robotics Autom. **6**(1), 10–19 (1990)

8.56 J.S. Reed, P.A. Ioannou: Instability analysis and robust adaptive control of robotic manipulators, IEEE Trans. Autom. Control **5**(3), 74–92 (1989)

8.57 G. Tao: On robust adaptive control of robot manipulators, Automatica **28**(4), 803–807 (1992)

8.58 H. Berghuis, R. Ogata, H. Nijmeijer: A robust adaptive controller for robot manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1992) pp. 1876–1881

8.59 R. Johansson: Quadratic optimization of motion coordination and control, IEEE Trans. Autom. Control **35**(11), 1197–1208 (1990)

8.60 Z. Qu, D.M. Dawson: *Robust Tracking Control of Robot Manipulators* (IEEE, Piscataway 1996)

8.61 P. Dorato, C. Abdallah, V. Cerone: *Linear-Quadratic Control* (Prentice Hall, Upper Saddle River 1995)

8.62 A. Locatelli: *Optimal Control: An Introduction* (Birkhäuser, Basel 2001)

8.63 A. Isidori: Feedback control of nonlinear systems, Int. J. Robust Nonlin. Control **2**, 291–311 (1992)

8.64 A.J. der van Schaft: Nonlinear state space $\mathcal{H}_\infty$ control theory. In: *Essays on Control: Perspective in Theory and its Applications*, ed. by H.L. Trentelman, J.C. Willems (Birkhäuser, Basel 1993) pp. 153–190

8.65 A.J. der van Schaft: L_2-gain analysis of nonlinear systems and nonlinear state feedback $\mathcal{H}_\infty$ control, IEEE Trans. Autom. Control **37**(6), 770–784 (1992)

8.66 J. Park, W.K. Chung, Y. Youm: Analytic nonlinear $\mathcal{H}_\infty$ inverse-optimal control for Euler–Lagrange system, IEEE Trans. Robotics Autom. **16**(6), 847–854 (2000)

8.67 B.S. Chen, T.S. Lee, J.H. Feng: A nonlinear $\mathcal{H}_\infty$ control design in robotics systems under parametric perturbation and external disturbance, Int. J. Control **59**(12), 439–461 (1994)

8.68 J. Park, W.K. Chung: Design of a robust $\mathcal{H}_\infty$ PID control for industrial manipulators, ASME J. Dyn. Syst. Meas. Control **122**(4), 803–812 (2000)

8.69 R.H. Castain, R.P. Paul: An on-line dynamic trajectory generator, Int. J. Robotics Res. **3**(1), 68–72 (1984)

8.70 T. Kröger: *On-Line Trajectory Generation in Robotic Systems*, Springer Tracts in Advanced Robotics, Vol. 58 (Springer, Berlin, Heidelberg 2010)

8.71 D. Simon, C. Isik: A trigonometric trajectory generator for robotic arms, Int. J. Control **57**(3), 505–517 (1993)

8.72 L. Biagiotti, C. Melchiorri: *Trajectory Planning for Automatic Machines and Robots* (Springer, Berlin, Heidelberg 2008)

8.73 J.E. Bobrow: Optimal robot path planning using the minimum-time criterion, IEEE J. Robotics Autom. **4**(4), 443–450 (1988)

8.74 K.G. Shin, N.D. McKay: Minimum-time control of robotic manipulators with geometric path constraints, IEEE Trans. Autom. Control **30**(5), 531–541 (1985)

8.75 F. Pfeiffer, R. Johanni: A concept for manipulator trajectory planning, Proc. Int. IEEE Conf. Robotics Autom. (ICRA) (1986) pp. 1399–1405

8.76 W. Khalil, E. Dombre: Trajectory generation. In: *Modeling, Identification and Control of Robots*, ed. by W. Khalil, E. Dombre (Butterworth-Heinemann, Oxford 2004)

8.77 A.I. Kostrikin, Y.I. Manin: *Linear Algebra and Geometry* (Gordon and Breach Sci. Publ., Amsterdam 1997)

8.78 M.E. Kahn, B. Roth: The near-minimum-time control of open-loop articulated kinematic chains, ASME J. Dyn. Syst. Meas. Control **93**, 164–172 (1971)

8.79 M. Brady: Trajectory planning. In: *Robot Motion: Planning and Control*, ed. by M. Brady, J.M. Hollerbach, T.L. Johnson, T. Lozano-Pérez, M.T. Mason (MIT Press, Cambridge 1982)

8.80 R.P.C. Paul: Manipulator cartesian path control. In: *Robot Motion: Planning and Control*, ed. by M. Brady, J.M. Hollerbach, T.L. Johnson, T. Lozano-Pérez, M.T. Mason (MIT Press, Cambridge 1982)

8.81 R.H. Taylor: Planning and execution of straight-line manipulator trajectories. In: *Robot Motion: Planning and Control*, ed. by M. Brady, J.M. Hollerbach, T.L. Johnson, T. Lozano-Pérez, M.T. Mason (MIT Press, Cambridge 1982)

8.82 C.-S. Lin, P.-R. Chang, J.Y.S. Luh: Formulation and optimization of cubic polynomial joint trajectories for industrial robots, IEEE Trans. Autom. Control **28**(12), 1066–1074 (1983)

8.83 J.M. Hollerbach: Dynamic scaling of manipulator trajectories, ASME J. Dyn. Syst. Meas. Control **106**(1), 102–106 (1984)

8.84 K.J. Kyriakopoulos, G.N. Sridis: Minimum jerk path generation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1988) pp. 364–369

8.85 J.-J.E. Slotine, H.S. Yang: Improving the efficiency of time-optimal path-following algorithms, IEEE Trans. Robotics Autom. **5**(1), 118–124 (1989)

8.86 Z. Shiller, H.-H. Lu: Computation of path constrained time optimal motions with dynamic singularities, ASME J. Dyn. Syst. Meas. Control **114**(1), 34–40 (1992)

8.87 P. Fiorini, Z. Shiller: Time optimal trajectory planning in dynamic environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 1553–1558

8.88 O. Dahl, L. Nielsen: Torque limited path following by on-line trajectory time scaling, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 1122–1128

8.89 B. Cao, G.I. Dodds, G.W. Irwin: Time-optimal and smooth constrained path planning for robot manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 1853–1858

8.90 B. Cao, G.I. Dodds, G.W. Irwin: A practical approach to near time-optimal inspection-task-sequence

planning for two cooperative industrial robot arms, Int. J. Robotics Res. **17**(8), 858–867 (1998)
8.91 D. Constantinescu, E.A. Croft: Smooth and time-optimal trajectory planning for industrial manipulators along specified paths, J. Robotics Syst. **17**(5), 233–249 (2000)
8.92 S. Macfarlane, E.A. Croft: Jerk-bounded manipulator trajectory planning: Design for real-time applications, IEEE Trans. Robotics Autom. **19**(1), 42–52 (2003)
8.93 R.L. Andersson: *A Robot Ping-Pong Player: Experiment in Real-Time Intelligent Control* (MIT Press, Cambridge 1988)
8.94 R.L. Andersson: Aggressive trajectory generator for a robot ping-pong player, IEEE Control Syst. Mag. **9**(2), 15–21 (1989)
8.95 J. Lloyd, V. Hayward: Trajectory generation for sensor-driven and time-varying tasks, Int. J. Robotics Res. **12**(4), 380–393 (1993)
8.96 K. Ahn, W.K. Chung, Y. Yourn: Arbitrary states polynomial-like trajectory (ASPOT) generation, Proc. IEEE 30th Annu. Conf. Ind. Electron. Soc. (2004) pp. 123–128
8.97 X. Broquère, D. Sidobre, I. Herrera-Aguilar: Soft motion trajectory planner for service manipulator robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 2808–2813
8.98 R. Haschke, E. Weitnauer, H. Ritter: On-line planning of time-optimal, jerk-limited trajectories, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 3248–3253
8.99 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Prentice Hall, Upper Saddle River 2003)
8.100 K.S. Fu, R.C. Gonzalez, C.S.G. Lee: *Robotics: Control, Sensing, Vision and Intelligence* (McGraw-Hill, New York 1988)
8.101 M.W. Spong, S.A. Hutchinson, M. Vidyasagar: *Robot Modeling and Control* (Wiley, New York 2006)
8.102 S. Arimoto: Mathematical theory or learning with application to robot control. In: *Adaptive and Learning Control*, ed. by K.S. Narendra (Plenum, New York 1986) pp. 379–388
8.103 S. Kawamura, F. Miyazaki, S. Arimoto: Realization of robot motion based on a learning method, IEEE Trans. Syst. Man. Cybern. **18**(1), 126–134 (1988)
8.104 G. Heinzinger, D. Frewick, B. Paden, F. Miyazaki: Robust learning control, Proc. IEEE Int. Conf. Decis. Control (1989)
8.105 S. Arimoto: Robustness of learning control for robot manipulators, Proc. IEEE Int. Conf. Decis. Control (1990) pp. 1523–1528
8.106 S. Arimoto, T. Naiwa, H. Suzuki: Selective learning with a forgetting factor for robotic motion control, Proc. IEEE Int. Conf. Decis. Control (1991) pp. 728–733

Multimedia Contents

9. Force Control

Luigi Villani, Joris De Schutter

A fundamental requirement for the success of a manipulation task is the capability to handle the physical contact between a robot and the environment. Pure motion control turns out to be inadequate because the unavoidable modeling errors and uncertainties may cause a rise of the contact force, ultimately leading to an unstable behavior during the interaction, especially in the presence of rigid environments. Force feedback and force control becomes mandatory to achieve a robust and versatile behavior of a robotic system in poorly structured environments as well as safe and dependable operation in the presence of humans. This chapter starts from the analysis of *indirect force control* strategies, conceived to keep the contact forces limited by ensuring a suitable compliant behavior to the end effector, without requiring an accurate model of the environment. Then the problem of *interaction tasks* modeling is analyzed, considering both the case of a rigid environment and the case of a compliant environment. For the specification of an interaction task, natural constraints set by the task geometry and artificial constraints set by the control strategy are established, with respect to suitable task frames. This formulation is the essential premise to the synthesis of *hybrid force/motion control* schemes.

9.1 **Background** 195
9.1.1 From Motion Control to Interaction Control 196
9.1.2 From Indirect Force Control to Hybrid Force/Motion Control 197

9.2 **Indirect Force Control** 198
9.2.1 Stiffness Control 198
9.2.2 Impedance Control 201

9.3 **Interaction Tasks** 205
9.3.1 Rigid Environment 205
9.3.2 Compliant Environment 208
9.3.3 Task Specification 209
9.3.4 Sensor-Based Contact Model Estimation 211

9.4 **Hybrid Force/Motion Control** 211
9.4.1 Acceleration-Resolved Approach 211
9.4.2 Passivity-Based Approach 214
9.4.3 Velocity-Resolved Approach 215

9.5 **Conclusions and Further Reading** 216
9.5.1 Indirect Force Control 216
9.5.2 Task Specification 217
9.5.3 Hybrid Force/Motion Control 217

Video-References 217

References 218

9.1 Background

Research on robot force control has flourished in the past three decades. Such a wide interest is motivated by the general desire of providing robotic systems with enhanced sensory capabilities. Robots using force, touch, distance, and visual feedback are expected to autonomously operate in unstructured environments other than the typical industrial shop floor.

Since the early work on telemanipulation, the use of force feedback was conceived to assist the human operator in the remote handling of objects with a slave manipulator. More recently, cooperative robot systems have been developed where two or more manipulators (viz. the fingers of a dexterous robot hand) are to be controlled so as to limit the exchanged forces

and avoid squeezing of a commonly held object. Force control plays a fundamental role also in the achievement of robust and versatile behavior of robotic systems in open-ended environments, providing intelligent response in unforeseen situations and enhancing human–robot interaction.

9.1.1 From Motion Control to Interaction Control

Control of the physical interaction between a robot manipulator and the environment is crucial for the successful execution of a number of practical tasks where the robot end-effector has to manipulate an object or perform some operation on a surface. Typical examples in industrial settings include polishing, deburring, machining or assembly. A complete classification of possible robot tasks, considering also nonindustrial applications, is practically infeasible in view of the large variety of cases that may occur, nor would such a classification be really useful to find a general strategy to control the interaction with the environment.

During contact, the environment may set constraints on the geometric paths that can be followed by the end-effector, denoted as *kinematic constraints*. This situation, corresponding to the contact with a stiff surface, is generally referred to as *constrained motion*. In other cases, the contact task is characterized by a dynamic interaction between the robot and the environment that can be inertial (as in pushing a block), dissipative (as in sliding on a surface with friction) or elastic (as in pushing against an elastically compliant wall). In all these cases, the use of a pure motion control strategy for controlling interaction is prone to failure, as explained below.

Successful execution of an interaction task with the environment by using motion control could be obtained only if the task were accurately planned. This would in turn require an accurate model of both the robot manipulator (kinematics and dynamics) and the environment (geometry and mechanical features). A manipulator model may be known with sufficient precision, but a detailed description of the environment is difficult to obtain.

To understand the importance of task planning accuracy, it is sufficient to observe that in order to perform a mechanical part mating with a positional approach the relative positioning of the parts should be guaranteed with an accuracy of an order of magnitude greater than part mechanical tolerance. Once the absolute position of one part is exactly known, the manipulator should guide the motion of the other with the same accuracy.

In practice, the planning errors may give rise to a contact force and moment, causing a deviation of the end-effector from the desired trajectory. On the other hand, the control system reacts to reduce such deviations. This ultimately leads to a build-up of the contact force until saturation of the joint actuators is reached or breakage of the parts in contact occurs.

The higher the environment stiffness and position control accuracy are, the more easily a situation like the one just described can occur. This drawback can be overcome if a *compliant behavior* is ensured during the interaction. This compliant behavior can be achieved either in a passive or in an active fashion.

Passive Interaction Control

In passive interaction control the trajectory of the robot end-effector is modified by the interaction forces due to the inherent compliance of the robot. The compliance may be due to the structural compliance of the links, joints, and end-effector, or to the compliance of the position servo. Soft robot arms with elastic joints or links are purposely designed for intrinsically safe interaction with humans. In industrial applications, a mechanical device with passive compliance, known as the remote center of compliance (RCC) device [9.1], is widely adopted. An RCC is a compliant end-effector mounted on a rigid robot, designed and optimized for peg-into-hole assembly operations.

The passive approach to interaction control is very simple and cheap, because it does not require force/torque sensors; also, the preprogrammed trajectory of the end-effector must not be changed at execution time; moreover, the response of a passive compliance mechanism is much faster than active repositioning by a computer control algorithm. However, the use of passive compliance in industrial applications lacks flexibility, since for every robotic task a special-purpose compliant end-effector has to be designed and mounted. Also, it can only deal with small position and orientation deviations of the programmed trajectory. Finally, since no forces are measured, it can not guarantee that high contact forces will never occur.

Active Interaction Control

In active interaction control, the compliance of the robotic system is mainly ensured by a purposely designed control system. This approach usually requires the measurement of the contact force and moment, which are fed back to the controller and used to modify or even generate online the desired trajectory of the robot end-effector.

Active interaction control may overcome the aforementioned disadvantages of passive interaction control, but it is usually slower, more expensive, and more sophisticated. To obtain a reasonable task execution speed and disturbance rejection capability, active interaction

control has to be used in combination with some degree of passive compliance [9.2]: feedback, by definition, always comes after a motion and force error has occurred, hence some passive compliance is needed in order to keep the reaction forces below an acceptable threshold.

Force Measurements

For a general force-controlled task, six force components are required to provide complete contact force information: three translational force components and three torques. Often, a force/torque sensor is mounted at the robot wrist [9.3], but other possibilities exist, for example, force sensors can be placed on the fingertips of robotic hands [9.4]; also, external forces and moments can be estimated via shaft torque measurements of joint torque sensors [9.5, 6]. However, the majority of the applications of force control (including industrial applications) is concerned with wrist force/torque sensors. In this case, the weight and inertia of the tool mounted between the sensor and the environment (i. e., the robot end-effector) is assumed to be negligible or suitably compensated from the force/torque measurements. The force signals may be obtained using strain measurements, which results in a stiff sensor, or deformation measurements (e.g., optically), resulting in a compliant sensor. The latter approach has an advantage if additional passive compliance is desired.

9.1.2 From Indirect Force Control to Hybrid Force/Motion Control

Active interaction control strategies can be grouped into two categories: those performing indirect force control and those performing direct force control. The main difference between the two categories is that the former achieve force control via motion control, without explicit closure of a force feedback loop; the latter instead offer the possibility of controlling the contact force and moment to a desired value, thanks to the closure of a force feedback loop.

To the first category belongs *impedance control* (or *admittance control*) [9.7, 8], where the deviation of the end-effector motion from the desired motion due to the interaction with the environment is related to the contact force through a mechanical impedance/admittance with adjustable parameters. A robot manipulator under impedance (or admittance) control is described by an equivalent mass–spring–damper system with adjustable parameters. This relationship is an impedance if the robot control reacts to the motion deviation by generating forces, while it corresponds to an admittance if the robot control reacts to interaction forces by imposing a deviation from the desired motion. Special cases of impedance and admittance control are *stiffness control* and *compliance control* [9.9], respectively, where only the static relationship between the end-effector position and orientation deviation from the desired motion and the contact force and moment is considered. Notice that, in the robot control literature, the terms impedance control and admittance control are often used to refer to the same control scheme; the same happens for stiffness and compliance control. Moreover, if only the relationship between the contact force and moment and the end-effector linear and angular velocity is of interest, the corresponding control scheme is referred to as *damping control* [9.10].

Indirect force control schemes do not require, in principle, measurements of contact forces and moments; the resulting impedance or admittance is typically nonlinear and coupled. However, if a force/torque sensor is available, then force measurements can be used in the control scheme to achieve a linear and decoupled behavior.

Differently from indirect force control, direct force control requires an explicit model of the interaction task. In fact, the user has to specify the desired motion and the desired contact force and moment in a consistent way with respect to the constraints imposed by the environment. A widely adopted strategy belonging to this category is *hybrid force/motion control*, which aims at controlling the motion along the unconstrained task directions and force (and moment) along the constrained task directions. The starting point is the observation that, for many robotic tasks, it is possible to introduce an orthogonal reference frame, known as the compliance frame [9.11] (or task frame [9.12]) which allows one to specify the task in terms of natural and artificial constrains acting along and about the three orthogonal axes of this frame. Based on this decomposition, hybrid force/motion control allows simultaneous control of both the contact force and the end-effector motion in two mutually independent subspaces. Simple selection matrices acting on both the desired and feedback quantities serve this purpose for planar contact surfaces [9.13], whereas suitable projection matrices must be used for general contact tasks, which can also be derived from the explicit constraint equations [9.14–16]. Several implementation of hybrid motion control schemes are available, e.g., based on inverse dynamics control in the operational space [9.17], passivity-based control [9.18], or outer force control loops closed around inner motion loops, typically available in industrial robots [9.2].

If an accurate model of the environment is not available, the force control action and the motion control action can be superimposed, resulting in a *parallel force/position control* scheme. In this approach, the

force controller is designed so as to dominate the motion controller; hence, a position error would be tolerated along the constrained task directions in order to ensure force regulation [9.19].

9.2 Indirect Force Control

To gain insight into the problems arising at the interaction between the end-effector of a robot manipulator and the environment, it is worth analyzing the effects of a motion control strategy in the presence of a contact force and moment. To this aim, assume that a reference frame Σ_e is attached to the end-effector, and let $\boldsymbol{p}_e$ denote the position vector of the origin and $\mathbf{R}_e$ the rotation matrix with respect to a fixed base frame. The end-effector velocity is denoted by the 6×1 twist vector $\boldsymbol{v}_e = (\dot{\boldsymbol{p}}_e^T \ \boldsymbol{\omega}_e^T)^T$ where $\dot{\boldsymbol{p}}_e$ is the translational velocity and $\boldsymbol{\omega}_e$ the angular velocity, and can be computed from the $n\times 1$ joint velocity vector $\dot{\boldsymbol{q}}$ using the linear mapping

$$\boldsymbol{v}_e = \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \,. \tag{9.1}$$

The matrix $\mathbf{J}$ is the $6\times n$ end-effector geometric Jacobian. For simplicity, the case of nonredundant nonsingular manipulators is considered; therefore, $n = 6$ and the Jacobian is a square nonsingular matrix. The force $\boldsymbol{f}_e$ and moment $\boldsymbol{m}_e$ applied by the end-effector to the environment are the components of the wrench $\boldsymbol{h}_e = (\boldsymbol{f}_e^T \ \boldsymbol{m}_e^T)^T$.

It is useful to consider the operational space formulation of the dynamic model of a rigid robot manipulator in contact with the environment

$$\boldsymbol{\Lambda}(\boldsymbol{q})\dot{\boldsymbol{v}}_e + \boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{v}_e + \boldsymbol{\eta}(\boldsymbol{q}) = \boldsymbol{h}_c - \boldsymbol{h}_e \,, \tag{9.2}$$

where

$$\boldsymbol{\Lambda}(\boldsymbol{q}) = (\mathbf{J}\mathbf{H}(\boldsymbol{q})^{-1}\mathbf{J}^T)^{-1}$$

is the 6×6 operational space inertia matrix,

$$\boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \mathbf{J}^{-T}\mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\mathbf{J}^{-1} - \boldsymbol{\Lambda}(\boldsymbol{q})\dot{\mathbf{J}}\mathbf{J}^{-1}$$

is the wrench including centrifugal and Coriolis effects, and $\boldsymbol{\eta}(\boldsymbol{q}) = \mathbf{J}^{-T}\boldsymbol{g}(\boldsymbol{q})$ is the wrench of the gravitational effects; $\mathbf{H}(\boldsymbol{q})$, $\mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ and $\boldsymbol{g}(\boldsymbol{q})$ are the corresponding quantities defined in the joint space. The vector $\boldsymbol{h}_c = \mathbf{J}^{-T}\boldsymbol{\tau}$ is the equivalent end-effector wrench corresponding to the input joint torques $\boldsymbol{\tau}$.

9.2.1 Stiffness Control

In the classical operational space formulation, the end-effector position and orientation is described by a 6×1 vector $\boldsymbol{x}_e = (\boldsymbol{p}_e^T \ \boldsymbol{\varphi}_e^T)^T$, where $\boldsymbol{\varphi}_e$ is a set of Euler angles extracted from $\mathbf{R}_e$. Hence, a desired end-effector position and orientation can be assigned in terms of a vector $\boldsymbol{x}_d$, corresponding to the position of the origin $\boldsymbol{p}_d$ and the rotation matrix $\mathbf{R}_d$ of a desired frame Σ_d. The end-effector error can be denoted as $\Delta\boldsymbol{x}_{de} = \boldsymbol{x}_d - \boldsymbol{x}_e$, and the corresponding velocity error, assuming a constant $\boldsymbol{x}_d$, can be expressed as $\Delta\dot{\boldsymbol{x}}_{de} = -\dot{\boldsymbol{x}}_e = -\mathbf{A}^{-1}(\boldsymbol{\varphi}_e)\boldsymbol{v}_e$, with

$$\mathbf{A}(\boldsymbol{\varphi}_e) = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}(\boldsymbol{\varphi}_e) \end{pmatrix} ,$$

where $\mathbf{I}$ is the 3×3 identity matrix, $\mathbf{0}$ is a 3×3 null matrix, and $\mathbf{T}$ is the 3×3 matrix of the mapping $\boldsymbol{\omega}_e = \mathbf{T}(\boldsymbol{\varphi}_e)\dot{\boldsymbol{\varphi}}_e$, depending on the particular choice of the Euler angles.

Consider the motion control law

$$\boldsymbol{h}_c = \mathbf{A}^{-T}(\boldsymbol{\varphi}_e)\mathbf{K}_P\Delta\boldsymbol{x}_{de} - \mathbf{K}_D\boldsymbol{v}_e + \boldsymbol{\eta}(\boldsymbol{q}) \,, \tag{9.3}$$

corresponding to a simple proportional–derivative (PD) + gravity compensation control in the operational space, where $\mathbf{K}_P$ and $\mathbf{K}_D$ are symmetric and positive-definite 6×6 matrices.

In the absence of interaction with the environment (i. e., when $\boldsymbol{h}_e = \mathbf{0}$), the equilibrium $\boldsymbol{v}_e = \mathbf{0}$, $\Delta\boldsymbol{x}_{de} = \mathbf{0}$ for the closed-loop system, corresponding to the desired position and orientation for the end-effector, is asymptotically stable. The stability proof is based on the positive-definite Lyapunov function

$$V = \frac{1}{2}\boldsymbol{v}_e^T\boldsymbol{\Lambda}(\boldsymbol{q})\boldsymbol{v}_e + \frac{1}{2}\Delta\boldsymbol{x}_{de}\mathbf{K}_P\Delta\boldsymbol{x}_{de} \,,$$

whose time derivative along the trajectories of the closed-loop system is the negative semidefinite function

$$\dot{V} = -\boldsymbol{v}_e^T\mathbf{K}_D\boldsymbol{v}_e \,. \tag{9.4}$$

In the presence of a constant wrench $\boldsymbol{h}_e$, using a similar Lyapunov argument, a different asymptotically stable equilibrium can be found, with a nonnull $\Delta\boldsymbol{x}_{de}$. The new equilibrium is the solution of the equation

$$\mathbf{A}^{-T}(\boldsymbol{\varphi}_e)\mathbf{K}_P\Delta\boldsymbol{x}_{de} - \boldsymbol{h}_e = \mathbf{0} \,,$$

which can be written in the form

$$\Delta \boldsymbol{x}_{\mathrm{de}} = \mathbf{K}_{\mathrm{P}}^{-1}\mathbf{A}^{\mathrm{T}}(\boldsymbol{\varphi}_{\mathrm{e}})\boldsymbol{h}_{\mathrm{e}} \,, \tag{9.5}$$

or, equivalently, as

$$\boldsymbol{h}_{\mathrm{e}} = \mathbf{A}^{-\mathrm{T}}(\boldsymbol{\varphi}_{\mathrm{e}})\mathbf{K}_{\mathrm{P}}\Delta \boldsymbol{x}_{\mathrm{de}} \,. \tag{9.6}$$

Equation (9.6) shows that in the steady state the end-effector, under a proportional control action on the position and orientation error, behaves as a six-degree-of-freedom (DOF) spring in respect of the external force and moment $\boldsymbol{h}_{\mathrm{e}}$. Thus, the matrix $\mathbf{K}_{\mathrm{P}}$ plays the role of an *active stiffness*, meaning that it is possible to act on the elements of $\mathbf{K}_{\mathrm{P}}$ so as to ensure a suitable elastic behavior of the end-effector during the interaction. Analogously, (9.5) represents a compliance relationship, where the matrix $\mathbf{K}_{\mathrm{P}}^{-1}$ plays the role of an *active compliance*. This approach, consisting of assigning a desired position and orientation and a suitable static relationship between the deviation of the end-effector position and orientation from the desired motion and the force exerted on the environment, is known as *stiffness control*.

The selection of the stiffness/compliance parameters is not easy, and strongly depends on the task to be executed. A higher value of the active stiffness means a higher accuracy of the position control at the expense of higher interaction forces. Hence, if it is expected to meet some physical constraint in a particular direction, the end-effector stiffness in that direction should be made low to ensure low interaction forces. Conversely, along the directions where physical constraints are not expected, the end-effector stiffness should be made high so as to follow closely the desired position. This allows discrepancies between the desired and achievable positions due to the constraints imposed by the environment to be resolved without excessive contact forces and moments.

It must be pointed out, however, that a selective stiffness behavior along different directions cannot be effectively assigned in practice on the basis of (9.6). This can easily be understood by using the classical definition of a mechanical stiffness for two bodies connected by a 6-DOF spring, in terms of the linear mapping between the infinitesimal twist displacement of the two bodies at an unloaded equilibrium and the elastic wrench.

In the case of the active stiffness, the two bodies are, respectively, the end-effector, with the attached frame Σ_{e}, and a virtual body, attached to the desired frame Σ_{d}. Hence, from (9.6), the following mapping can be derived

$$\boldsymbol{h}_{\mathrm{e}} = \mathbf{A}^{-\mathrm{T}}(\boldsymbol{\varphi}_{\mathrm{e}})\mathbf{K}_{\mathrm{P}}\mathbf{A}^{-1}(\boldsymbol{\varphi}_{\mathrm{e}})\delta \boldsymbol{x}_{\mathrm{de}} \,, \tag{9.7}$$

in the case of an infinitesimal twist displacement $\delta \boldsymbol{x}_{\mathrm{de}}$ defined as

$$\delta \boldsymbol{x}_{\mathrm{de}} = \begin{pmatrix} \delta \boldsymbol{p}_{\mathrm{de}} \\ \delta \boldsymbol{\theta}_{\mathrm{de}} \end{pmatrix} = \begin{pmatrix} \Delta \dot{\boldsymbol{p}}_{\mathrm{de}} \\ \Delta \boldsymbol{\omega}_{\mathrm{de}} \end{pmatrix} \mathrm{d}t = - \begin{pmatrix} \dot{\boldsymbol{p}}_{\mathrm{e}} \\ \boldsymbol{\omega}_{\mathrm{e}} \end{pmatrix} \mathrm{d}t \,,$$

where $\Delta \dot{\boldsymbol{p}}_{\mathrm{de}} = \dot{\boldsymbol{p}}_{\mathrm{d}} - \dot{\boldsymbol{p}}_{\mathrm{e}}$ is the time derivative of the position error $\Delta \boldsymbol{p}_{\mathrm{de}} = \boldsymbol{p}_{\mathrm{d}} - \boldsymbol{p}_{\mathrm{e}}$ and $\Delta \boldsymbol{\omega}_{\mathrm{de}} = \boldsymbol{\omega}_{\mathrm{d}} - \boldsymbol{\omega}_{\mathrm{e}}$ is the angular velocity error. Equation (9.7) shows that the actual stiffness matrix is $\mathbf{A}^{-\mathrm{T}}(\boldsymbol{\varphi}_{\mathrm{e}})\mathbf{K}_{\mathrm{P}}\mathbf{A}^{-1}(\boldsymbol{\varphi}_{\mathrm{e}})$, which depends on the end-effector orientation through the vector $\boldsymbol{\varphi}_{\mathrm{e}}$, so that, in practice, the selection of the stiffness parameters is quite difficult.

This problem can be overcome by defining a *geometrically consistent active stiffness*, with the same structure and properties as ideal mechanical springs.

Mechanical Springs

Consider two elastically coupled rigid bodies A and B and two reference frames Σ_{a} and Σ_{b}, attached to A and B, respectively. Assuming that at equilibrium frames Σ_{a} and Σ_{b} coincide, the compliant behavior near the equilibrium can be described by the linear mapping

$$\boldsymbol{h}_{\mathrm{b}}^{\mathrm{b}} = \mathbf{K}\delta \boldsymbol{x}_{\mathrm{ab}}^{\mathrm{b}} = \begin{pmatrix} \mathbf{K}_{\mathrm{t}} & \mathbf{K}_{\mathrm{c}} \\ \mathbf{K}_{\mathrm{c}}^{\mathrm{T}} & \mathbf{K}_{\mathrm{o}} \end{pmatrix} \delta \boldsymbol{x}_{\mathrm{ab}}^{\mathrm{b}} \,, \tag{9.8}$$

where $\boldsymbol{h}_{\mathrm{b}}^{\mathrm{b}}$ is the elastic wrench applied to body B, expressed in frame B, in the presence of an infinitesimal twist displacement $\delta \boldsymbol{x}_{\mathrm{ab}}^{\mathrm{b}}$ of frame Σ_{a} with respect to frame Σ_{b}, expressed in frame B. The elastic wrench and the infinitesimal twist displacement in (9.8) can also be expressed equivalently in frame Σ_{a}, since Σ_{a} and Σ_{b} coincide at equilibrium. Therefore, $\boldsymbol{h}_{\mathrm{b}}^{\mathrm{b}} = \boldsymbol{h}_{\mathrm{b}}^{\mathrm{a}}$ and $\delta \boldsymbol{x}_{\mathrm{ab}}^{\mathrm{b}} = \delta \boldsymbol{x}_{\mathrm{ab}}^{\mathrm{a}}$; moreover, for the elastic wrench applied to body A, $\boldsymbol{h}_{\mathrm{a}}^{\mathrm{a}} = \mathbf{K}_{\mathrm{t}}\delta \boldsymbol{x}_{\mathrm{ba}}^{\mathrm{a}} = -\boldsymbol{h}_{\mathrm{b}}^{\mathrm{b}}$ being $\delta \boldsymbol{x}_{\mathrm{ba}}^{\mathrm{a}} = -\delta \boldsymbol{x}_{\mathrm{ab}}^{\mathrm{b}}$. This property of the mapping (9.8) is known as *port symmetry*.

In (9.8), $\mathbf{K}$ is the 6×6 symmetric positive-semidefinite stiffness matrix. The 3×3 matrices $\mathbf{K}_{\mathrm{t}}$ and $\mathbf{K}_{\mathrm{o}}$, called respectively the *translational stiffness* and *rotational stiffness*, are also symmetric. It can be shown that, if the 3×3 matrix $\mathbf{K}_{\mathrm{c}}$, called the *coupling stiffness* is symmetric, there is maximum decoupling between rotation and translation. In this case, the point corresponding to the coinciding origins of the frames Σ_{a} and Σ_{b} is called the *center of stiffness*. Similar definitions and results can be formulated for the case of the compliance matrix $\mathbf{C} = \mathbf{K}^{-1}$. In particular, it is possible to define a *center of compliance* in the case that the off-diagonal blocks of the compliance matrix are symmetric. The center of stiffness and compliance do not necessarily coincide.

There are special cases in which no coupling exists between translation and rotation, i. e., a relative translation of the bodies results in a wrench corresponding to a pure force along an axis through the center of stiffness; also, a relative rotation of the bodies results in a wrench that is equivalent to a pure torque about an axis through the centers of stiffness. In these cases, the center of stiffness and compliance coincide. Mechanical systems with completely decoupled behavior are, e.g., the remote center of compliance (RCC) devices.

Since $\mathbf{K}_t$ is symmetric, there exists a rotation matrix $\mathbf{R}_t$ with respect to the frame $\Sigma_a = \Sigma_b$ at equilibrium, such that $\mathbf{K}_t = \mathbf{R}_t\,\mathbf{\Gamma}_t\,\mathbf{R}_t^T$, and $\mathbf{\Gamma}_t$ is a diagonal matrix whose diagonal elements are the *principal translational stiffnessess* in the directions corresponding to the columns of the rotation matrix $\mathbf{R}_t$, known as the *principal axes of translational stiffness*. Analogously, $\mathbf{K}_o$ can be expressed as $\mathbf{K}_o = \mathbf{R}_o\mathbf{\Gamma}_o\mathbf{R}_o^T$, where the diagonal elements of $\mathbf{\Gamma}_o$ are the *principal rotational stiffnesses* about the axes corresponding to the columns of rotation matrix $\mathbf{R}_o$, known as the *principal axes of rotational stiffness*. Moreover, assuming that the origins of Σ_a and Σ_b at equilibrium coincide with the center of stiffness, the expression $\mathbf{K}_c = \mathbf{R}_c\mathbf{\Gamma}_c\mathbf{R}_c^T$ can be found, where the diagonal elements of $\mathbf{\Gamma}_c$ are the *principal coupling stiffnesses* along the directions corresponding to the columns of the rotation matrix $\mathbf{R}_c$, known as the *principal axes of coupling stiffness*. In sum, a 6×6 stiffness matrix can be specified, with respect to a frame with origin in the center of stiffness, in terms of the principal stiffness parameters and principal axes.

Notice that the mechanical stiffness defined by (9.8) describes the behavior of an ideal 6-DOF spring which stores potential energy. The potential energy function of an ideal stiffness depends only on the relative position and orientation of the two attached bodies and is port symmetric. A physical 6-DOF spring has a predominant behavior similar to the ideal one, but nevertheless it always has parasitic effects causing energy dissipation.

Geometrically Consistent Active Stiffness

To achieve a geometrically consistent 6-DOF active stiffness, a suitable definition of the proportional control action in control law (9.3) is required. This control action can be interpreted as the elastic wrench applied to the end-effector, in the presence of a *finite displacement* of the desired frame Σ_d with respect to the end-effector frame Σ_e. Hence, the properties of the ideal mechanical stiffness for small displacements should be extended to the case of finite displacements. Moreover, to guarantee asymptotic stability in the sense of Lyapunov, a suitable potential elastic energy function must be defined.

For simplicity, it is assumed that the coupling stiffness matrix is zero. Hence, the potential elastic energy can be computed as the sum of a translational potential energy and a rotational potential energy.

The translational potential energy can be defined as

$$V_t = \frac{1}{2}\Delta\boldsymbol{p}_{de}^T\mathbf{K}'_{Pt}\Delta\boldsymbol{p}_{de}\,, \tag{9.9}$$

with

$$\mathbf{K}'_{Pt} = \frac{1}{2}\mathbf{R}_d\mathbf{K}_{Pt}\mathbf{R}_d^T + \frac{1}{2}\mathbf{R}_e\mathbf{K}_{Pt}\mathbf{R}_e^T\,,$$

where $\mathbf{K}_{Pt}$ is a 3×3 symmetric positive-definite matrix. The use of $\mathbf{K}'_{Pt}$ in lieu of $\mathbf{K}_{Pt}$ in (9.9) guarantees that the potential energy is port symmetric also in the case of finite displacements. Matrices $\mathbf{K}'_{Pt}$ and $\mathbf{K}_{Pt}$ coincide at equilibrium (i. e., when $\mathbf{R}_d = \mathbf{R}_e$) and in the case of isotropic translational stiffness (i. e., when $\mathbf{K}_{Pt} = k_{Pt}\mathbf{I}$).

The computation of the power $\dot{V}_t$ yields

$$\dot{V}_t = \Delta\dot{\boldsymbol{p}}_{de}^{eT}\boldsymbol{f}_{\Delta t}^e + \Delta\boldsymbol{\omega}_{de}^{eT}\boldsymbol{m}_{\Delta t}^e\,,$$

where $\Delta\dot{\boldsymbol{p}}_{de}^e$ is the time derivative of the position displacement $\Delta\boldsymbol{p}_{de}^e = \mathbf{R}_e^T(\boldsymbol{p}_d - \boldsymbol{p}_e)$, while $\Delta\boldsymbol{\omega}_{de}^e = \mathbf{R}_e^T(\boldsymbol{\omega}_d - \boldsymbol{\omega}_e)$. The vectors $\boldsymbol{f}_{\Delta t}^e$ and $\boldsymbol{\mu}_{\Delta t}^e$ are, respectively, the elastic force and moment applied to the end-effector in the presence of the finite position displacement $\Delta\boldsymbol{p}_{de}^e$. These vectors have the following expressions when computed in the base frame

$$\boldsymbol{f}_{\Delta t} = \mathbf{K}'_{Pt}\Delta\boldsymbol{p}_{de} \qquad \boldsymbol{m}_{\Delta t} = \mathbf{K}''_{Pt}\Delta\boldsymbol{p}_{de}\,, \tag{9.10}$$

with

$$\mathbf{K}''_{Pt} = \frac{1}{2}\mathbf{S}(\Delta\boldsymbol{p}_{de})\mathbf{R}_d\mathbf{K}_{Pt}\mathbf{R}_d^T\,,$$

where $\mathbf{S}(\cdot)$ is the skew-symmetric operator performing the vector product. The vector $\boldsymbol{h}_{\Delta t} = (\boldsymbol{f}_{\Delta t}^T\ \boldsymbol{m}_{\Delta t}^T)^T$ is the elastic wrench applied to the end-effector in the presence of a finite position displacement $\Delta\boldsymbol{p}_{de}$ and a null orientation displacement. The moment $\boldsymbol{m}_{\Delta t}$ is null in the case of isotropic translational stiffness.

To define the rotational potential energy, a suitable definition of the orientation displacement between the frames Σ_d and Σ_e has to be adopted. A possible choice is the vector part of the unit quaternion $\{\eta_{de}, \boldsymbol{\epsilon}_{de}^e\}$ that can be extracted from matrix $\mathbf{R}_d^e = \mathbf{R}_e^T\mathbf{R}_d$. Hence, the orientation potential energy has the form

$$V_o = 2\boldsymbol{\epsilon}_{de}^{eT}\mathbf{K}_{Po}\boldsymbol{\epsilon}_{de}^e\,, \tag{9.11}$$

where $\mathbf{K}_{\mathrm{Po}}$ is a 3×3 symmetric positive-definite matrix. The function V_{o} is port symmetric because $\boldsymbol{\epsilon}^{\mathrm{e}}_{\mathrm{de}} = -\boldsymbol{\epsilon}^{\mathrm{d}}_{\mathrm{ed}}$.

The computation of the power $\dot{V}_{\mathrm{o}}$ yields

$$\dot{V}_{\mathrm{o}} = \Delta\boldsymbol{\omega}^{\mathrm{eT}}_{\mathrm{de}}\boldsymbol{m}^{\mathrm{e}}_{\Delta\mathrm{o}} ,$$

where

$$\boldsymbol{m}_{\Delta\mathrm{o}} = \mathbf{K}'_{\mathrm{Po}}\boldsymbol{\epsilon}_{\mathrm{de}} , \tag{9.12}$$

with

$$\mathbf{K}'_{\mathrm{Po}} = 2\mathbf{E}^{\mathrm{T}}(\boldsymbol{\eta}_{\mathrm{de}}, \boldsymbol{\epsilon}_{\mathrm{de}})\mathbf{R}_{\mathrm{e}}\mathbf{K}_{\mathrm{Po}}\mathbf{R}^{\mathrm{T}}_{\mathrm{e}} ,$$

and $\mathbf{E}(\boldsymbol{\eta}_{\mathrm{de}}, \boldsymbol{\epsilon}_{\mathrm{de}}) = \boldsymbol{\eta}_{\mathrm{de}}\mathbf{I} - \mathbf{S}(\boldsymbol{\epsilon}_{\mathrm{de}})$. The above equations show that a finite orientation displacement $\boldsymbol{\epsilon}_{\mathrm{de}} = \mathbf{R}^{\mathrm{T}}_{\mathrm{e}}\boldsymbol{\epsilon}^{\mathrm{e}}_{\mathrm{de}}$ produces an elastic wrench $\boldsymbol{h}_{\Delta\mathrm{o}} = (\mathbf{0}^{\mathrm{T}} \;\; \boldsymbol{m}^{\mathrm{T}}_{\Delta\mathrm{o}})^{\mathrm{T}}$ which is equivalent to a pure moment.

Hence, the total elastic wrench in the presence of a finite position and orientation displacement of the desired frame Σ_{d} with respect to the end-effector frame Σ_{e} can be defined in the base frame as

$$\boldsymbol{h}_{\Delta} = \boldsymbol{h}_{\Delta\mathrm{t}} + \boldsymbol{h}_{\Delta\mathrm{o}} . \tag{9.13}$$

where $\boldsymbol{h}_{\Delta\mathrm{t}}$ and $\boldsymbol{h}_{\Delta\mathrm{o}}$ are computed according to (9.10) and (9.12), respectively.

Using (9.13) for the computation of the elastic wrench in the case of an infinitesimal twist displacement $\delta\boldsymbol{x}^{\mathrm{e}}_{\mathrm{de}}$ near the equilibrium, and discarding the high-order infinitesimal terms, yields the linear mapping

$$\boldsymbol{h}^{\mathrm{e}}_{\mathrm{e}} = \mathbf{K}_{\mathrm{P}}\delta\boldsymbol{x}^{\mathrm{e}}_{\mathrm{de}} = \begin{pmatrix} \mathbf{K}_{\mathrm{Pt}} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{\mathrm{Po}} \end{pmatrix} \delta\boldsymbol{x}^{\mathrm{e}}_{\mathrm{de}} . \tag{9.14}$$

Therefore, $\mathbf{K}_{\mathrm{P}}$ represents the stiffness matrix of an ideal spring with respect to a frame Σ_{e} (coinciding with Σ_{d} at equilibrium) with the origin at the center of stiffness. Moreover, it can be shown, using definition (9.13), that the physical/geometrical meaning of the principal stiffnesses and of the principal axes for the matrices $\mathbf{K}_{\mathrm{Pt}}$ and $\mathbf{K}_{\mathrm{Po}}$ are preserved also in the case of large displacements.

The above results imply that the active stiffness matrix $\mathbf{K}_{\mathrm{P}}$ can be set in a geometrically consistent way with respect to the task at hand.

Notice that geometrical consistency can also be ensured with different definitions of the orientation error in the potential orientation energy (9.11), for example, any error based on the angle/axis representation of $\mathbf{R}^{\mathrm{d}}_{\mathrm{e}}$ can be adopted (the unit quaternion belongs to this category), or, more generally, homogeneous matrices or exponential coordinates (for the case of both position and orientation errors). Also, the *XYZ* Euler angles extracted from the matrix $\mathbf{R}^{\mathrm{d}}_{\mathrm{e}}$ could be used; however, in this case, it can be shown that the principal axes of rotational stiffness cannot be set arbitrarily but must coincide with those of the end-effector frame.

Stiffness control with a geometrically consistent active stiffness can be defined using the control law

$$\boldsymbol{h}_{\mathrm{c}} = \boldsymbol{h}_{\Delta} - \mathbf{K}_{\mathrm{D}}\boldsymbol{v}_{\mathrm{e}} + \boldsymbol{\eta}(\boldsymbol{q}) , \tag{9.15}$$

with $\boldsymbol{h}_{\Delta}$ in (9.13). The asymptotic stability about the equilibrium in the case $\boldsymbol{h}_{\mathrm{e}} = \mathbf{0}$ can be proven using the Lyapunov function

$$V = \frac{1}{2}\boldsymbol{v}^{\mathrm{T}}_{\mathrm{e}}\boldsymbol{\Lambda}(\boldsymbol{q})\boldsymbol{v}_{\mathrm{e}} + V_{\mathrm{t}} + V_{\mathrm{o}} ,$$

with V_{t} and V_{o} given in (9.9) and (9.11), respectively, whose time derivative along the trajectories of the closed-loop system, in case the frame Σ_{d} is motionless, has the same expression as in (9.4). When $\boldsymbol{h}_{\mathrm{e}} \neq \mathbf{0}$, a different asymptotically stable equilibrium can be found, corresponding to a nonnull displacement of the desired frame Σ_{d} with respect to the end-effector frame Σ_{e}. The new equilibrium is the solution of the equation $\boldsymbol{h}_{\Delta} = \boldsymbol{h}_{\mathrm{e}}$.

Stiffness control allows to keep the interaction force and moment limited at the expense of the end-effector position and orientation error, with a proper choice of the stiffness matrix, without the need of a force/torque sensor. However, in the presence of disturbances (e.g., joint friction) which can be modeled as an equivalent end-effector wrench, the adoption of low values for the active stiffness may produce large deviations with respect to the desired end-effector position and orientation, also in the absence of interaction with the environment.

9.2.2 Impedance Control

Stiffness control is designed to achieve a desired *static* behavior of the interaction. In fact, the dynamics of the controlled system depends on that of the robot manipulator, which is nonlinear and coupled. A more demanding objective may be that of achieving a desired *dynamic* behavior for the end-effector, e.g., that of a second-order mechanical system with six degrees of freedom, characterized by a given mass, damping, and stiffness, known as mechanical *impedance*.

The starting point to pursue this goal may be the acceleration-resolved approach used for motion control, which is aimed at decoupling and linearizing the nonlinear robot dynamics at the acceleration level via an inverse dynamics control law. In the presence of interaction with the environment, the control law

$$\boldsymbol{h}_{\mathrm{c}} = \boldsymbol{\Lambda}(\boldsymbol{q})\boldsymbol{\alpha} + \boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\eta}(\boldsymbol{q}) + \boldsymbol{h}_{\mathrm{e}} \tag{9.16}$$

cast into the dynamic model (9.2) results in

$$\dot{\boldsymbol{v}}_{\mathrm{e}} = \boldsymbol{\alpha} \,, \tag{9.17}$$

where $\boldsymbol{\alpha}$ is a properly designed control input with the meaning of an acceleration referred to the base frame. Considering the identity $\dot{\boldsymbol{v}}_{\mathrm{e}} = \bar{\mathbf{R}}_{\mathrm{e}}^{\mathrm{T}} \dot{\boldsymbol{v}}_{\mathrm{e}}^{\mathrm{e}} + \dot{\bar{\mathbf{R}}}_{\mathrm{e}}^{\mathrm{T}} \boldsymbol{v}_{\mathrm{e}}^{\mathrm{e}}$, with

$$\bar{\mathbf{R}}_{\mathrm{e}} = \begin{pmatrix} \mathbf{R}_{\mathrm{e}} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathrm{e}} \end{pmatrix} \,,$$

the choice

$$\boldsymbol{\alpha} = \bar{\mathbf{R}}_{\mathrm{e}}^{\mathrm{T}} \boldsymbol{\alpha}^{\mathrm{e}} + \dot{\bar{\mathbf{R}}}_{\mathrm{e}}^{\mathrm{T}} \boldsymbol{v}_{\mathrm{e}}^{\mathrm{e}} \tag{9.18}$$

gives

$$\dot{\boldsymbol{v}}_{\mathrm{e}}^{\mathrm{e}} = \boldsymbol{\alpha}^{\mathrm{e}} \,, \tag{9.19}$$

where the control input $\boldsymbol{\alpha}^{\mathrm{e}}$ has the meaning of an acceleration referred to the end-effector frame Σ_{e}. Hence, setting

$$\boldsymbol{\alpha}^{\mathrm{e}} = \dot{\boldsymbol{v}}_{\mathrm{d}}^{\mathrm{e}} + \mathbf{K}_{\mathrm{M}}^{-1} (\mathbf{K}_{\mathrm{D}} \Delta \boldsymbol{v}_{\mathrm{de}}^{\mathrm{e}} + \boldsymbol{h}_{\Delta}^{\mathrm{e}} - \boldsymbol{h}_{\mathrm{e}}^{\mathrm{e}}) \,, \tag{9.20}$$

the following expression can be found for the closed-loop system

$$\mathbf{K}_{\mathrm{M}} \Delta \dot{\boldsymbol{v}}_{\mathrm{de}}^{\mathrm{e}} + \mathbf{K}_{\mathrm{D}} \Delta \boldsymbol{v}_{\mathrm{de}}^{\mathrm{e}} + \boldsymbol{h}_{\Delta}^{\mathrm{e}} = \boldsymbol{h}_{\mathrm{e}}^{\mathrm{e}} \,, \tag{9.21}$$

where $\mathbf{K}_{\mathrm{M}}$ and $\mathbf{K}_{\mathrm{D}}$ are 6×6 symmetric and positive-definite matrices, $\Delta \dot{\boldsymbol{v}}_{\mathrm{de}}^{\mathrm{e}} = \dot{\boldsymbol{v}}_{\mathrm{d}}^{\mathrm{e}} - \dot{\boldsymbol{v}}_{\mathrm{e}}^{\mathrm{e}}$, $\Delta \boldsymbol{v}_{\mathrm{de}}^{\mathrm{e}} = \boldsymbol{v}_{\mathrm{d}}^{\mathrm{e}} - \boldsymbol{v}_{\mathrm{e}}^{\mathrm{e}}$, $\dot{\boldsymbol{v}}_{\mathrm{d}}^{\mathrm{e}}$ and $\boldsymbol{v}_{\mathrm{d}}^{\mathrm{e}}$ are, respectively, the acceleration and the velocity of a desired frame Σ_{d} and $\boldsymbol{h}_{\Delta}^{\mathrm{e}}$ is the elastic wrench (9.13); all the quantities are referred to the end-effector frame Σ_{e}.

The above equation describing the dynamic behavior of the controlled end-effector can be interpreted as a generalized mechanical impedance. The asymptotic stability of the equilibrium in the case $\boldsymbol{h}_{\mathrm{e}} = \mathbf{0}$ can be proven by considering the Lyapunov function

$$V = \frac{1}{2} \Delta \boldsymbol{v}_{\mathrm{de}}^{\mathrm{eT}} \mathbf{K}_{\mathrm{M}} \Delta \boldsymbol{v}_{\mathrm{de}}^{\mathrm{e}} + V_{\mathrm{t}} + V_{\mathrm{o}} \,, \tag{9.22}$$

where V_{t} and V_{o} are defined in (9.9) and (9.11), respectively, and whose time derivative along the trajectories of system (9.21) is the negative semidefinite function

$$\dot{V} = -\Delta \boldsymbol{v}_{\mathrm{de}}^{\mathrm{eT}} \mathbf{K}_{\mathrm{D}} \Delta \boldsymbol{v}_{\mathrm{de}}^{\mathrm{e}} \,.$$

When $\boldsymbol{h}_{\mathrm{e}} \neq \mathbf{0}$, a different asymptotically stable equilibrium can be found, corresponding to a nonnull displacement of the desired frame Σ_{d} with respect to the end-effector frame Σ_{e}. The new equilibrium is the solution of the equation $\boldsymbol{h}_{\Delta} = \boldsymbol{h}_{\mathrm{e}}$.

In case Σ_{d} is constant, (9.21) has the meaning of a true 6-DOF mechanical impedance if $\mathbf{K}_{\mathrm{M}}$ is chosen as

$$\mathbf{K}_{\mathrm{M}} = \begin{pmatrix} m\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{pmatrix} \,,$$

where m is a mass and $\mathbf{M}$ is a 3×3 inertia tensor, and $\mathbf{K}_{\mathrm{D}}$ is chosen as a block-diagonal matrix with 3×3 blocks. The physically equivalent system is a body of mass m with inertia tensor $\mathbf{M}$ with respect to a frame Σ_{e} attached to the body, subject to an external wrench $\boldsymbol{h}_{\mathrm{e}}^{\mathrm{e}}$. This body is connected to a virtual body attached to frame Σ_{d} through a 6-DOF ideal spring with stiffness matrix $\mathbf{K}_{\mathrm{P}}$ and is subject to viscous forces and moments with damping $\mathbf{K}_{\mathrm{D}}$. The function V in (9.22) represents the total energy of the body: the sum of the kinetic and potential elastic energy.

A block diagram of the resulting impedance control is sketched in Fig. 9.1. The impedance control computes the acceleration input as in (9.18) and (9.20) on the basis of the position and orientation feedback as well as the force and moment measurements. Then, the inverse dynamics control law computes the torques for the joint actuators $\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}} \boldsymbol{h}_{\mathrm{c}}$ with $\boldsymbol{h}_{\mathrm{c}}$ in (9.16). This control scheme, in the absence of interaction, guarantees that the end-effector frame Σ_{e} asymptotically follows the desired frame Σ_{d}. In the presence of contact with the environment, a compliant dynamic behavior is imposed on the end-effector, according to the impedance (9.21), and the contact wrench is bounded at the expense of a finite position and orientation displacement between Σ_{d} and Σ_{e}. Differently from stiffness control, a force/torque sensor is required for the measurement of the contact force and moment.

In the case that the force/torque sensor is not available, the measure of the external wrench $\boldsymbol{h}_{\mathrm{e}}$ cannot be used in the controller and thus the configuration-

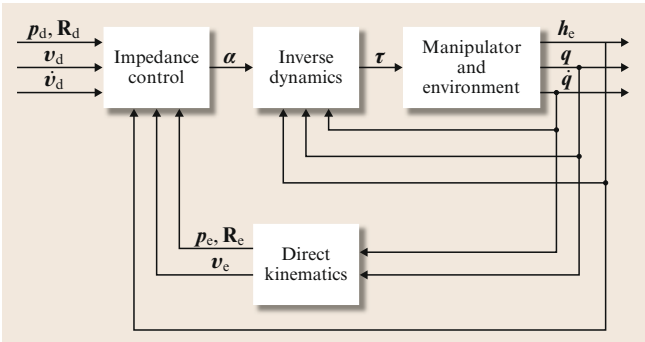


Fig. 9.1 Impedance control

independent impedance behaviour (9.21) cannot be obtained anymore. However, a desired impedance behaviour can be still achieved with the control law

$$\boldsymbol{h}_{\mathrm{c}} = \boldsymbol{\Lambda}(\boldsymbol{q})\dot{\boldsymbol{v}}_{\mathrm{d}} + \boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{v}_{\mathrm{d}} + \mathbf{K}_{\mathrm{D}}\Delta\boldsymbol{v}_{\mathrm{de}} + \boldsymbol{h}_{\Delta} + \boldsymbol{\eta}(\boldsymbol{q}) \tag{9.23}$$

in place of (9.16), where $\mathbf{K}_{\mathrm{D}}$ is a 6×6 positive-definite matrix and $\boldsymbol{h}_{\Delta}$ is the elastic wrench (9.13). The resulting closed-loop equation is

$$\boldsymbol{\Lambda}(\boldsymbol{q})\Delta\dot{\boldsymbol{v}}_{\mathrm{de}} + (\boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \mathbf{K}_{\mathrm{D}})\Delta\boldsymbol{v}_{\mathrm{de}} + \boldsymbol{h}_{\Delta} = \boldsymbol{h}_{\mathrm{e}},$$

representing an impedance behaviour which preserves the actual operational space inertia matrix $\boldsymbol{\Lambda}(\boldsymbol{q})$ of the robot. In the above equation, the centrifugal and Coriolis wrench $\boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\Delta\boldsymbol{v}_{\mathrm{de}}$ is required to preserve the mechanical properties of a configuration-dependent inertia and to prove the stability similarly to (9.21). In case Σ_{d} is constant, control law (9.23) reduces to the stiffness control law (9.15).

Pioneering experiments on stiffness control and impedance control and without without force sensing are presented in VIDEO 684. VIDEO 686 reports experiments on impedance control based on a geometrically consistent active stiffness.

Implementation Issues

The selection of good impedance parameters ensuring a satisfactory behavior is not an easy task. In fact, the dynamics of the closed-loop system is different in free space and during interaction. The control objectives are different as well, since motion tracking and disturbance rejection must be ensured in free space, while, during the interaction, the main goal is achieving a suitable compliant dynamic behavior for the end-effector. Notice also that the dynamics of the controlled system during the interaction depends on the dynamics of the environment.

To gain insight into these problems, assume that the interaction of the end-effector with the environment can be approximated by that derived from an ideal 6-DOF spring connecting end-effector frame Σ_{e} to the environment frame Σ_{o}. Therefore, according to (9.8), the elastic wrench exerted by the end-effector on the environment, in the presence of an infinitesimal twist displacement of Σ_{e} with respect to Σ_{o}, can be computed as

$$\boldsymbol{h}_{\mathrm{e}}^{\mathrm{e}} = \mathbf{K}\delta\boldsymbol{x}_{\mathrm{eo}}^{\mathrm{e}}\,, \tag{9.24}$$

where Σ_{e} and Σ_{o} coincide at equilibrium and $\mathbf{K}$ is a stiffness matrix. The above model holds only in the presence of interaction, while the contact wrench is null when the end-effector moves in free space.

The disturbances acting on the robot manipulator and the unmodeled dynamics (joint friction, modeling errors, etc.) may be taken into account by introducing an additive term on the right-hand side of the dynamic model of the robot manipulator (9.2), corresponding to an equivalent disturbance wrench acting on the end-effector. This term produces an additive acceleration disturbance $\boldsymbol{\gamma}^{\mathrm{e}}$ on the right-hand side of (9.19). Therefore, using the control law (9.20), the following closed-loop impedance equation can be found

$$\mathbf{K}_{\mathrm{M}}\Delta\dot{\boldsymbol{v}}_{\mathrm{de}}^{\mathrm{e}} + \mathbf{K}_{\mathrm{D}}\Delta\boldsymbol{v}_{\mathrm{de}}^{\mathrm{e}} + \boldsymbol{h}_{\Delta}^{\mathrm{e}} = \boldsymbol{h}_{\mathrm{e}}^{\mathrm{e}} + \mathbf{K}_{\mathrm{M}}\boldsymbol{\gamma}^{\mathrm{e}}\,. \tag{9.25}$$

The tuning procedure for the impedance parameters can be set up starting from the linearized model that can be computed from (9.25) in the case of infinitesimal displacements, i. e.,

$$\begin{aligned}&\mathbf{K}_{\mathrm{M}}\delta\ddot{\boldsymbol{x}}_{\mathrm{de}}^{\mathrm{e}} + \mathbf{K}_{\mathrm{D}}\delta\dot{\boldsymbol{x}}_{\mathrm{de}}^{\mathrm{e}} + (\mathbf{K}_{\mathrm{P}} + \mathbf{K})\delta\boldsymbol{x}_{\mathrm{de}}^{\mathrm{e}}\\ &= \mathbf{K}\delta\boldsymbol{x}_{\mathrm{do}}^{\mathrm{e}} + \mathbf{K}_{\mathrm{M}}\boldsymbol{\gamma}^{\mathrm{e}}\,,\end{aligned} \tag{9.26}$$

where (9.24) and the equality $\delta\boldsymbol{x}_{\mathrm{eo}}^{\mathrm{e}} = -\delta\boldsymbol{x}_{\mathrm{de}}^{\mathrm{e}} + \delta\boldsymbol{x}_{\mathrm{do}}^{\mathrm{e}}$ have been used. The above equation is valid both for constrained ($\mathbf{K}\neq\mathbf{0}$) and for free motion ($\mathbf{K}=\mathbf{0}$).

It is evident that suitable dynamics of the position and orientation errors can be set by suitably choosing the matrix gains $\mathbf{K}_{\mathrm{M}}$, $\mathbf{K}_{\mathrm{D}}$, and $\mathbf{K}_{\mathrm{P}}$. This task is easier under the hypothesis that all the matrices are diagonal, resulting in a decoupled behavior for the six components of the infinitesimal twist displacement. In this case, the transient behavior of each component can be set, e.g., by assigning the natural frequency and damping ratio with the relations

$$\omega_{\mathrm{n}} = \sqrt{\frac{< k_{\mathrm{P}} + k}{k_{\mathrm{M}}}}\,, \quad \zeta = \frac{1}{2}\frac{k_{\mathrm{D}}}{\sqrt{k_{\mathrm{M}}(k_{\mathrm{P}}+k)}}\,.$$

Hence, if the gains are chosen so that a given natural frequency and damping ratio are ensured during the interaction (i. e., for $k\neq 0$), a smaller natural frequency with a higher damping ratio will be obtained when the end-effector moves in free space (i. e., for $k=0$). As for the steady-state performance, the end-effector error for the generic component is

$$\delta x_{\mathrm{de}} = \frac{k}{(k_{\mathrm{P}}+k)}\delta x_{\mathrm{do}} + \frac{k_{\mathrm{M}}}{k_{\mathrm{P}}+k}\boldsymbol{\gamma}$$

and the corresponding interaction force is

$$h = \frac{k_{\mathrm{P}}k}{k_{\mathrm{P}}+k}\delta x_{\mathrm{do}} - \frac{k_{\mathrm{M}}k}{k_{\mathrm{P}}+k}\boldsymbol{\gamma}\,.$$

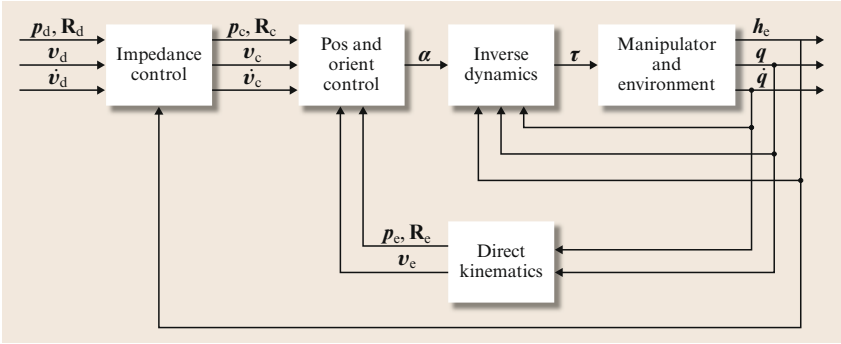


Fig. 9.2 Impedance control with inner motion control loop (admittance control)

The above relations show that, during interaction, the contact force can be made small at the expense of a large position error in steady state, as long as the active stiffness k_P is set low with respect to the stiffness of the environment k, and vice versa. However, both the contact force and the position error also depend on the external disturbance $\boldsymbol{\gamma}$; in particular, the lower k_P, the higher the influence of $\boldsymbol{\gamma}$ on both δx_{de} and h. Moreover, a low active stiffness k_P may result in a large position error also in the absence of interaction (i. e., when $k = 0$).

Admittance Control

A solution to this drawback can be devised by separating motion control from impedance control as follows. The motion control action is purposefully made stiff so as to enhance disturbance rejection but, rather than ensuring tracking of the desired end-effector position and orientation, it ensures tracking of a reference position and orientation resulting from the impedance control action. In other words, the desired position and orientation, together with the measured contact wrench, are input to the impedance equation which, via a suitable integration, generates the position and orientation to be used as a reference for the motion control.

To implement this solution, it is worth introducing a reference frame other than the desired frame Σ_d. This frame is referred to as the *compliant frame* Σ_c, and is specified by the quantities $\boldsymbol{p}_c$, $\mathbf{R}_c$, $\boldsymbol{v}_c$, and $\dot{\boldsymbol{v}}_c$ that are computed from $\boldsymbol{p}_d$, $\mathbf{R}_d$, $\boldsymbol{v}_d$, and $\dot{\boldsymbol{v}}_d$ and the measured wrench $\boldsymbol{h}_c$, by integrating the equation

$$\mathbf{K}_M \Delta \dot{\boldsymbol{v}}_{dc}^c + \mathbf{K}_D \Delta \boldsymbol{v}_{dc}^c + \boldsymbol{h}_\Delta^c = \boldsymbol{h}^c \,, \tag{9.27}$$

where $\boldsymbol{h}_\Delta^c$ is the elastic wrench in the presence of a finite displacement between the desired frame Σ_d and the compliant frame Σ_c. Then, a motion control strategy, based on inverse dynamics, is designed so that the end-effector frame Σ_e is taken to coincide with the compliant frame Σ_c. To guarantee the stability of the overall system, the bandwidth of the motion controller should be higher than the bandwidth of the impedance controller.

A block diagram of the resulting scheme is sketched in Fig. 9.2. It is evident that, in the absence of interaction, the compliant frame Σ_c coincides with the desired frame Σ_d and the dynamics of the position and orientation error, as well as the disturbance rejection capabilities, depend only on the gains of the inner motion control loop. On the other hand, the dynamic behavior in the presence of interaction is imposed by the impedance gains (9.27).

The control scheme of Fig. 9.2 is also known as *admittance control* because, in (9.27), the measured force (the input) is used to compute the motion of the compliant frame (the output), given the motion of the desired frame; a mapping with a force as input and a position or velocity as output corresponds to a mechanical *admittance*. Vice versa, (9.21), mapping the end-effector displacement (the input) from the desired motion trajectory into the contact wrench (the output), has the meaning of a mechanical *impedance*. Experiments on admittance control are reported in VIDEO 685.

Simplified Schemes

The inverse dynamics control is model based and requires modification of current industrial robot controllers, which are usually equipped with independent proportional–integral (PI) joint velocity controllers with very high bandwidth. These controllers are able to decouple the robot dynamics to a large extent, especially in the case of slow motion, and to mitigate the effects of external forces on the manipulator motion if the environment is sufficiently compliant. Hence, the closed-loop dynamics of the controlled robot can be approximated by

$$\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}_r$$

in joint space, or equivalently

$$\dot{\boldsymbol{v}}_e = \boldsymbol{v}_r \tag{9.28}$$

in the operational space, where $\dot{\boldsymbol{q}}_r$ and $\boldsymbol{v}_r$ are the control signals for the inner velocity motion loop generated by

a suitably designed outer control loop. These control signals are related by

$$\dot{\boldsymbol{q}}_{\mathrm{r}} = \mathbf{J}^{-1}(\boldsymbol{q})\boldsymbol{v}_{\mathrm{r}} \,.$$

The velocity $\boldsymbol{v}_{\mathrm{r}}$, corresponding to a *velocity-resolved control*, can be computed as

$$\boldsymbol{v}_{\mathrm{r}}^{\mathrm{e}} = \boldsymbol{v}_{\mathrm{d}}^{\mathrm{e}} + \mathbf{K}_{\mathrm{D}}^{-1}\left(\boldsymbol{h}_{\Delta}^{\mathrm{e}} - \boldsymbol{h}_{\mathrm{e}}^{\mathrm{e}}\right) \,,$$

where the control input has been referred to the end-effector frame, $\mathbf{K}_{\mathrm{D}}$ is a 6×6 positive-definite matrix and $\boldsymbol{h}_{\Delta}$ is the elastic wrench (9.13) with stiffness matrix $\mathbf{K}_{\mathrm{P}}$. The resulting closed-loop equation is

$$\mathbf{K}_{\mathrm{D}}\Delta\boldsymbol{v}_{\mathrm{de}}^{\mathrm{e}} + \boldsymbol{h}_{\Delta}^{\mathrm{e}} = \boldsymbol{h}_{\mathrm{e}}^{\mathrm{e}} \,,$$

corresponding to a compliant behavior of the end-effector characterized by a damping $\mathbf{K}_{\mathrm{D}}$ and a stiffness $\mathbf{K}_{\mathrm{P}}$. In the case $\mathbf{K}_{\mathrm{P}} = \mathbf{0}$, the resulting scheme is known as *damping control*.

Alternatively, an admittance-type control scheme can be adopted, where the motion of a compliant frame Σ_{c} can be computed as the solution of the differential equation

$$\mathbf{K}_{\mathrm{D}}\Delta\boldsymbol{v}_{\mathrm{dc}}^{\mathrm{c}} + \boldsymbol{h}_{\Delta}^{\mathrm{c}} = \boldsymbol{h}_{\mathrm{e}}^{\mathrm{c}}$$

in terms of the position $\boldsymbol{p}_{\mathrm{c}}$, orientation $\mathbf{R}_{\mathrm{c}}$, and velocity twist $\boldsymbol{v}_{\mathrm{c}}$, where the inputs are the motion variables of the desired frame Σ_{d} and the contact wrench $\boldsymbol{h}_{\mathrm{e}}^{\mathrm{c}}$. The motion variables of Σ_{c} are then input to an inner position and velocity controller. In the case $\mathbf{K}_{\mathrm{D}} = \mathbf{0}$, the resulting scheme is known as *compliance control*.

9.3 Interaction Tasks

Indirect force control does not require explicit knowledge of the environment, although to achieve a satisfactory dynamic behavior the control parameters have to be tuned for a particular task. On the other hand, a model of the interaction task is required for the synthesis of direct force control algorithms.

An interaction task is characterized by complex contact situations between the manipulator and the environment. To guarantee proper task execution, it is necessary to have an analytical description of the interaction force and moment, which is very demanding from a modeling viewpoint.

A real contact situation is a naturally distributed phenomenon in which the local characteristics of the contact surfaces as well as the global dynamics of the manipulator and environment are involved. In detail:

- The environment imposes kinematic constraints on the end-effector motion, due to one or more contacts of different type, and a reaction wrench arises when the end-effector tends to violate the constraints (e.g., the case of a robot sliding a rigid tool on a frictionless rigid surface).
- The end-effector, while being subject to kinematic constraints, may also exert a dynamic wrench on the environment, in the presence of environment dynamics (e.g., the case of a robot turning a crank, when the crank dynamics is relevant, or a robot pushing against a compliant surface).
- The contact wrench may depend on the structural compliance of the robot, due to the finite stiffness of the joints and links of the manipulator, as well as of the wrist force/torque sensor or of the tool (e.g., an end-effector mounted on an RCC device).
- Local deformation of the contact surfaces may occur during the interaction, producing distributed contact areas (e.g., the case of a soft contact surface of the tool or of the environment).
- Static and dynamic friction may occur in the case of non ideally smooth contact surfaces.

The design of the interaction control and the performance analysis are usually carried out under simplifying assumptions. The following two cases are considered:

1. The robot and the environment are perfectly rigid and purely kinematics constraints are imposed by the environment,
2. The robot is perfectly rigid, all the compliance in the system is localized in the environment, and the contact wrench is approximated by a linear elastic model.

In both cases, frictionless contact is assumed. It is obvious that these situations are only ideal. However, the robustness of the control should be able to cope with situations where some of the ieal assumptions are relaxed. In that case the control laws may be adapted to deal with nonideal characteristics.

9.3.1 Rigid Environment

The kinematic constraints imposed by the environment can be represented by a set of equations that the

variables describing the end-effector position and orientation must satisfy; since these variables depend on the joint variables through the direct kinematic equations, the constraint equations can also be expressed in the joint space as

$$\boldsymbol{\phi}(\boldsymbol{q}) = \mathbf{0} \,. \tag{9.29}$$

The vector $\boldsymbol{\phi}$ is an $m \times 1$ function, with $m < n$, where n is the number of joints of the manipulator, assumed to be nonredundant; without loss of generality, the case $n = 6$ is considered. Constraints of the form (9.29), involving only the generalized coordinates of the system, are known as holonomic constraints. The case of time-varying constraints of the form $\boldsymbol{\phi}(\boldsymbol{q}, t) = \mathbf{0}$ is not considered here but can be analyzed in a similar way. Moreover, only bilateral constraints expressed by equalities of the form (9.29) are of concern; this means that the end-effector always keeps contact with the environment. The analysis presented here is known as *kinetostatic analysis*.

It is assumed that the vector (9.29) is twice differentiable and that its m components are linearly independent at least locally in a neighborhood of the operating point. Hence, differentiation of (9.29) yields

$$\mathbf{J}_{\phi}(\boldsymbol{q})\dot{\boldsymbol{q}} = \mathbf{0} \,, \tag{9.30}$$

where $\mathbf{J}_{\phi}(\boldsymbol{q}) = \partial\boldsymbol{\phi}/\partial\boldsymbol{q}$ is the $m \times 6$ Jacobian of $\boldsymbol{\phi}(\boldsymbol{q})$, known as the *constraint Jacobian*. By virtue of the above assumption, $\mathbf{J}_{\phi}(\boldsymbol{q})$ is of rank m at least locally in a neighborhood of the operating point.

In the absence of friction, the generalized interaction forces are represented by a reaction wrench that tends to violate the constraints. This end-effector wrench produces reaction torques at the joints that can be computed using the principle of virtual work as

$$\boldsymbol{\tau}_{\mathrm{e}} = \mathbf{J}_{\phi}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{\lambda} \,,$$

where $\boldsymbol{\lambda}$ is an $m \times 1$ vector of Lagrange multipliers. The end-effector wrench corresponding to $\boldsymbol{\tau}_{\mathrm{e}}$ can be computed as

$$\boldsymbol{h}_{\mathrm{e}} = \mathbf{J}^{-\mathrm{T}}(\boldsymbol{q})\boldsymbol{\tau}_{\mathrm{e}} = \mathbf{S}_{\mathrm{f}}(\boldsymbol{q})\boldsymbol{\lambda} \,, \tag{9.31}$$

where

$$\mathbf{S}_{\mathrm{f}} = \mathbf{J}^{-\mathrm{T}}(\boldsymbol{q})\mathbf{J}_{\phi}^{\mathrm{T}}(\boldsymbol{q}) \,. \tag{9.32}$$

From (9.31) it follows that $\boldsymbol{h}_{\mathrm{e}}$ belongs to the m-dimensional vector space spanned by the columns of the $6 \times m$ matrix $\mathbf{S}_{\mathrm{f}}$. The inverse of the linear transformation (9.31) is computed as

$$\boldsymbol{\lambda} = \mathbf{S}_{\mathrm{f}}^{\dagger}(\boldsymbol{q})\boldsymbol{h}_{\mathrm{e}} \,, \tag{9.33}$$

where $\mathbf{S}_{\mathrm{f}}^{\dagger}$ denotes a weighted pseudoinverse of the matrix $\mathbf{S}_{\mathrm{f}}$, i. e.,

$$\mathbf{S}_{\mathrm{f}}^{\dagger} = \left(\mathbf{S}_{\mathrm{f}}^{\mathrm{T}}\mathbf{W}\mathbf{S}_{\mathrm{f}}\right)^{-1}\mathbf{S}_{\mathrm{f}}^{\mathrm{T}}\mathbf{W} \,, \tag{9.34}$$

where $\mathbf{W}$ is a suitable weighting matrix.

Notice that, while the range space of the matrix $\mathbf{S}_{\mathrm{f}}$ in (9.32) is uniquely defined by the geometry of the contact, the matrix $\mathbf{S}_{\mathrm{f}}$ itself is not unique; also, the constraint equations (9.29), the corresponding Jacobian $\mathbf{J}_{\phi}$ as well as the pseudoinverse $\mathbf{S}_{\mathrm{f}}^{\dagger}$ and the vector $\boldsymbol{\lambda}$ are not uniquely defined.

In general, the physical units of measure of the elements of $\boldsymbol{\lambda}$ are not homogeneous and the columns of the matrix $\mathbf{S}_{\mathrm{f}}$, as well as of the matrix $\mathbf{S}_{\mathrm{f}}^{\dagger}$, do not necessarily represent homogeneous entities. This may produce invariance problems in the transformation (9.33) if $\boldsymbol{h}_{\mathrm{e}}$ represents a measured wrench that is subject to disturbances and, as a result, may have components outside the range space of $\mathbf{S}_{\mathrm{f}}$. If a physical unit or a reference frame is changed, the matrix $\mathbf{S}_{\mathrm{f}}$ undergoes a transformation; however, the result of (9.33) with the transformed pseudoinverse in general depends on the adopted physical units or on the reference frame. The reason is that the pseudoinverse is the weighted least-squares solution of a minimization problem based on the norm of the vector $\boldsymbol{h}_{\mathrm{e}} - \mathbf{S}_{\mathrm{f}}(\boldsymbol{q})\boldsymbol{\lambda}$, and the invariance can be guaranteed only if a physically consistent norm of this vector is used. In the ideal case that $\boldsymbol{h}_{\mathrm{e}}$ is in the range space of $\mathbf{S}_{\mathrm{f}}$, there is a unique solution for $\boldsymbol{\lambda}$ in (9.33), regardless of the weighting matrix, and hence the invariance problem does not appear.

A possible solution consists of choosing $\mathbf{S}_{\mathrm{f}}$ so that its columns represent linearly independent wrenches. This implies that (9.31) gives $\boldsymbol{h}_{\mathrm{e}}$ as a linear combination of wrenches and $\boldsymbol{\lambda}$ is a dimensionless vector. A physically consistent norm on the wrench space can be defined based on the quadratic form $\boldsymbol{h}_{\mathrm{e}}^{\mathrm{T}}\mathbf{K}^{-1}\boldsymbol{h}_{\mathrm{e}}$, which has the meaning of an elastic energy if $\mathbf{K}$ is a positive-definite matrix corresponding to a stiffness. Hence, the choice $\mathbf{W} = \mathbf{K}^{-1}$ can be made for the weighting matrix of the pseudoinverse.

Notice that, for a given $\mathbf{S}_{\mathrm{f}}$, the constraint Jacobian can be computed from (9.32) as $\mathbf{J}_{\phi}(\boldsymbol{q}) = \mathbf{S}_{\mathrm{f}}^{\mathrm{T}}\mathbf{J}(\boldsymbol{q})$; moreover, the constraint equations can be derived by integrating (9.30).

Using (9.1) and (9.32), the equality (9.30) can be rewritten in the form

$$\mathbf{J}_{\phi}(\boldsymbol{q})\mathbf{J}^{-1}(\boldsymbol{q})\mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} = \mathbf{S}_{\mathrm{f}}^{\mathrm{T}}\boldsymbol{v}_{\mathrm{e}} = \mathbf{0} \,, \tag{9.35}$$

which, by virtue of (9.31), is equivalent to

$$\boldsymbol{h}_{\mathrm{e}}^{\mathrm{T}}\boldsymbol{v}_{\mathrm{e}} = 0 \,. \tag{9.36}$$

Equation (9.36) represents the kinetostatic relationship, known as *reciprocity*, between the ideal reaction wrench $\boldsymbol{h}_e$ (belonging to the so-called *force-controlled* subspace) and the end-effector twist that obeys the constraints (belonging to the so-called *velocity-controlled* subspace). The concept of reciprocity, expressing the physical fact that, in the hypothesis of rigid and frictionless contact, the wrench does not cause any work against the twist, is often confused with the concept of *orthogonality*, which makes no sense in this case because twists and wrenches belong to different spaces.

Equations ((9.35)) and (9.36) imply that the velocity-controlled subspace is the reciprocal complement of the m-dimensional force-controlled subspace, identified by the range of matrix $\mathbf{S}_f$. Hence, the dimension of the velocity-controlled subspace is $6-m$ and a $6\times(6-m)$ matrix $\mathbf{S}_v$ can be defined, whose columns span the velocity-controlled subspace, i. e.,

$$\boldsymbol{v}_e = \mathbf{S}_v(\boldsymbol{q})\boldsymbol{\nu}\,, \tag{9.37}$$

where $\boldsymbol{\nu}$ is a suitable $(6-m)\times 1$ vector. From (9.35) and (9.37) the following equality holds

$$\mathbf{S}_f^T(\boldsymbol{q})\mathbf{S}_v(\boldsymbol{q}) = \mathbf{0}\,; \tag{9.38}$$

moreover, the inverse of the linear transformation (9.37) can be computed as

$$\boldsymbol{\nu} = \mathbf{S}_v^\dagger(\boldsymbol{q})\boldsymbol{v}_e\,, \tag{9.39}$$

where $\mathbf{S}_v^\dagger$ denotes a suitable weighted pseudoinverse of the matrix $\mathbf{S}_v$, computed as in (9.34).

Notice that, as for the case of $\mathbf{S}_f$, although the range space of the matrix $\mathbf{S}_v$ is uniquely defined, the choice of the matrix $\mathbf{S}_v$ itself is not unique. Moreover, the columns of $\mathbf{S}_v$ are not necessarily twists and the scalar $\boldsymbol{\nu}$ may have nonhomogeneous physical dimensions. However, in order to avoid invariance problems analogous to that considered for the case of $\mathbf{S}_f$, it is convenient to select the columns of $\mathbf{S}_v$ as twists so that the vector $\boldsymbol{\nu}$ is dimensionless; moreover, the weighting matrix used to compute the pseudoinverse in (9.39) can be set as $\mathbf{W}=\mathbf{M}$, being $\mathbf{M}$ a 6×6 inertia matrix; this corresponds to defining a norm in the space of twists based on the kinetic energy. It is worth observing that the transformation matrices of twists and wrenches, corresponding to a change of reference frame, are different; however, if twists are defined with angular velocity on top and translational velocity on bottom, then their transformation matrix is the same as for wrenches.

The matrix $\mathbf{S}_v$ may also have an interpretation in terms of Jacobians, as for $\mathbf{S}_f$ in (9.32). Due to the presence of m independent holonomic constraints (9.29), the configuration of the robot in contact with the environment can be described in terms of a $(6-m)\times 1$ vector $\boldsymbol{r}$ of independent variables. From the implicit function theorem, this vector can be defined as

$$\boldsymbol{r} = \boldsymbol{\psi}(\boldsymbol{q})\,, \tag{9.40}$$

where $\boldsymbol{\psi}(\boldsymbol{q})$ is any $(6-m)\times 1$ twice-differentiable vector function such that the m components of $\boldsymbol{\phi}(\boldsymbol{q})$ and the $n-m$ components of $\boldsymbol{\psi}(\boldsymbol{q})$ are linearly independent at least locally in a neighborhood of the operating point. This means that the mapping (9.40), together with the constraint (9.29), is locally invertible, with inverse defined as

$$\boldsymbol{q} = \boldsymbol{\rho}(\boldsymbol{r})\,, \tag{9.41}$$

where $\boldsymbol{\rho}(\boldsymbol{r})$ is a 6×1 twice-differentiable vector function. Equation (9.41) explicitly provides all the joint vectors which satisfy the constraint (9.29). Moreover, the joint velocity vectors that satisfy (9.30) can be computed as

$$\dot{\boldsymbol{q}} = \mathbf{J}_\rho(\boldsymbol{r})\dot{\boldsymbol{r}}\,,$$

where $\mathbf{J}_\rho(\boldsymbol{r}) = \partial\boldsymbol{\rho}/\partial\boldsymbol{r}$ is a $6\times(6-m)$ full-rank Jacobian matrix. Therefore, the following equality holds

$$\mathbf{J}_\phi(\boldsymbol{q})\mathbf{J}_\rho(\boldsymbol{r}) = \mathbf{0}\,,$$

which can be interpreted as a reciprocity condition between the subspace of the reaction torques spanned by the columns of the matrix $\mathbf{J}_\phi^T$ and the subspace of the constrained joint velocities spanned by the columns of the matrix $\mathbf{J}_\rho$.

Rewriting the above equation as

$$\mathbf{J}_\phi(\boldsymbol{q})\mathbf{J}(\boldsymbol{q})^{-1}\mathbf{J}(\boldsymbol{q})\mathbf{J}_\rho(\boldsymbol{r}) = \mathbf{0}\,,$$

and taking into account (9.32) and (9.38), the matrix $\mathbf{S}_v$ can be expressed as

$$\mathbf{S}_v = \mathbf{J}(\boldsymbol{q})\mathbf{J}_\rho(\boldsymbol{r})\,, \tag{9.42}$$

which, by virtue of (9.40) and (9.41), it can be equivalently expressed as a function of either $\boldsymbol{q}$ or $\boldsymbol{r}$.

The matrices $\mathbf{S}_f$ and $\mathbf{S}_v$, and their pseudoinverse $\mathbf{S}_f^\dagger$ and $\mathbf{S}_v^\dagger$ are known as selection matrices. They play a fundamental role for the task specification, i. e., the specification of the desired end-effector motion and interaction forces and moments, as well as for the control synthesis.

9.3.2 Compliant Environment

In many applications, the interaction wrench between the end-effector and a compliant environment can be approximated by an ideal elastic model of the form (9.24). However, since the stiffness matrix $\mathbf{K}$ is positive definite, this model describes a fully constrained case, when the environment deformation coincides with the infinitesimal twist displacement of the end-effector. In general, however, the end-effector motion is only partially constrained by the environment and this situation can be modeled by introducing a suitable positive-semidefinite stiffness matrix.

The stiffness matrix describing the partially constrained interaction between the end-effector and the environment can be computed by modeling the environment as a couple of rigid bodies, S and O, connected through an ideal 6-DOF spring of compliance $\mathbf{C} = \mathbf{K}^{-1}$. Body S is attached to a frame Σ_s and is in contact with the end-effector; body O is attached to a frame Σ_o which, at equilibrium, coincides with frame Σ_s. The environment deformation about the equilibrium, in the presence of a wrench $\boldsymbol{h}_s$, is represented by the infinitesimal twist displacement $\delta\boldsymbol{x}_{so}$ between frames Σ_s and Σ_o, which can be computed as

$$\delta\boldsymbol{x}_{so} = \mathbf{C}\,\boldsymbol{h}_s \,. \tag{9.43}$$

All the quantities hereafter are referred to frame Σ_s but the superscript s is omitted for brevity.

For the considered contact situation, the end-effector twist does not completely belong to the ideal velocity subspace, corresponding to a rigid environment, because the environment can deform. Therefore, the infinitesimal twist displacement of the end-effector frame Σ_e with respect to Σ_o can be decomposed as

$$\delta\boldsymbol{x}_{eo} = \delta\boldsymbol{x}_v + \delta\boldsymbol{x}_f \,, \tag{9.44}$$

where $\delta\boldsymbol{x}_v$ is the end-effector infinitesimal twist displacement in the velocity-controlled subspace, defined as the $6-m$ reciprocal complement of the force-controlled subspace, while $\delta\boldsymbol{x}_f$ is the end-effector infinitesimal twist displacement corresponding to the environment deformation. Hence,

$$\delta\boldsymbol{x}_v = \mathbf{P}_v\delta\boldsymbol{x}_{eo} \,, \tag{9.45}$$

$$\delta\boldsymbol{x}_f = (\mathbf{I}-\mathbf{P}_v)\delta\boldsymbol{x}_{eo} = (\mathbf{I}-\mathbf{P}_v)\delta\boldsymbol{x}_{so} \,, \tag{9.46}$$

where $\mathbf{P}_v = \mathbf{S}_v\mathbf{S}_v^{\dagger}$ and $\mathbf{S}_v$ and $\mathbf{S}_v^{\dagger}$ are defined as in the rigid environment case. Being $\mathbf{P}_v\mathbf{P}_v = \mathbf{P}_v$, the matrix $\mathbf{P}_v$ is a projection matrix that filters out all the end-effector twists (and infinitesimal twist displacements) that are not in the range space of $\mathbf{S}_v$. Moreover, $\mathbf{I}-\mathbf{P}_v$ is a projection matrix that filters out all the end-effector twists (and infinitesimal twist displacements) that are in the range space of $\mathbf{S}_v$. The twists $\mathbf{P}_v\boldsymbol{v}$ are denoted as *twists of freedom* while the twists $(\mathbf{I}-\mathbf{P}_v)\boldsymbol{v}$ are denoted as *twists of constraint*.

In the hypothesis of frictionless contact, the interaction wrench between the end-effector and the environment is restricted to a force-controlled subspace defined by the m-dimensional range space of a matrix $\mathbf{S}_f$, as for the rigid environment case, i. e.,

$$\boldsymbol{h}_e = \mathbf{S}_f\boldsymbol{\lambda} = \boldsymbol{h}_s \,, \tag{9.47}$$

where $\boldsymbol{\lambda}$ is an $m\times 1$ dimensionless vector. Premultiplying both sides of (9.44) by $\mathbf{S}_f^{\mathrm{T}}$ and using (9.43), (9.45), (9.46), and (9.47) yields

$$\mathbf{S}_f^{\mathrm{T}}\delta\boldsymbol{x}_{eo} = \mathbf{S}_f^{\mathrm{T}}\mathbf{C}\,\mathbf{S}_f\boldsymbol{\lambda} \,,$$

where the identity $\mathbf{S}_f^{\mathrm{T}}\mathbf{P}_v = \mathbf{0}$ has been exploited. Therefore, the following elastic model can be found

$$\boldsymbol{h}_e = \mathbf{S}_f\boldsymbol{\lambda} = \mathbf{K}'\delta\boldsymbol{x}_{eo} \,, \tag{9.48}$$

where $\mathbf{K}' = \mathbf{S}_f(\mathbf{S}_f^{\mathrm{T}}\mathbf{C}\mathbf{S}_f)^{-1}\mathbf{S}_f^{\mathrm{T}}$ is the positive-semidefinite stiffness matrix corresponding to the partially constrained interaction.

If the compliance matrix $\mathbf{C}$ is adopted as a weighting matrix for the computation of $\mathbf{S}_f^{\dagger}$, then $\mathbf{K}'$ can be expressed as

$$\mathbf{K}' = \mathbf{P}_f\mathbf{K} \,, \tag{9.49}$$

where $\mathbf{P}_f = \mathbf{S}_f\mathbf{S}_f^{\dagger}$. Being $\mathbf{P}_f\mathbf{P}_f = \mathbf{P}_f$, matrix $\mathbf{P}_f$ is a projection matrix that filters out all the end-effector wrenches that are not in the range space of $\mathbf{S}_f$.

The compliance matrix for the partially constrained interaction cannot be computed as the inverse of $\mathbf{K}'$, since this matrix is of rank $m<6$. However, using (9.46), (9.43), and (9.47), the following equality can be found

$$\delta\boldsymbol{x}_f = \mathbf{C}'\boldsymbol{h}_e \,,$$

where the matrix

$$\mathbf{C}' = (\mathbf{I}-\mathbf{P}_v)\mathbf{C} \,, \tag{9.50}$$

of rank $6-m$, is positive semidefinite. If the stiffness matrix $\mathbf{K}$ is adopted as a weighting matrix for the computation of $\mathbf{S}_v^{\dagger}$, then the matrix $\mathbf{C}'$ has the noticeable expression $\mathbf{C}' = \mathbf{C}-\mathbf{S}_v(\mathbf{S}_v^{\mathrm{T}}\mathbf{K}\,\mathbf{S}_v)^{-1}\mathbf{S}_v^{\mathrm{T}}$, showing that $\mathbf{C}'$ is symmetric.

9.3.3 Task Specification

An interaction task can be assigned in terms of a desired end-effector wrench $\boldsymbol{h}_\mathrm{d}$ and twist $\boldsymbol{v}_\mathrm{d}$. In order to be consistent with the constraints, these vectors must lie in the force- and velocity-controlled subspaces, respectively. This can be guaranteed by specifying vectors $\boldsymbol{\lambda}_\mathrm{d}$ and $\boldsymbol{\nu}_\mathrm{d}$ and computing $\boldsymbol{h}_\mathrm{d}$ and $\boldsymbol{v}_\mathrm{d}$ as

$$\boldsymbol{h}_\mathrm{d} = \mathbf{S}_\mathrm{f}\boldsymbol{\lambda}_\mathrm{d}, \quad \boldsymbol{v}_\mathrm{d} = \mathbf{S}_\mathrm{v}\boldsymbol{\nu}_\mathrm{d} ,$$

where $\mathbf{S}_\mathrm{f}$ and $\mathbf{S}_\mathrm{v}$ have to be suitably defined on the basis of the geometry of the task, and so that invariance with respect to the choice of the reference frame and change of physical units is guaranteed.

Many robotic tasks have a set of orthogonal reference frames in which the task specification is very easy and intuitive. Such frames are called *task frames* or *compliance frames*. An interaction task can be specified by assigning a desired force/torque or a desired linear/angular velocity along/about each of the frame axes. The desired quantities are known as *artificial constraints* because they are imposed by the controller; these constraints, in the case of rigid contact, are complementary to those imposed by the environment, known as *natural constraints*.

Some examples of task frame definition and task specification are given below.

Peg-in-Hole

The goal of this task is to push the peg into the hole while avoiding wedging and jamming. The peg has two degrees of motion freedom, hence the dimension of the velocity-controlled subspace is $6-m=2$, while the dimension of the force-controlled subspace is $m=4$. The task frame can be chosen as shown in Fig. 9.3 and the task can be achieved by assigning the following desired forces and torques:

- Zero forces along the x_t and y_t axes
- Zero torques about the x_t and y_t axes,

and the desired velocities:

- A nonzero linear velocity along the z_t-axis
- An arbitrary angular velocity about the z_t-axis.

The task continues until a *large* reaction force in the z_t direction is measured, indicating that the peg has hit the bottom of the hole, not represented in the figure. Hence, the matrices $\mathbf{S}_\mathrm{f}$ and $\mathbf{S}_\mathrm{v}$ can be chosen as

$$\mathbf{S}_\mathrm{f} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \qquad \mathbf{S}_\mathrm{v} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix},$$

where the columns of $\mathbf{S}_\mathrm{f}$ have the dimensions of wrenches and those of $\mathbf{S}_\mathrm{v}$ have the dimensions of twists, defined in the task frame, and they transform accordingly when changing the reference frame. The task frame can be chosen either attached to the end-effector or to the environment.

Turning a Crank

The goal of this task is turning a crank with an idle handle. The handle has two degrees of motion freedom, corresponding to the rotation about the z_t-axis and to the rotation about the rotation axis of the crank. Hence the dimension of the velocity-controlled subspace is $6-m=2$, while the dimension of the force-controlled subspace is $m=4$. The task frame can be assumed as in the Fig. 9.4, attached to the crank. The task can be achieved by assigning the following desired forces and torques:

- Zero forces along the x_t and z_t axes
- Zero torques about the x_t and y_t axes.

and the following desired velocities:

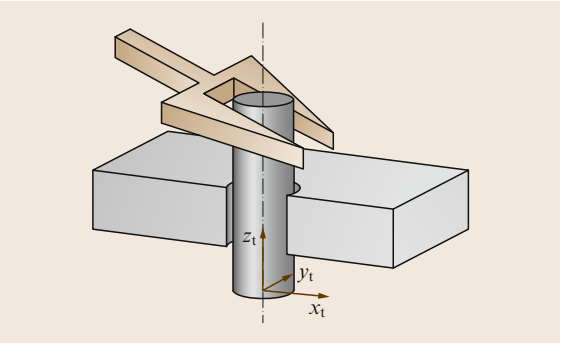


Fig. 9.3 Insertion of a cylindrical peg into a hole

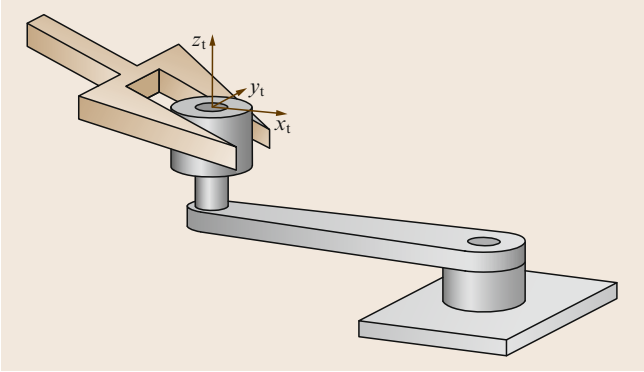


Fig. 9.4 Turning a crank with an idle handle

- A nonzero linear velocity along the y_t-axis
- An arbitrary angular velocity about the z_t-axis.

Hence, the matrices $\mathbf{S}_f$ and $\mathbf{S}_v$ can be chosen as

$$\mathbf{S}_f = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad \mathbf{S}_v = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix},$$

referred to the task frame. In this case, the task frame is fixed with respect to the crank, but in motion with respect both the end-effector frame (fixed to the handle) and to the base frame of the robot. Hence, the matrices $\mathbf{S}_f$ and $\mathbf{S}_v$ are time variant when referred either to the end-effector frame or to the base frame.

Sliding a Block on a Planar Elastic Surface

The goal of this task is to slide a prismatic block over a planar surface along the x_t-axis, while pushing with a given force against the elastic planar surface. The object has three degrees of motion freedom, hence the dimension of the velocity-controlled subspace is $6 - m = 3$ while the dimension of the force-controlled subspace is $m = 3$. The task frame can be chosen attached to the environment, as shown in Fig. 9.5, and the task can be achieved by assigning the desired velocities:

- A nonzero velocity along the x_t-axis
- A zero velocity along the y_t-axis
- A zero angular velocity about the z_t-axis,

and the desired forces and torques:

- A nonzero force along the z_t-axis
- Zero torques about the x_t and y_t axes.

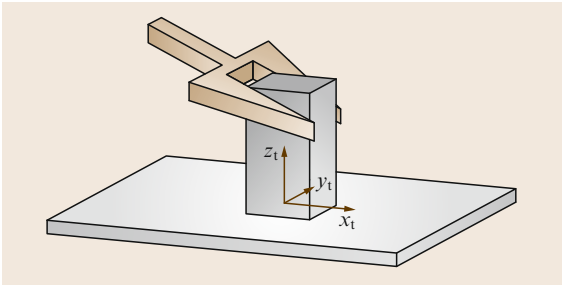


Fig. 9.5 Sliding of a prismatic object on a planar elastic surface

Hence, the matrices $\mathbf{S}_f$ and $\mathbf{S}_v$ can be chosen as

$$\mathbf{S}_f = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \quad \mathbf{S}_v = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The elements of the 6×6 stiffness matrix $\mathbf{K}'$, corresponding to the partially constrained interaction of the end-effector with the environment, are all zero except those of the 3×3 principal minor $\mathbf{K}'_m$ formed by rows $3, 4, 5$ and columns $3, 4, 5$ of $\mathbf{K}'$, which can be computed as

$$\mathbf{K}'_m = \begin{pmatrix} c_{3,3} & c_{3,4} & c_{3,5} \\ c_{4,3} & c_{4,4} & c_{4,5} \\ c_{5,3} & c_{5,4} & c_{5,5} \end{pmatrix}^{-1},$$

where $c_{i,j} = c_{j,i}$ are the elements of the compliance matrix $\mathbf{C}$.

General Contact Model

The task frame concept has proven to be very useful for the specification of a variety of practical robotic tasks. However, it only applies to task geometries with limited complexity, for which separate control modes can be assigned independently to three pure translational and three pure rotational directions along the axes of a single frame. For more complex situations, as in the case of multiple-point contact, a task frame may not exist and more complex contact models have to be adopted. A possible solution is represented by the *virtual contact manipulator* model, where each individual contact is modeled by a virtual kinematic chain between the manipulated object and the environment, giving the manipulated object (instantaneously) the same motion freedom as the contact. The velocities and force kinematics of the parallel manipulator, formed by the virtual manipulators of all individual contacts, can be derived using the standard kinematics procedures of real manipulators and allow the construction of bases for the twist and wrench spaces of the total motion constraint.

A more general approach, known as *constraint-based task specification* opens up new applications involving complex geometries and/or the use of multiple sensors (force/torque, distance, visual sensors) for controlling different directions in space simultaneously. The concept of task frame is extended to that of multiple *feature frames*. Each of the feature frames makes it possible to model part of the task geometry using translational and rotational directions along the axes of a frame; also, part of the constraints is specified in each

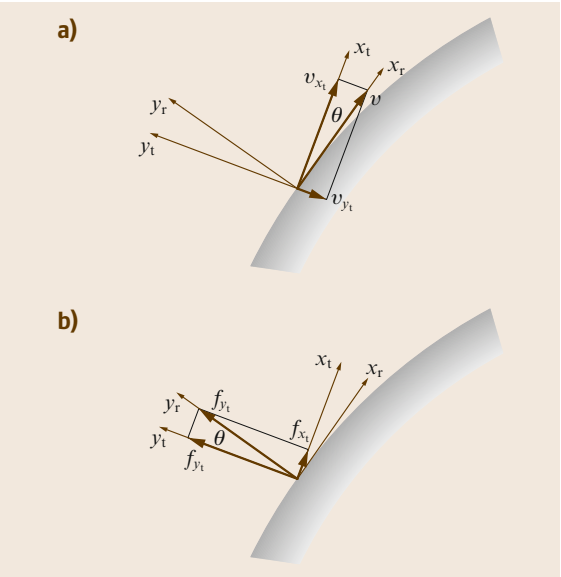


Fig.9.6a,b Estimation of orientation error: **(a)** velocity-based approach, **(b)** force-based approach

of the feature frames. The total model and the total set of constraints are achieved by collecting the partial task and constraints descriptions, each expressed in the individual feature frames.

9.3.4 Sensor-Based Contact Model Estimation

The task specification relies on the definition of the velocity-controlled subspace and of the force-controlled subspace assuming that an accurate model of the contact is available all the time. On the other hand, in most practical implementations, the selection matrices $\mathbf{S}_f$ and $\mathbf{S}_v$ are not exactly known, however many interaction control strategies turn out to be rather robust against modeling errors. In fact, to cope reliably with these situations is exactly why force control is used. The robustness of the force controller increases if the matrices $\mathbf{S}_f$ and $\mathbf{S}_v$ can be continuously updated, using motion and/or force measurements, during task execution.

In detail, a nominal model is assumed to be available; when the contact situation evolves differently from what the model predicts, the measured and predicted motion and force will begin to deviate. These small incompatibilities can be measured and can then be used to adapt the model online, using algorithms derived from classical state-space *prediction–correction* estimators such as the Kalman filter.

Figure 9.6 reports an example of error between nominal and measured motion and force variables, typical of a two-dimensional contour-following task. The orientation of the contact normal changes if the environment is notplanar. Hence an angular error $\boldsymbol{\theta}$ appears between the nominal contact normal, aligned to the y_t-axis of the task frame (the frame with axes x_t and y_t), and the real contact normal, aligned to the y_r-axis of the real task frame (the frame with axes x_r and y_r). This angle can be estimated with either velocity or force measurements only:

- *Velocity-based approach*: the executed linear velocity $\boldsymbol{v}$, which is tangent to the real contour (aligned to the x_r-axis), does not completely lie along the x_t-axis, but has a small component v_{y_t} along the y_t-axis. The orientation error $\boldsymbol{\theta}$ can then be approximated by $\boldsymbol{\theta} = \tan^{-1}(v_{y_t}/v_{x_t})$.
- *Force-based approach*: the measured (ideal) contact force $\boldsymbol{f}$ does not completely lie along the nominal normal direction, aligned to the y_t-axis, but has a small component f_{x_t} along the x_t-axis. The orientation error $\boldsymbol{\theta}$ can then be approximated by $\boldsymbol{\theta} = \tan^{-1}(f_{x_t}/f_{y_t})$.

The velocity-based approach is disturbed by mechanical compliance in the system; the force-based approach is disturbed by contact friction.

9.4 Hybrid Force/Motion Control

The aim of hybrid force/motion control is to split up simultaneous control of both end-effector motion and contact forces into two separate decoupled subproblems. In the following, the main control approaches in the hybrid framework are presented for the cases of both rigid and compliant environments.

9.4.1 Acceleration-Resolved Approach

As for the case of motion control, the acceleration-resolved approach is aimed at decoupling and linearizing the nonlinear robot dynamics at the acceleration level, via an inverse dynamics control law. In the presence of interaction with the environment, a complete decoupling between the force- and velocity-controlled subspaces is sought. The basic idea is that of designing a model-based *inner control loop* to compensate for the nonlinear dynamics of the robot manipulator and decouple the force and velocity subspaces; hence an *outer control loop* is designed to ensure disturbance rejection and tracking of the desired end-effector force and motion.

Rigid Environment

In the case of a rigid environment, the external wrench can be written in the form $\boldsymbol{h}_e = \mathbf{S}_f \boldsymbol{\lambda}$. The force multiplier $\boldsymbol{\lambda}$ can be computed from (9.2) by multiplying both sides of (9.2) by the weighted pseudo-inverse $\mathbf{S}_f^{\dagger}$ with weight $\boldsymbol{\Lambda}^{-1}(\boldsymbol{q})$ and using the time derivative of the last equality (9.35). This yields

$$\boldsymbol{\lambda} = \mathbf{S}_f^{\dagger}(\boldsymbol{q})\,[\boldsymbol{h}_c - \boldsymbol{\mu}(\boldsymbol{q}, \dot{\boldsymbol{q}})] + \boldsymbol{\Lambda}_f(\boldsymbol{q})\dot{\mathbf{S}}_f^{\mathrm{T}} \boldsymbol{v}_e \,, \tag{9.51}$$

where $\boldsymbol{\Lambda}_f(\boldsymbol{q}) = (\mathbf{S}_f^{\mathrm{T}} \boldsymbol{\Lambda}^{-1} \mathbf{S}_f)^{-1}$ and $\boldsymbol{\mu}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\Gamma}\dot{\boldsymbol{q}} + \boldsymbol{\eta}$. Replacing again (9.51) into (9.2) gives

$$\boldsymbol{\Lambda}(\boldsymbol{q})\dot{\boldsymbol{v}}_e + \mathbf{S}_f \boldsymbol{\Lambda}_f(\boldsymbol{q})\dot{\mathbf{S}}_f^{\mathrm{T}} \boldsymbol{v}_e = \mathbf{N}_f(\boldsymbol{q})[\boldsymbol{h}_c - \boldsymbol{\mu}(\boldsymbol{q}, \dot{\boldsymbol{q}})] \,, \tag{9.52}$$

with $\mathbf{N}_f = \mathbf{I} - \mathbf{S}_f \mathbf{S}_f^{\dagger}$. Notice that $\mathbf{N}_f \mathbf{N}_f = \mathbf{N}_f$ and $\mathbf{N}_f \mathbf{S}_f = \mathbf{0}$, hence the 6×6 matrix $\mathbf{N}_f$ is a projection matrix that filters out all the end-effector wrenches lying in the range of $\mathbf{S}_f$. These correspond to wrenches that tend to violate the constraints. Equation (9.52) represents a set of six second-order differential equations whose solution, if initialized on the constraints, automatically satisfy (9.29) at all times.

The reduced-order dynamics of the constrained system is described by $6-m$ second-order equations that are obtained by premultiplying both sides of (9.52) by the matrix $\mathbf{S}_v^{\mathrm{T}}$ and substituting the acceleration $\dot{\boldsymbol{v}}_e$ with

$$\dot{\boldsymbol{v}}_e = \mathbf{S}_v \dot{\boldsymbol{\nu}} + \dot{\mathbf{S}}_v \boldsymbol{\nu} \,.$$

The resulting equations are

$$\boldsymbol{\Lambda}_v(\boldsymbol{q})\dot{\boldsymbol{\nu}} = \mathbf{S}_v^{\mathrm{T}} \left[\boldsymbol{h}_c - \boldsymbol{\mu}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \boldsymbol{\Lambda}(\boldsymbol{q})\dot{\mathbf{S}}_v \boldsymbol{\nu} \right] \,, \tag{9.53}$$

where

$$\boldsymbol{\Lambda}_v = \mathbf{S}_v^{\mathrm{T}} \boldsymbol{\Lambda} \mathbf{S}_v$$

and the identities (9.38) and

$$\mathbf{S}_v^{\mathrm{T}} \mathbf{N}_f = \mathbf{S}_v^{\mathrm{T}}$$

have been used. Moreover, expression (9.51) can be rewritten as

$$\boldsymbol{\lambda} = \mathbf{S}_f^{\dagger}(\boldsymbol{q}) \left[\boldsymbol{h}_c - \boldsymbol{\mu}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \boldsymbol{\Lambda}(\boldsymbol{q})\dot{\mathbf{S}}_v \boldsymbol{\nu} \right] \,, \tag{9.54}$$

where the identity $\dot{\mathbf{S}}_f^{\mathrm{T}} \mathbf{S}_v = -\mathbf{S}_f^{\mathrm{T}} \dot{\mathbf{S}}_v$ has been exploited.

Equation (9.54) reveals that the force multiplier vector $\boldsymbol{\lambda}$ instantaneously depends also on the applied input wrench $\boldsymbol{h}_c$. Hence, by suitably choosing $\boldsymbol{h}_c$, it is possible to directly control the m independent components of the end-effector wrench that tend to violate the constraints; these components can be computed from the m force multipliers through (9.31). On the other hand, the $6-m$ independent components of the end-effector velocity twist can be controlled through $\boldsymbol{h}_c$ via (9.53).

An inverse-dynamics *inner control loop* can be designed by choosing the control wrench $\boldsymbol{h}_c$ as

$$\boldsymbol{h}_c = \boldsymbol{\Lambda}(\boldsymbol{q})\mathbf{S}_v \boldsymbol{\alpha}_v + \mathbf{S}_f \boldsymbol{f}_\lambda + \boldsymbol{\mu}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\Lambda}(\boldsymbol{q})\dot{\mathbf{S}}_v \boldsymbol{\nu} \,, \tag{9.55}$$

where $\boldsymbol{\alpha}_v$ and $\boldsymbol{f}_\lambda$ are properly designed control inputs.

Substituting (9.55) into (9.53) and (9.51) yields

$$\dot{\boldsymbol{\nu}} = \boldsymbol{\alpha}_v \,,$$
$$\boldsymbol{\lambda} = \boldsymbol{f}_\lambda \,,$$

showing that control law (9.55) allows complete decoupling between the force- and velocity-controlled subspaces.

It is worth noticing that, for the implementation of the control law (9.55), the constraint (9.29) as well as (9.40) defining the vector of the configuration variables for the constrained system are not required, provided that the matrices $\mathbf{S}_f$ and $\mathbf{S}_v$ are known or estimated online. In these cases, the task can easily be assigned by specifying a desired force, in terms of the vector $\boldsymbol{\lambda}_d(t)$, and a desired velocity, in terms of the vector $\boldsymbol{\nu}_d(t)$; moreover, a force/velocity control is implemented.

The desired force $\boldsymbol{\lambda}_d(t)$ can be achieved by setting

$$\boldsymbol{f}_\lambda = \boldsymbol{\lambda}_d(t) \,, \tag{9.56}$$

but this choice is very sensitive to disturbance forces, since it contains no force feedback. Alternative choices are

$$\boldsymbol{f}_\lambda = \boldsymbol{\lambda}_d(t) + \mathbf{K}_{P\lambda} [\boldsymbol{\lambda}_d(t) - \boldsymbol{\lambda}(t)] \,, \tag{9.57}$$

or

$$\boldsymbol{f}_\lambda = \boldsymbol{\lambda}_d(t) + \mathbf{K}_{I\lambda} \int_0^t [\boldsymbol{\lambda}_d(\tau) - \boldsymbol{\lambda}(\tau)] \, \mathrm{d}\tau \,, \tag{9.58}$$

where $\mathbf{K}_{P\lambda}$ and $\mathbf{K}_{I\lambda}$ are suitable positive-definite matrix gains. The proportional feedback is able to reduce the force error due to disturbance forces, while the integral action is able to compensate for constant bias disturbances.

The implementation of force feedback requires the computation of the force multiplier λ from the measurement of the end-effector wrench $\boldsymbol{h}_e$, which can be achieved by using (9.33).

Velocity control is achieved by setting

$$\boldsymbol{\alpha}_v = \dot{\boldsymbol{\nu}}_d(t) + \mathbf{K}_{P\nu} [\boldsymbol{\nu}_d(t) - \boldsymbol{\nu}(t)] + \mathbf{K}_{I\nu} \int_0^t [\boldsymbol{\nu}_d(\tau) - \boldsymbol{\nu}(\tau)] \, \mathrm{d}\tau \,, \tag{9.59}$$

where $\mathbf{K}_{P\nu}$ and $\mathbf{K}_{I\nu}$ are suitable matrix gains. It is straightforward to show that asymptotic tracking of $\boldsymbol{\nu}_d(t)$ and $\dot{\boldsymbol{\nu}}_d(t)$ is ensured with exponential convergence for any choice of positive-definite matrices $\mathbf{K}_{P\nu}$ and $\mathbf{K}_{I\nu}$.

The computation of the vector $\boldsymbol{\nu}$ from the available measurements can be achieved using (9.39), where the end-effector twist is computed from joint position and velocity measurements through (9.1).

Equations (9.57) or (9.58) and (9.59) represent the *outer control loop* ensuring *force/velocity* control and disturbance rejection.

When (9.29) and (9.40) are known, the matrices $\mathbf{S}_f$ and $\mathbf{S}_v$ can be computed according to (9.32) and (9.42) and a *force/position* control can be designed by specifying a desired force $\boldsymbol{\lambda}_d(t)$ and a desired position $\boldsymbol{r}_d(t)$.

Force control can be designed as above, while position control can be achieved by setting

$$\boldsymbol{\alpha}_\nu = \ddot{\boldsymbol{r}}_d(t) + \mathbf{K}_{Dr}\left[\dot{\boldsymbol{r}}_d(t) - \boldsymbol{\nu}(t)\right] + \mathbf{K}_{Pr}\left[\boldsymbol{r}_d(t) - \boldsymbol{r}(t)\right] .$$

Asymptotic tracking of $\boldsymbol{r}_d(t)$, $\dot{\boldsymbol{r}}_d(t)$ and $\ddot{\boldsymbol{r}}_d(t)$ is ensured with exponential convergence for any choice of positive-definite matrices $\mathbf{K}_{Dr}$ and $\mathbf{K}_{Pr}$. The vector $\boldsymbol{r}$ required for position feedback can be computed from joint position measurements via (9.40).

Compliant Environment

In the case of a compliant environment, according to the decomposition (9.44) of the end-effector displacement, the end-effector twist can be decomposed as

$$\boldsymbol{v}_e = \mathbf{S}_v\boldsymbol{\nu} + \mathbf{C}'\mathbf{S}_f\dot{\boldsymbol{\lambda}} , \tag{9.60}$$

where the first term is a twist of freedom, the second term is a twist of constraint, the vector $\boldsymbol{\nu}$ is defined as in (9.42), and $\mathbf{C}'$ is defined in (9.50). Assuming a constant contact geometry and compliance, i. e., $\dot{\mathbf{S}}_v = \mathbf{0}$, $\dot{\mathbf{C}}' = \mathbf{0}$, and $\dot{\mathbf{S}}_f = \mathbf{0}$, a similar decomposition holds in terms of acceleration

$$\dot{\boldsymbol{v}}_e = \mathbf{S}_v\dot{\boldsymbol{\nu}} + \mathbf{C}'\mathbf{S}_f\ddot{\boldsymbol{\lambda}} . \tag{9.61}$$

An inverse-dynamics control law (9.16) can be adopted, resulting in the closed loop (9.17), where $\boldsymbol{\alpha}$ is a properly designed control input.

In view of the acceleration decomposition (9.61), the choice

$$\boldsymbol{\alpha} = \mathbf{S}_v\boldsymbol{\alpha}_\nu + \mathbf{C}'\mathbf{S}_f\boldsymbol{f}_\lambda \tag{9.62}$$

allows decoupling of the force control from velocity control. In fact, substituting (9.61) and (9.62) into (9.17) and premultiplying both sides of the resulting equation once by $\mathbf{S}_v^\dagger$ and once by $\mathbf{S}_f^T$, the following decoupled equations can be derived

$$\dot{\boldsymbol{\nu}} = \boldsymbol{\alpha}_\nu , \tag{9.63}$$

$$\ddot{\boldsymbol{\lambda}} = \boldsymbol{f}_\lambda . \tag{9.64}$$

Hence, by choosing $\boldsymbol{\alpha}_\nu$ according to (9.59) as for the rigid environment case, asymptotic tracking of a desired velocity $\boldsymbol{\nu}_d(t)$ and acceleration $\dot{\boldsymbol{\nu}}_d(t)$ is ensured, with exponential convergence. The control input $\boldsymbol{f}_\lambda$ can be chosen as

$$\begin{aligned}\boldsymbol{f}_\lambda = {} & \ddot{\boldsymbol{\lambda}}_d(t) + \mathbf{K}_{D\lambda}\left[\dot{\boldsymbol{\lambda}}_d(t) - \dot{\boldsymbol{\lambda}}(t)\right] \\ & + \mathbf{K}_{P\lambda}\left[\boldsymbol{\lambda}_d(t) - \boldsymbol{\lambda}(t)\right] ,\end{aligned} \tag{9.65}$$

ensuring asymptotic tracking of a desired force trajectory $(\boldsymbol{\lambda}_d(t), \dot{\boldsymbol{\lambda}}_d(t), \ddot{\boldsymbol{\lambda}}_d(t))$ with exponential convergence for any choice of positive-definite matrices $\mathbf{K}_{D\lambda}$ and $\mathbf{K}_{P\lambda}$.

Differently from the rigid environment case, feedback of $\dot{\boldsymbol{\lambda}}$ is required for the implementation of the force control law (9.65). This quantity could be computed from end-effector wrench measurements $\boldsymbol{h}_e$ as

$$\dot{\boldsymbol{\lambda}} = \mathbf{S}_f^\dagger\dot{\boldsymbol{h}}_e .$$

However, since the wrench measurement signal is often noisy, the feedback of $\dot{\boldsymbol{\lambda}}$ is often replaced by

$$\dot{\boldsymbol{\lambda}} = \mathbf{S}_f^\dagger\mathbf{K}'\mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} , \tag{9.66}$$

where joint velocities are measured using tachometers or computed from joint positions via numerical differentiation and $\mathbf{K}'$ is the positive-semidefinite stiffness matrix (9.49) describing the partially constrained interaction. For the computation of (9.66), only knowledge (or an estimate) of $\mathbf{K}'$ is required, and not that of the full stiffness matrix $\mathbf{K}$. Also, the implementation of the control law (9.62) requires knowledge (or an estimate) of the compliance matrix $\mathbf{C}'$ of the partially constrained interaction and not that of the full compliance matrix $\mathbf{C}$.

If the contact geometry is known, but only an estimate of the stiffness/compliance of the environment is available, the control law (9.62), with (9.65), may still guarantee the convergence of the force error, if a constant desired force $\boldsymbol{\lambda}_d$ is assigned. In this case, the control law (9.62) has the form

$$\boldsymbol{\alpha} = \mathbf{S}_v\boldsymbol{\alpha}_\nu + \widehat{\mathbf{C}}'\mathbf{S}_f\boldsymbol{f}_\lambda ,$$

where $\widehat{\mathbf{C}}' = (\mathbf{I} - \mathbf{P}_v)\widehat{\mathbf{C}}$ and $\widehat{\mathbf{C}}$ is an estimate of the compliance matrix. Hence, (9.63) still holds, while, in lieu of (9.64), the following equality can be found

$$\ddot{\boldsymbol{\lambda}} = \mathbf{L}_f\boldsymbol{f}_\lambda ,$$

where $\mathbf{L}_f = (\mathbf{S}_f^T \mathbf{C} \mathbf{S}_f)^{-1} \mathbf{S}_f^T \widehat{\mathbf{C}} \mathbf{S}_f$ is a nonsingular matrix. Thus, the force- and velocity-controlled subspaces remain decoupled and the velocity control law (9.59) does not need to be modified. On the other hand, if the feedback of the time derivative of $\boldsymbol{\lambda}$ is computed using (9.66), only an estimate $\dot{\hat{\boldsymbol{\lambda}}}$ can be obtained. Using (9.66), (9.60) and (9.48), the following equality can be found

$$\dot{\hat{\boldsymbol{\lambda}}} = \mathbf{L}_f^{-1} \dot{\boldsymbol{\lambda}} \,.$$

Therefore, computing the force control law $\boldsymbol{f}_\lambda$ as in (9.65) with a constant $\boldsymbol{\lambda}_d$, $\dot{\hat{\boldsymbol{\lambda}}}$ in lieu of $\dot{\boldsymbol{\lambda}}$ and $\mathbf{K}_{D\lambda} = \mathbf{K}_{D\lambda} \mathbf{I}$, the dynamics of the closed-loop system is

$$\ddot{\boldsymbol{\lambda}} + \mathbf{K}_{D\lambda} \dot{\boldsymbol{\lambda}} + \mathbf{L}_f \mathbf{K}_{P\lambda} \boldsymbol{\lambda} = \mathbf{L}_f \mathbf{K}_{P\lambda} \boldsymbol{\lambda}_d \,,$$

showing that exponential asymptotic stability of the equilibrium $\boldsymbol{\lambda} = \boldsymbol{\lambda}_d$ can be ensured, also in the presence of the uncertain matrix $\mathbf{L}_f$, with a suitable choice of the gains $\mathbf{K}_{D\lambda}$ and $\mathbf{K}_{P\lambda}$.

9.4.2 Passivity-Based Approach

The passivity-based approach exploits the passivity properties of the dynamic model of the manipulator, which hold also for the constrained dynamic model (9.2). It can be easily shown that the choice of the matrix $\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ that guarantees the skew symmetry of the matrix $\dot{\mathbf{H}}(\boldsymbol{q}) - 2\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ in the joint space, also makes the matrix $\dot{\boldsymbol{\Lambda}}(\boldsymbol{q}) - 2\boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ skew symmetric. This fundamental property of Lagrangian systems is at the base of passivity-based control algorithms.

Rigid Environment

The control wrench $\boldsymbol{h}_c$ can be chosen as

$$\boldsymbol{h}_c = \boldsymbol{\Lambda}(\boldsymbol{q}) \mathbf{S}_v \dot{\boldsymbol{\nu}}_r + \boldsymbol{\Gamma}'(\boldsymbol{q}, \dot{\boldsymbol{q}}) \boldsymbol{\nu}_r + (\mathbf{S}_v^\dagger)^T \mathbf{K}_\nu (\boldsymbol{\nu}_r - \boldsymbol{\nu}) + \boldsymbol{\eta}(\boldsymbol{q}) + \mathbf{S}_f \boldsymbol{f}_\lambda \,, \tag{9.67}$$

where $\boldsymbol{\Gamma}'(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\Gamma}\, \mathbf{S}_v + \boldsymbol{\Lambda} \dot{\mathbf{S}}_v$, $\mathbf{K}_\nu$ is a suitable symmetric and positive-definite matrix, and $\boldsymbol{\nu}_r$ and $\boldsymbol{f}_\lambda$ are properly designed control inputs.

Substituting (9.67) into (9.2) yields

$$\boldsymbol{\Lambda}(\boldsymbol{q}) \mathbf{S}_v \dot{\boldsymbol{s}}_\nu + \boldsymbol{\Gamma}'(\boldsymbol{q}, \dot{\boldsymbol{q}}) \boldsymbol{s}_\nu + (\mathbf{S}_v^\dagger)^T \mathbf{K}_\nu \boldsymbol{s}_\nu + \mathbf{S}_f (\boldsymbol{f}_\lambda - \boldsymbol{\lambda}) = \mathbf{0} \,, \tag{9.68}$$

with $\dot{\boldsymbol{s}}_\nu = \dot{\boldsymbol{\nu}}_r - \dot{\boldsymbol{\nu}}$ and $\boldsymbol{s}_\nu = \boldsymbol{\nu}_r - \boldsymbol{\nu}$, showing that the closed-loop system remains nonlinear and coupled.

Premultiplying both sides of (9.68) by the matrix $\mathbf{S}_v$, the following expression for the reduced-order dynamics is achieved

$$\boldsymbol{\Lambda}_v(\boldsymbol{q}) \dot{\boldsymbol{s}}_\nu + \boldsymbol{\Gamma}_v(\boldsymbol{q}, \dot{\boldsymbol{q}}) \boldsymbol{s}_\nu + \mathbf{K}_\nu \boldsymbol{s}_\nu = \mathbf{0} \,, \tag{9.69}$$

with $\boldsymbol{\Gamma}_v = \mathbf{S}_v^T \boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \mathbf{S}_v + \mathbf{S}_v^T \boldsymbol{\Lambda}(\boldsymbol{q}) \dot{\mathbf{S}}_v$; it can easily be shown that the skew symmetry of the matrix $\dot{\boldsymbol{\Lambda}}(\boldsymbol{q}) - 2\boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ implies that the matrix $\dot{\boldsymbol{\Lambda}}_v(\boldsymbol{q}) - 2\boldsymbol{\Gamma}_v(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is skew symmetric as well.

On the other hand, premultiplying both sides of (9.68) by the matrix $\mathbf{S}_f^T \boldsymbol{\Lambda}^{-1}(\boldsymbol{q})$, the following expression for the force dynamics can be found

$$\boldsymbol{f}_\lambda - \boldsymbol{\lambda} = -\mathbf{S}_f^\dagger(\boldsymbol{q}) \left[\boldsymbol{\Gamma}'(\boldsymbol{q}, \dot{\boldsymbol{q}}) - (\mathbf{S}_v^\dagger)^T \mathbf{K}_\nu \right] \boldsymbol{s}_\nu \,, \tag{9.70}$$

being $\mathbf{S}_f^\dagger$ the weighted pseudo-inverse of $\mathbf{S}_f$ with weight $\boldsymbol{\Lambda}^{-1}(\boldsymbol{q})$. The above equation shows that the force multiplier $\boldsymbol{\lambda}$ instantaneously depends on the control input $\boldsymbol{f}_\lambda$ but also on the error $\boldsymbol{s}_\nu$ in the velocity-controlled subspace.

The asymptotic stability of the reduced-order system (9.69) can be ensured with the choices

$$\dot{\boldsymbol{\nu}}_r = \dot{\boldsymbol{\nu}}_d + \boldsymbol{\alpha} \Delta \boldsymbol{\nu} \,, \tag{9.71}$$

$$\boldsymbol{\nu}_r = \boldsymbol{\nu}_d + \boldsymbol{\alpha} \Delta \boldsymbol{x}_\nu \,, \tag{9.72}$$

where $\boldsymbol{\alpha}$ is a positive gain, $\dot{\boldsymbol{\nu}}_d$ and $\boldsymbol{\nu}_d$ are the desired acceleration and velocity, respectively, $\Delta \boldsymbol{\nu} = \boldsymbol{\nu}_d - \boldsymbol{\nu}$, and

$$\Delta \boldsymbol{x}_\nu = \int_0^t \Delta \boldsymbol{\nu}(\tau) \, \mathrm{d}\tau \,.$$

The stability proof is based on the positive-definite Lyapunov function

$$V = \frac{1}{2} \boldsymbol{s}_\nu^T \boldsymbol{\Lambda}_v(\boldsymbol{q}) \boldsymbol{s}_\nu + \boldsymbol{\alpha} \Delta \boldsymbol{x}_\nu^T \mathbf{K}_\nu \Delta \boldsymbol{x}_\nu \,,$$

whose time derivative along the trajectories of (9.69),

$$\dot{V} = -\Delta \boldsymbol{\nu}^T \mathbf{K}_\nu \Delta \boldsymbol{\nu} - \boldsymbol{\alpha}^2 \Delta \boldsymbol{x}_\nu^T \mathbf{K}_\nu \Delta \boldsymbol{x}_\nu \,,$$

is a definite semi-negative function. Hence, $\Delta \boldsymbol{\nu} = \mathbf{0}$, $\Delta \boldsymbol{x}_\nu = \mathbf{0}$, and $\boldsymbol{s}_\nu = \mathbf{0}$ asymptotically. Therefore, tracking of the desired velocity $\boldsymbol{\nu}_d(t)$ is ensured. Moreover, the right-hand side of (9.70) remains bounded and vanishes asymptotically. Hence, tracking of the desired force $\boldsymbol{\lambda}_d(t)$ can be ensured by setting $\boldsymbol{f}_\lambda$ as for the acceleration-resolved approach, according to the choices (9.56)–(9.58).

Notice that position control can be achieved if a desired position $\boldsymbol{r}_d(t)$ is assigned for the vector $\boldsymbol{r}$ in (9.40), provided that the matrices $\mathbf{S}_f$ and $\mathbf{S}_v$ are computed according to (9.32) and (9.42), and the vectors $\dot{\boldsymbol{\nu}}_d = \ddot{\boldsymbol{r}}_d$, $\boldsymbol{\nu}_d = \dot{\boldsymbol{r}}_d$, and $\Delta \boldsymbol{x}_\nu = \boldsymbol{r}_d - \boldsymbol{r}$ are used in (9.71) and (9.72).

Compliant Environment

The control wrench $\boldsymbol{h}_c$ can be chosen as

$$\boldsymbol{h}_c = \boldsymbol{\Lambda}(\boldsymbol{q})\dot{\boldsymbol{v}}_r + \boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{v}_r + \mathbf{K}_s(\boldsymbol{v}_r - \boldsymbol{v}_e) + \boldsymbol{h}_e + \boldsymbol{\eta}(\boldsymbol{q}) , \tag{9.73}$$

where $\mathbf{K}_s$ is a suitable symmetric positive-definite matrix while $\boldsymbol{v}_r$ and its time derivative $\dot{\boldsymbol{v}}_r$ are chosen as

$$\boldsymbol{v}_r = \boldsymbol{v}_d + \boldsymbol{\alpha}\Delta\boldsymbol{x} ,$$
$$\dot{\boldsymbol{v}}_r = \dot{\boldsymbol{v}}_d + \boldsymbol{\alpha}\Delta\boldsymbol{v} ,$$

where $\boldsymbol{\alpha}$ is the positive gain, $\boldsymbol{v}_d$ and its time derivative $\dot{\boldsymbol{v}}_d$ are properly designed control inputs, $\Delta\boldsymbol{v} = \boldsymbol{v}_d - \boldsymbol{v}_e$, and

$$\Delta\boldsymbol{x} = \int_0^t \Delta\boldsymbol{v}\,\mathrm{d}\boldsymbol{\tau} .$$

Substituting (9.73) into (9.2) yields

$$\boldsymbol{\Lambda}(\boldsymbol{q})\dot{\boldsymbol{s}} + \boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{s} + \mathbf{K}_s\boldsymbol{s} = \mathbf{0} , \tag{9.74}$$

with $\dot{\boldsymbol{s}} = \dot{\boldsymbol{v}}_r - \dot{\boldsymbol{v}}_e$ and $\boldsymbol{s} = \boldsymbol{v}_r - \boldsymbol{v}_e$.

The asymptotic stability of system (9.74) can be ensured by setting

$$\boldsymbol{v}_d = \mathbf{S}_v\boldsymbol{\nu}_d + \mathbf{C}'\mathbf{S}_f\dot{\boldsymbol{\lambda}}_d ,$$

where $\boldsymbol{\nu}_d(t)$ is a desired velocity trajectory and $\boldsymbol{\lambda}_d(t)$ is the desired force trajectory. The stability proof is based on the positive-definite Lyapunov function

$$V = \frac{1}{2}\boldsymbol{s}^{\mathrm{T}}\boldsymbol{\Lambda}(\boldsymbol{q})\boldsymbol{s} + \boldsymbol{\alpha}\Delta\boldsymbol{x}^{\mathrm{T}}\mathbf{K}_s\Delta\boldsymbol{x} ,$$

whose time derivative along the trajectories of (9.74),

$$\dot{V} = -\Delta\boldsymbol{v}^{\mathrm{T}}\mathbf{K}_s\Delta\boldsymbol{v} - \boldsymbol{\alpha}^2\Delta\boldsymbol{x}^{\mathrm{T}}\mathbf{K}_s\Delta\boldsymbol{x} ,$$

is a negative-definite function. Hence, $\Delta\boldsymbol{v} = \mathbf{0}$ and $\Delta\boldsymbol{x} = \mathbf{0}$, asymptotically. In the case of constant contact geometry and stiffness, the following equalities hold

$$\Delta\boldsymbol{v} = \mathbf{S}_v(\boldsymbol{\nu}_d - \boldsymbol{\nu}) + \mathbf{C}'\mathbf{S}_f(\dot{\boldsymbol{\lambda}}_d - \dot{\boldsymbol{\lambda}}) ,$$
$$\Delta\boldsymbol{x} = \mathbf{S}_v\int_0^t(\boldsymbol{\nu}_d - \boldsymbol{\nu})\,\mathrm{d}\boldsymbol{\tau} + \mathbf{C}'\mathbf{S}_f(\boldsymbol{\lambda}_d - \boldsymbol{\lambda}) ,$$

showing that both the velocity and force tracking errors, belonging to reciprocal subspaces, converge asymptotically to zero.

9.4.3 Velocity-Resolved Approach

The acceleration-resolved approach, as well as the passivity-based approach, require modification of the current industrial robot controllers. As for the case of impedance control, if the contact is sufficiently compliant, the closed-loop dynamics of a motion-controlled robot can be approximated by (9.28), corresponding to a velocity-resolved control.

To achieve force and velocity control, according to the end-effector twist decomposition (9.60), the control input $\boldsymbol{v}_r$ can be chosen as

$$\boldsymbol{v}_r = \mathbf{S}_v\boldsymbol{v}_\nu + \mathbf{C}'\mathbf{S}_f\boldsymbol{f}_\lambda , \tag{9.75}$$

with

$$\boldsymbol{v}_\nu = \boldsymbol{\nu}_d(t) + \mathbf{K}_{I\nu}\int_0^t[\boldsymbol{\nu}_d(\boldsymbol{\tau}) - \boldsymbol{\nu}(\boldsymbol{\tau})]\,\mathrm{d}\boldsymbol{\tau} , \tag{9.76}$$

and

$$\boldsymbol{f}_\lambda = \dot{\boldsymbol{\lambda}}_d(t) + \mathbf{K}_{P\lambda}[\boldsymbol{\lambda}_d(t) - \boldsymbol{\lambda}(t)] , \tag{9.77}$$

where $\mathbf{K}_{I\nu}$ and $\mathbf{K}_{P\lambda}$ are suitable symmetric and positive-definite matrix gains. Decoupling between velocity- and force-controlled subspaces and exponential asymptotic stability of the closed-loop system can be proven as for the acceleration-resolved approach. Also, since the force error has second-order dynamics, an integral action can be added to (9.77) to improve the disturbance rejection capabilities, i. e.,

$$\boldsymbol{f}_\lambda = \dot{\boldsymbol{\lambda}}_d(t) + \mathbf{K}_{P\lambda}[\boldsymbol{\lambda}_d(t) - \boldsymbol{\lambda}(t)] + \mathbf{K}_{I\lambda}\int_0^t[\boldsymbol{\lambda}_d(\boldsymbol{\tau}) - \boldsymbol{\lambda}(\boldsymbol{\tau})]\,\mathrm{d}\boldsymbol{\tau} , \tag{9.78}$$

and the exponential asymptotic stability is guaranteed if the matrices $\mathbf{K}_{P\lambda}$ and $\mathbf{K}_{I\lambda}$ are symmetric and positive definite.

If an estimate $\widehat{\mathbf{C}}$ of the stiffness matrix of the environment is used in (9.75), as for the acceleration-resolved approach, the exponential convergence of $\boldsymbol{\lambda}$ to a constant $\boldsymbol{\lambda}_d$ can still be ensured for both (9.77) and (9.78).

In some applications, besides the stiffness matrix, also the geometry of the environment is uncertain. In these cases, a force/motion control law similar to (9.75) can be implemented, without using the selection matrices $\mathbf{S}_v$ and $\mathbf{S}_f$ to separate the force-controlled subspace from the velocity-controlled subspace. The motion control law can be set as in (9.76),

but using full velocity feedback. Also, the force control law can be set as in (9.78), but using full force and moment feedback. That is, both motion control and force control are applied in all directions of the six-dimensional (6-D) space. The resulting control, known as *force control with feedforward motion* or *parallel force/position control* guarantees force regulation at the expense of position errors along the constrained task directions, thanks to the dominance of the force controller over the position controller ensured by the presence of the integral action on the force error. This approach is tested in the experiments reported in VIDEO 687 and VIDEO 691 where the concept of task frame is also exploited. A framework for robotic assembly, where a standard position-based robot controller is integrated with an external controller performing force-controlled skills is presented in VIDEO 692.

9.5 Conclusions and Further Reading

This chapter has summarized the main approaches to force control in a unifying perspective. However, there are many aspects that have not been considered and that must be carefully taken into account when dealing with interaction robotic tasks. The two major paradigms of force control (impedance and hybrid force/motion control) are based on several simplifying assumptions that are only partially satisfied in practical implementations. In fact, the performance of a force-controlled robotic system depends on the interaction with a changing environment which is very difficult to model and identify correctly. A general contact situation is far from being completely predictable, not only quantitatively, but also qualitatively: the contact configuration can change abruptly, or be of a different type than expected. Hence, the standard performance indices used to evaluate a control system, i. e., stability, bandwidth, accuracy, and robustness, cannot be defined by considering the robotic system alone, as for the case of robot motion control, but must be always referred to the particular contact situation at hand. Also, a classification of all these different situations is not easy, especially in the case of dynamics environments and when the task involves multiple contacts acting in parallel.

Due to the inherent complexity of the force control problem, a large number of research papers on this topic have been published in the past three decades. A description of the state of the art of the first decade is provided in [9.20], whereas the progress of the second decade is surveyed in [9.21] and [9.22]. More recently, two monographs on force control [9.23, 24] have appeared. In the following, a list of references is provided, where more details on the arguments presented in the chapter, as well as topics not covered here, can be found.

9.5.1 Indirect Force Control

The concept of generalized spring and damping for force control in joint coordinates was originally proposed in [9.3] and the implementation discussed in [9.10]. Stiffness control in Cartesian coordinates was proposed in [9.9]. Devices based on the remote center of compliance were discussed in [9.25] for successful mating of rigid parts. The original idea of a mechanical impedance model used for controlling the interaction between the manipulator and the environment is presented in [9.7], and a similar formulation is given in [9.8]. Stability of impedance control was analyzed in [9.26] and the problems of interaction with a stiff environment were considered in [9.27].

Adaptive impedance control algorithms [9.28, 29] have been proposed to overcome uncertainties in the dynamic parameters of the robot manipulator, while robust control schemes can be found in [9.30]. Impedance control has also been used in the hybrid force/motion control framework [9.31].

A reference work on modeling 6-DOF (spatial) stiffness is [9.32], while the properties of spatial compliance have been analyzed in detail in [9.33–35]; a 6-DOF variable compliant wrist was proposed in [9.36], while several studies concerning the construction of programmed compliances, optimized for specific tasks, have been proposed [9.37, 38]. The energy-based approach to derive a spatial impedance was introduced in [9.39], using rotation matrices; various 6-DOF impedance control schemes based on different representations of end-effector orientation, including the unit quaternion, can be found in [9.40]. The quaternion-based formulation is extended to the case of non-block-diagonal stiffness matrix in [9.41]. A rigorous treatment of spatial impedance control in a passivity framework can be found in [9.42].

More recently, impedance control was proposed as an effective approach to enhance safety in applications where humans and robots share the same workspace and may have physical interaction. To this end, the passive compliance of lightweight robots is combined with the active compliance ensured by impedance control [9.43, 44].

9.5.2 Task Specification

The concepts of natural and artificial constraints and of compliance frame were introduced in [9.11]. These ideas have been systematically developed in [9.12, 45] within the *task frame formalism*. Theoretical issues on the reciprocity of generalized force and velocity directions are discussed in [9.46, 47], while invariance in computing the generalized inverse in robotics is addressed in [9.48]. The issue of partially constrained tasks is considered in [9.49], where the models of positive-semidefinite stiffness and compliance matrices are developed. The problem of estimating geometric uncertainties is considered in [9.50, 51], as well as the issue of linking constraint-based task specification with real-time task execution control. This approach is generalized in [9.52], where a systematic constraint-based methodology to specify complex tasks is presented.

9.5.3 Hybrid Force/Motion Control

Early works on force control can be found in [9.10]. The original hybrid force/positon control concept was introduced in [9.13], based on the natural and artificial constraint task formulation [9.11]. The explicit inclusion of the manipulator dynamic model was presented in [9.17], and a systematic approach to modeling the interaction with a dynamic environment was developed in [9.53]. The constrained formulation with inverse dynamic controllers is treated in [9.14, 54] in the Cartesian space as well as in [9.15] joint space. The constrained approach was also used in [9.16] with a controller based on linearized equations. The invariance problems pointed out in [9.47] were correctly addressed, among other papers, in [9.46, 55]. Transposition of model-based schemes from unconstrained motion control to constrained cases was accomplished for adaptive control in [9.18, 56, 57] and for robust control in [9.58].

Approaches designed to cope with uncertainties in the environment geometry are the force control with feedforward motion scheme proposed in [9.2] and the parallel force/position control [9.19], based on the concept of dominance of force control on motion control, thanks to the use of an integral action on the force error. A parallel force/position regulator was developed in [9.59]. The integral action for removing steady-state force errors has traditionally been used; its stability was proven in [9.60], while robustness with respect to force measurement delays was investigated in [9.61, 62].

In the absence of force/torque sensors, suitable algorithms can be adopted to estimate the contact force, based, e.g., on disturbance observers [9.63], or on the error of a position control [9.64].

It has generally been recognized that force control may cause unstable behavior during contact with environment. Dynamic models for explaining this phenomenon were introduced in [9.65] and experimental investigations can be found in [9.66] and [9.67]. Moreover, control schemes are usually derived on the assumption that the manipulator end-effector is in contact with the environment and that this contact is not lost. Impact phenomena may occur and deserve careful consideration, and there is a need for global analysis of control schemes including the transition from noncontact to contact situations and vice versa, see e.g., [9.68–70].

Video-References

VIDEO 684 Recent research in impedance control
available from http://handbookofrobotics.org/view-chapter/09/videodetails/684

VIDEO 685 Integration of force strategies and natural admittance control
available from http://handbookofrobotics.org/view-chapter/09/videodetails/685

VIDEO 686 Experiments of spatial impedance control
available from http://handbookofrobotics.org/view-chapter/09/videodetails/686

VIDEO 687 Compliant robot motion; Control and task specification
available from http://handbookofrobotics.org/view-chapter/09/videodetails/687

VIDEO 691 COMRADE: Compliant motion research and development environment
available from http://handbookofrobotics.org/view-chapter/09/videodetails/691

VIDEO 692 Robotic assembly of emergency stop buttons
available from http://handbookofrobotics.org/view-chapter/09/videodetails/692

References

9.1 T.L. De Fazio, D.S. Seltzer, D.E. Whitney: The instrumented remote center of compliance, Ind. Robot **11**(4), 238–242 (1984)

9.2 J. De Schutter, H. Van Brussel: Compliant robot motion II. A control approach based on external control loops, Int. J. Robotics Res. **7**(4), 18–33 (1988)

9.3 I. Nevins, D.E. Whitney: The force vector assembler concept, Proc. 1 CISM-IFToMM Symp. Theory Pract. Robotics Manip., Udine (1973)

9.4 M.T. Mason, J.K. Salisbury: *Robot Hands and Mechanics of Manipulation* (MIT Press, Cambridge 1985)

9.5 J.Y.S. Luh, W.D. Fisher, R.P.C. Paul: Joint torque control by direct feedback for industrial robots, IEEE Trans. Autom. Control **28**, 153–161 (1983)

9.6 G. Hirzinger, N. Sporer, A. Albu-Schäffer, M. Hähnle, R. Krenn, A. Pascucci, R. Schedl: DLR's torque-controlled light weight robot III – Are we reaching the technological limits now?, Proc. IEEE Int. Conf. Robotics Autom., Washington (2002) pp. 1710–1716

9.7 N. Hogan: Impedance control: An approach to manipulation: Parts I–III, ASME J. Dyn. Syst. Meas. Control **107**, 1–24 (1985)

9.8 H. Kazerooni, T.B. Sheridan, P.K. Houpt: Robust compliant motion for manipulators. Part I: The fundamental concepts of compliant motion, IEEE J. Robotics Autom. **2**, 83–92 (1986)

9.9 J.K. Salisbury: Active stiffness control of a manipulator in Cartesian coordinates, 19th IEEE Conf. Decis. Control, Albuquerque (1980) pp. 95–100

9.10 D.E. Whitney: Force feedback control of manipulator fine motions, ASME J. Dyn. Syst. Meas. Control **99**, 91–97 (1977)

9.11 M.T. Mason: Compliance and force control for computer controlled manipulators, IEEE Trans. Syst. Man Cybern. **11**, 418–432 (1981)

9.12 J. De Schutter, H. Van Brussel: Compliant robot motion I. A formalism for specifying compliant motion tasks, Int. J. Robotics Res. **7**(4), 3–17 (1988)

9.13 M.H. Raibert, J.J. Craig: Hybrid position/force control of manipulators, ASME J. Dyn. Syst. Meas. Control **103**, 126–133 (1981)

9.14 T. Yoshikawa: Dynamic hybrid position/force control of robot manipulators – Description of hand constraints and calculation of joint driving force, IEEE J. Robotics Autom. **3**, 386–392 (1987)

9.15 N.H. McClamroch, D. Wang: Feedback stabilization and tracking of constrained robots, IEEE Trans. Autom. Control **33**, 419–426 (1988)

9.16 J.K. Mills, A.A. Goldenberg: Force and position control of manipulators during constrained motion tasks, IEEE Trans. Robotics Autom. **5**, 30–46 (1989)

9.17 O. Khatib: A unified approach for motion and force control of robot manipulators: The operational space formulation, IEEE J. Robotics Autom. **3**, 43–53 (1987)

9.18 L. Villani, C. Canudas de Wit, B. Brogliato: An exponentially stable adaptive control for force and position tracking of robot manipulators, IEEE Trans. Autom. Control **44**, 798–802 (1983)

9.19 S. Chiaverini, L. Sciavicco: The parallel approach to force/position control of robotic manipulators, IEEE Trans. Robotics Autom. **9**, 361–373 (1993)

9.20 D.E. Whitney: Historical perspective and state of the art in robot force control, Int. J. Robotics Res. **6**(1), 3–14 (1987)

9.21 M. Vukobratović, Y. Nakamura: Force and contact control in robotic systems, Proc. IEEE Int. Conf. Robotics Autom., Atlanta (1993)

9.22 J. De Schutter, H. Bruyninckx, W.H. Zhu, M.W. Spong: Force control: A bird's eye view. In: *Control Problems in Robotics and Automation*, ed. by K.P. Valavanis, B. Siciliano (Springer, London 1998) pp. 1–17

9.23 D.M. Gorinevski, A.M. Formalsky, A.Y. Schneider: *Force Control of Robotics Systems* (CRC, Boca Raton 1997)

9.24 B. Siciliano, L. Villani: *Robot Force Control* (Kluwer, Boston 1999)

9.25 D.E. Whitney: Quasi-static assembly of compliantly supported rigid parts, ASME J. Dyn. Syst. Meas. Control **104**, 65–77 (1982)

9.26 N. Hogan: On the stability of manipulators performing contact tasks, IEEE J. Robotics Autom. **4**, 677–686 (1988)

9.27 H. Kazerooni: Contact instability of the direct drive robot when constrained by a rigid environment, IEEE Trans. Autom. Control **35**, 710–714 (1990)

9.28 R. Kelly, R. Carelli, M. Amestegui, R. Ortega: Adaptive impedance control of robot manipulators, IASTED Int. J. Robotics Autom. **4**(3), 134–141 (1989)

9.29 R. Colbaugh, H. Seraji, K. Glass: Direct adaptive impedance control of robot manipulators, J. Robotics Syst. **10**, 217–248 (1993)

9.30 Z. Van Lu, A.A. Goldenberg: Robust impedance control and force regulation: Theory and experiments, Int. J. Robotics Res. **14**, 225–254 (1995)

9.31 R.J. Anderson, M.W. Spong: Hybrid impedance control of robotic manipulators, IEEE J. Robotics Autom. **4**, 549–556 (1986)

9.32 J. Lončarić: Normal forms of stiffness and compliance matrices, IEEE J. Robotics Autom. **3**, 567–572 (1987)

9.33 T. Patterson, H. Lipkin: Structure of robot compliance, ASME J. Mech. Design **115**, 576–580 (1993)

9.34 E.D. Fasse, P.C. Breedveld: Modelling of elastically coupled bodies: Part I – General theory and geometric potential function method, ASME J. Dyn. Syst. Meas. Control **120**, 496–500 (1998)

9.35 E.D. Fasse, P.C. Breedveld: Modelling of elastically coupled bodies: Part II – Exponential and generalized coordinate method, ASME J. Dyn. Syst. Meas. Control **120**, 501–506 (1998)

9.36 R.L. Hollis, S.E. Salcudean, A.P. Allan: A six-degree-of-freedom magnetically levitated variable compliance fine-motion wrist: Design, modeling and control, IEEE Trans. Robotics Autom. **7**, 320–333 (1991)

9.37 M.A. Peshkin: Programmed compliance for error corrective assembly, IEEE Trans. Robotics Autom. **6**, 473–482 (1990)

9.38 J.M. Shimmels, M.A. Peshkin: Admittance matrix design for force-guided assembly, IEEE Trans. Robotics Autom. **8**, 213–227 (1992)

9.39 E.D. Fasse, J.F. Broenink: A spatial impedance controller for robotic manipulation, IEEE Trans. Robotics Autom. **13**, 546–556 (1997)

9.40 F. Caccavale, C. Natale, B. Siciliano, L. Villani: Six-DOF impedance control based on angle/axis representations, IEEE Trans. Robotics Autom. **15**, 289–300 (1999)

9.41 F. Caccavale, C. Natale, B. Siciliano, L. Villani: Robot impedance control with nondiagonal stiffness, IEEE Trans. Autom. Control **44**, 1943–1946 (1999)

9.42 S. Stramigioli: *Modeling and IPC Control of Interactive Mechanical Systems – A Coordinate Free Approach* (Springer, London 2001)

9.43 C. Ott: *Cartesian Impedance Control of Redundant and Flexible-Joint Robots*, Springer Tracts in Advanced Robotics (STAR) (Springer, Berlin, Heidelberg 2008)

9.44 C. Ott, A. Albu-Schäffer, A. Kugi, G. Hirzinger: On the passivity based impedance control of flexible joint robots, IEEE Trans. Robotics **24**, 416–429 (2008)

9.45 H. Bruyninckx, J. De Schutter: Specification of Force-controlled actions in the *task frame formalism* – A synthesis, IEEE Trans. Robotics Autom. **12**, 581–589 (1996)

9.46 H. Lipkin, J. Duffy: Hybrid twist and wrench control for a robotic manipulator, ASME J. Mech. Design **110**, 138–144 (1988)

9.47 J. Duffy: The fallacy of modern hybrid control theory that is based on *orthogonal complements* of twist and wrench spaces, J. Robotics Syst. **7**, 139–144 (1990)

9.48 K.L. Doty, C. Melchiorri, C. Bonivento: A theory of generalized inverses applied to robotics, Int. J. Robotics Res. **12**, 1–19 (1993)

9.49 T. Patterson, H. Lipkin: Duality of constrained elastic manipulation, Proc. IEEE Conf. Robotics Autom., Sacramento (1991) pp. 2820–2825

9.50 J. De Schutter, H. Bruyninckx, S. Dutré, J. De Geeter, J. Katupitiya, S. Demey, T. Lefebvre: Estimation first-order geometric parameters and monitoring contact transitions during force-controlled compliant motions, Int. J. Robotics Res. **18**(12), 1161–1184 (1999)

9.51 T. Lefebvre, H. Bruyninckx, J. De Schutter: Polyedral contact formation identification for auntonomous compliant motion, IEEE Trans. Robotics Autom. **19**, 26–41 (2007)

9.52 J. De Schutter, T. De Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aerbeliën, K. Claes, H. Bruyninckx: Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty, Int. J. Robotics Res. **26**(5), 433–455 (2007)

9.53 A. De Luca, C. Manes: Modeling robots in contact with a dynamic environment, IEEE Trans. Robotics Autom. **10**, 542–548 (1994)

9.54 T. Yoshikawa, T. Sugie, N. Tanaka: Dynamic hybrid position/force control of robot manipulators – Controller design and experiment, IEEE J. Robotics Autom. **4**, 699–705 (1988)

9.55 J. De Schutter, D. Torfs, H. Bruyninckx, S. Dutré: Invariant hybrid force/position control of a velocity controlled robot with compliant end effector using modal decoupling, Int. J. Robotics Res. **16**(3), 340–356 (1997)

9.56 R. Lozano, B. Brogliato: Adaptive hybrid force-position control for redundant manipulators, IEEE Trans. Autom. Control **37**, 1501–1505 (1992)

9.57 L.L. Whitcomb, S. Arimoto, T. Naniwa, F. Ozaki: Adaptive model-based hybrid control if geometrically constrained robots, IEEE Trans. Robotics Autom. **13**, 105–116 (1997)

9.58 B. Yao, S.P. Chan, D. Wang: Unified formulation of variable structure control schemes for robot manipulators, IEEE Trans. Autom. Control **39**, 371–376 (1992)

9.59 S. Chiaverini, B. Siciliano, L. Villani: Force/position regulation of compliant robot manipulators, IEEE Trans. Autom. Control **39**, 647–652 (1994)

9.60 J.T.-Y. Wen, S. Murphy: Stability analysis of position and force control for robot arms, IEEE Trans. Autom. Control **36**, 365–371 (1991)

9.61 R. Volpe, P. Khosla: A theoretical and experimental investigation of explicit force control strategies for manipulators, IEEE Trans. Autom. Control **38**, 1634–1650 (1993)

9.62 L.S. Wilfinger, J.T. Wen, S.H. Murphy: Integral force control with robustness enhancement, IEEE Control Syst. Mag. **14**(1), 31–40 (1994)

9.63 S. Katsura, Y. Matsumoto, K. Ohnishi: Modeling of force sensing and validation of disturbance observer for force control, IEEE Trans. Ind. Electron. **54**, 530–538 (2007)

9.64 A. Stolt, M. Linderoth, A. Robertsson, R. Johansson: Force controlled robotic assembly without a force sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1538–1543

9.65 S.D. Eppinger, W.P. Seering: Introduction to dynamic models for robot force control, IEEE Control Syst. Mag. **7**(2), 48–52 (1987)

9.66 C.H. An, J.M. Hollerbach: The role of dynamic models in Cartesian force control of manipulators, Int. J. Robotics Res. **8**(4), 51–72 (1989)

9.67 R. Volpe, P. Khosla: A theoretical and experimental investigation of impact control for manipulators, Int. J. Robotics Res. **12**, 351–365 (1993)

9.68 J.K. Mills, D.M. Lokhorst: Control of robotic manipulators during general task execution: A discontinuous control approach, Int. J. Robotics Res. **12**, 146–163 (1993)

9.69 T.-J. Tarn, Y. Wu, N. Xi, A. Isidori: Force regulation and contact transition control, IEEE Control Syst. Mag. **16**(1), 32–40 (1996)

9.70 B. Brogliato, S. Niculescu, P. Orhant: On the control of finite dimensional mechanical systems with unilateral constraints, IEEE Trans. Autom. Control **42**, 200–215 (1997)

Multimedia Contents

10. Redundant Robots

Stefano Chiaverini, Giuseppe Oriolo, Anthony A. Maciejewski

This chapter focuses on redundancy resolution schemes, i. e., the techniques for exploiting the redundant degrees of freedom in the solution of the inverse kinematics problem. This is obviously an issue of major relevance for motion planning and control purposes.

In particular, task-oriented kinematics and the basic methods for its inversion at the velocity (first-order differential) level are first recalled, with a discussion of the main techniques for handling kinematic singularities. Next, different first-order methods to solve kinematic redundancy are arranged in two main categories, namely those based on the optimization of suitable performance criteria and those relying on the augmentation of the task space. Redundancy resolution methods at the acceleration (second-order differential) level are then considered in order to take into account dynamics issues, e.g., torque minimization. Conditions under which a cyclic task motion results in a cyclic joint motion are also discussed; this is a major issue when a redundant manipulator is used to execute a repetitive task, e.g., in industrial applications. The use of kinematic redundancy for fault tolerance is analyzed in detail. Suggestions for further reading are given in a final section.

10.1 **Overview** 221
10.2 **Task-Oriented Kinematics** 224
10.2.1 Task-Space Formulation 224
10.2.2 Singularities 225
10.3 **Inverse Differential Kinematics** 227
10.3.1 The General Solution 227
10.3.2 Singularity Robustness 227
10.3.3 Joint Trajectory Reconstruction ... 231
10.4 **Redundancy Resolution via Optimization** 232
10.4.1 Performance Criteria 232
10.4.2 Local Optimization 232
10.4.3 Global Optimization 233
10.5 **Redundancy Resolution via Task Augmentation** 233
10.5.1 Extended Jacobian 233
10.5.2 Augmented Jacobian 234
10.5.3 Algorithmic Singularities 234
10.5.4 Task Priority 235
10.6 **Second-Order Redundancy Resolution** .. 236
10.7 **Cyclicity** 237
10.8 **Fault Tolerance** 237
10.9 **Conclusion and Further Reading** 239
Video-References 239
References 240

10.1 Overview

Kinematic redundancy occurs when a robotic manipulator has more degrees of freedom than those strictly required to execute a given task. This means that, in principle, no manipulator is inherently redundant; rather, there are certain tasks with respect to which it may become redundant. Because it is widely recognized that a fully general spatial task consists of following an end-effector motion trajectory requiring six degrees of freedom, a robot arm with seven or more joints is frequently considered as a typical example of an inherently redundant manipulator. However, even robot arms with fewer degrees of freedom, like conventional six-joint industrial manipulators, may become kinematically redundant for specific tasks, such as simple end-effector positioning without constraints on the orientation.

The motivation for introducing kinematic redundancy in the mechanical structure of a manipulator goes beyond that for using redundancy in traditional engineering design, namely, increasing robustness to faults so as to improve reliability (e.g., redundant processors

or sensors). In fact, endowing robotic manipulators with kinematic redundancy is mainly aimed at increasing dexterity, as is the case with a human arm.

The minimal-complexity approach that characterized early manipulator designs had the objective of minimizing cost and maintenance. For example, this led to the development of selective compliance assembly robot arm (SCARA) robots for pick-and-place operations where products had been designed for assembly, i. e., using a single axis of insertion. However, giving a manipulator the minimum number of joints required to execute its task results in a serious limitation in real-world applications where, in addition to the singularity problem, joint limits or workspace obstacles are present. All of these give rise to forbidden regions in the joint space that must be avoided during operation, thus requiring a carefully structured (and static) workspace where the motion of the manipulator can be planned in advance; this is the typical situation for workcells in traditional industrial applications.

On the other hand, the presence of additional degrees of freedom besides those strictly required to execute the task allows motions of the manipulator that do not displace the end effector (the so-called self-motions or internal motions); this implies that the same task at the end-effector level can be executed in several ways at the joint level, giving the possibility of avoiding the forbidden regions and ultimately resulting in a more versatile mechanism. Such a feature is key to allowing operation in unstructured or dynamically vary-

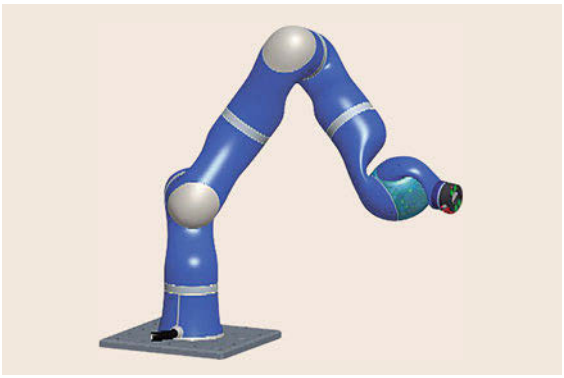


Fig. 10.1 The 7-DOF DLR Lightweight Robot

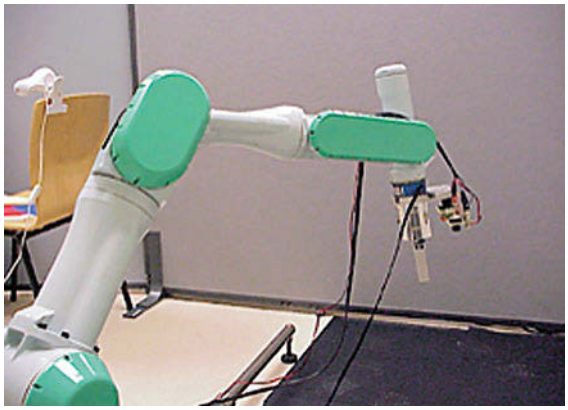


Fig. 10.2 The 7-DOF Mitsubishi PA-10 manipulator

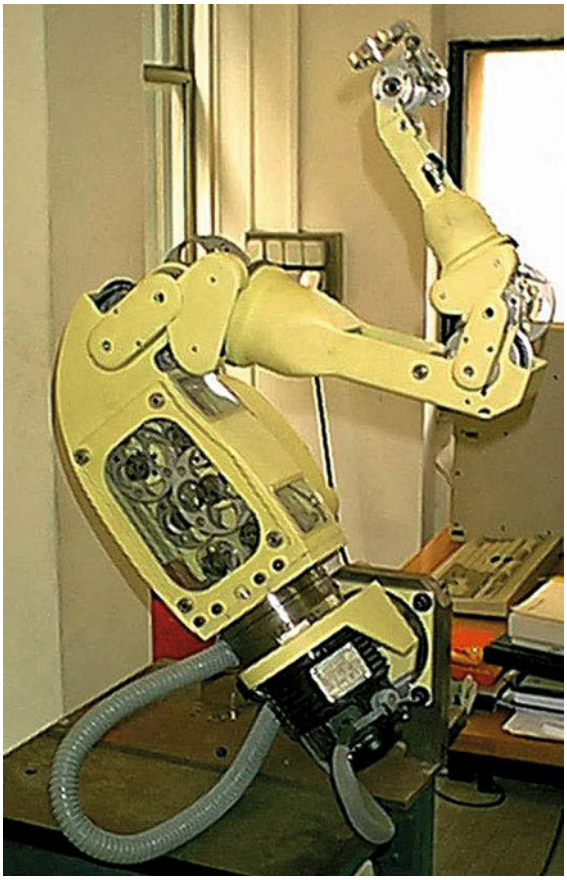


Fig. 10.3 The 8-DOF DEXTER by Scienzia Machinale

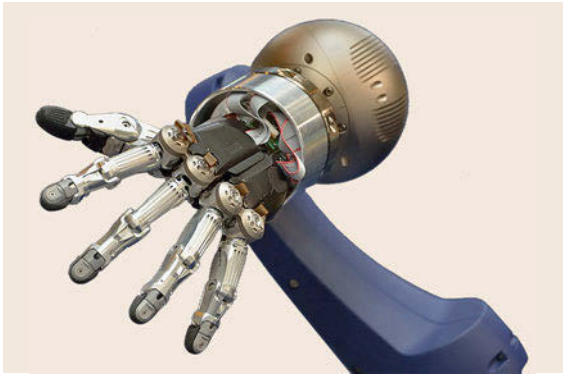


Fig. 10.4 A 5-finger hand by Schunk

ing environments that characterize advanced industrial applications and service robotics scenarios.

In practice, if properly managed, the increased dexterity characterizing kinematically redundant robots may allow them to not only avoid singularities, joint limits, and workspace obstacles, but also to minimize torque/energy over a given task, ultimately meaning that the robotic manipulator can achieve a higher degree of autonomy.

The biological archetype of a kinematically redundant manipulator is the human arm, which, not surprisingly, also inspires the terminology used to characterize the structure of serial-chain manipulators. In fact, the human arm has three degrees of freedom at the *shoulder*, one degree of freedom at the *elbow* and three degrees of freedom at the *wrist*. The available redundancy can be easily verified by locking one's wrist, e.g., on a table and moving the elbow while keeping the shoulder still. The kinematic arrangement of the human arm has been replicated in a number of robots often termed as *human-arm-like* manipulators. This family of 7-DOF (degree of freedom) manipulators includes the DLR (Deutsches Zentrum für Luft- und Raumfahrt) Lightweight Robot (Fig. 10.1) and the Mitsubishi PA-10 robot (Fig. 10.2). An example of an 8-DOF robot is the DEXTER by Scienzia Machinale (Fig. 10.3). Manipulators with a larger number of joints are often called *hyperredundant* robots, and include many snake-like robots described in the literature.

The use of two or more robotic structures to execute a task (as in the case of multifingered hands, humanoid mockups or cooperating manipulators, Figs. 10.4–10.7, respectively) also gives rise to kinematic redundancy. Redundant mechanisms also include vehicle-manipulator systems (Fig. 10.8); in this case, however, the possible presence of nonholonomic constraints on the

Fig. 10.5 The Baxter working robot by Rethink Robotics

Fig. 10.6 The NASA (National Aeronautics and Space Administration) Robonaut

Fig. 10.7 A cooperating dual-arm concept robot by Asea Brown Boveri (ABB)

Fig. 10.8 A huge vehicle-manipulator system by KUKA Aktiengesellschaft

motion of the base must be properly taken into account in order to determine the actual degree of redundancy.

Although the realization of a kinematically redundant structure raises a number of issues from the point of view of mechanical design, this chapter focuses on redundancy resolution schemes, i. e., techniques for exploiting the redundant degrees of freedom in the solution of the inverse kinematics problem. This is an issue of major relevance for motion planning and control purposes.

10.2 Task-Oriented Kinematics

The relationship between the variables representing the configuration of an articulated manipulator and those describing an assigned task in the appropriate space can be established at the position, velocity or acceleration level. In particular, consideration of the first-order task kinematics inevitably results in discussion of the task Jacobian matrix, which is central to many redundancy resolution techniques.

10.2.1 Task-Space Formulation

A manipulator consists of a chain of rigid bodies articulated by joints. If q_i denotes the variable characterizing the relative displacement of body i with respect to body $i-1$, the vector $\boldsymbol{q} = (q_1 \dots q_N)^{\mathrm{T}}$ uniquely describes the configuration of an N-joint serial-chain manipulator. Joint i may be either prismatic or revolute, in which case q_i measures the relative translation or rotation of the attached links, respectively.

While the manipulator is naturally described and actuated in the *joint space*, its operation is conveniently specified in terms of the vector $\boldsymbol{t} = (t_1 \dots t_M)^{\mathrm{T}}$, which typically defines the location of the manipulator end-effector in a suitably defined *task space*. In the general case, it is of dimension $M = 6$ and $\boldsymbol{t}$ is chosen so that its first three components represent the position of the end effector, while its last three components represent a minimal description of the end-effector orientation (such as Euler angles or the roll–pitch–yaw representation), i. e.,

$$\boldsymbol{t} = \begin{bmatrix} p_x & p_y & p_z & \alpha & \beta & \gamma \end{bmatrix}^{\mathrm{T}} .$$

Typically, one has $N \geq M$, so that the joints can provide at least the number of degrees of freedom required for the end-effector task. If $N > M$ strictly, the manipulator is *kinematically redundant*.

The relationship between the joint-space coordinate vector $\boldsymbol{q}$ and the task-space coordinate vector $\boldsymbol{t}$ is expressed by the *direct kinematics* equation

$$\boldsymbol{t} = \boldsymbol{k}_{\mathrm{t}}(\boldsymbol{q}) \,, \tag{10.1}$$

where $\boldsymbol{k}_{\mathrm{t}}$ is a nonlinear vector function.

Task Jacobian and Geometric Jacobian

It is useful to consider the *first-order differential kinematics* [10.1]

$$\dot{\boldsymbol{t}} = \mathbf{J}_{\mathrm{t}}(\boldsymbol{q})\, \dot{\boldsymbol{q}} \,, \tag{10.2}$$

that can be obtained by differentiating (10.1) with respect to time. In (10.2), $\dot{\boldsymbol{t}}$ is the task-space velocity vector, $\dot{\boldsymbol{q}}$ is the joint-space velocity vector, and $\mathbf{J}_{\mathrm{t}}(\boldsymbol{q}) = \partial \boldsymbol{k}_{\mathrm{t}} / \partial \boldsymbol{q}$ is the $M \times N$ *task Jacobian* matrix (also called the *analytic* Jacobian).

Remarkably, the components of $\dot{\boldsymbol{t}}$ relative to the end-effector orientation express the rate of change of the parameters characterizing the adopted minimal representation; they are not the components of the angular velocity vector of the end-effector. Indeed, denoting by $\mathbf{v}_N$ the 3×1 translational velocity vector and by $\boldsymbol{\omega}_N$ the 3×1 angular velocity vector of the end-effector, and defining the end-effector spatial velocity $\boldsymbol{v}_N$ as

$$\boldsymbol{v}_N = \begin{pmatrix} \mathbf{v}_N \\ \boldsymbol{\omega}_N \end{pmatrix} \,, \tag{10.3}$$

the following relationship holds

$$\dot{\boldsymbol{t}} = \mathbf{T}(\boldsymbol{t})\, \boldsymbol{v}_N \,, \tag{10.4}$$

where $\mathbf{T}$ is an $M \times 6$ transformation matrix that is a function of $\boldsymbol{t}$ only. In the case of $M = 6$, the transformation matrix $\mathbf{T}$ takes the form

$$\mathbf{T} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{pmatrix} \,, \tag{10.5}$$

where $\mathbf{I}$ and $\mathbf{0}$ are, respectively, the identity and null matrix of proper dimensions, and $\mathbf{R}$ is a 3×3 matrix that specifically depends on the minimal representation used to describe the end-effector orientation.

For a given manipulator, the mapping

$$\boldsymbol{v}_N = \mathbf{J}(\boldsymbol{q})\, \dot{\boldsymbol{q}} \tag{10.6}$$

relates a joint-space velocity to the corresponding end-effector velocity through the $6 \times N$ *geometric Jacobian*

matrix $\mathbf{J}$. The geometric Jacobian matrix is of major concern in the kinematic analysis of a manipulator because it allows description of the motion capabilities of the end-effector (in terms of its free rigid-body spatial velocity) as a result of the velocity commands at the joints in the current configuration. As a matter of fact, if $\boldsymbol{\mathcal{J}}_{Pi}\dot{q}_i$ denotes the contribution of $\dot{q}_i$ to $\mathbf{v}_N$, z_{i-1} is the unit vector of the axis of joint i and $\boldsymbol{r}_{i-1,e}$ is the vector expressing the position of the end effector frame with respect to the joint i frame, it is

$$\boldsymbol{\mathcal{J}}_{Pi} = \begin{cases} z_{i-1} & \text{if joint } i \text{ is prismatic} \\ z_{i-1} \times \boldsymbol{r}_{i-1,e} & \text{if joint } i \text{ is revolute} \end{cases},$$

whereas if $\boldsymbol{\mathcal{J}}_{Oi}\dot{q}_i$ denotes the contribution of $\dot{q}_i$ to $\boldsymbol{\omega}_N$, it is

$$\boldsymbol{\mathcal{J}}_{Oi} = \begin{cases} \mathbf{0} & \text{if joint } i \text{ is prismatic} \\ z_{i-1} & \text{if joint } i \text{ is revolute} \end{cases}.$$

By comparing (10.2), (10.4), and (10.6), the relation between the geometric Jacobian and the task Jacobian can be found as

$$\mathbf{J}_{\mathrm{t}}(\boldsymbol{q}) = \mathbf{T}(\boldsymbol{t})\,\mathbf{J}(\boldsymbol{q})\,. \tag{10.7}$$

Second-Order Differential Kinematics

While the first-order differential kinematics (10.2) relates task-space to joint-space velocities, further differentiation with respect to time provides an analogous relationship between accelerations

$$\ddot{\boldsymbol{t}} = \mathbf{J}_{\mathrm{t}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\mathbf{J}}_{\mathrm{t}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}\,. \tag{10.8}$$

This equation is also known as the *second-order differential kinematics*.

10.2.2 Singularities

In this section the occurrence of singular configurations is considered to properly analyze their effect on inverse kinematics solutions.

Representation and Kinematic Singularities

A robot configuration $\boldsymbol{q}$ is *singular* if the task Jacobian matrix $\mathbf{J}_{\mathrm{t}}$ is rank-deficient there. Considering the role of $\mathbf{J}_{\mathrm{t}}$ in (10.2) and (10.8), it is easy to realize that at a singular configuration it is impossible to generate end-effector task velocities or accelerations in certain directions. Further insight can be gained by looking at (10.7), which indicates that a singularity may be due to a loss of rank of the transformation matrix $\mathbf{T}$ and/or of the geometric Jacobian matrix $\mathbf{J}$.

Rank deficiencies of $\mathbf{T}$ are only related to the mathematical relationship established by $\mathbf{R}$ between the angular velocity vector of the end-effector and the components of $\dot{\boldsymbol{t}}$ relative to the end-effector orientation; because the expression of $\mathbf{R}$ depends on the adopted minimal representation of the orientation, a configuration at which $\mathbf{T}$ is singular is then referred to as a *representation singularity*. Remarkably, any minimal description of the end-effector orientation experiences the occurrence of representation singularities. Furthermore, a given configuration may or may not yield a representation singularity depending on which description of orientation is used.

A representation singularity is not directly related to the true motion capabilities of the manipulator structure, which can instead be inferred by the analysis of the geometric Jacobian matrix. Rank deficiencies of $\mathbf{J}$ are in fact related to loss of mobility of the manipulator end-effector; indeed, end-effector velocities exist in this case that are unfeasible for any velocity commanded at the joints. A configuration at which $\mathbf{J}$ is singular is referred to as a *kinematic singularity*.

Because this chapter focuses on the inversion of the task differential kinematics (10.2) and (10.8), in the following the task Jacobian matrix and its singularities (which include representation and kinematic singularities) are studied in detail. The case $N \geq M$ is considered, which encompasses conventional as well as redundant manipulators.

Singular Value Decomposition of the Jacobian

To analyze the linear mapping (10.2), the *singular value decomposition* (SVD) of the Jacobian matrix is adopted; remarkably, this powerful numerical tool is the sole reliable method to compute the rank of a matrix and to study near-singular linear mappings. The classic *Golub–Reinsch* algorithm [10.2], which is the most efficient and numerically stable algorithm to compute the SVD of an arbitrary matrix, may however be computationally demanding in view of real-time applications. A faster algorithm that takes advantage of the nature of robotic matrix calculations has been proposed [10.3]; this makes it possible to improve real-time kinematic control techniques.

The SVD of the task Jacobian matrix can be written in the form

$$\mathbf{J}_{\mathrm{t}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}} = \sum_{i=1}^{M} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathrm{T}}\,, \tag{10.9}$$

where $\mathbf{U}$ is the $M \times M$ orthonormal matrix of the output singular vectors $\boldsymbol{u}_i$, $\mathbf{V}$ is the $N \times N$ orthonormal matrix of the input singular vectors $\boldsymbol{v}_i$, and $\boldsymbol{\Sigma} = (\mathbf{S}\ \mathbf{0})$ is the $M \times N$ matrix whose $M \times M$ diagonal submatrix $\mathbf{S}$

contains the singular values σ_i of the matrix $\mathbf{J}_\mathrm{t}$. It is worth noting that the SVD is a well-behaved function of its matrix argument; therefore, the singular values and associated subspaces do not change radically in the neighborhood of the current configuration. Letting rank $(\mathbf{J}_\mathrm{t}) = R$, the following hold

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_R > \sigma_{R+1} = \cdots = 0 \,,$$
$$\mathcal{R}(\mathbf{J}_\mathrm{t}) = \mathrm{span}\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_R\} \,,$$
$$\mathcal{N}(\mathbf{J}_\mathrm{t}) = \mathrm{span}\{\boldsymbol{v}_{R+1}, \ldots, \boldsymbol{v}_N\} \,.$$

If the task Jacobian is full-rank ($R = M$), all the singular values are nonzero, the range space of $\mathbf{J}_\mathrm{t}$ is the entire $\mathbb{R}^M$, and the null space of $\mathbf{J}_\mathrm{t}$ has dimension $N-M$. In a singular configuration, $R < M$; thus, the last $M-R$ singular values are zero, the range space of $\mathbf{J}_\mathrm{t}$ is an R-dimensional subspace of $\mathbb{R}^M$, and the dimension of the null space of $\mathbf{J}_\mathrm{t}$ increases to $N-R$. An interpretation of this from a kinematic viewpoint is presented in the following.

Feasible Velocities. At each configuration, the range space of $\mathbf{J}_\mathrm{t}$ is the set of task-space velocities that can be obtained as a result of all possible joint-space velocities $\dot{\boldsymbol{q}}$; therefore, it constitutes the so-called subspace of *feasible velocities* for the end-effector task. A basis for $\mathcal{R}(\mathbf{J}_\mathrm{t})$ is given by the first R output singular vectors; accordingly, the effect of a singularity is to decrease the dimension of the range space of $\mathbf{J}_\mathrm{t}$ that corresponds to a reduction of the space of feasible velocities.

Null-Space Velocities. At each configuration, the null space of $\mathbf{J}_\mathrm{t}$ is the set of joint-space velocities that yield zero task velocity; these are thus shortly called *null-space velocities*. A basis for $\mathcal{N}(\mathbf{J}_\mathrm{t})$ is given by the $N-R$ last input singular vectors, which represent independent linear combinations of the velocities at each joint. From this viewpoint, the effect of a singularity is to increase the dimension of the null space of $\mathbf{J}_\mathrm{t}$ by introducing a further independent linear combination of joint velocities that produces a zero task velocity.

According to (10.2) and (10.9), a joint velocity along the i-th input singular vector results in a task velocity that lies along the i-th output singular vector

$$\forall \rho \in \mathbb{R} \quad \dot{\boldsymbol{q}} = \rho\, \boldsymbol{v}_i \quad \Rightarrow \quad \boldsymbol{t} = \sigma_i \rho\, \boldsymbol{u}_i \,.$$

Thus, the i-th singular value of $\mathbf{J}_\mathrm{t}$ can be viewed as a gain factor relating motion along the $\boldsymbol{v}_i$ direction of the joint velocity space to the resulting motion along the $\boldsymbol{u}_i$ direction of the task velocity space. When a singularity is approached, the R-th singular value σ_R tends to zero and the task velocity produced by a fixed joint velocity along $\boldsymbol{v}_R$ is decreased proportionally. At a singular configuration, the joint-space velocity along $\boldsymbol{v}_R$ belongs to the null-space velocities and task velocities along $\boldsymbol{u}_R$ become unfeasible for the manipulator.

In the generic case, the joint-space velocity $\dot{\boldsymbol{q}}$ is an arbitrary linear combination of individual joint velocities with nonzero components along all the $\boldsymbol{v}_i$. Its effect can be analyzed by combining the single effects of the above components. Remarkably, the components of $\dot{\boldsymbol{q}}$ in the null space of $\mathbf{J}_\mathrm{t}$ produce a change in the configuration of the manipulator without affecting its task velocity. This can be exploited to achieve additional goals – such as obstacle or singularity avoidance – in addition to the realization of a desired task motion, and constitutes the core of redundancy resolution approaches.

Distance from Singularities

Clearly, the effect of a singularity is experienced not only at the singular configuration itself but also in its neighborhood (Sect. 10.3.2). For this reason, it is important to be able to characterize the distance from singularities through suitable measures; these can then be exploited to counteract undesirable effects.

Because each singularity is associated with a reduction in the rank of $\mathbf{J}_\mathrm{t}$, one conceptually simple possibility in the case of a square Jacobian matrix ($M = N$) is to compute its determinant. A generalization of this idea that works also for nonsquare Jacobians is the *manipulability measure* [10.4], defined as

$$\mu = \sqrt{|\mathbf{J}_\mathrm{t}\, \mathbf{J}_\mathrm{t}^\mathrm{T}|} \,.$$

It can be recognized that the manipulability measure is equal to the product of the singular values of $\mathbf{J}_\mathrm{t}$, i. e.,

$$\mu = \prod_{i=1}^{M} \sigma_i \,,$$

and thus its zeros coincide with the singularities.

Another possible measure of distance from a singular configuration is the *condition number* of the Jacobian matrix [10.5], defined as

$$\kappa = \frac{\sigma_1}{\sigma_M} \,.$$

The condition number has values ranging from 1, at configurations where all the singular values are equal, to ∞, at singular configurations. Note that when $\kappa = 1$ the end effector has the same motion capability in all task space directions because all the singular values are equal – i. e., the arm is at an *isotropic configuration* – whereas at a singularity it loses mobility in some task-space direction.

An even more direct measure of the distance from singular configurations is the *smallest singular value* of the Jacobian matrix [10.5], i. e.,

$$\sigma_{\min} = \sigma_M \, .$$

A computationally light estimate of the smallest singular value can be obtained either via numerical methods [10.3, 6, 7] or based on a kinematic analysis of the robot structure [10.8].

It must be noted that the manipulability measure may remain constant even in the presence of variations of either the condition number or the smallest singular value of $\mathbf{J}_t$. On the other hand, because the smallest singular value changes more radically near singularities than the other singular values, it dominates the behavior of the determinant and the condition number of the Jacobian matrix; therefore, the most effective measure of distance from singular configurations is the smallest singular value of $\mathbf{J}_t$ [10.5].

10.3 Inverse Differential Kinematics

In order to accomplish a task, a proper joint motion must be commanded to the manipulator; therefore, it is necessary to derive mathematical relations that allow one to compute joint-space variables corresponding to the assigned task-space variables. This is the objective of the inverse kinematics problem.

The inverse kinematics problem can be solved by inverting the direct kinematics equation (10.1), the first-order differential kinematics (10.2) or the second-order differential kinematics (10.8). If the task is time-varying (i. e., if a desired trajectory $\boldsymbol{t}(t)$ is assigned), it is convenient to solve the differential kinematic relationships because these represent linear equations using the task Jacobian matrix [10.9].

10.3.1 The General Solution

Under the assumption that the manipulator is kinematically redundant (i. e., $M < N$), one solution of (10.2) or (10.8) can be expressed by resorting to the pseudoinverse $\mathbf{J}_t^\dagger$ of the task Jacobian matrix [10.1, 10]; this is the unique matrix satisfying the Moore–Penrose conditions [10.11–13]

$$\begin{aligned}
&\mathbf{J}_t \, \mathbf{J}_t^\dagger \mathbf{J}_t = \mathbf{J}_t \, , \\
&\mathbf{J}_t^\dagger \mathbf{J}_t \, \mathbf{J}_t^\dagger = \mathbf{J}_t^\dagger \, , \\
&(\mathbf{J}_t \, \mathbf{J}_t^\dagger)^{\mathrm{T}} = \mathbf{J}_t \, \mathbf{J}_t^\dagger \, , \\
&(\mathbf{J}_t^\dagger \mathbf{J}_t)^{\mathrm{T}} = \mathbf{J}_t^\dagger \mathbf{J}_t \, .
\end{aligned} \tag{10.10}$$

If $\mathbf{J}_t$ is low-rectangular and full-rank, its pseudoinverse can be expressed as

$$\mathbf{J}_t^\dagger = \mathbf{J}_t^{\mathrm{T}} (\mathbf{J}_t \, \mathbf{J}_t^{\mathrm{T}})^{-1} \, . \tag{10.11}$$

If $\mathbf{J}_t$ is square, expression (10.11) reduces to the standard inverse matrix.

The general solution of (10.2) can be written as

$$\dot{\boldsymbol{q}} = \mathbf{J}_t^\dagger \dot{\boldsymbol{t}} + (\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t) \, \dot{\boldsymbol{q}}_0 \, . \tag{10.12}$$

Here, $\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t$ represents the orthogonal projection matrix into the null space of $\mathbf{J}_t$, and $\dot{\boldsymbol{q}}_0$ is an arbitrary joint-space velocity; the second part of the solution is therefore a null-space velocity. Equation (10.12) provides all least-squares solution to the end-effector task constraint (10.2), i. e., it minimizes $\|\dot{\boldsymbol{t}} - \mathbf{J}\dot{\boldsymbol{q}}\|$. In particular, if $\mathbf{J}_t$ is low-rectangular and full-rank, all joint velocities in the form (10.12) exactly realize the assigned task velocity. By acting on $\dot{\boldsymbol{q}}_0$, one can still obtain different joint velocities that give the same end-effector task velocity; therefore, as will be discussed later in detail, solutions in the form of (10.12) are typically used to represent redundancy resolution techniques.

The particular solution obtained by setting $\dot{\boldsymbol{q}}_0 = 0$ in (10.12),

$$\dot{\boldsymbol{q}} = \mathbf{J}_t^\dagger \, \dot{\boldsymbol{t}} \, , \tag{10.13}$$

provides the least-squares solution of (10.2) with minimum norm, and is known as the *pseudoinverse solution*. In terms of the inverse differential kinematics problem, the least-squares property quantifies the accuracy of the end-effector task realization, while the minimum norm property may be relevant for the feasibility of the joint-space velocities.

As for the second-order kinematics (10.8), its least-squares solutions can be expressed in the general form

$$\ddot{\boldsymbol{q}} = \mathbf{J}_t^\dagger (\ddot{\boldsymbol{t}} - \dot{\mathbf{J}}_t \dot{\boldsymbol{q}}) + (\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t) \, \ddot{\boldsymbol{q}}_0 \, , \tag{10.14}$$

where $\ddot{\boldsymbol{q}}_0$ is an arbitrary joint-space acceleration. As above, choosing $\ddot{\boldsymbol{q}}_0 = 0$ in (10.14) gives the minimum-norm acceleration solution

$$\ddot{\boldsymbol{q}} = \mathbf{J}_t^\dagger (\ddot{\boldsymbol{t}} - \dot{\mathbf{J}}_t \, \dot{\boldsymbol{q}}) \, . \tag{10.15}$$

10.3.2 Singularity Robustness

We now investigate the kinematics aspects involved in the first-order inverse mappings (10.12) and (10.13)

with respect to the handling of singularities. With reference to the SVD of $\mathbf{J}_t$ in (10.9), let us consider the following decomposition of the matrix $\mathbf{J}_t^\dagger$

$$\mathbf{J}_t^\dagger = \mathbf{V}\boldsymbol{\Sigma}^\dagger\mathbf{U}^T = \sum_{i=1}^{R} \frac{1}{\sigma_i}\boldsymbol{v}_i\boldsymbol{u}_i^T \,, \tag{10.16}$$

where, as above, R denotes the rank of the task Jacobian matrix. Analogously to (10.9), the following hold

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_R > \sigma_{R+1} = \cdots = 0 \,,$$
$$\mathcal{R}(\mathbf{J}_t^\dagger) = \mathcal{N}^\perp(\mathbf{J}_t) = \text{span}\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_R\} \,,$$
$$\mathcal{N}(\mathbf{J}_t^\dagger) = \mathcal{R}^\perp(\mathbf{J}_t) = \text{span}\{\boldsymbol{u}_{R+1}, \ldots, \boldsymbol{u}_M\} \,.$$

Notice that, if the Jacobian matrix is full-rank, the range space of $\mathbf{J}_t^\dagger$ is an M-dimensional subspace of $\mathbb{R}^N$ and the null space of $\mathbf{J}_t^\dagger$ is trivial. In a singular configuration ($R < M$), the range space of $\mathbf{J}_t^\dagger$ is an R-dimensional subspace of $\mathbb{R}^N$, and an $M-R$-dimensional null space of $\mathbf{J}_t^\dagger$ exists.

The range space of $\mathbf{J}_t^\dagger$ is the set of joint-space velocities $\dot{\boldsymbol{q}}$ that can be obtained via the inverse kinematic mapping (10.13) as a result of all possible task velocities $\dot{\boldsymbol{t}}$. Because these $\dot{\boldsymbol{q}}$ belong to the orthogonal complement of the null space of $\mathbf{J}_t$, the pseudoinverse solution (10.13) satisfies the least-squares condition, as expected.

The null space of $\mathbf{J}_t^\dagger$ is the set of the task velocities $\dot{\boldsymbol{t}}$ that yield null joint-space velocity in the current configuration; these $\dot{\boldsymbol{t}}$ belong to the orthogonal complement of the space of feasible task velocities. Therefore, one effect of the pseudoinverse solution (10.13) is to filter out the unfeasible components of the commanded task velocity while allowing exact tracking of the feasible components; this is related to the minimum-norm property.

If the assigned task velocity is aligned with $\boldsymbol{u}_i$, the corresponding joint-space velocity – computed via (10.13) – is obtained along $\boldsymbol{v}_i$ modulated by the factor $1/\sigma_i$. When a singularity is approached, the R-th singular value tends to zero and a fixed task velocity command along $\boldsymbol{u}_R$ requires joint-space velocities along $\boldsymbol{v}_R$ that grow unboundedly in proportion to the factor $1/\sigma_R$. At the singular configuration, the $\boldsymbol{u}_R$ direction becomes unfeasible for the task variables and $\boldsymbol{v}_R$ adds to the null-space velocities of the arm.

According to the above, two main problems are inherently related to the basic inverse differential kinematics solution (10.13), namely:

- At near-singular configurations, excessive joint-space velocities may result, due to the component of $\dot{\boldsymbol{t}}$ along the direction which becomes unfeasible at the singularity.
- At the singular configuration, discontinuity of the joint-space solution is experienced if $\dot{\boldsymbol{t}}$ has a non-null unfeasible component.

The same is obviously true for the complete inverse solution (10.12).

Both the above problems are of major concern for kinematic control of manipulators, where the computed joint-space velocities must actually be executed by the robot arm. This has motivated the development of modified inverse differential mappings, so as to ensure proper behavior of the manipulator throughout its workspace independent of the configuration of the robot. A reasonable approach is to preserve the mapping (10.13) far from singularities and to set up local modifications of it only inside a region enclosing the singular configuration; the definition of the region depends on a suitable measure of distance from the singularity, while the modified mapping must ensure feasible and continuous joint velocities.

Modification of the Planned Trajectory

One way to tackle the problem of singularities is to act at the planning level by either designing trajectories that do not incur in unavoidable singularities or assigning task-space motion commands that are feasible for the robot arm. However, these solutions rely on perfect task planning and cannot be used in real-time sensory control applications, in which motion commands are generated online.

Because singularities are a function of configuration and not workspace location, in general, it is not possible to avoid singularities by restricting the workspace trajectory of the manipulator. There do exist cases where avoidance of singular configurations at the motion planning level is relatively simple, e.g., when dealing with those associated with workspace boundary such as the elbow singularity of an anthropomorphic arm. However, this approach may be difficult for those singularities, like the wrist one, that may occur everywhere in the workspace.

Another possibility is to perform a joint-space interpolation when the planned trajectory is close to a singularity [10.14]; however, this may cause large errors in tracking the originally assigned task-space motion.

Acting in the task space, a method based on time-scale transformation is presented in [10.15], which slows down the manipulator's motion when a singularity is approached; this technique, however, fails at the singularity.

Because a task-space robot control system must be able to guide the motion of the manipulator safely through singularities, considerable research effort has been devoted to the derivation of well-defined and

continuous inverse kinematic mappings instead of the above ad hoc approaches.

Removal of Dependent Rows or Columns of the Jacobian Matrix

The first requirement of solution (10.13) is the availability of a general algorithm to compute the pseudoinverse of $\mathbf{J}_t$ also when the Jacobian is singular. Several techniques presented in the literature can be arranged in this framework; they consist either in removing the unfeasible end-effector reference components [10.16] or in using nonsingular blocks of the Jacobian matrix [10.10]. The main problem with this type of solution is the specification of the directions of unfeasible velocities in a systematic way and the need to have smooth transitions between the usual inverse kinematic algorithm and the algorithms used close to the singularities.

A systematic procedure to compute the pseudoinverse of the Jacobian matrix can be devised by taking advantage of a kinematic analysis of the manipulator structure because for many manipulators it is possible to identify and describe classes of singular configurations in suitably defined link-fixed frames. This approach has been demonstrated for a six-degree-of-freedom elbow geometry in [10.17, 18].

As for the continuity of the solution across a singularity, it must be observed that, while the singular vectors change very little in the neighborhood of the singularity, the term

$$\frac{1}{\sigma_M}\, \boldsymbol{v}_M \boldsymbol{u}_M^{\mathrm{T}}\, \dot{\boldsymbol{t}}$$

suddenly disappears from (10.16) when R becomes less than M at the singular configuration.

One possibility to avoid this problem is to make

$$\boldsymbol{u}_M^{\mathrm{T}}\, \dot{\boldsymbol{t}} \approx 0$$

in the neighborhood of the singularity, which means avoiding commanding task velocities along the direction that becomes unfeasible at the singular configuration. This would require some method for efficiently computing the singular subspace associated with near-zero singular value(s) as described in the upcoming section on numerical filtering.

Independently from the assigned $\dot{\boldsymbol{t}}$, continuity of the pseudoinverse solution can be ensured by treating the manipulator as singular in a suitably defined region around each singularity through a modified Jacobian $\bar{\mathbf{J}}_t$, so that $M-R$ extra degrees of freedom are available [10.17–19]; these can be used without significantly affecting the end-effector velocity, because the modified Jacobian $\bar{\mathbf{J}}_t$ approximates $\mathbf{J}_t$ inside the region.

The described approach is difficult to generalize for multiple singularities. For the typical anthropomorphic geometry of industrial manipulators, however, only the wrist singularity is of primary concern because it may occur everywhere in the workspace, while the elbow and shoulder singularities are naturally characterized in the task space and thus can be avoided during planning.

Regularization/Damped Least-Squares Technique

The use of the damped least-squares technique in the inverse differential kinematics problem has been independently proposed in [10.9, 20]. The method corresponds to solving the optimization problem

$$\min_{\dot{\boldsymbol{q}}} \left(\|\dot{\boldsymbol{t}} - \mathbf{J}_t \dot{\boldsymbol{q}}\|^2 + \lambda^2 \|\dot{\boldsymbol{q}}\|^2 \right) , \tag{10.17}$$

which results in a solution $\dot{\boldsymbol{q}}$ that minimizes the end-effector tracking error from the set of all joint velocities that do not exceed $\|\dot{\boldsymbol{q}}\|$. Here, $\lambda \in \mathbb{R}$ is the damping factor that represents the trade-off between the least-squares and the mimimum-norm properties. When λ is zero, the solution to (10.17) and (10.13) become identical.

The solution to (10.17) can be written in two equivalent forms

$$\dot{\boldsymbol{q}} = \mathbf{J}_t^{\mathrm{T}} (\mathbf{J}_t \mathbf{J}_t^{\mathrm{T}} + \lambda^2 \mathbf{I})^{-1} \dot{\boldsymbol{t}} , \tag{10.18}$$

$$\dot{\boldsymbol{q}} = (\mathbf{J}_t^{\mathrm{T}} \mathbf{J}_t + \lambda^2 \mathbf{I})^{-1} \mathbf{J}_t^{\mathrm{T}} \dot{\boldsymbol{t}} . \tag{10.19}$$

The computational load of (10.18) is lower than that of (10.19), because usually $N \geq M$. In the remainder, we will refer to the damped least-squares solution as

$$\dot{\boldsymbol{q}} = \mathbf{J}_t^{*}(\boldsymbol{q})\, \dot{\boldsymbol{t}} , \tag{10.20}$$

whenever explicit specification of the computation used is not essential.

Condition (10.17) implies consideration of accuracy and feasibility at the same time when choosing the joint-space velocity required to match the given $\dot{\boldsymbol{t}}$. In this regard it is essential to suitably select the value to be assigned to the damping factor: small values of λ give accurate solutions but low robustness to the occurrence of singular and near-singular configurations; high values of λ result in low tracking accuracy even when a feasible and accurate solution would be possible.

In the framework of singular value decomposition, the solution (10.20) can be written as

$$\dot{\boldsymbol{q}} = \sum_{i=1}^{R} \frac{\sigma_i}{\sigma_i^2 + \lambda^2}\, \boldsymbol{v}_i \boldsymbol{u}_i^{\mathrm{T}}\, \dot{\boldsymbol{t}} . \tag{10.21}$$

Remarkably, we have

$$\begin{aligned}\mathcal{R}(\mathbf{J}_t^*) &= \mathcal{R}(\mathbf{J}_t^\dagger) = \mathcal{N}^\perp(\mathbf{J}_t)\\ &= \text{span}\{\boldsymbol{v}_1,\ldots,\boldsymbol{v}_R\}\,,\\ \mathcal{N}(\mathbf{J}_t^*) &= \mathcal{N}(\mathbf{J}_t^\dagger) = \mathcal{R}^\perp(\mathbf{J}_t)\\ &= \text{span}\{\boldsymbol{u}_{R+1},\ldots,\boldsymbol{u}_M\}\,.\end{aligned}$$

As with $\mathbf{J}_t^\dagger$, if the Jacobian matrix is full-rank, the range space of $\mathbf{J}_t^*$ is an M-dimensional subspace of $\mathbb{R}^N$ and the null space of $\mathbf{J}_t^*$ is trivial; in a singular configuration, the range space of $\mathbf{J}_t^*$ is an R-dimensional subspace of $\mathbb{R}^N$, and an $M-R$-dimensional null space of $\mathbf{J}_t^*$ exists.

It is clear that, with respect to the pure least-squares solution (10.13), in (10.21) the components for which $\sigma_i \gg \lambda$ are little influenced by the damping factor, because in this case

$$\frac{\sigma_i}{\sigma_i^2+\lambda^2} \approx \frac{1}{\sigma_i}\,.$$

On the other hand, when a singularity is approached, the smallest singular value tends to zero while the associated component of the solution is driven to zero by the factor σ_i/λ^2; this progressively reduces the joint velocity required to achieve near-degenerate components of the commanded $\dot{\boldsymbol{t}}$. At the singularity, the solutions (10.20) and (10.13) behave identically as long as the remaining singular values are significantly larger than the damping factor. One can show that the maximum gain factor of $1/(2\lambda)$ relating the task velocity component along $\boldsymbol{u}_i$ to the resulting joint velocity along $\boldsymbol{v}_i$ occurs when $\sigma_i = \lambda$.

Selection of the Damping Factor. According to above, λ determines the degree of approximation introduced with respect to the pure least-squares solution given by the pseudoinverse. An optimal choice for λ requires consideration of the smallest non-null singular value experienced along the given trajectory and of the minimum damping needed to ensure feasible joint velocities.

To achieve good performance in the entire manipulator's workspace the use of a configuration-varying damping factor has been proposed [10.9]. The natural choice is to adjust λ as a function of some measure of distance from the singularity at the current configuration of the robot arm. Far from singular configurations feasible joint velocities are obtained; thus the accuracy requirement prevails and low damping must be used. Close to a singularity, task velocity commands along the unfeasible directions give large joint velocities and the accuracy requirement must be relaxed; in these cases, high damping is needed.

A first proposal was to adjust the damping factor as a function of the manipulability measure [10.9]. Because a more effective measure of distance from singular configurations is the smallest singular value of the Jacobian matrix [10.5], its use has been considered later for building a variable damping factor.

If an estimate $\tilde{\sigma}_M$ of the smallest singular value is available, the following choice for the damping factor can be adopted [10.21]

$$\lambda^2 = \begin{cases} 0 & \text{when } \tilde{\sigma}_M \geq \varepsilon \\ \left[1-\left(\frac{\tilde{\sigma}_M}{\varepsilon}\right)^2\right]\lambda_{\max}^2 & \text{otherwise,}\end{cases} \qquad (10.22)$$

which ensures continuity and good shaping of the solution. In (10.22), ε defines the size of the singular region, in which damping is used, and $\lambda_{\max}$ sets the maximum value of the damping factor, which occurs just at the singularity.

Numerical Filtering. As can be recognized from (10.21), the damping factor affects the accuracy of the solution along each end-effector velocity component; however, the sole unfeasible direction is responsible for the (possible) loss of tracking ability. To overcome this problem, selective filtering of the end-effector velocity components was proposed in [10.6]. The method can be generalized as follows; if estimates $\tilde{\boldsymbol{u}}_i$ of the output singular vectors associated with the $M-K$ smallest singular values – which give the components to become unfeasible – are available, the solution can be written in the form

$$\dot{\boldsymbol{q}} = \mathbf{J}_t^{\mathrm{T}}\left(\mathbf{J}_t\,\mathbf{J}_t^{\mathrm{T}} + \lambda^2\mathbf{I} + \beta^2\sum_{i=K+1}^{M}\tilde{\boldsymbol{u}}_i\,\tilde{\boldsymbol{u}}_i^{\mathrm{T}}\right)^{-1}\dot{\boldsymbol{t}}\,, \qquad (10.23)$$

where β provides the largest contribution to damping only along the unfeasible components. This solution is an approximation to applying a greater selective filtering to the near singular components, i. e.,

$$\begin{aligned}\dot{\boldsymbol{q}} \approx &\sum_{i=1}^{K}\frac{\sigma_i}{\sigma_i^2+\lambda^2}\boldsymbol{v}_i\,\boldsymbol{u}_i^{\mathrm{T}}\dot{\boldsymbol{t}}\\ &+\sum_{i=K+1}^{R}\frac{\sigma_i}{\sigma_i^2+\lambda^2+\beta^2}\boldsymbol{v}_i\,\boldsymbol{u}_i^{\mathrm{T}}\dot{\boldsymbol{t}}\,,\end{aligned} \qquad (10.24)$$

where the quality of the approximation is based on the accuracy of the estimates of $\tilde{\boldsymbol{u}}$ in (10.23). Notice that $K \leq R$; nevertheless, a nonzero value of λ is kept to guarantee satisfactory conditioning of the mapping (10.23) even for incorrect estimates of the output singular vectors.

Similarly, additional damping can be provided to the joint-space velocity components given by the in-

put singular vectors associated to the smallest singular values, since they are close to becoming the null-space velocities [10.22]. The solution can be generalized in the form

$$\dot{\boldsymbol{q}} = \left(\mathbf{J}_{\mathrm{t}}^{\mathrm{T}} \mathbf{J}_{\mathrm{t}} + \lambda^2 \mathbf{I} + \beta^2 \sum_{i=K+1}^{N} \tilde{\boldsymbol{v}}_i \tilde{\boldsymbol{v}}_i^{\mathrm{T}} \right)^{-1} \mathbf{J}_{\mathrm{t}}^{\mathrm{T}} \dot{\boldsymbol{t}} \,, \quad (10.25)$$

where β provides the largest contribution to damping only along the $N-K$ estimated null-space velocity components $\tilde{\boldsymbol{v}}_i$. Once again, this solution is an approximation to applying a greater selective filtering to the near singular components, i. e.,

$$\dot{\boldsymbol{q}} \approx \sum_{i=1}^{K} \frac{\sigma_i}{\sigma_i^2 + \lambda^2} \boldsymbol{v}_i \boldsymbol{u}_i^{\mathrm{T}} \dot{\boldsymbol{t}} + \sum_{i=K+1}^{R} \frac{\sigma_i}{\sigma_i^2 + \lambda^2 + \beta^2} \boldsymbol{v}_i \boldsymbol{u}_i^{\mathrm{T}} \dot{\boldsymbol{t}} \,, \quad (10.26)$$

where the quality of the approximation is based on the accuracy of the estimates $\tilde{\boldsymbol{v}}_i$ in (10.25). Also in this case a nonzero value of λ is kept to guarantee satisfactory conditioning of the mapping (10.25) even for incorrect estimates of the input singular vectors.

Comparison of (10.26) with (10.24) shows that, in the case of accurate estimate of the singular vectors, the solutions (10.23) and (10.25) behave identically.

10.3.3 Joint Trajectory Reconstruction

When solving the inverse kinematics problem at the first-order differential level, one obtains the joint velocity profile $\dot{\boldsymbol{q}}(t)$ that corresponds to an assigned end-effector task velocity profile $\dot{\boldsymbol{t}}(t)$; however, the robot motion controller needs reference position trajectories in addition to the velocity references for the joints. The reconstruction of the joint-position profile from the joint-velocity profile obtained from an end-effector task velocity profile $\dot{\boldsymbol{t}}(t)$ realizes a kinematic inversion that can be seen as an *inverse kinematics algorithm*.

If the joint-velocity profile were completely specified (e.g., by its analytical expression) the corresponding joint-position profile could be obtained from time integration

$$\boldsymbol{q}(t) = \boldsymbol{q}(t_0) + \int_{t_0}^{t} \dot{\boldsymbol{q}}(\boldsymbol{\tau}) \, \mathrm{d}\boldsymbol{\tau} \,. \quad (10.27)$$

However, digital implementation of the robot control system makes it more likely that a discrete-time sequence of samples $\dot{\boldsymbol{q}}_k$ of the computed joint velocities at the time instants t_k will be available, i. e.,

$$\dot{\boldsymbol{q}}_k = \dot{\boldsymbol{q}}(t_k) \,.$$

For this reason, a discrete-time numerical approximation of the continuous-time integral (10.27) must be suitably devised.

Accurate discrete-time approximations of a continuous-time integral usually require a trade-off between the complexity of the interpolating algorithm and the length of the time step. In real-time applications, such as robot motion control, high-order interpolation results in large finite-time delays that degrade the dynamic performance of the control loop. This time delay can be reduced by suitably shortening the time step; in fact, if the time step is sufficiently short even a low-order interpolation may give acceptable accuracy of the numerical integration. It is typical to use first-order interpolation, i. e., an Euler forward rectangular rule that transforms the integral (10.27) into

$$\boldsymbol{q}_k = \boldsymbol{q}_0 + \sum_{h=0}^{k-1} \dot{\boldsymbol{q}}_h \, \Delta t \,, \quad (10.28)$$

where Δt is the time step. Equation (10.28) is usually written in the more effective recursive form

$$\boldsymbol{q}_k = \boldsymbol{q}_{k-1} + \dot{\boldsymbol{q}}_{k-1} \, \Delta t \,.$$

Closed-Loop Inverse Kinematics (CLIK)

No matter what kind of interpolation is used, the small though unavoidable error suffered at each numerical integration step accumulates, leading to long-term drifting of the reconstructed profile from the exact joint-position profile. Another source of error affecting any integral reconstruction method is the possible uncertainty in the initial value of the joint position.

Algorithmic solutions that overcome these problems are based on the use of a feedback correction term; these are termed *closed-loop inverse kinematics* (CLIK) algorithms [10.23]. Considering, e.g., the case of first-order kinematics, the joint velocity at the k-th time instant is computed as (compare with (10.12))

$$\dot{\boldsymbol{q}}_k = \mathbf{J}_{\mathrm{t}}^{\dagger}(\boldsymbol{q}_k) \left\{ \dot{\boldsymbol{t}}_{\mathrm{k}} + \mathbf{K} \left[\boldsymbol{t}_{\mathrm{k}} - \boldsymbol{k}_{\mathrm{t}}(\boldsymbol{q}_k) \right] \right\} + \left[\mathbf{I} - \mathbf{J}_{\mathrm{t}}^{\dagger}(\boldsymbol{q}_k) \mathbf{J}_{\mathrm{t}}(\boldsymbol{q}_k) \right] \dot{\boldsymbol{q}}_{0k} \,, \quad (10.29)$$

where $\mathbf{K}$ is a constant positive-definite gain matrix.

Second-order CLIK algorithms are also available that solve for joint positions, velocities, and accelerations [10.24, 25]. The CLIK approach was originally proposed in [10.26] and [10.27] based on the Jacobian transpose in lieu of the pseudoinverse, which provides remarkable computational savings and provides inherent singularity handling capabilities [10.28].

10.4 Redundancy Resolution via Optimization

For a kinematically redundant manipulator, the inverse kinematics problem admits an infinite number of solutions, so that a criterion to select one of them is needed. In this section we consider the problem of redundancy resolution via optimization at the first-order differential kinematics level. Before discussing algorithmic schemes for computing joint velocities, we provide a short review of possible performance criteria.

10.4.1 Performance Criteria

The availability of degrees of freedom in excess of those strictly needed to execute a given task can be used to improve the value of performance criteria during the motion. These criteria may depend on the robot joint configuration only, or also involve velocities and/or accelerations.

Among the additional objectives that can be pursued by defining a suitable criterion, arguably the most popular is singularity avoidance. In fact, one of the original motivations for introducing kinematic redundancy was to increase the size of the workspace the manipulator could reach without being in a singular configuration (*unavoidable* singularities) for achieving general motion trajectory tracking tasks. A discussion of avoidable and unavoidable singularities in redundant manipulators can be found in [10.29]. If the assigned end-effector task does not pass through unavoidable singularities, it is in principle always possible to compute a joint trajectory along which the task Jacobian $\mathbf{J}_\mathrm{t}$ is continuously full-rank. To this end, possible performance criteria are the configuration-dependent functions introduced in Sect. 10.2.2 that characterize the distance from singularities, i. e., the manipulability measure, the condition number, and the smallest singular value of $\mathbf{J}_\mathrm{t}$. Maximizing (or keeping as large as possible) these functions during the motion is a reasonable plan in order to avoid singular configurations during the motion.

Because kinematic inversion can produce arbitrarily large joint velocities in the vicinity of singular configurations, a conceptually different possibility is to minimize the norm of the joint velocity generated by the redundancy resolution scheme. However, this approach guarantees singularity avoidance only if such a norm is minimized in an integral sense along the motion of the manipulator. Local minimization of the norm [10.10] does not lead to singularity avoidance in any practical sense [10.29].

Redundancy can be also used to keep the manipulator linkage away from undesired regions of the joint space, for example, mechanical joint limits that are typically present in robot manipulators may be avoided by minimizing the cost function [10.30]

$$\mathbf{H}(\boldsymbol{q}) = \frac{1}{2}\sum_{i=1}^{N}\left(\frac{q_i - q_{i,\mathrm{mid}}}{q_{i,\max} - q_{i,\min}}\right)^2 ,$$

where $[q_{i,\min}, q_{i,\max}]$ is the available range for joint i and $q_{i,\mathrm{mid}}$ is its midpoint. Another interesting application is obstacle avoidance, which can be enforced by minimizing suitable *artificial potential* functions defined on the basis of the image of the obstacle region in the configuration space [10.31, 32].

Many other performance criteria have been proposed in the literature; some of them are mentioned in Sects. 10.5 and 10.8.

10.4.2 Local Optimization

The simplest form of local optimization is represented by the pseudoinverse solution (10.13), which provides the joint velocity with the minimum norm among those that realize the task constraint. Clearly, the joint movement generated by this locally optimal solution does not provide global velocity minimization along the entire manipulator motion. This means that, despite the local minimization of joint velocities, singularity avoidance is not guaranteed [10.29].

Another possibility is to use the general solution (10.12), choosing the arbitrary joint velocity $\dot{\boldsymbol{q}}_0$ in the direction of the antigradient of a scalar configuration-dependent performance criteria $\mathbf{H}(\boldsymbol{q})$ that one would like to minimize

$$\dot{\boldsymbol{q}}_0 = -k_\mathbf{H}\,\nabla\mathbf{H}(\boldsymbol{q}) , \tag{10.30}$$

where $k_\mathbf{H}$ is a scalar step size and $\nabla\mathbf{H}(\boldsymbol{q})$ denotes the gradient of $\mathbf{H}$ at the current joint configuration. This leads to the following redundancy resolution scheme [10.30]

$$\dot{\boldsymbol{q}} = \mathbf{J}_\mathrm{t}^\dagger\dot{\boldsymbol{t}} - k_\mathbf{H}(\mathbf{I} - \mathbf{J}_\mathrm{t}^\dagger\mathbf{J}_\mathrm{t})\,\nabla\mathbf{H}(\boldsymbol{q}) . \tag{10.31}$$

Because its second term is the projection of the antigradient of $\mathbf{H}$ in the null space of the task Jacobian, the above expression is reminiscent of the projected gradient method for constrained minimization [10.33]. In particular, it can be shown [10.34] that the inverse kinematic solution (10.31) minimizes the complete quadratic function

$$L(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{1}{2}\dot{\boldsymbol{q}}^\mathrm{T}\dot{\boldsymbol{q}} + k_\mathbf{H}\dot{\boldsymbol{q}}^\mathrm{T}\nabla\mathbf{H}(\boldsymbol{q})$$

at the current configuration $\boldsymbol{q}$. Thus, (10.31) represents a natural trade-off between the unconstrained local

minimization of the performance criteria $\mathbf{H}$ (which would lead to choose $\dot{\boldsymbol{q}} = -k_{\mathbf{H}} \nabla \mathbf{H}(\boldsymbol{q})$) and the satisfaction of constraint (10.2) by a minimum-norm joint velocity.

The choice of the step size $k_{\mathbf{H}}$ is critical for the performance of the redundancy resolution scheme (10.31). In particular, a small value of the step size may slow down the minimization of the performance criteria, but on the other hand a large value may even lead to an increase of $\mathbf{H}$ (recall that the antigradient is the *local* direction of maximum decrease). In practice, one may use a simplified *line search* technique such as Armijo's rule [10.33] to identify at each configuration an appropriate value of $k_{\mathbf{H}}$ in a reasonable time.

10.4.3 Global Optimization

The main advantage of the redundancy resolution scheme (10.31) is its simplicity: if the computation of $\nabla \mathbf{H}(\boldsymbol{q})$ and $k_{\mathbf{H}}$ is efficient, such a scheme is a realistic option for real-time kinematic inversion. Its disadvantage is to be found in the local nature of the optimization process, which can lead to unsatisfactory performance over long tasks, for example, use of (10.31) with $\mathbf{H} = -\mu$ (the manipulability measure) will perform better than the simple pseudoinverse solution, but still cannot guarantee singularity avoidance.

It is therefore natural to consider the possibility of selecting $\dot{\boldsymbol{q}}_0$ in (10.12) so as to minimize integral criteria of the form

$$\int_{t_\mathrm{i}}^{t_\mathrm{f}} \mathbf{H}(\boldsymbol{q}) \, \mathrm{d}t \, .$$

defined over the duration $[t_\mathrm{i}, t_\mathrm{f}]$ of the entire task (e.g., the integral of manipulability along the motion). Unfortunately, the solution of this problem (naturally formulated within the framework of calculus of variations) may not exist, and, in general, admits no closed form. One way to make the problem solvable is to include a quadratic form in the joint velocities or accelerations inside the integral. However, this is more easily done at the second-order kinematic level (Sect. 10.5).

10.5 Redundancy Resolution via Task Augmentation

Another approach to redundancy resolution consists of augmenting the task vector so as to tackle additional objectives expressed as constraints. In this section, we review the basic task augmentation techniques for solving the first-order differential kinematics (10.2).

10.5.1 Extended Jacobian

The extended Jacobian technique, proposed by *Baillieul* [10.35] and later revisited by *Chang* [10.36], enforces an appropriate number of functional constraints to be fulfilled along with the original end-effector task so as to identify a single solution among the infinite ones that are compatible with the end-effector task.

Let us consider an objective function $g(\boldsymbol{q})$ to be optimized and let also $\mathbf{N}_{\mathbf{J}_\mathrm{t}}(\boldsymbol{q})$ be a matrix spanning the null space of $\mathbf{J}_\mathrm{t}$ at a nonsingular configuration $\boldsymbol{q}$, e.g.,

$$\mathbf{N}_{\mathbf{J}_\mathrm{t}} = \mathbf{I} - \mathbf{J}_\mathrm{t}^\dagger \mathbf{J}_\mathrm{t} \, .$$

It can be recognized that, for a given $\boldsymbol{t}_0$, if $\boldsymbol{q}_0$ is a configuration at which the function $g(\boldsymbol{q})$ is at an extreme under the constraint $\boldsymbol{t}_0 = \boldsymbol{k}_\mathrm{t}(\boldsymbol{q}_0)$, one has

$$\left. \frac{\partial g(\boldsymbol{q})}{\partial \boldsymbol{q}} \right|_{\boldsymbol{q}=\boldsymbol{q}_0} \mathbf{N}_{\mathbf{J}_\mathrm{t}}(\boldsymbol{q}_0) = \mathbf{0}^\mathrm{T} \, . \tag{10.32}$$

If the Jacobian $\mathbf{J}_\mathrm{t}$ has full rank M, then $\mathbf{N}_{\mathbf{J}_\mathrm{t}}$ has rank $N-M$; therefore, (10.32) yields a set of $N-M$ independent constraints that can be written in vector form as

$$\boldsymbol{h}(\boldsymbol{q}) = \mathbf{0} \, ;$$

for example, these can be obtained by taking the scalar product of the gradient $\partial g(\boldsymbol{q})/\partial \boldsymbol{q}$ by each of the $N-M$ vectors constituting a base of the null space of $\mathbf{J}_\mathrm{t}$, i. e.,

$$\boldsymbol{h}(\boldsymbol{q}) = \left(\frac{\partial g(\boldsymbol{q})}{\partial \boldsymbol{q}} \; (\boldsymbol{v}_{M+1}(\boldsymbol{q}) \ldots \boldsymbol{v}_N(\boldsymbol{q})) \right)^\mathrm{T} \, .$$

At this point, the condition (10.32) implies that the equation

$$\begin{pmatrix} \boldsymbol{k}_\mathrm{t}(\boldsymbol{q}_0) \\ \boldsymbol{h}(\boldsymbol{q}_0) \end{pmatrix} = \begin{pmatrix} \boldsymbol{t}_0 \\ \mathbf{0} \end{pmatrix}$$

is satisfied.

For motion starting from $\boldsymbol{t}_0$ with $\boldsymbol{q}_0$ that tracks a trajectory $\boldsymbol{t}(t)$ by keeping $g(\boldsymbol{q})$ extremized at each time one has

$$\begin{pmatrix} \boldsymbol{k}_\mathrm{t}(\boldsymbol{q}(t)) \\ \boldsymbol{h}(\boldsymbol{q}(t)) \end{pmatrix} = \begin{pmatrix} \boldsymbol{t}(t) \\ \mathbf{0} \end{pmatrix} \, .$$

By differentiating both sides with respect to time one obtains

$$\begin{pmatrix} \mathbf{J}_t(\boldsymbol{q}) \\ \dfrac{\partial \boldsymbol{h}(\boldsymbol{q})}{\partial \boldsymbol{q}} \end{pmatrix} \dot{\boldsymbol{q}} = \begin{pmatrix} \dot{\boldsymbol{t}} \\ \mathbf{0} \end{pmatrix} , \tag{10.33}$$

where the matrix premultiplying the vector $\dot{\boldsymbol{q}}$ is square and is called the *extended Jacobian* $\mathbf{J}_{\text{ext}}$.

Therefore, if the initial configuration $\boldsymbol{q}_0$ extremizes $g(\boldsymbol{q})$ and provided that $\mathbf{J}_{\text{ext}}$ does not become singular, the time integral of the inverse mapping

$$\dot{\boldsymbol{q}} = \mathbf{J}_{\text{ext}}^{-1}(\boldsymbol{q}) \begin{pmatrix} \dot{\boldsymbol{t}} \\ 0 \end{pmatrix} \tag{10.34}$$

tracks the assigned end-effector trajectory $\dot{\boldsymbol{t}}(t)$ propagating joint configurations that extremize $g(\boldsymbol{q})$.

The extended Jacobian method has a major advantage over pseudoinverse techniques of the form (10.13) in that it is *cyclic* (Sect. 10.6). Moreover, solution (10.34) can be made equivalent to (10.12) via suitable choice of the vector $\dot{\boldsymbol{q}}_0$ [10.35, 37]. One relative disadvantage is its performance near algorithmic singularities (Sect. 10.5.3).

10.5.2 Augmented Jacobian

Another approach, the so-called task-space augmentation, introduces a constraint task to be fulfilled along with the end-effector task; then, an augmented Jacobian matrix is set-up whose inverse gives the sought joint velocity solution. The concept of task-space augmentation has been independently introduced by *Sciavicco* and *Siciliano* [10.28, 38, 39] and *Egeland* [10.40] and later revisited by *Seraji* in the framework of the configuration control method [10.41].

In detail, let us consider the vector $\boldsymbol{t}_c = (t_{c,1} \dots t_{c,P})^T$ that describes the tasks to be fulfilled in addition to the M-dimensional end-effector task $\boldsymbol{t}$. In the general case $P \leq N - M$, although full redundancy exploitation suggests the consideration of exactly as many additional tasks as the number of redundant degrees of freedom, i. e., $P = N - M$.

The relation between the joint-space coordinate vector $\boldsymbol{q}$ and the *constraint-task* vector $\boldsymbol{t}_c$ can be considered as a direct kinematics equation

$$\boldsymbol{t}_c = \boldsymbol{k}_c(\boldsymbol{q}) , \tag{10.35}$$

where $\boldsymbol{k}_c$ is a continuous nonlinear vector function. Accordingly, it is useful to consider the mapping

$$\dot{\boldsymbol{t}}_c = \mathbf{J}_c(\boldsymbol{q})\, \dot{\boldsymbol{q}} , \tag{10.36}$$

which can be obtained by differentiating Eq. (10.35). In (10.36), $\dot{\boldsymbol{t}}_c$ is the constraint-task velocity vector, and $\mathbf{J}_c(\boldsymbol{q}) = \partial \boldsymbol{k}_c / \partial \boldsymbol{q}$ is the $P \times N$ *constraint-task Jacobian* matrix.

At this point, an *augmented-task* vector can be defined by stacking the end-effector task vector with the constraint-task vector as

$$\boldsymbol{t}_a = \begin{pmatrix} \boldsymbol{t} \\ \boldsymbol{t}_c \end{pmatrix} = \begin{pmatrix} \boldsymbol{k}_t(\boldsymbol{q}) \\ \boldsymbol{k}_c(\boldsymbol{q}) \end{pmatrix} .$$

According to this definition, finding a joint configuration $\boldsymbol{q}$ that results in some desired value for $\boldsymbol{t}_a$ means satisfying both the end-effector and the constraint task at the same time.

A solution to this problem can be found at the differential level by inverting the mapping

$$\dot{\boldsymbol{t}}_a = \mathbf{J}_a(\boldsymbol{q})\, \dot{\boldsymbol{q}} , \tag{10.37}$$

where the matrix

$$\mathbf{J}_a = \begin{pmatrix} \mathbf{J}_t \\ \mathbf{J}_c \end{pmatrix}$$

is referred to as the *augmented Jacobian*.

A particular choice for the constraint-task vector is $\boldsymbol{t}_c = \boldsymbol{h}(\boldsymbol{q})$, with $\boldsymbol{h}$ defined as explained in 10.4.1, which allows the augmented Jacobian method to embed the extended Jacobian one.

10.5.3 Algorithmic Singularities

The specification of additional goals besides tracking the end-effector velocity raises the possibility that configurations exist at which the augmented kinematics problem is singular while the sole end-effector task kinematics is not; these configurations are referred to as *algorithmic singularities* [10.35]. In terms of the velocity mappings (10.33) and (10.37), an algorithmically singular configuration is one at which the extended and the augmented Jacobians, respectively, are singular while $\mathbf{J}_t$ is full-rank.

When he introduced task-augmentation redundancy resolution methods, *Baillieul* pointed out that algorithmic singularities are not a specific problem of the extended Jacobian technique but that they arise from the way in which the constraint task conflicts with the end-effector task [10.35, 37]. This is easily understandable in the simple problem of trajectory tracking with obstacle avoidance: if the assigned trajectory passes through an obstacle either the trajectory is tracked or the obstacle is avoided so that both tasks cannot be achieved. If the origin of the conflict between the two tasks has

a clear meaning the algorithmic singularity may then be avoided by keenly specifying the constraint task case by case [10.23]. In more general situations analytical tools may be useful in finding algorithmic singularities and guiding the choice of the constraint function [10.42] or in finding configurations that better harmonize the two tasks [10.43].

In terms of the tasks definitions given by (10.2) and (10.36), one can see that the two tasks are in conflict when

$$\mathcal{R}\left(\mathbf{J}_{\mathrm{c}}^{\mathrm{T}}\right) \cap \mathcal{R}\left(\mathbf{J}_{\mathrm{t}}^{\mathrm{T}}\right) \neq \{\mathbf{0}\}\,, \tag{10.38}$$

which is the condition for the occurrence of an algorithmic singularity. On the other hand, when

$$\mathcal{R}\left(\mathbf{J}_{\mathrm{c}}^{\mathrm{T}}\right) \cap \mathcal{R}\left(\mathbf{J}_{\mathrm{t}}^{\mathrm{T}}\right) = \{\mathbf{0}\}\,,$$

the two tasks are compatible becuse the two inverse mappings are linearly independent; a special case of task compatibility is when

$$\mathcal{R}\left(\mathbf{J}_{\mathrm{c}}^{\mathrm{T}}\right) \equiv \mathcal{R}^{\perp}\left(\mathbf{J}_{\mathrm{t}}^{\mathrm{T}}\right)\,, \tag{10.39}$$

and the two mappings are orthogonal.

At the algorithmic singularities the augmented Jacobian cannot be inverted but singularity-robust techniques can be adopted. Because the exact solution does not exist there are reconstruction errors and these affect both the task vectors. To counteract this problem the use of weighted damped least-squares has been considered to invert the augmented Jacobian matrix [10.44, 45]. A different approach is the so-called task priority inverse kinematics.

10.5.4 Task Priority

Conflicts between the end-effector task and the constraint task are handled in the framework of the task-priority strategy by suitably assigning an order of priority to the given tasks and then satisfying the lower-priority task only in the null space of the higher-priority task [10.46, 47]. In the typical case, an end-effector task is considered as the primary task, although examples can occur when it becomes the secondary task [10.23]. The idea is that, when an exact solution does not exist, the reconstruction error should only affect the lower-priority task.

With reference to solution (10.12), the task-priority method consists of computing $\dot{\boldsymbol{q}}_0$ to suitably achieve the P-dimensional constraint-task velocity $\dot{\boldsymbol{t}}_{\mathrm{c}}$. The projection of $\dot{\boldsymbol{q}}_0$ onto the null space of $\mathbf{J}_{\mathrm{t}}$ ensures that the lower priority of the constraint task will not affect the higher priority end-effector task [10.48].

When the secondary task $\dot{\boldsymbol{t}}_{\mathrm{c}}$ is orthogonal to the primary task $\dot{\boldsymbol{t}}$ (in the sense of (10.39)) the joint velocity

$$\dot{\boldsymbol{q}}_0 = \mathbf{J}_{\mathrm{c}}^{\dagger}(\boldsymbol{q})\dot{\boldsymbol{t}}_{\mathrm{c}} \tag{10.40}$$

would easily solve the problem, because it would already be a null-space velocity for the primary-task velocity mapping (10.2), i. e., projection through $\mathbf{N}_{\mathbf{J}_{\mathrm{t}}}$ would not be needed. However, in the general case the two tasks may be compatible but not orthogonal or may conflict and there may not exist a joint velocity solution that ensures the achievement of both $\dot{\boldsymbol{t}}$ and $\dot{\boldsymbol{t}}_{\mathrm{c}}$. To be consistent with the defined order of priority between the two tasks, a reasonable choice is then to guarantee exact tracking of the primary-task velocity while minimizing the constraint-task velocity reconstruction error $\dot{\boldsymbol{t}}_{\mathrm{c}} - \mathbf{J}_{\mathrm{c}}\dot{\boldsymbol{q}}$; this gives [10.49]

$$\dot{\boldsymbol{q}}_0 = \left[\mathbf{J}_{\mathrm{c}}(\mathbf{I} - \mathbf{J}_{\mathrm{t}}^{\dagger}\mathbf{J}_{\mathrm{t}})\right]^{\dagger}(\dot{\boldsymbol{t}}_{\mathrm{c}} - \mathbf{J}_{\mathrm{c}}\mathbf{J}_{\mathrm{t}}^{\dagger}\dot{\boldsymbol{t}})\,. \tag{10.41}$$

Finally, by observing that the null-space projection operator is both hermitian and idempotent, the solution given by (10.12) and (10.41) can be simplified to [10.46]

$$\dot{\boldsymbol{q}} = \mathbf{J}_{\mathrm{t}}^{\dagger}\dot{\boldsymbol{t}} + \left[\mathbf{J}_{\mathrm{c}}(\mathbf{I} - \mathbf{J}_{\mathrm{t}}^{\dagger}\mathbf{J}_{\mathrm{t}})\right]^{\dagger}(\dot{\boldsymbol{t}}_{\mathrm{c}} - \mathbf{J}_{\mathrm{c}}\mathbf{J}_{\mathrm{t}}^{\dagger}\dot{\boldsymbol{t}})\,. \tag{10.42}$$

It can be recognized that the problem of algorithmic singularities still remains; in fact, when condition (10.38) is satisfied the matrix $\mathbf{J}_{\mathrm{c}}(\mathbf{I} - \mathbf{J}_{\mathrm{t}}^{\dagger}\mathbf{J}_{\mathrm{t}})$ loses rank with full-rank $\mathbf{J}_{\mathrm{t}}$ and $\mathbf{J}_{\mathrm{c}}$. However, in contrast to the task-space augmentation approach, correct primary-task solutions are expected as long as the sole primary-task Jacobian matrix is full-rank. Away from the algorithmic singularities the task-priority strategy gives the same solution as the task-space augmentation approach; this implies that close to an algorithmic singularity the solution becomes ill-conditioned and large joint velocities may result. This problem can be solved by using a continuous version of the truncated SVD as described in [10.3].

Another approach is to relax minimization of the secondary-task velocity reconstruction constraint and simply pursue tracking of the components of (10.40) that do not conflict with the primary task [10.50, 51], namely

$$\dot{\boldsymbol{q}} = \mathbf{J}_{\mathrm{t}}^{\dagger}\dot{\boldsymbol{t}} + (\mathbf{I} - \mathbf{J}_{\mathrm{t}}^{\dagger}\mathbf{J}_{\mathrm{t}})\mathbf{J}_{\mathrm{c}}^{\dagger}\dot{\boldsymbol{t}}_{\mathrm{c}}\,. \tag{10.43}$$

An intuitive justification of this solution can be given as follows: The pseudoinverses $\mathbf{J}_t^\dagger$ and $\mathbf{J}_c^\dagger$ are used to solve separately for the joint velocities associated with the respective task velocities; the joint velocity associated with the (secondary) constraint task is then projected onto the null space of $\mathbf{J}_t$ to remove the components that would interfere with the (primary) end-effector task and finally added to the joint velocity associated with the end-effector task. As a result, a nice property of solution (10.43) is that algorithmic singularities are decoupled from the singularities of $\mathbf{J}_c$.

By construction, the solution (10.43) leads to larger constraint-task reconstruction errors than solution (10.42); this is the price to be paid for smoother trajectories for the joint velocity when tracking conflicting tasks. In any case, a CLIK implementation will allow one to recover from the secondary-task tracking error due to the algorithmic singularities. In this case, the CLIK formulation becomes

$$\dot{\boldsymbol{q}} = \mathbf{J}_t^\dagger \boldsymbol{w}_t + (\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t)\mathbf{J}_c^\dagger \boldsymbol{w}_c \,,$$

with

$$\begin{aligned} \boldsymbol{w}_t &= \dot{\boldsymbol{t}} + \mathbf{K}_t \left(\boldsymbol{t} - \boldsymbol{k}_t(\boldsymbol{q})\right) \,, \\ \boldsymbol{w}_c &= \dot{\boldsymbol{t}}_c + \mathbf{K}_c \left(\boldsymbol{t}_c - \boldsymbol{k}_c(\boldsymbol{q})\right) \,. \end{aligned}$$

10.6 Second-Order Redundancy Resolution

Redundancy resolution at the acceleration level allows the consideration of dynamic performance along the manipulator motion. Moreover, the obtained acceleration profiles can be directly used as reference signals (together with the corresponding positions and velocities) of a task-space dynamic controller. On the other hand, a second-order redundancy resolution scheme is invariably more demanding in terms of computational load.

The simplest scheme operating at the acceleration level is represented by (10.15), i. e., the solution with the minimum norm among those that realize the task constraint (10.8). Similar to the velocity-level pseudoinverse solution, the joint movement generated by this locally optimal solution does not result in global acceleration minimization along the entire manipulator motion. Remarkably, however, the use of (10.15) does lead to the minimization of the integral index

$$\int_{t_i}^{t_f} \dot{\boldsymbol{q}}^T \dot{\boldsymbol{q}} \, \mathrm{d}t \,,$$

over $[t_i, t_f]$ provided that the appropriate boundary conditions are satisfied [10.52]. For example, in the case of free endpoints (neither joint positions nor velocities specified at t_i and t_f) the boundary conditions to be satisfied are split and expressed as

$$\dot{\boldsymbol{q}}(\bar{t}) = \mathbf{J}_t^\dagger \dot{\boldsymbol{t}}(\bar{t}) \,, \quad \bar{t} = t_i, t_f \,.$$

Thus, in spite of the apparent simplicity and elegance of the solution (10.15), actual minimization of the above integral requires the solution of a two-point boundary value problem (TPBVP), which is a computationally intensive numerical procedure impractical for real-time kinematic control. However, this may be perfectly acceptable for offline redundancy resolution in an industrial setting.

More flexibility in the choice of (both local and global) performance criteria is obviously obtained by considering the full second-order solution (10.14). Let the manipulator dynamic model be expressed as

$$\boldsymbol{\tau} = \mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\tau}_g(\boldsymbol{q}) \,, \tag{10.44}$$

where $\boldsymbol{\tau}$ is the vector of actuator torques, $\mathbf{H}$ is the manipulator inertia matrix, $\boldsymbol{c}$ is the vector of centrifugal/Coriolis terms, and $\boldsymbol{\tau}_g$ is the gravitational torque vector. Choosing the null-space acceleration in (10.14) as

$$\ddot{\boldsymbol{q}}_0 = -\left[\mathbf{H}(\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t)\right]^\dagger \tilde{\boldsymbol{\tau}} \,, \tag{10.45}$$

with

$$\tilde{\boldsymbol{\tau}} = \mathbf{H}\mathbf{J}_t^\dagger (\ddot{\boldsymbol{t}} - \dot{\mathbf{J}}_t \dot{\boldsymbol{q}}) + \boldsymbol{c} + \boldsymbol{\tau}_g$$

leads to the local minimization of the actuator torque norm $\boldsymbol{\tau}^T\boldsymbol{\tau}$ [10.53]. This particular redundancy resolution scheme performs reasonably over short tasks, but may lead to instability (more precisely, very high joint torques) in the long run, essentially due to the build-up of null-space joint velocities. Note also that the matrix product $\mathbf{H}(\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t)$ in (10.45) is not full-rank; however, its pseudoinverse can be efficiently calculated using the procedure given in [10.54]. Minimization of the integral joint torque is also possible [10.55]; this solution obviously avoids the instability problem, but once again the solution of a TPBVP is required.

Another interesting inverse solution is the following

$$\ddot{\boldsymbol{q}} = \mathbf{J}_{\mathrm{t},\mathbf{H}}^{\dagger}(\ddot{\boldsymbol{t}} - \dot{\mathbf{J}}_{\mathrm{t}}\,\dot{\boldsymbol{q}}) + (\mathbf{I} - \mathbf{J}_{\mathrm{t},\mathbf{H}}^{\dagger}\mathbf{J}_{\mathrm{t}})\,\mathbf{H}^{-1}\boldsymbol{c}\,, \tag{10.46}$$

which is slightly modified with respect to the general formula (10.14) in view of the use of a weighted pseudoinverse. In particular, $\mathbf{J}_{\mathrm{t},\mathbf{H}}^{\dagger}$ is the inertia-weighted task Jacobian pseudoinverse, given in the full-rank case by the following expression

$$\mathbf{J}_{\mathrm{t},\mathbf{H}}^{\dagger} = \mathbf{H}^{-1}\mathbf{J}_{\mathrm{t}}^{\mathrm{T}}(\mathbf{J}_{\mathrm{t}}\,\mathbf{H}^{-1}\mathbf{J}_{\mathrm{t}}^{\mathrm{T}})^{-1}\,.$$

The solution (10.46) minimizes

$$\int_{t_{\mathrm{i}}}^{t_{\mathrm{f}}} \frac{1}{2}\,\dot{\boldsymbol{q}}^{\mathrm{T}}\mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{q}}\;\mathrm{d}t\,,$$

i. e., the integral over $[t_{\mathrm{i}}, t_{\mathrm{f}}]$ of the manipulator kinetic energy [10.52]. Once again, the correct boundary conditions must be used; for example, the case of free endpoints leads to

$$\dot{\boldsymbol{q}}(\bar{t}) = \mathbf{J}_{\mathrm{t},\mathbf{H}}^{\dagger}\dot{\boldsymbol{t}}(\bar{t})\,, \quad \bar{t} = t_{\mathrm{i}}, t_{\mathrm{f}}\,.$$

10.7 Cyclicity

A common drawback of redundancy resolution schemes based on differential kinematics is the lack of *cyclicity* (also called *repeatability*): in general, the joint-space trajectory corresponding to a cyclic task space trajectory is not cyclic itself (i. e., the final position of the joints does not coincide with the initial position). This phenomenon is clearly undesirable, because it basically means that the behavior of the manipulator along periodic tasks to be repeated over and over is unpredictable.

A mathematical condition for cyclicity exists for a particular class of redundancy resolution methods [10.56]. In particular, consider any scheme of the form

$$\dot{\boldsymbol{q}} = \mathbf{G}_{\mathrm{t}}(\boldsymbol{q})\dot{\boldsymbol{t}}\,, \tag{10.47}$$

where $\mathbf{G}_{\mathrm{t}}$ is any *generalized inverse* of the task Jacobian $\mathbf{J}_{\mathrm{t}}$, i. e., an $N \times M$ matrix such that $\mathbf{J}_{\mathrm{t}}\mathbf{G}_{\mathrm{t}}\mathbf{J}_{\mathrm{t}} = \mathbf{J}_{\mathrm{t}}$ (the pseudoinverse matrix $\mathbf{J}_{\mathrm{t}}^{\dagger}$ in (10.13) is a particular generalized inverse). Assume that the assigned task $\boldsymbol{t}(t)$ describes a cyclic trajectory in a simply connected region of the task space, and denote by $\boldsymbol{g}_{\mathrm{t}i}(\boldsymbol{q})$ the i-th column of $\mathbf{G}_{\mathrm{t}}$. A necessary and sufficient condition for (10.47) to generate cyclic joint trajectories is that the *distribution*

$$\Delta_{\mathbf{G}_{\mathrm{t}}}(\boldsymbol{q}) = \mathrm{span}\{\boldsymbol{g}_{\mathrm{t}1}(\boldsymbol{q}), \ldots, \boldsymbol{g}_{\mathrm{t}M}(\boldsymbol{q})\}$$

is involutive (i. e., it is closed under the Lie bracket operation).

It should be emphasized that the involutivity of $\Delta_{\mathbf{G}_{\mathrm{t}}}$ is a strong condition, because it must be satisfied at any configuration. This suggests that most generalized inverses are not cyclic. Note also that the above condition for cyclicity depends on the chosen generalized inverse (i. e., pseudoinverse, weighted pseudoinverse, and so on) as well as on the form of $\mathbf{J}_{\mathrm{t}}$, which in turn is related to the mechanical structure of the manipulator. This means that cyclicity must be established on a case-by-case basis. However, it is possible to design a repeatable inverse that can approximate any desired generalized inverse over a specified subset of the workspace [10.57].

As for redundancy resolution schemes which do not fall in the class (10.47) – namely, those entailed by the general inverse solution (10.12) – they are in general not cyclic. In particular, this is true for when local optimization is used to solve redundancy, as in (10.31). A notable exception is the extended Jacobian method, which is always cyclic.

10.8 Fault Tolerance

One natural use for kinematic redundancy is that of fault tolerance, i. e., if there are more degrees of freedom than the minimum required to complete a task then one should, theoretically, be able to tolerate a joint failure and still complete the desired task. While component failures for robots employed in structured and benign environments where regular maintenance can be performed are relatively rare, there are many important applications, although less common, where this is not true, e.g., in space or underwater exploration and in nuclear environments. The failure rates for components in such harsh environments are relatively high [10.58] and maintenance is not possible. Many of these component failures will result in a robot's joint becoming immobilized, i. e., a locked joint failure mode [10.59]. Component failures that result in other common fail-

ure modes, e.g., free-swinging joint failures [10.60], are also frequently transformed into the locked joint failure mode by failure recovery mechanisms that employ fail safe brakes [10.61].

A large body of work on fault-tolerant manipulators has focused on the kinematic properties of redundant robots, both of serial and parallel structure [10.62]. These analyses have been performed both on the local properties associated with the manipulator Jacobian [10.63, 64] as well as the global characteristics such as the resulting workspace following a particular failure [10.65–67]. Clearly both local and global kinematic properties are related, e.g., if a robot is at a workspace boundary then the Jacobian is singular.

The effect of locked joint failures on the motion capabilities of a robotic manipulator are most easily characterized when the manipulator Jacobian is represented as a collection of columns

$$\mathbf{J}_{M\times N} = \begin{pmatrix} \mathcal{J}_1 & \mathcal{J}_2 & \dots & \mathcal{J}_N \end{pmatrix} ,$$

where $\mathcal{J}_i$ represents the end-effector velocity due to the velocity of joint i. For an arbitrary single joint failure at joint f, assuming that the failed joint can be locked, the resulting $(M\times(N-1))$ Jacobian will be missing the f-th column, where f can range from 1 to N. This Jacobian will be denoted by a preceding superscript so that in general

$${}^{f}\mathbf{J}_{M\times(N-1)} = \begin{pmatrix} \mathcal{J}_1 & \mathcal{J}_2 & \dots & \mathcal{J}_{f-1} & \mathcal{J}_{f+1} & \dots & \mathcal{J}_N \end{pmatrix}.$$

As noted above, the local velocity properties of a manipulator Jacobian are frequently quantified in terms of the singular values. Most local dexterity measures can be defined in terms of simple combinations of these singular values such as their product (determinant), sum (trace), or ratio (condition number). The most significant of the singular values is σ_M, the minimum singular value, because it is by definition the measure of proximity to a singularity and tends to dominate the behavior of both the manipulability (determinant) and the condition number. The minimum singular value is also a measure of the worst-case dexterity over all possible end-effector motions. One definition of failure tolerance is based on the worst-case dexterity following an arbitrary locked joint failure. Because ${}^{f}\sigma_M$ denotes the minimum singular value of ${}^{f}\mathbf{J}$, it is a measure of the worst-case dexterity if joint f fails. If all joints are equally likely to fail, then a measure of the worst-case failure tolerance is given by

$$K = \min_{f=1}^{N} {}^{f}\sigma_M \, .$$

Physically, this amounts to minimizing the worst-case increase in joint velocity when a joint is locked and the others must accelerate to maintain the desired end effector trajectory. In addition, maximizing K is equivalent to locally maximizing the distance to a post-failure workspace boundary [10.68]. To insure that manipulator performance is optimal prior to a failure, one may want to further define an optimally failure tolerant Jacobian as having all equal singular values due to the desirable properties of isotropic manipulator configurations. Under these conditions, to guarantee that the minimum ${}^{f}\sigma_M$ is as large as possible they should all be equal. It is easy to show that the worst-case dexterity of an isotropic manipulator that experiences a single joint failure is governed by the inequality

$$K = \min_{f=1}^{N} {}^{f}\sigma_M \leq \sigma\sqrt{\frac{(N-M)}{N}} \, ,$$

where σ denotes the norm of the original Jacobian. The best case of equality occurs if the manipulator is in an optimally failure tolerant configuration. The above inequality makes sense from a physical point of view because it represents the ratio of the degree of redundancy to the original number of degrees of freedom. Using the above definition of an optimally failure tolerant configuration one can identify the structure of the Jacobian required to obtain this property [10.63]. In particular, one can show that the optimally failure tolerant criteria requires that each joint contributes equally to the null space of the Jacobian transformation. Physically, this means that the redundancy of the robot is uniformly distributed among all the joints so that a fail-

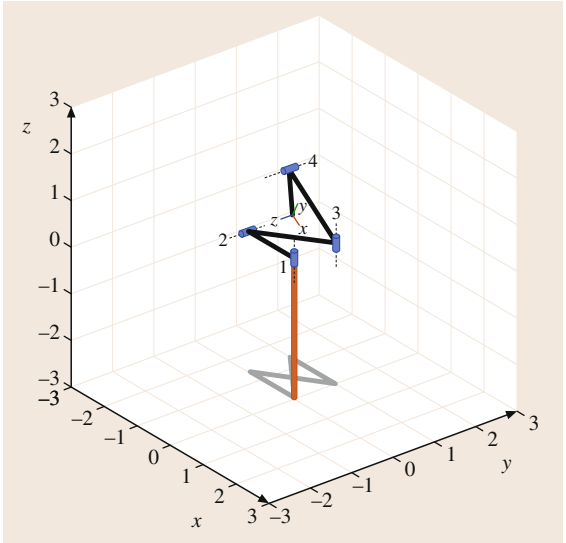


Fig. 10.9 A four degree-of-freedom spatial positioning manipulator in a configuration that is optimally fault tolerant

ure at any one joint can be compensated for by the remaining joints.

A simple example of an optimally failure tolerant configuration that is easy to visualize is that of a spatial positioning manipulator with four joints whose Jacobian is given by

$$\mathbf{J}_{\mathrm{v}} = \begin{pmatrix} -\sqrt{3/4} & \sqrt{1/12} & \sqrt{1/12} & \sqrt{1/12} \\ 0 & -\sqrt{2/3} & \sqrt{1/6} & \sqrt{1/6} \\ 0 & 0 & -\sqrt{1/2} & \sqrt{1/2} \end{pmatrix} ,$$

where $\mathbf{J}_{\mathrm{v}}$ represents the linear velocity portion of a manipulator Jacobian. The null space at this configuration is given by $(1/2)[1\,1\,1\,1]^{\mathrm{T}}$, which illustrates that each joint contributes equally to the null space motion, thus distributing the redundancy proportionally to all degrees of freedom. If the four possible single locked joint failures are considered, one can show that

$$^{f}\sigma_3 = K_{\max} = \frac{1}{2} ,$$
$$f = 1, \ldots, 4 ,$$

which satisfies the optimally failure tolerant criterion. It is then possible to design physical robot manipulators that possess this Jacobian along with its associated optimal kinematic properties. An example of one such robot is shown in Fig. 10.9 [10.69].

10.9 Conclusion and Further Reading

Research on kinematically redundant robots has been flourishing for over two decades now, and is still very active. The number of papers dealing with this subject is therefore enormous. The few works cited below are simply a small addition to the fundamental contributions already referenced so far, and constitute by no means an exhaustive list.

The mechanical design of kinematically redundant manipulators has been studied in many papers [10.70–74]. In particular, the superiority of human-arm-like manipulators over conventional 6-DOF robots was first advocated in [10.75]. Reconfiguration of this arm so as to ensure full mobility in the whole workspace is studied in [10.76, 77].

A general analysis of the inverse kinematic problem for redundant manipulators is presented in [10.78]. In particular, the geometric structure of self-motions is analyzed in [10.79].

Weighting the damped least-squares solution for guaranteed singularity avoidance in anisotropic end-effector tasks was proposed in [10.9, 21, 80].

In addition to singularity avoidance, redundancy has also been exploited to achieve obstacle avoidance [10.37, 46, 81, 82], minimization of the effects of joint elasticity [10.83], fault tolerance [10.60], reduction of impact force [10.84, 85], and maximization of various dexterity measures [10.4, 5, 43, 86]. A completely different approach for obstacle avoidance with redundant robots is presented in [10.87].

A review of redundancy resolution via local optimization is given in [10.88]. A numerically efficient alternative for redundancy resolution via local optimization is proposed in [10.89]. A redundancy resolution approach with somehow intermediate characteristics between local and global optimization is presented in [10.90]. Other methods for second-order redundancy resolution are discussed, e.g., in [10.91]. Also worth citing is the dynamically consistent generalized inverse of [10.92].

The cyclicity issue was first pointed out in [10.93], and further investigated, e.g., in [10.94, 95]. General formalisms for redundancy resolution of vehicle-manipulator systems subject to nonholonomic constraints are described in [10.96, 97].

Video-References

VIDEO 813 KUKA LBR iiwa – Kinematic redundancy
available from http://handbookofrobotics.org/view-chapter/10/videodetails/813

VIDEO 814 Free floating autonomous valve turning (task priority redundancy control + task concurrence)
available from http://handbookofrobotics.org/view-chapter/10/videodetails/814

VIDEO 815 Human inspired tele-impedance and minimum effort controller for improved manipulation performance
available from http://handbookofrobotics.org/view-chapter/10/videodetails/815

VIDEO 816 Human motion mapping to a robot arm with redundancy resolution
available from http://handbookofrobotics.org/view-chapter/10/videodetails/816

VIDEO 817 Configuration space control of KUKA lightweight robot LWR with EXARM exoskeleton
available from http://handbookofrobotics.org/view-chapter/10/videodetails/817

VIDEO 818 FlexIRob – Teaching nullspace constraints in physical human-robot interaction
available from http://handbookofrobotics.org/view-chapter/10/videodetails/818

VIDEO 819 Visual servoing control of baxter robot arms with obstacle avoidance using kinematic redundancy
available from http://handbookofrobotics.org/view-chapter/10/videodetails/819

References

10.1 D.E. Whitney: Resolved motion rate control of manipulators and human prostheses, IEEE Trans. Man-Mach. Syst. **10**(2), 47–53 (1969)

10.2 G.H. Golub, C. Reinsch: Singular value decomposition and least-squares solutions, Numer. Math. **14**, 403–420 (1970)

10.3 A.A. Maciejewski, C.A. Klein: The singular value decomposition: computation and applications to robotics, Int. J. Robotics Res. **8**(6), 63–79 (1989)

10.4 T. Yoshikawa: Manipulability of robotic mechanisms, Int. J. Robotics Res. **4**(2), 3–9 (1985)

10.5 C.A. Klein, B.E. Blaho: Dexterity measures for the design and control of kinematically redundant manipulators, Int. J. Robotics Res. **6**(2), 72–83 (1987)

10.6 A.A. Maciejewski, C.A. Klein: Numerical filtering for the operation of robotic manipulators through kinematically singular configurations, J. Robotics Syst. **5**, 527–552 (1988)

10.7 S. Chiaverini: Estimate of the two smallest singular values of the Jacobian matrix: application to damped least-squares inverse kinematics, J. Robotics Syst. **10**, 991–1008 (1993)

10.8 O. Egeland, M. Ebdrup, S. Chiaverini: Sensory control in singular configurations – Application to visual servoing, IEEE Int. Workshop Intell. Motion Control, Istanbul (1990) pp. 401–405

10.9 Y. Nakamura, H. Hanafusa: Inverse kinematic solutions with singularity robustness for robot manipulator control, Trans. ASME J. Dyn. Syst. Meas., Control **108**, 163–171 (1986)

10.10 D.E. Whitney: The mathematics of coordinated control of prosthetic arms and manipulators, Trans. ASME J. Dyn. Syst. Meas., Control **94**, 303–309 (1972)

10.11 T.L. Boullion, P.L. Odell: *Generalized Inverse Matrices* (Wiley, New York 1971)

10.12 C.R. Rao, S.K. Mitra: *Generalized Inverse of Matrices and its Applications* (Wiley, New York 1971)

10.13 A. Ben-Israel, T.N.E. Greville: *Generalized Inverses: Theory and Applications* (Wiley, New York 1974)

10.14 R.H. Taylor: Planning and execution of straight-line manipulator trajectories, IBM J. Res. Dev. **23**, 424–436 (1979)

10.15 M. Sampei, K. Furuta: Robot control in the neighborhood of singular points, IEEE J. Robotics Autom. **4**, 303–309 (1988)

10.16 E.W. Aboaf, R.P. Paul: Living with the singularity of robot wrists, IEEE Int. Conf. Robotics Autom. (ICRA), Raleigh (1987) pp. 1713–1717

10.17 S. Chiaverini, O. Egeland: A solution to the singularity problem for six-joint manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), Cincinnati (1990) pp. 644–649

10.18 S. Chiaverini, O. Egeland: An efficient pseudo-inverse solution to the inverse kinematic problem for six-joint manipulators, Model. Identif. Control **11**(4), 201–222 (1990)

10.19 O. Khatib: A unified approach for motion and force control of robot manipulators: The operational space formulation, IEEE J. Robotics Autom. **3**, 43–53 (1987)

10.20 C.W. Wampler: Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods, IEEE Trans. Syst. Man Cybern. **16**, 93–101 (1986)

10.21 S. Chiaverini, O. Egeland, R.K. Kanestrøm: Achieving user-defined accuracy with damped least-squares inverse kinematics, 5th Int. Conf. Adv. Robotics, Pisa (1991) pp. 672–677

10.22 J.R. Sagli: Coordination of Motion in Manipulators with Redundant Degrees of Freedom, Ph.D. Thesis (Norwegian University of Science and Technology, Trondheim 1991)

10.23 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy, Int. J. Robotics Res. **10**(4), 410–425 (1991)

10.24 B. Siciliano: A closed-loop inverse kinematic scheme for on-line joint-based robot control, Robotica **8**, 231–243 (1990)

10.25 Z.R. Novaković, B. Siciliano: A new second-order inverse kinematics solution for redundant manipulators. In: *Advances in Robot Kinematics*, ed. by S. Stifter, J. Lenarčič (Springer, New York 1991) pp. 408–415

10.26 A. Balestrino, G. De Maria, L. Sciavicco: Robust control of robotic manipulators, 9th IFAC World Cong., Budapest (1984) pp. 80–85

10.27 W.A. Wolovich, H. Elliott: A computational technique for inverse kinematics, 23rd IEEE Conf. Decis. Control, Las Vegas (1984) pp. 1359–1363

10.28 L. Sciavicco, B. Siciliano: A solution algorithm to the inverse kinematic problem for redundant manipulators, IEEE J. Robotics Autom. **4**, 403–410 (1988)

10.29 J. Baillieul, J. Hollerbach, R.W. Brockett: Programming and control of kinematically redundant manipulators, 23rd IEEE Conf. Decis. Control, Las Vegas (1984) pp. 768–774

10.30 A. Liégeois: Automatic supervisory control of the configuration and behavior of multibody mech-

anisms, IEEE Trans. Syst. Man Cybern. **7**, 868–871 (1977)
10.31 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, IEEE Int. Conf. Robotics Autom. (ICRA), St. Louis (1985) pp. 500–505
10.32 J.C. Latombe: *Robot Motion Planning* (Kluwer, Boston 1991)
10.33 D.G. Luenberger: *Linear and Nonlinear Programming* (Addison-Wesley, Reading 1984)
10.34 A. De Luca, G. Oriolo: Issues in acceleration resolution of robot redundancy, 3rd IFAC Symp. Robot Control, Vienna (1991) pp. 665–670
10.35 J. Baillieul: Kinematic programming alternatives for redundant manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), St. Louis (1985) pp. 722–728
10.36 P.H. Chang: A closed-form solution for inverse kinematics of robot manipulators with redundancy, IEEE J. Robotics Autom. **3**, 393–403 (1987)
10.37 J. Baillieul: Avoiding obstacles and resolving kinematic redundancy, IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 1698–1704
10.38 L. Sciavicco, B. Siciliano: Solving the inverse kinematic problem for robotic manipulators, 6th CISM-IFToMM Symp. Theory Pract. Robotics Manip., Kraków (1986) pp. 107–114
10.39 L. Sciavicco, B. Siciliano: A dynamic solution to the inverse kinematic problem for redundant manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), Raleigh (1987) pp. 1081–1087
10.40 O. Egeland: Task-space tracking with redundant manipulators, IEEE J. Robotics Autom. **3**, 471–475 (1987)
10.41 H. Seraji: Configuration control of redundant manipulators: theory and implementation, IEEE J. Robotics Autom. **5**, 472–490 (1989)
10.42 J. Baillieul: A constraint oriented approach to inverse problems for kinematically redundant manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), Raleigh (1987) pp. 1827–1833
10.43 S.L. Chiu: Control of redundant manipulators for task compatibility, IEEE Int. Conf. Robotics Autom. (ICRA), Raleigh (1987) pp. 1718–1724
10.44 H. Seraji, R. Colbaugh: Improved configuration control for redundant robots, J. Robotics Syst. **7**, 897–928 (1990)
10.45 O. Egeland, J.R. Sagli, I. Spangelo, S. Chiaverini: A damped least-squares solution to redundancy resolution, IEEE Int. Conf. Robotics Autom. (ICRA), Sacramento (1991) pp. 945–950
10.46 A.A. Maciejewski, C.A. Klein: Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments, Int. J. Robotics Res. **4**(3), 109–117 (1985)
10.47 Y. Nakamura, H. Hanafusa, T. Yoshikawa: Task-priority based redundancy control of robot manipulators, Int. J. Robotics Res. **6**(2), 3–15 (1987)
10.48 H. Hanafusa, T. Yoshikawa, Y. Nakamura: Analysis and control of articulated robot arms with redundancy, IFAC 8th Trienn. World Congr., Kyoto (1981) pp. 78–83
10.49 Y. Nakamura, H. Hanafusa: Task priority based redundancy control of robot manipulators. In: *Robotics Research – The Second International Symposium*, ed. by H. Hanafusa, H. Hinoue (MIT Press, Cambridge 1985) pp. 155–162
10.50 S. Chiaverini: Task-priority redundancy resolution with robustness to algorithmic singularities, 4th IFAC Symp. Robot Control, Capri (1994) pp. 393–399
10.51 S. Chiaverini: Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators, IEEE Trans. Robotics Autom. **13**, 398–410 (1997)
10.52 K. Kazerounian, Z. Wang: Global versus local optimization in redundancy resolution of robotic manipulators, Int. J. Robotics Res. **7**(5), 312 (1988)
10.53 J.M. Hollerbach, K.C. Suh: Redundancy resolution of manipulators through torque optimization, IEEE J. Robotics Autom. **3**, 308–316 (1987)
10.54 A.A. Maciejewski: Kinetic limitations on the use of redundancy in robotic manipulators, IEEE Trans. Robotics Autom. **7**(2), 205–210 (1991)
10.55 J.M. Hollerbach, K.C. Suh: Local versus global torque optimization of redundant manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), Raleigh (1987) pp. 619–624
10.56 T. Shamir, Y. Yomdin: Repeatability of redundant manipulators: Mathematical solution of the problem, IEEE Trans. Autom. Control **33**, 1004–1009 (1988)
10.57 R.G. Roberts, A.A. Maciejewski: Repeatable generalized inverse control strategies for kinematically redundant manipulators, IEEE Trans. Autom. Control **38**(5), 689–699 (1993)
10.58 B.S. Dhillon, A.R.M. Fashandi, K.L. Liu: Robot systems reliability and safety: A review, J. Qual. Maint. Eng. **8**(3), 170–212 (2002)
10.59 M.L. Visinsky, J.R. Cavallaro, I.D. Walker: A dynamic fault tolerance framework for remote robots, IEEE Trans. Robotics Autom. **11**(4), 477–490 (1995)
10.60 J.D. English, A.A. Maciejewski: Fault tolerance for kinematically redundant manipulators: Anticipating free-swinging joint failures, IEEE Trans. Robotics Autom. **14**, 566–575 (1998)
10.61 P. Nieminen, S. Esque, A. Muhammad, J. Mattila, J. Väyrynen, M. Siuko, M. Vilenius: Water hydraulic manipulator for fail safe and fault tolerant remote handling operations at ITER, Fusion Eng. Des. **84**(7), 1420–1424 (2009)
10.62 Y. Yi, J.E. McInroy, Y. Chen: Fault tolerance of parallel manipulators using task space and kinematic redundancy, IEEE Trans. Robotics **22**(5), 1017–1021 (2006)
10.63 A.A. Maciejewski, R.G. Roberts: On the existence of an optimally failure tolerant 7R manipulator Jacobian, Appl. Math. Comp. Sci. **5**(2), 343–357 (1995)
10.64 R.G. Roberts, A.A. Maciejewski: A local measure of fault tolerance for kinematically redundant manipulators, IEEE Trans. Robotics Autom. **12**(4), 543–552 (1996)
10.65 C.J.J. Paredis, P.K. Khosla: Fault tolerant task execution through global trajectory planning, Rel. Eng. Syst. Saf. **53**, 225–235 (1996)

10.66 C.L. Lewis, A.A. Maciejewski: Fault tolerant operation of kinematically redundant manipulators for locked joint failures, IEEE Trans. Robotics Autom. **13**(4), 622–629 (1997)

10.67 R.S. Jamisola Jr., A.A. Maciejewski, R.G. Roberts: Failure tolerant path planning for kinematically redundant manipulators anticipating locked-joint failures, IEEE Trans. Robotics **22**(4), 603–612 (2006)

10.68 K.N. Groom, A.A. Maciejewski, V. Balakrishnan: Real-time failure-tolerant control of kinematically redundant manipulators, IEEE Trans. Robotics Autom. **15**(6), 1109–1116 (1999)

10.69 K.M. Ben-Gharbia, A.A. Maciejewski, R.G. Roberts: Kinematic design of redundant robotic manipulators for spatial positioning that are optimally fault tolerant, IEEE Trans. Robotics Autom. **29**(5), 1300–1307 (2013)

10.70 J. Salisbury, J. Abramowitz: Design and control of a redundant mechanism for small motion, IEEE Int. Conf. Robotics Autom. (ICRA), St. Louis (1985) pp. 323–328

10.71 J. Baillieul: Design of kinematically redundant mechanisms, 24th IEEE Conf. Decis. Control, Ft. Lauderdale (1985) pp. 18–21

10.72 G.S. Chirikjian, J.W. Burdick: Design and experiments with a 30 DoF robot, IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 113–119

10.73 J. Angeles: The design of isotropic manipulator architectures in the presence of redundancies, Int. J. Robotics Res. **11**(3), 196–201 (1992)

10.74 A. Bowling, O. Khatib: Design of macro/mini manipulators for optimal dynamic performance, IEEE Int. Conf. Robotics Autom. (ICRA), Albuquerqe (1997) pp. 449–454

10.75 J.M. Hollerbach: Optimum kinematic design for a seven degree of freedom manipulator. In: *Robotics Research – The Second International Symposium*, ed. by H. Hanafusa, H. Hinoue (MIT Press, Cambridge 1985) pp. 216–222

10.76 O. Egeland, J.R. Sagli, S. Hendseth, F. Wilhelmsen: Dynamic coordination in a manipulator with seven joints, IEEE Int. Conf. Robotics Autom. (ICRA), Scottsdale (1989) pp. 125–130

10.77 S. Chiaverini, B. Siciliano, O. Egeland: Kinematic analysis and singularity avoidance for a seven-joint manipulator, Am. Control Conf., San Diego (1990) pp. 2300–2305

10.78 D.R. Baker, C.W. Wampler: On the inverse kinematics of redundant manipulators, Int. J. Robotics Res. **7**(2), 3–21 (1988)

10.79 J.W. Burdick: On the inverse kinematics of redundant manipulators: Characterization of the self-motion manifolds, IEEE Int. Conf. Robotics Autom. (ICRA), Scottsdale (1989) pp. 264–270

10.80 S. Chiaverini, O. Egeland, R.K. Kanestrøm: Weighted damped least-squares in kinematic control of robotic manipulators, Adv. Robotics **7**, 201–218 (1993)

10.81 M. Kirćanski, M. Vukobratović: Trajectory planning for redundant manipulators in presence of obstacles, 5th CISM-IFToMM Symp. Theor. Pract. Robots Manip., Udine (1984) pp. 43–58

10.82 C.A. Klein: Use of redundancy in the design of robotic systems. In: *Robotics Research – The Second International Symposium*, ed. by H. Hanafusa, H. Hinoue (MIT Press, Cambridge 1985) pp. 207–214

10.83 J. Baillieul: Kinematic redundancy and the control of robots with flexible components, IEEE Int. Conf. Robotics Autom. (ICRA), Nice (1992) pp. 715–721

10.84 I.D. Walker: The use of kinematic redundancy in reducing impact and contact effects in manipulation, IEEE Int. Conf. Robotics Autom. (ICRA), Cincinnati (1990) pp. 434–439

10.85 M.W. Gertz, J.O. Kim, P.K. Khosla: Exploiting redundancy to reduce impact force, IEEE/RSJ Int. Workshop Intell. Robots Syst., Osaka (1991) pp. 179–184

10.86 T. Yoshikawa: Dynamic manipulability of robot manipulators, J. Robotics Syst. **2**(1), 113–124 (1985)

10.87 G. Oriolo, M. Ottavi, M. Vendittelli: Probabilistic motion planning for redundant robots along given end-effector paths, IEEE/RSJ Int. Conf. Intell. Robots Syst., Lausanne (2002) pp. 1657–1662

10.88 D.N. Nenchev: Redundancy resolution through local optimization: A review, J. Robotics Syst. **6**, 6 (1989)

10.89 A. De Luca, G. Oriolo: The reduced gradient method for solving redundancy in robot arms, Robotersysteme **7**(2), 117–122 (1991)

10.90 S. Seereeram, J.T. Wen: A global approach to path planning for redundant manipulators, IEEE Trans. Robotics Autom. **11**(1), 152–160 (1995)

10.91 A. De Luca, G. Oriolo, B. Siciliano: Robot redundancy resolution at the acceleration level, Lab. Robotics Autom. **4**(2), 97–106 (1992)

10.92 O. Khatib: Inertial properties in robotics manipulation: An object-level framework, Int. J. Robotics Res. **14**(1), 19–36 (1995)

10.93 C.A. Klein, C.H. Huang: Review of pseudoinverse control for use with kinematically redundant manipulators, IEEE Trans. Syst. Man Cybern. **13**, 245–250 (1983)

10.94 R. Mukherjee: Design of holonomic loops for repeatability in redundant manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), Nagoya (1985) pp. 2785–2790

10.95 A. De Luca, G. Oriolo: Nonholonomic behavior in redundant robots under kinematic control, IEEE Trans. Robotics Autom. **13**(5), 776–782 (1997)

10.96 B. Bayle, J.Y. Fourquet, M. Renaud: Manipulability of wheeled-mobile manipulators: Application to motion generation, Int. J. Robotics Res. **22**(7/8), 565–581 (2003)

10.97 A. De Luca, G. Oriolo, P.R. Giordano: Kinematic modeling and redundancy resolution for nonholonomic mobile manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), Orlando (2006) pp. 1867–1873

Multimedia Contents

11. Robots with Flexible Elements

Alessandro De Luca, Wayne J. Book

Design issues, dynamic modeling, trajectory planning, and feedback control problems are presented for robot manipulators having components with mechanical flexibility, either concentrated at the joints or distributed along the links. The chapter is divided accordingly into two main parts. Similarities or differences between the two types of flexibility are pointed out wherever appropriate.

For robots with flexible joints, the dynamic model is derived in detail by following a Lagrangian approach and possible simplified versions are discussed. The problem of computing the nominal torques that produce a desired robot motion is then solved. Regulation and trajectory tracking tasks are addressed by means of linear and nonlinear feedback control designs.

For robots with flexible links, relevant factors that lead to the consideration of distributed flexibility are analyzed. Dynamic models are presented, based on the treatment of flexibility through lumped elements, transfer matrices, or assumed modes. Several specific issues are then highlighted, including the selection of sensors, the model order used for control design, and the generation of effective commands that reduce or eliminate residual vibrations in rest-to-rest maneuvers. Feedback control alternatives are finally discussed.

In each of the two parts of this chapter, a section is devoted to the illustration of the original references and to further readings on the subject.

11.1 **Robots with Flexible Joints** 244
11.1.1 Dynamic Modeling 245
11.1.2 Inverse Dynamics 249
11.1.3 Regulation Control 252
11.1.4 Trajectory Tracking 257
11.1.5 Further Reading 261

11.2 **Robots with Flexible Links** 263
11.2.1 Design Issues 263
11.2.2 Modeling of Flexible Link Arms 265
11.2.3 Control 269
11.2.4 Further Reading 278

Video-References 279

References 279

The standard assumption underlying robot kinematics, dynamics, and control design is that manipulators consist only of rigid bodies (links and motion transmission components). This is, however, an ideal situation that may be considered valid only for slow motion and small interacting forces. In practice, mechanical flexibility in robot manipulators is present for two main reasons: the use of compliant transmission elements and the reduction of the mass of the moving links through the use of lightweight material and slender design. These two kinds of flexibility introduce static and dynamic deflections between the position of the driving actuators and the position of the manipulator end-effector. If flexibility is not taken into account when considering robot design and control, a degradation of the overall expected performance of the robot typically occurs.

From a modeling point of view, flexibility can be assumed as concentrated at the robot joints or distributed (in different ways) along the robot links. The dynamic modeling steps are similar, with the need to introduce additional generalized coordinates besides those used to describe the rigid motion of the robot arm. However, the properties of the resulting models are quite different from a control point of view. Therefore, the case of *robots with flexible joints* and of *robots with flexible links* are presented in this chapter mostly as

separate items, pointing out similarities or structural differences wherever appropriate. For both classes of flexible robots, the relevant design issues, dynamic modeling, inverse dynamics algorithms, and control laws for set-point regulation and trajectory tracking will be discussed. Indeed, joint and link flexibilities may be present (and dynamically interact) at the same time. Many of the presented results can be extended to this case as well.

In the following, it is assumed that the reader is already familiar with basic issues on kinematics, dynamics, and control of rigid robots (Chaps. 2, 3, 8).

11.1 Robots with Flexible Joints

The presence of joint flexibility is common in current industrial robots, when motion transmission/reduction elements such as belts – (as in the selective compliant assembly robot arm (SCARA) family), long shafts (as in the forearm of the Unimation Puma arm), cables, harmonic drives, or cycloidal gears are used. The purpose of these components is to allow relocation of the actuators next to the robot base, thus improving dynamic efficiency, or to guarantee high reduction ratios with power-efficient compact inline devices.

However, when subject to the forces/torques arising in normal robot operation, these components are intrinsically flexible (e.g., the flexspline in harmonic drives; Fig. 11.1) and introduce a time-varying displacement between the position of the actuators and that of the driven links. Without a specific control action, an oscillatory behavior, typically of small magnitude but at a relatively high frequency, is observed at the robot end-effector level during free motion. Moreover, some form of instability (e.g., chattering) may occur in tasks involving contact with the environment.

Recently, flexible actuation/transmission elements have been deliberately selected in soft robots (such as those driven by serial elastic actuation (SEA) or variable stiffness actuation (VSA), Chap. 21) intended for physical human–robot interaction and for achieving more natural motion characteristics. In fact, this form of mechanical compliance guarantees an inertial decoupling between the actuator and the (possibly, lightweight) link, thus reducing the kinetic energy involved in undesired collisions with humans. Such a safety-oriented mechanical design should be traded-off with a more complex control design aimed at preserving the same performance of rigid robots in terms of speed and accuracy of the end-effector motion (Chap. 69).

Figures 11.2 and 11.3 show two examples of robot manipulators with flexible joints, the 8*R* robot *Dex-*

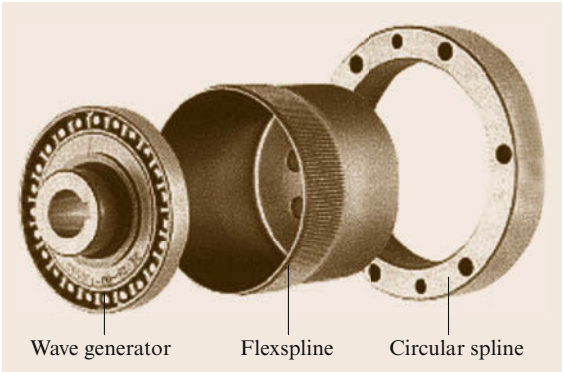


Fig. 11.1 The components of a harmonic drive

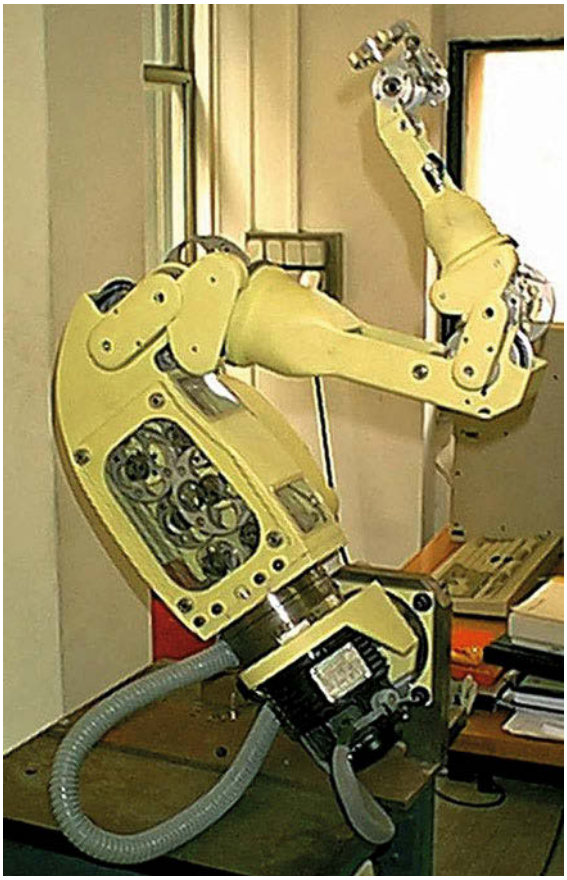


Fig. 11.2 The cable-driven robot *Dexter* by Scienzia Machinale

ter and the 7*R* lightweight manipulator *DLR LWR-III*. The first robot has motors 3–8 located inside the second link, with motion transmitted to the distal links through steel cables and pulleys; the second robot has a modular structure, with each motor integrated in the associated joint and using an harmonic drive as reduction element. The joint stiffnesses of the *Dexter* robot vary between 120 and 6300 N m/rad depending on the joint, while for the *DLR LWR-III* manipulator, the range is 6000–15 000 N m/rad. These numerical values refer to deflections and associated torques evaluated after the reduction gears.

Deflection of the flexible transmission elements can be modeled as being *concentrated at the joints* of the robot, thus limiting the complexity of the associated equations of motion. When compared to the rigid case, the dynamic model of robots with flexible joints (and rigid links) requires *twice* the number of generalized coordinates to completely characterize the configuration of all rigid bodies (motors and links) constituting the arm.

As a result, the case of flexible joint robots is one example in which the number of control inputs is strictly less than the number of mechanical degrees of freedom. This suggests that the design of control schemes realizing standard motion tasks is expected to be more difficult than for rigid robots. Moreover, the implementation of a full state feedback law will require twice the number of sensors, measuring quantities that are *before* and *after* (or across) the joint deformation. On the other hand, the motor torques used to command the robot as well as the disturbance torques due to joint flexibility perform work on the same joint axes (they are *physically collocated*). It should be noted that this is one relevant difference with respect to the case of flexibility distributed along the links. This characteristic is very helpful for rejecting vibrations and controlling the overall robot motion. Roughly speaking, we will be able to use input commands acting before the source of flexibility to ensure that output variables defined beyond the flexibility behave in a desired way.

Fig. 11.3 The lightweight manipulator *DLR LWR-III* by the German Aerospace Center

11.1.1 Dynamic Modeling

Dynamic models that explicitly include joint flexibility are used to evaluate quantitatively vibratory effects superposed on the rigid motion, verify whether the control laws obtained under the assumption of joint rigidity can still work in practice (or to what extent they should be modified), and enable the design of new model-based feedforward and feedback controllers.

We consider a robot with flexible joints as an open kinematic chain having $N+1$ rigid bodies, the base and the N links, interconnected by N (rotary or prismatic) joints undergoing deflection, and actuated by N electrical drives. From a mechanical point of view, each motor (with its stator and rotor) is an additional rigid body with its inertial properties. All joints are considered to be flexible, though mixed situations may be encountered due to the use of different transmission devices. When reduction gearings are present, they are modeled as being placed *before* the joint deflection occurs.

The following standard assumptions are made:

A1 Joint deflections are small, so that flexibility effects are limited to the domain of linear elasticity.
A2 The actuators' rotors are modeled as uniform bodies having their center of mass on the rotation axis.
A3 Each motor is located on the robot arm in a position preceding the driven link. (This can be generalized to the case of multiple motors simultaneously driving multiple distal links.)

The first assumption supports the terminology of robots with *elastic* joints often used in the literature. The elasticity at joint i is modeled by a spring of stiff-

ness $K_i > 0$, which is torsional for rotational joints and linear for translational ones. Figure 11.4 shows a single link driven by a motor through a rotational elastic joint. Nonlinear stiffness characteristics for the flexible joint can also be considered, provided that the map from deflection to force is smooth and invertible. Assumption **A2** is a basic requirement for the long life of an electrical drive, and thus very reasonable. As we will see, it implies that the inertia matrix and the gravity term in the robot dynamic model will be independent of the angular position of the motors. A typical arrangement of motors satisfying assumption **A3** is shown in Fig. 11.5. The simplest situation is when the i-th motor moves link i and is mounted on link $i-1$ with its rotation axis aligned with the i-th joint. This is, for instance, the case for the *LWR-III* manipulator. The dislocation of the actuating motors along the structure has a great influence on the structure of the dynamic equations of motion.

For kinematic and dynamic analysis, $2N$ frames are attached to the $2N$ moving rigid bodies (links and motors) in the robot chain: the link frames L_i and the motor frames R_i, for $i = 1, \ldots, N$ (Fig. 11.5). For the definition of the link frames L_i, the standard Denavit–Hartenberg convention can be used. The frames R_i are attached to the motor stators, aligned with the axes of symmetry of the motors and with the z-axis along the spinning direction of the rotor.

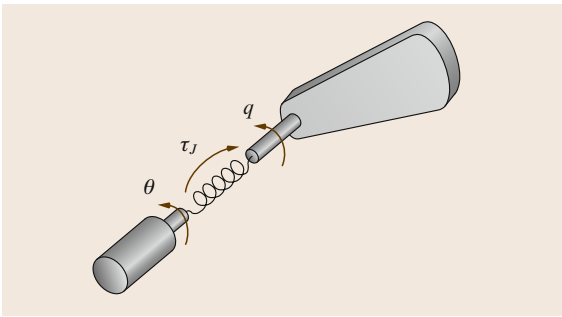


Fig. 11.4 Schematic representation of an elastic joint

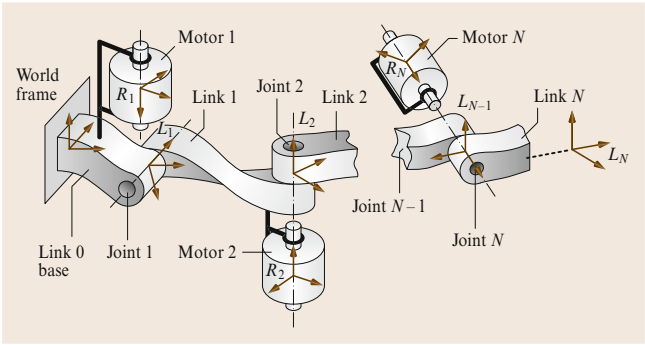


Fig. 11.5 Arrangement of motors and links in an open kinematic chain

Accordingly, $2N$ generalized coordinates will be needed. A possible set of coordinates is given by

$$\boldsymbol{\Theta} = \begin{pmatrix} \boldsymbol{q} \\ \boldsymbol{\theta} \end{pmatrix} \in \mathbb{R}^{2N} ,$$

where $\boldsymbol{q}$ is the N-vector of link positions and $\boldsymbol{\theta}$ is the N-vector of motor (i. e., rotor) positions, as reflected through the transmission/reduction gears. This choice of variables is quite convenient because:

1. The model will be formally independent of the reduction ratios.
2. These position variables will have a similar dynamic range.
3. The kinematics of the robot will be a function of the link variables $\boldsymbol{q}$ only (these variables are already *beyond* the joint flexibility) so that all issues related to direct/inverse kinematics will be identical to the case of fully rigid robots.

For some considerations, it is also useful to define the variable $\boldsymbol{\theta}_{\mathrm{m}}$, namely the N-vector of motor positions before reduction, which are the quantities directly measured by encoders mounted on the motors. In the case of a motor directly placed on the i-th joint axis, one has $\dot{\theta}_{\mathrm{m},i} = n_i \dot{\theta}_i$, where $n_i \geq 1$ is the reduction ratio at the i-th joint. In addition, for $i = 1, \ldots, N$, the difference

$$\delta_i = q_i - \theta_i$$

is the deflection at the i-th joint, while

$$\tau_{\mathrm{J},i} = K_i(\theta_i - q_i)$$

is the torque transmitted to the i-th link through the associated elastic spring (Fig. 11.4) – the quantity measured by a joint torque sensor, whenever present. Note that for robots with flexible links, the set $(\boldsymbol{\theta}, \boldsymbol{\delta})$ is typically used in the dynamic modeling, where $\boldsymbol{\delta}$ is the vector of link deflection coordinates.

Following a Lagrangian approach, the single energy contributions to the Lagrangian

$$\mathcal{L} = \mathcal{T}(\boldsymbol{\Theta}, \dot{\boldsymbol{\Theta}}) - \mathcal{U}(\boldsymbol{\Theta})$$

will be derived next.

The *potential energy* of the robot is due to gravity and joint elasticity. The gravity part is related to the position of the barycenter of the links (each of mass m_i) and of the motors (of mass m_{r_i}). Because of assumption **A2**, the latter will be independent of $\boldsymbol{\theta}$. Thus,

$$\mathcal{U}_{\mathrm{grav}} = \mathcal{U}_{\mathrm{grav,link}}(\boldsymbol{q}) + \mathcal{U}_{\mathrm{grav,motor}}(\boldsymbol{q}) .$$

For the elastic part of the potential, because of assumption **A1**, we have

$$\mathcal{U}_{\text{elas}} = \frac{1}{2}(\boldsymbol{q}-\boldsymbol{\theta})^{\mathrm{T}}\mathbf{K}(\boldsymbol{q}-\boldsymbol{\theta}) ,$$
$$\mathbf{K} = \operatorname{diag}(K_1,\dots,K_N) .$$

As a result,

$$\mathcal{U}(\boldsymbol{\Theta}) = \mathcal{U}_{\text{grav}}(\boldsymbol{q}) + \mathcal{U}_{\text{elas}}(\boldsymbol{q}-\boldsymbol{\theta}) .$$

The *kinetic energy* of the robot is the sum of the link and rotor contributions. For the links, there is no difference with respect to the standard rigid robot case and it will be sufficient to write in general

$$\mathcal{T}_{\text{link}} = \frac{1}{2}\dot{\boldsymbol{q}}^{\mathrm{T}}\mathbf{M}_{\mathrm{L}}(\boldsymbol{q})\dot{\boldsymbol{q}} , \tag{11.1}$$

with the positive-definite symmetric link inertia matrix $\mathbf{M}_{\mathrm{L}}(\boldsymbol{q})$. For the rotors, some more details are needed

$$\begin{aligned}\mathcal{T}_{\text{rotor}} &= \sum_{i=1}^{N}\mathcal{T}_{\text{rotor}_i} \\ &= \sum_{i=1}^{N}\left(\frac{1}{2}m_{r_i}\boldsymbol{v}_{r_i}^{\mathrm{T}}\boldsymbol{v}_{r_i} + \frac{1}{2}{}^{R_i}\boldsymbol{\omega}_{r_i}^{\mathrm{T}}{}^{R_i}\mathbf{I}_{r_i}{}^{R_i}\boldsymbol{\omega}_{r_i}\right),\end{aligned} \tag{11.2}$$

where $\boldsymbol{v}_{r_i}$ is the linear velocity of the center of mass of the i-th rotor and $\boldsymbol{\omega}_{r_i}$ is the angular velocity of the i-th rotor body. All quantities in the angular contributions in (11.2) are conveniently expressed in the local frame R_i. According to assumption **A2**, the rotor inertia matrix is then diagonal

$${}^{R_i}\mathbf{I}_{r_i} = \operatorname{diag}(I_{r_{ixx}}, I_{r_{iyy}}, I_{r_{izz}}) ,$$

with $I_{r_{ixx}} = I_{r_{iyy}}$, while $\boldsymbol{v}_{r_i}$ can be expressed as a function of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ only. Moreover, due to assumption **A3**, the angular velocity of the i-th rotor has the general expression

$${}^{R_i}\boldsymbol{\omega}_{r_i} = \sum_{j=1}^{i-1}\mathbf{J}_{r_i,j}(\boldsymbol{q})\dot{q}_j + \begin{pmatrix}0\\0\\ \dot{\theta}_{m,i}\end{pmatrix}, \tag{11.3}$$

where $\mathbf{J}_{r_i,j}(\boldsymbol{q})$ is the j-th column of the Jacobian relating the link velocities $\dot{\boldsymbol{q}}$ to the angular velocity of the i-th rotor in the robot chain. By substituting (11.3) into (11.2) and expressing $\dot{\boldsymbol{\theta}}_m$ in terms of $\dot{\boldsymbol{\theta}}$, it can be shown that

$$\begin{aligned}\mathcal{T}_{\text{rotor}} = &\frac{1}{2}\dot{\boldsymbol{q}}^{\mathrm{T}}\left[\mathbf{M}_{\mathrm{R}}(\boldsymbol{q}) + \mathbf{S}(\boldsymbol{q})\mathbf{B}^{-1}\mathbf{S}^{\mathrm{T}}(\boldsymbol{q})\right]\dot{\boldsymbol{q}} \\ &+ \dot{\boldsymbol{q}}^{\mathrm{T}}\mathbf{S}(\boldsymbol{q})\dot{\boldsymbol{\theta}} + \frac{1}{2}\dot{\boldsymbol{\theta}}^{\mathrm{T}}\mathbf{B}\,\dot{\boldsymbol{\theta}} ,\end{aligned} \tag{11.4}$$

where $\mathbf{B}$ is the constant diagonal inertia matrix collecting the rotors inertial components $I_{r_{izz}}$ around their spinning axes, as reflected through the reduction ratios, $\mathbf{M}_{\mathrm{R}}(\boldsymbol{q})$ contains the rotor masses (and, possibly, the rotor inertial components along the other principal axes), and the square matrix $\mathbf{S}(\boldsymbol{q})$ expresses the inertial couplings between the rotors and the previous links in the robot chain.

A simple example illustrates the derivation and the actual expressions of the terms in (11.4). We also include the reduction elements to show how the rotor inertial component around the spinning axis will appear in the kinetic energy, weighted by different powers of the reduction ratios. Consider a planar robot with two rotational elastic joints, a first link of length ℓ_1, and motors mounted directly on the joint axes. The kinetic energies of the two rotors are

$$\begin{aligned}\mathcal{T}_{\text{rotor}_1} &= \frac{1}{2}I_{r_{1zz}}\dot{\theta}_{m,1}^2 = \frac{1}{2}I_{r_{1zz}}n_1^2\dot{\theta}_1^2 \\ \mathcal{T}_{\text{rotor}_2} &= \frac{1}{2}m_{r_2}\ell_1^2\dot{q}_1^2 + \frac{1}{2}I_{r_{2zz}}(\dot{q}_1+\dot{\theta}_{m,2})^2 \\ &= \frac{1}{2}m_{r_2}\ell_1^2\dot{q}_1^2 + \frac{1}{2}I_{r_{2zz}}(\dot{q}_1^2 + 2n_2\dot{q}_1\dot{\theta}_2 + n_2^2\dot{\theta}_2^2) ,\end{aligned}$$

leading to

$$\mathbf{B} = \begin{pmatrix}I_{r_{1zz}}n_1^2 & 0\\ 0 & I_{r_{2zz}}n_2^2\end{pmatrix}, \qquad \mathbf{S} = \begin{pmatrix}0 & I_{r_{2zz}}n_2\\ 0 & 0\end{pmatrix},$$
$$\mathbf{M}_{\mathrm{R}} = \begin{pmatrix}m_{r_2}\ell_1^2 & 0\\ 0 & 0\end{pmatrix}, \qquad \mathbf{S}\mathbf{B}^{-1}\mathbf{S}^{\mathrm{T}} = \begin{pmatrix}I_{r_{2zz}} & 0\\ 0 & 0\end{pmatrix}.$$

In this specific case, the matrix $\mathbf{S}$ (as well as $\mathbf{M}_{\mathrm{R}}$) is constant. Note that for large reduction ratios n_i, the dominant inertial effect due to the rotors is given by the matrix $\mathbf{B}$. Also, if the second motor was mounted remotely on the first joint (as it is often the case in the class of SCARA arms), or still close to the second joint but with the spinning axis orthogonal to the axis of this joint, then the matrix $\mathbf{S}$ would be zero.

In general, as a consequence of assumption **A3**, the matrix $\mathbf{S}(\boldsymbol{q})$ has always a strictly upper triangular structure with a cascaded dependence of its nonzero elements

$$\mathbf{S}(\boldsymbol{q}) = \begin{pmatrix}
0 & S_{12} & S_{13}(q_2) & S_{14}(q_2,q_3) & \dots & \dots & S_{1N}(q_2,\dots,q_{N-1})\\
0 & 0 & S_{23} & S_{24}(q_3) & \dots & \dots & S_{2N}(q_3,\dots,q_{N-1})\\
0 & 0 & 0 & S_{34} & \dots & \dots & S_{3N}(q_4,\dots,q_{N-1})\\
\vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots\\
0 & 0 & 0 & \dots & 0 & S_{N-2,N-1} & S_{N-2,N}(q_{N-1})\\
0 & 0 & 0 & \dots & 0 & 0 & S_{N-1,N}\\
0 & 0 & 0 & \dots & 0 & 0 & 0
\end{pmatrix}. \tag{11.5}$$

Summing up, the total kinetic energy of the robot is

$$\mathcal{T} = \frac{1}{2}\dot{\boldsymbol{\Theta}}^{\mathrm{T}}\boldsymbol{\mathcal{M}}(\boldsymbol{\Theta})\dot{\boldsymbol{\Theta}} = \frac{1}{2}\begin{pmatrix}\dot{\boldsymbol{q}}^{\mathrm{T}} & \dot{\boldsymbol{\theta}}^{\mathrm{T}}\end{pmatrix}\begin{pmatrix}\mathbf{M}(\boldsymbol{q}) & \mathbf{S}(\boldsymbol{q})\\ \mathbf{S}^{\mathrm{T}}(\boldsymbol{q}) & \mathbf{B}\end{pmatrix}\begin{pmatrix}\dot{\boldsymbol{q}}\\ \dot{\boldsymbol{\theta}}\end{pmatrix},$$

with

$$\mathbf{M}(\boldsymbol{q}) = \mathbf{M}_{\mathrm{L}}(\boldsymbol{q}) + \mathbf{M}_{\mathrm{R}}(\boldsymbol{q}) + \mathbf{S}(\boldsymbol{q})\mathbf{B}^{-1}\mathbf{S}^{\mathrm{T}}(\boldsymbol{q})\,. \quad (11.6)$$

As anticipated, the total inertia matrix $\boldsymbol{\mathcal{M}}$ of the robot depends only on $\boldsymbol{q}$.

Using the Lagrange equations, finally yields the complete dynamic model

$$\begin{pmatrix}\mathbf{M}(\boldsymbol{q}) & \mathbf{S}(\boldsymbol{q})\\ \mathbf{S}^{\mathrm{T}}(\boldsymbol{q}) & \mathbf{B}\end{pmatrix}\begin{pmatrix}\ddot{\boldsymbol{q}}\\ \ddot{\boldsymbol{\theta}}\end{pmatrix} + \begin{pmatrix}\boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{c}_1(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{\theta}})\\ \boldsymbol{c}_2(\boldsymbol{q},\dot{\boldsymbol{q}})\end{pmatrix} + \begin{pmatrix}\boldsymbol{g}(\boldsymbol{q}) + \mathbf{K}(\boldsymbol{q}-\boldsymbol{\theta})\\ \mathbf{K}(\boldsymbol{\theta}-\boldsymbol{q})\end{pmatrix} = \begin{pmatrix}\mathbf{0}\\ \boldsymbol{\tau}\end{pmatrix}, \quad (11.7)$$

where the inertial terms (related to the total inertia matrix $\boldsymbol{\mathcal{M}}(\boldsymbol{q})$), the Coriolis and centrifugal terms (collectively denoted by $\boldsymbol{c}_{\mathrm{tot}}(\boldsymbol{\Theta},\dot{\boldsymbol{\Theta}})$), and the potential terms $(\partial\mathcal{U}(\boldsymbol{\Theta})/\partial\boldsymbol{\Theta})^{\mathrm{T}}$ have been written separately. In particular, $\boldsymbol{g}(\boldsymbol{q}) = (\partial\mathcal{U}_{\mathrm{grav}}(\boldsymbol{q})/\partial\boldsymbol{q})^{\mathrm{T}}$ while $\boldsymbol{\tau}_{\mathrm{J}} = \mathbf{K}(\boldsymbol{\theta}-\boldsymbol{q})$ is the elastic torque transmitted through the joints.

The first N and the last N equations of the dynamic model (11.7) are referred to as the *link* and the *motor equations*, respectively.

On the right-hand side of (11.7), all nonconservative generalized forces should appear. When dissipative effects are not considered, only the motor torque $\boldsymbol{\tau}$ producing work on the $\boldsymbol{\theta}$ variable is present (i. e., the torques on the motor output shafts, amplified by the reduction ratios) in the motor equations. If the robot end-effector is in contact with the environment, the zero on the right-hand side of the link equations should be replaced by $\boldsymbol{\tau}_{\mathrm{ext}} = \mathbf{J}^{\mathrm{T}}(\boldsymbol{q})\mathbf{F}$, where $\mathbf{J}(\boldsymbol{q})$ is the robot Jacobian and $\mathbf{F}$ the forces/torques acting from the environment on the robot.

In the presence of energy-dissipating effects, additional terms appear on the right-hand side of (11.7). For instance, viscous friction at both sides of the transmissions and spring damping of the (visco)elastic joints give rise to the vector term

$$\begin{pmatrix}-\mathbf{F}_{\boldsymbol{q}}\dot{\boldsymbol{q}} - \mathbf{D}(\dot{\boldsymbol{q}}-\dot{\boldsymbol{\theta}})\\ -\mathbf{F}_{\boldsymbol{\theta}}\dot{\boldsymbol{\theta}} - \mathbf{D}(\dot{\boldsymbol{\theta}}-\dot{\boldsymbol{q}})\end{pmatrix}, \quad (11.8)$$

where the diagonal, positive-definite matrices $\mathbf{F}_{\boldsymbol{q}}$, $\mathbf{F}_{\boldsymbol{\theta}}$, and $\mathbf{D}$ contain, respectively, the viscous coefficients on the link side and on the motor side, and the damping of the elastic springs at the joints. More general forms of nonlinear friction $\boldsymbol{\tau}_{\mathrm{F}}$ can be considered. Note that, in principle, friction acting on the motor side can be fully compensated by a suitable choice of the control torque $\boldsymbol{\tau}$, while this is not true for friction acting on the link side, due to the noncollocation.

Model Properties

All elements in the $2N$-vector $\boldsymbol{c}_{\mathrm{tot}}(\boldsymbol{\Theta},\dot{\boldsymbol{\Theta}})$ of velocity-dependent terms in (11.7) are independent of the motor position $\boldsymbol{\theta}$. The specific dependence of the N-vectors $\boldsymbol{c}$, $\boldsymbol{c}_1$, and $\boldsymbol{c}_2$ follows from the general expression of the components of $\boldsymbol{c}_{\mathrm{tot}}$ based on Christoffel symbols

$$c_{\mathrm{tot},i}(\boldsymbol{\Theta},\dot{\boldsymbol{\Theta}}) = \frac{1}{2}\dot{\boldsymbol{\Theta}}^{\mathrm{T}}\left[\frac{\partial\boldsymbol{\mathcal{M}}_i}{\partial\boldsymbol{\Theta}} + \left(\frac{\partial\boldsymbol{\mathcal{M}}_i}{\partial\boldsymbol{\Theta}}\right)^{\mathrm{T}} - \frac{\partial\boldsymbol{\mathcal{M}}}{\partial\boldsymbol{\Theta}_i}\right]\dot{\boldsymbol{\Theta}}\,,$$

for $i = 1,\ldots,2N$, where $\boldsymbol{\mathcal{M}}_i$ is the i-th column of the total inertia matrix $\boldsymbol{\mathcal{M}}(\boldsymbol{\Theta})$. In particular, the velocity vectors $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ arise only in the presence of a configuration-dependent $\mathbf{S}(\boldsymbol{q})$ matrix. Performing calculations, it can be shown that:

1. The vector $\boldsymbol{c}_1$ does not contain quadratic velocity terms in $\dot{\boldsymbol{q}}$ or $\dot{\boldsymbol{\theta}}$, but only mixed quadratic terms $\dot{\theta}_i\dot{q}_j$.
2. When the matrix $\mathbf{S}$ is constant (in particular, zero), both $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ vanish.

The dynamic model (11.7) also shares some properties of the rigid case, such as:

- The model equations admit a linear parametrization in terms of a suitable set of dynamic coefficients, including joint stiffnesses and motor inertias, which is useful for model identification and adaptive control.
- The Coriolis and centrifugal terms can always be factorized as $\boldsymbol{c}_{\mathrm{tot}}(\boldsymbol{\Theta},\dot{\boldsymbol{\Theta}}) = \boldsymbol{C}(\boldsymbol{\Theta},\dot{\boldsymbol{\Theta}})\dot{\boldsymbol{\Theta}}$, in such a way that matrix $\dot{\boldsymbol{\mathcal{M}}} - 2\boldsymbol{C}$ is skew-symmetric – a property used in control analysis.
- For robots having only rotational joints, the gradient of the gravity vector $\boldsymbol{g}(\boldsymbol{q})$ is globally bounded in norm by a constant.

Finally, when the joint stiffness is extremely large ($\mathbf{K}\to\infty$), then $\boldsymbol{\theta}\to\boldsymbol{q}$ while $\boldsymbol{\tau}_{\mathrm{J}}\to\boldsymbol{\tau}$. It is easy to check that the dynamic model (11.7) collapses in the limit into the standard model of fully rigid robots (including links and motors).

Reduced Model

In general, the link and motor equations in (11.7) are dynamically coupled through the elastic torque $\boldsymbol{\tau}_{\mathrm{J}}$ at the joints, but also (at the acceleration level) via the inertial components of matrix $\mathbf{S}(\boldsymbol{q})$ – usually a path of low energy transfer. The presence and actual relevance

of these inertial couplings depend on the kinematic arrangement of the manipulator and, in particular, on the specific placement of the motors and transmission devices. There are cases in which the matrix $\mathbf{S}$ is constant (e.g., the planar case, as in the previous example, with any number of links) or zero (e.g., for a single link with elastic joint or for a robot with $N = 2$ links having the two joint axes orthogonal and the motors mounted at the joints). As a result, the dynamic equations will simplify considerably.

For a generic robot with elastic joints, one can take advantage of the presence of large reduction ratios (with the n_i's of the order of 100–150) and simply neglect energy contributions due to the inertial couplings between the motors and the links (see again the $2R$ planar example). This is equivalent to considering the following simplifying assumption:

A4 The angular velocity of the rotors is due only to their own spinning, i. e.,

$$^{R_i}\boldsymbol{\omega}_{r_i} = \begin{pmatrix} 0 & 0 & \dot{\theta}_{m,i} \end{pmatrix}^{\mathrm{T}} , \quad i = 1, \ldots, N ,$$

instead of the complete expression (11.3).

As a result, the total angular kinetic energy of the rotors is just $\frac{1}{2}\dot{\boldsymbol{\theta}}^{\mathrm{T}}\mathbf{B}\dot{\boldsymbol{\theta}}$ (or $\mathbf{S} \equiv \mathbf{0}$), and the dynamic model (11.7) reduces to

$$\begin{aligned} \mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) + \mathbf{K}(\boldsymbol{q} - \boldsymbol{\theta}) &= \mathbf{0} \\ \mathbf{B}\ddot{\boldsymbol{\theta}} + \mathbf{K}(\boldsymbol{\theta} - \boldsymbol{q}) &= \boldsymbol{\tau} , \end{aligned} \qquad (11.9)$$

with $\mathbf{M}(\boldsymbol{q}) = \mathbf{M}_{\mathrm{L}}(\boldsymbol{q}) + \mathbf{M}_{\mathrm{R}}(\boldsymbol{q})$. The main feature of this model is that the link and motor equations are dynamically coupled only through the elastic torque $\boldsymbol{\tau}_J$. In addition, the motor equations are now fully linear.

We note that the complete model (11.7) and the reduced model (11.9) display different characteristics with respect to certain control problems. In fact, the reduced model is always feedback linearizable by *static* state feedback, whereas this is never the case for the complete model as soon as a coupling $\mathbf{S} \neq \mathbf{0}$ is present.

Singular Perturbation Model

It is interesting to show the *two-time-scale* dynamic behavior that it is induced in robots with elastic joints when the joint stiffness $\mathbf{K}$ is relatively large, but still finite. This behavior can be made explicit by a simple linear change of coordinates, namely replacing $\boldsymbol{\theta}$ with the joint torque $\boldsymbol{\tau}_{\mathrm{J}}$. For the sake of simplicity, this is illustrated on the reduced model only (without dissipation).

Since the diagonal joint stiffness matrix is assumed to have large but similar elements, a common scalar factor $1/\epsilon^2 \gg 1$ can be extracted as

$$\mathbf{K} = \frac{1}{\epsilon^2}\hat{\mathbf{K}} = \frac{1}{\epsilon^2}\,\mathrm{diag}\left(\hat{K}_1, \ldots, \hat{K}_N\right) .$$

The *slow* subsystem is then given by the link equations, once they are rewritten as

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}_{\mathrm{J}} . \qquad (11.10)$$

To obtain the dynamics of the *fast* subsystem, the joint torque is differentiated twice, the motor and link accelerations are replaced from (11.9), and the above definition of $\hat{\mathbf{K}}$ is used. This leads to

$$\begin{aligned} \epsilon^2\ddot{\boldsymbol{\tau}}_{\mathrm{J}} = \hat{\mathbf{K}} \{ & \mathbf{B}^{-1}\boldsymbol{\tau} - \left[\mathbf{B}^{-1} + \mathbf{M}^{-1}(\boldsymbol{q})\right]\boldsymbol{\tau}_{\mathrm{J}} \\ & + \mathbf{M}^{-1}(\boldsymbol{q})[\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})] \} . \end{aligned} \qquad (11.11)$$

For small ϵ, (11.10) and (11.11) represent a singularly perturbed system. The two separate time scales governing the slow and fast dynamics are t and $\sigma = t/\epsilon$, since

$$\epsilon^2\,\ddot{\boldsymbol{\tau}}_{\mathrm{J}} = \epsilon^2\,\frac{\mathrm{d}^2\boldsymbol{\tau}_{\mathrm{J}}}{\mathrm{d}t^2} = \frac{\mathrm{d}^2\boldsymbol{\tau}_{\mathrm{J}}}{\mathrm{d}\sigma^2} .$$

This model serves as a basis for composite control schemes, where the control torque has the general form

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{\mathrm{s}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) + \epsilon\boldsymbol{\tau}_{\mathrm{f}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau}_{\mathrm{J}}, \dot{\boldsymbol{\tau}}_{\mathrm{J}}) . \qquad (11.12)$$

This includes a slow action $\boldsymbol{\tau}_{\mathrm{s}}$, designed when neglecting joint elasticity, and an additional action $\boldsymbol{\tau}_{\mathrm{f}}$ for locally stabilizing the fast flexible dynamics around a suitable manifold in the state space. It can be verified that, when setting $\epsilon = 0$ in (11.10)–(11.12), the equivalent rigid robot model is recovered as

$$[\mathbf{M}(\boldsymbol{q}) + \mathbf{B}]\,\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}_{\mathrm{s}} .$$

A similar singular perturbation model (and control design) can also be derived for robot manipulators with flexible links.

11.1.2 Inverse Dynamics

Given a desired motion of a robot, we wish to compute the nominal torque needed to reproduce this motion in ideal conditions (the inverse dynamics problem). Such nominal torque can be used as the feedforward term in a trajectory-tracking control law.

For rigid robots, the inverse dynamics is a straightforward algebraic computation obtained by replacing the desired motion of the generalized coordinates in the dynamic model. The minimum requirement for exact

reproduction is that the planned motion has a continuously differentiable desired velocity. For robots with elastic joints, a motion task can be conveniently expressed in terms of a desired link trajectory $\boldsymbol{q} = \boldsymbol{q}_\mathrm{d}(t)$ (possibly obtained from the kinematic inversion of a desired motion in Cartesian space). The additional complexity lies in the fact that not all robot coordinates are directly assigned in this way, so that additional derivations should be performed. This will require some higher degree of smoothness of the desired trajectory $\boldsymbol{q}_\mathrm{d}(t) \in [0, T]$, where the final time T may or may not be finite.

Reduced Model

Consider first the case of the reduced model (11.7) and set $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})$ for compactness. By evaluating the link equations on the desired link motion

$$\mathbf{M}(\boldsymbol{q}_\mathrm{d})\ddot{\boldsymbol{q}}_\mathrm{d} + \boldsymbol{n}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) + \mathbf{K}\boldsymbol{q}_\mathrm{d} = \mathbf{K}\boldsymbol{\theta}_\mathrm{d}\,, \tag{11.13}$$

the nominal position $\boldsymbol{\theta}_\mathrm{d}$ of the motors associated with the desired link motion is readily obtained. The nominal elastic torque at the joints is $\boldsymbol{\tau}_{\mathrm{J,d}} = \mathbf{K}(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{q}_\mathrm{d})$; note that, from (11.13), this quantity can be expressed as a function of $\boldsymbol{q}_\mathrm{d}$, $\dot{\boldsymbol{q}}_\mathrm{d}$, and $\ddot{\boldsymbol{q}}_\mathrm{d}$, which is independent of $\mathbf{K}$. Differentiating (11.13) leads to the expression for the nominal motor velocity $\dot{\boldsymbol{\theta}}_\mathrm{d}$,

$$\mathbf{M}(\boldsymbol{q}_\mathrm{d})\boldsymbol{q}_\mathrm{d}^{[3]} + \dot{\mathbf{M}}(\boldsymbol{q}_\mathrm{d})\ddot{\boldsymbol{q}}_\mathrm{d} + \dot{\boldsymbol{n}}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) + \mathbf{K}\dot{\boldsymbol{q}}_\mathrm{d} = \mathbf{K}\dot{\boldsymbol{\theta}}_\mathrm{d}\,, \tag{11.14}$$

where the notation $\boldsymbol{y}^{[i]} = \mathrm{d}^i\boldsymbol{y}/\mathrm{d}t^i$ is used. Differentiating once more, we obtain

$$\begin{aligned}\mathbf{M}(\boldsymbol{q}_\mathrm{d})\boldsymbol{q}_\mathrm{d}^{[4]} + 2\dot{\mathbf{M}}(\boldsymbol{q}_\mathrm{d})\boldsymbol{q}_\mathrm{d}^{[3]} + \ddot{\boldsymbol{n}}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) \\ + \left[\ddot{\mathbf{M}}(\boldsymbol{q}_\mathrm{d}) + \mathbf{K}\right]\ddot{\boldsymbol{q}}_\mathrm{d} = \mathbf{K}\ddot{\boldsymbol{\theta}}_\mathrm{d}\,,\end{aligned} \tag{11.15}$$

to be used in the motor equations evaluated along the desired motion. After simplifications, the nominal torque is obtained as

$$\begin{aligned}\boldsymbol{\tau}_\mathrm{d} = {}& \left[\mathbf{M}(\boldsymbol{q}_\mathrm{d}) + \mathbf{B}\right]\ddot{\boldsymbol{q}}_\mathrm{d} + \boldsymbol{n}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) \\ & + \mathbf{B}\mathbf{K}^{-1}\left[\mathbf{M}(\boldsymbol{q}_\mathrm{d})\boldsymbol{q}_\mathrm{d}^{[4]} + 2\dot{\mathbf{M}}(\boldsymbol{q}_\mathrm{d})\boldsymbol{q}_\mathrm{d}^{[3]}\right. \\ & \left. + \ddot{\mathbf{M}}(\boldsymbol{q}_\mathrm{d})\ddot{\boldsymbol{q}}_\mathrm{d} + \ddot{\boldsymbol{n}}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right)\right],\end{aligned} \tag{11.16}$$

where the extra contributions to the nominal torque of the rigid case due to joint elasticity can be clearly recognized. The evaluation of $\boldsymbol{\tau}_\mathrm{d}$ involves the computation of first- and second-order partial derivatives of the dynamic model terms. For instance, one needs to compute

$$\dot{\mathbf{M}}[\boldsymbol{q}_\mathrm{d}(t)] = \sum_{i=1}^{N} \left.\frac{\partial \mathbf{M}_i(\boldsymbol{q})}{\partial \boldsymbol{q}}\right|_{\boldsymbol{q}=\boldsymbol{q}_\mathrm{d}(t)} \dot{\boldsymbol{q}}_\mathrm{d}(t)\boldsymbol{e}_i^\mathrm{T}\,,$$

where $\boldsymbol{e}_i$ is the i-th unit vector and $\boldsymbol{M}_i$ is the i-th column of matrix $\mathbf{M}(\boldsymbol{q})$. This and other similar expressions can be obtained by symbolic manipulation software. From (11.16), it follows that the minimum requirement for the exact reproducibility of the desired motion is that $\boldsymbol{q}_\mathrm{d}(t)$ admits a continuously differentiable jerk (i. e., that $\boldsymbol{q}_\mathrm{d}^{[4]}(t)$ exists in the time interval $[0, T]$). Such a smoother requirement should not come unexpected, in view of the flexible nature of the system.

Similarly to the rigid case, a recursive numerical Newton–Euler algorithm can be defined, which computes efficiently the inverse dynamics torque (11.16) in $O(N)$ asymptotic complexity. For the reduced model of robots with elastic joints, the algorithm involves forward recursion of motion variables up to the fourth differential order and backward recursion of second time derivatives of forces and torques.

Complete Model

Some more analysis is needed for the model (11.9). For ease of exposition, and without loss of generality, the case of a constant matrix $\mathbf{S}$ is considered. When evaluating the link equations on the desired link motion

$$\mathbf{M}(\boldsymbol{q}_\mathrm{d})\ddot{\boldsymbol{q}}_\mathrm{d} + \mathbf{S}\ddot{\boldsymbol{\theta}}_\mathrm{d} + \boldsymbol{n}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) + \mathbf{K}\boldsymbol{q}_\mathrm{d} = \mathbf{K}\boldsymbol{\theta}_\mathrm{d}\,, \tag{11.17}$$

the additional presence of the motor acceleration on the left-hand side does not allow solving directly for the motor position $\boldsymbol{\theta}_\mathrm{d}$ as a function of $(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ddot{\boldsymbol{q}}_\mathrm{d})$ only. However, the strictly upper triangular structure (11.5) of $\mathbf{S}$ allows one to define $\boldsymbol{\theta}_\mathrm{d}$ componentwise, using the scalar equations in (11.17) recursively. The N-th equation is in fact independent of $\ddot{\boldsymbol{\theta}}_\mathrm{d}$,

$$\mathbf{M}_N^\mathrm{T}(\boldsymbol{q}_\mathrm{d})\ddot{\boldsymbol{q}}_\mathrm{d} + \mathbf{0}^\mathrm{T}\ddot{\boldsymbol{\theta}}_\mathrm{d} + n_N\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) + K_N q_{\mathrm{d},N} = K_N\theta_{\mathrm{d},N}\,,$$

so that this equation can be used to define

$$\theta_{\mathrm{d},N} = f_N\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ddot{\boldsymbol{q}}_\mathrm{d}\right)$$

and, after double differentiation, its second time derivative

$$\ddot{\theta}_{\mathrm{d},N} = f_N''\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ldots, \boldsymbol{q}_\mathrm{d}^{[4]}\right).$$

In the $(N-1)$-th equation,

$$\begin{aligned}&\mathbf{M}_{N-1}^\mathrm{T}(\boldsymbol{q}_\mathrm{d})\ddot{\boldsymbol{q}}_\mathrm{d} + S_{N-1,N}\ddot{\theta}_{\mathrm{d},N} \\ &\quad + n_{N-1}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) + K_{N-1}q_{\mathrm{d},N-1} = K_{N-1}\theta_{\mathrm{d},N-1}\,,\end{aligned}$$

the acceleration $\ddot{\theta}_{\mathrm{d},N}$ has already been determined in the previous step. Therefore, this equation can be used similarly to define

$$\theta_{\mathrm{d},N-1} = f_{N-1}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ldots, \boldsymbol{q}_\mathrm{d}^{[4]}\right)$$

and, after two time differentiations, also

$$\ddot{\theta}_{\mathrm{d},N-1} = f''_{N-1}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ldots, \boldsymbol{q}_\mathrm{d}^{[6]}\right).$$

Note that whenever $S_{N-1,N} = 0$, there will be no increase in the degree of the highest order derivatives of $\boldsymbol{q}_\mathrm{d}$ within the functional dependence of $\theta_{\mathrm{d},N-1}$. This argument also applies recursively to the following steps. Proceeding backward through the link equations, the scalar computations end up with the definition of

$$\theta_{\mathrm{d},1} = f_1\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ldots, \boldsymbol{q}_\mathrm{d}^{[2N]}\right)$$

and

$$\ddot{\theta}_{\mathrm{d},1} = f''_1\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ldots, \boldsymbol{q}_\mathrm{d}^{[2(N+1)]}\right),$$

where the dependence on the highest possible differential degree of $\boldsymbol{q}_\mathrm{d}$ is shown. With $\ddot{\boldsymbol{\theta}}_\mathrm{d} = \boldsymbol{f}''(\cdot)$ made available in this way, the nominal torque is finally computed from the motor equations, again evaluated along the desired motion. Using (11.17) to substitute for $\mathbf{K}(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{q}_\mathrm{d})$ yields

$$\begin{aligned}\boldsymbol{\tau}_\mathrm{d} = &\left[\mathbf{M}(\boldsymbol{q}_\mathrm{d}) + \mathbf{S}^\mathrm{T}\right]\ddot{\boldsymbol{q}}_\mathrm{d} + \boldsymbol{n}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) \\ &+ (\mathbf{B} + \mathbf{S})\ \ddot{\boldsymbol{\theta}}_\mathrm{d}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \ldots, \boldsymbol{q}_\mathrm{d}^{[2(N+1)]}\right).\end{aligned} \tag{11.18}$$

As a result, the presence of inertial motor–link couplings considerably increases the complexity of the solution to the inverse dynamics problem. The exact reproduction of a desired link trajectory by the nominal torque in (11.18) requires that $\boldsymbol{q}_\mathrm{d}(t)$ has a continuously differentiable $(2N+1)$-th time derivative (i. e., that $\boldsymbol{q}_\mathrm{d}^{[2(N+1)]}(t)$ exists in the time interval $[0, T]$). For rest-to-rest motions, this implies a trajectory plan that imposes a very slow start and ending phases.

We finally remark that the possibility of expressing the evolution of the state and input of a system algebraically in terms of the evolution of an output variable (in our case $\boldsymbol{q}$) and a finite number of its derivatives is sometimes referred to as the *flatness* property. The above inverse dynamics analysis shows that $\boldsymbol{q}$ is a flat output for robots with elastic joints modeled either by (11.7) or (11.9).

Note also that, when a *constant* $\boldsymbol{q}_\mathrm{d}$ is considered, these computations all provide the same condition for the associated motor position

$$\boldsymbol{\theta}_\mathrm{d} = \boldsymbol{q}_\mathrm{d} + \mathbf{K}^{-1}\boldsymbol{g}(\boldsymbol{q}_\mathrm{d}) \tag{11.19}$$

and nominal static torque

$$\boldsymbol{\tau}_\mathrm{d} = \boldsymbol{g}(\boldsymbol{q}_\mathrm{d}). \tag{11.20}$$

Presence of Dissipative Terms

Inclusion of dissipative terms in the inverse dynamics deserves some additional comments. Any model of frictional effects acting on the motor side of the transmissions can be included in the computation without extra requirements. Friction at the link side should instead be described by a smooth model, because of the need to differentiate the link equations. Thus, some functional approximations may be needed (e.g., replacing discontinuous *sign* functions with hyperbolic tangents) before including this term in (11.13) or (11.17).

On the other hand, the presence of a nonnegligible spring damping $\mathbf{D}$ in (11.8) changes the structure of the computations. While this reduces the order of the derivatives of $\boldsymbol{q}_\mathrm{d}$ involved, the problem itself will become nonalgebraic; in fact, the inverse system will require the use of the solution of a dynamical system, albeit a simple one.

Consider for instance the model (11.9), including now all the dissipative terms given in (11.8). When evaluating the link equations,

$$\begin{aligned}\mathbf{M}(\boldsymbol{q}_\mathrm{d})\ddot{\boldsymbol{q}}_\mathrm{d} + \boldsymbol{n}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) + \left(\mathbf{D} + \mathbf{F}_q\right)\dot{\boldsymbol{q}}_\mathrm{d} + \mathbf{K}\boldsymbol{q}_\mathrm{d} \\ = \mathbf{D}\dot{\boldsymbol{\theta}}_\mathrm{d} + \mathbf{K}\boldsymbol{\theta}_\mathrm{d},\end{aligned} \tag{11.21}$$

the motor velocity $\dot{\boldsymbol{\theta}}_\mathrm{d}$ additionally appears on the right-hand side. Differentiating (11.21) leads to

$$\mathbf{D}\ddot{\boldsymbol{\theta}}_\mathrm{d} + \mathbf{K}\dot{\boldsymbol{\theta}}_\mathrm{d} = \boldsymbol{w}_\mathrm{d}, \tag{11.22}$$

with

$$\begin{aligned}\boldsymbol{w}_\mathrm{d} = &\mathbf{M}(\boldsymbol{q}_\mathrm{d})\boldsymbol{q}_\mathrm{d}^{[3]} + \left[\dot{\mathbf{M}}(\boldsymbol{q}_\mathrm{d}) + \mathbf{D} + \mathbf{F}_\mathrm{q}\right]\ddot{\boldsymbol{q}}_\mathrm{d} \\ &+ \dot{\boldsymbol{n}}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) + \mathbf{K}\dot{\boldsymbol{q}}_\mathrm{d}.\end{aligned}$$

Equation (11.22) represents a first-order linear *asymptotically stable* dynamical system (internal dynamics), with state $\dot{\boldsymbol{\theta}}_\mathrm{d}$ and forcing signal $\boldsymbol{w}_\mathrm{d}(t)$. For a given $\dot{\boldsymbol{\theta}}_\mathrm{d}(0)$, its solution $\dot{\boldsymbol{\theta}}_\mathrm{d}(t)$ and the associated solution derivative $\ddot{\boldsymbol{\theta}}_\mathrm{d}(t)$ are needed to evaluate the nominal torque in the motor equations. This yields

$$\boldsymbol{\tau}_\mathrm{d} = \mathbf{M}(\boldsymbol{q}_\mathrm{d})\ddot{\boldsymbol{q}}_\mathrm{d} + \boldsymbol{n}\left(\boldsymbol{q}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}\right) + \mathbf{F}_\mathrm{q}\dot{\boldsymbol{q}}_\mathrm{d} + \mathbf{B}\ddot{\boldsymbol{\theta}}_\mathrm{d} + \mathbf{F}_{\boldsymbol{\theta}}\dot{\boldsymbol{\theta}}_\mathrm{d},$$

where (11.21) has been used to replace the term $\mathbf{D}(\dot{\boldsymbol{\theta}}_\mathrm{d} - \dot{\boldsymbol{q}}_\mathrm{d}) + \mathbf{K}(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{q}_\mathrm{d})$. In this case, the desired link trajectory $\boldsymbol{q}_\mathrm{d}(t)$ should have a continuously differentiable acceleration ($\boldsymbol{q}_\mathrm{d}^{[3]}$ should exist, since it is used in the definition of $\boldsymbol{w}_\mathrm{d}$). Note that any initialization $\dot{\boldsymbol{\theta}}_\mathrm{d}(0)$ of the internal dynamics (11.22) is feasible (i. e., it produces a specific torque profile $\boldsymbol{\tau}_\mathrm{d}(t)$ yielding the same link

motion $\boldsymbol{q}_{\mathrm{d}}(t)$), with the associated motor position $\boldsymbol{\theta}_{\mathrm{d}}(0)$ initialized from (11.21). Indeed, we should match the actual initial state of the robot. For instance, starting from an equilibrium state implies the unique choice $\dot{\boldsymbol{\theta}}_{\mathrm{d}}(0) = \mathbf{0}$.

A similar procedure can also be applied to the complete model (11.7) in the presence of spring damping. Again a dynamical inverse system will be needed, while the smoothness requirement on $\boldsymbol{q}_{\mathrm{d}}(t)$ would be even more dramatically reduced in that case.

We finally remark that, for robots with flexible links, the inverse dynamics problem gives also rise to an internal dynamics, independently of the presence of modal damping. Moreover, when specifying a desired motion for the tip of the flexible arm, the associated internal dynamics will be *unstable* and this critical issue has to be tackled to determine a feasible solution.

11.1.3 Regulation Control

We now consider the problem of controlling the motion of a robot with joint elasticity to a *constant* configuration. In this problem, no trajectory planning is involved and a feedback law should be found that achieves asymptotic stabilization of a desired closed-loop equilibrium. Global solutions (i. e., those valid when starting from any initial state) are indeed preferred.

From the analysis in the previous section, it should be clear that one needs to define only a constant reference $\boldsymbol{q}_{\mathrm{d}}$ (with $\dot{\boldsymbol{q}}_{\mathrm{d}}(t) \equiv \mathbf{0}$) for the link coordinates.

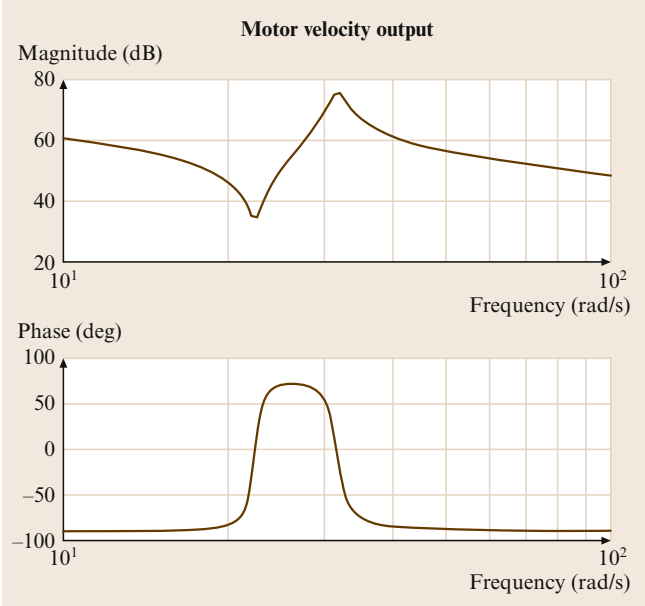


Fig. 11.6 Bode diagrams of the torque to motor velocity transfer function

From (11.19), a unique reference $\boldsymbol{\theta}_{\mathrm{d}}$ for the motor variables is in fact associated with the desired $\boldsymbol{q}_{\mathrm{d}}$ (which, in turn, may result from a desired pose of the robot end-effector). Furthermore, (11.20) provides the needed static torque to be applied at steady state by any feasible controller.

A major aspect of the presence of joint elasticity is that the feedback part of the control law can depend in general on *four* variables for each joint: the motor and link position, and the motor and link velocity. However, in most robots where joint elasticity is not explicitly considered in the system design, at most two sensors are available for measurement at the joints: a position sensor (e.g., an encoder) and, in some cases, a tachometer as a velocity sensor. When no velocity sensors are present, velocity is typically reconstructed by suitable numerical differentiation of high-resolution position measurements. Due to the presence of joint elasticity, the position/velocity quantities that are actually measured depend on where these sensors are mounted in the motor/transmission assembly.

A single link driven through an elastic joint is introduced as a paradigmatic case study to show what can be achieved using different sets of partial state measurements. This simple situation provides indications on how to handle the general multilink case.

In the absence of gravity, it will be shown that a proportional–derivative (PD) controller based only on motor measurements is sufficient to achieve the desired regulation task. In the presence of gravity, various gravity compensation schemes can be added to the PD feedback controller. These control results mimic the situation of robots with rigid joints, once the convenient quantities for use in the feedback have been identified.

Single Elastic Joint Example

Consider a single link rotating on a horizontal plane (thus, without gravity) and actuated with a motor through an elastic joint coupling (Fig. 11.4). When viscous friction on motor and link side as well as damping across the spring are included, the dynamic model is

$$M\ddot{q} + D(\dot{q} - \dot{\theta}) + K(q - \theta) + F_q\dot{q} = 0\,,$$
$$B\ddot{\theta} + D(\dot{\theta} - \dot{q}) + K(\theta - q) + F_\theta\dot{\theta} = \tau\,,$$

where the same, now scalar, notation of Sect. 11.1.1 has been used. Since this system is described by linear equations, Laplace transforms are used to derive the transfer functions of interest, namely from input torque to motor position

$$\frac{\theta(s)}{\tau(s)} = \frac{Ms^2 + (D + F_q)s + K}{\mathrm{den}(s)}\,,$$

and from input torque to link position

$$\frac{q(s)}{\tau(s)} = \frac{Ds + K}{\mathrm{den}(s)} ,$$

with the common denominator den(s) given by

$$\begin{aligned}\mathrm{den}(s) = \Big\{ & MB\, s^3 + \big[M(D + F_\theta) + B(D + F_q)\big]\, s^2 \\ & + \big[(M + B)K + (F_q + F_\theta)D + F_q F_\theta\big]\, s \\ & + (F_q + F_\theta)K \Big\} s .\end{aligned}$$

In the case of link position output, the transfer function has a larger relative degree (or pole-zero excess). Figures 11.6 and 11.7 show typical frequency responses in the two cases. For clarity, the outputs have been chosen at the velocity level. In the Bode diagram of the magnitude for the motor velocity output, note the presence of an antiresonance/resonance behavior. Similarly, there is a pure resonance (more pronounced for weak or zero spring damping D) for the link velocity output. The phase profile in Fig. 11.7, with a high-frequency lag of 270°, indicates the greater control difficulty encountered when closing the feedback loop on link quantities. In experimental tests on a robot joint, such plots are quite characteristic of the presence of concentrated joint elasticity and can be used to assess the relevance of this phenomenon and for identifying model parameters.

For the analysis of the stabilizing properties of different feedback arrangements, all dissipative effects will be neglected in the following ($D = F_q = F_\theta = 0$ is anyway the worst case). Consider first the transfer function relative to the motor position output

$$\left.\frac{\theta(s)}{\tau(s)}\right|_{\text{no diss}} = \frac{Ms^2 + K}{[MB\, s^2 + (M + B)K]\, s^2} . \qquad (11.23)$$

Apart from a double pole at the origin, this transfer function has a pair of imaginary zeros and poles, with the zero pair occurring at the so-called locked frequency

$$\omega_1 = \sqrt{\frac{K}{M}} ,$$

which characterizes the oscillations occurring when the motor is locked ($\theta \equiv 0$), e.g., by a high-gain positional feedback. This frequency is used to assess the performance limit of a simple PD control for the motor variables. In order to achieve enough damping in the closed-loop system, the bandwidth should be limited, as a rule of thumb, to one-third of ω_1. Faster transients can be achieved only by taking into account the fourth-order dynamics of the elastic joint assembly. Note also

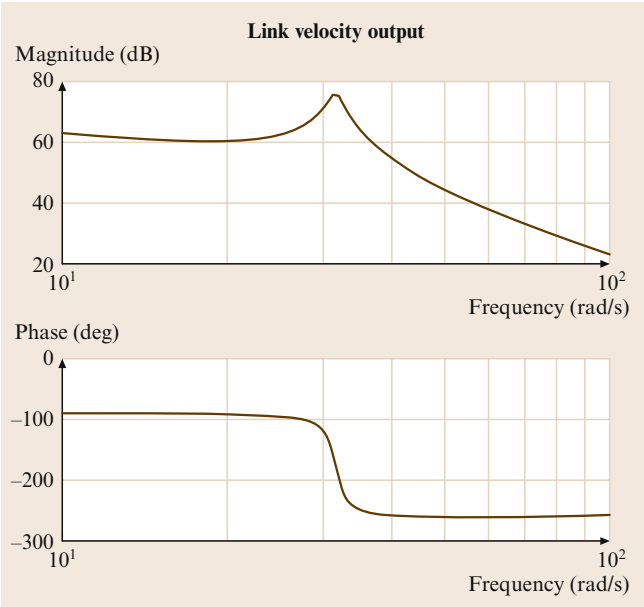


Fig. 11.7 Bode diagrams of the torque to link velocity transfer function

that the frequency of the zero pair is always *lower* than that of the pole pair in (11.23). This is related to the *passivity* of the mapping from torque to motor velocity, a property useful for stability and robust or adaptive control design.

Since the control objective is to regulate the link position output, we are also interested in the open-loop transfer function

$$\left.\frac{q(s)}{\tau(s)}\right|_{\text{no diss}} = \frac{K}{[MB\, s^2 + (M + B)K]\, s^2} . \qquad (11.24)$$

This transfer function has *no zeros* (in fact, for this to happen it is sufficient that $D = 0$) so that the relative degree is now four, the maximum possible. We shall see that the nonlinear counterpart of the absence of zeros in (11.24) will play a relevant role also for trajectory control in the multilink case of robots with elastic joints.

It is worth mentioning that this situation is completely different from the case in which the elastic spring would be moved out of the joint and placed anywhere along the link – a simple one-mode approximate model of link flexibility. In that case, the analogous transfer function will possess a negative and a positive real zero, symmetrically placed w.r.t. the origin (a *nonminimum-phase* system). This indicates the criticality of a direct inversion of the system input–output map for the execution of a desired link trajectory.

With the desired position given in terms of the link variable q_d, the most natural choice for the design of

a linear stabilizing feedback using one position and one velocity variable, is to close a PD loop from the link variables,

$$\tau = u_q - \left(K_{P,q} q + K_{D,q} \dot{q}\right) , \tag{11.25}$$

with position and velocity gains $K_{P,q}$ and $K_{D,q}$, and with $u_q = K_{P,q} q_\mathrm{d}$ being the external input used for defining the set point. It is easy to verify that the closed-loop poles will be unstable no matter how the gains are chosen, so that error feedback from link variables alone should be avoided. In a similar way, also the combination of motor position and link velocity feedback is always unstable.

Another mixed feedback strategy is to use link position and motor velocity

$$\tau = u_q - \left(K_{P,q} q + K_{D,\mathrm{m}} \dot{\theta}\right) . \tag{11.26}$$

This combination corresponds, e.g., to the case of a tachometer integrated in a direct-current (DC) motor and of an optical encoder placed on the load shaft for sensing its position (without any knowledge about the relevance of joint elasticity). Use of (11.26) leads to the closed-loop characteristic equation

$$\begin{aligned} &BMs^4 + MK_{D,\mathrm{m}} s^3 + (B+M)Ks^2 \\ &\quad + KK_{D,\mathrm{m}} s + KK_{P,q} = 0 . \end{aligned}$$

Using Routh's criterion, asymptotic stability occurs if and only if the motor velocity gain $K_{D,\mathrm{m}} > 0$ and the link position gain satisfies $0 < K_{P,q} < K$, i. e., the proportional feedback should not *override* the spring stiffness. The existence of such an upper bound limits the usefulness of this scheme.

Finally, consider performing feedback from the motor variables

$$\tau = u_\theta - \left(K_{P,\mathrm{m}} \theta + K_{D,\mathrm{m}} \dot{\theta}\right) , \tag{11.27}$$

with $u_\theta = K_{P,\mathrm{m}} \theta_d = K_{P,\mathrm{m}} q_d$ (due to the absence of gravity). The closed-loop system will be asymptotically stable provided that both $K_{P,\mathrm{m}}$ and $K_{D,\mathrm{m}}$ are strictly positive (and otherwise arbitrarily large). This favorable situation lends itself to a convenient generalization in the multilink case.

Note that other partial state feedback combinations would be possible, depending on the available sensing devices. For instance, mounting a strain gauge on the transmission shaft provides a direct measure of the elastic torque $\tau_\mathrm{J} = K(\theta - q)$ for control use. Strain gauges are also useful sensors for flexible links. Indeed, full-state feedback may be designed so as to guarantee asymptotic stability and improve the transient behavior considerably; however, this would be obtained at the cost of additional sensors and requires proper tuning of the four gains.

PD Control Using only Motor Variables

For the general multilink case in the absence of gravity, consider the PD control law based on motor position and velocity feedback

$$\boldsymbol{\tau} = \mathbf{K}_P(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta}) - \mathbf{K}_D \dot{\boldsymbol{\theta}} , \tag{11.28}$$

with symmetric (typically, diagonal) positive-definite gain matrices $\mathbf{K}_P$ and $\mathbf{K}_D$. Since $\boldsymbol{g}(\boldsymbol{q}) \equiv \mathbf{0}$, it follows from (11.19) that the reference value for the motor position is $\boldsymbol{\theta}_\mathrm{d} = \boldsymbol{q}_\mathrm{d}$ (neither a joint deflection is present nor an input torque is needed at steady state).

The control law (11.28) globally asymptotically stabilizes the desired equilibrium state $\boldsymbol{q} = \boldsymbol{\theta} = \boldsymbol{q}_\mathrm{d}$, $\dot{\boldsymbol{q}} = \dot{\boldsymbol{\theta}} = \mathbf{0}$. This can be shown through a Lyapunov argument, completed by the application of La Salle's theorem. In fact, a candidate Lyapunov function is given by the sum of the total energy of the system (kinetic plus elastic potential) and of the control energy due to the proportional term (a virtual elastic potential energy)

$$\begin{aligned} V = &\frac{1}{2} \dot{\boldsymbol{\Theta}}^\mathrm{T} \boldsymbol{\mathcal{M}}(\boldsymbol{\Theta}) \dot{\boldsymbol{\Theta}} + \frac{1}{2} (\boldsymbol{q} - \boldsymbol{\theta})^\mathrm{T} \mathbf{K} (\boldsymbol{q} - \boldsymbol{\theta}) \\ &+ \frac{1}{2} (\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta})^\mathrm{T} \mathbf{K}_P (\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta}) \geq 0 . \end{aligned} \tag{11.29}$$

Computing the time derivative of V along the trajectories of the closed-loop system given by (11.7) (or (11.9)) and (11.28), and taking into account the skew-symmetry of $\dot{\boldsymbol{\mathcal{M}}} - 2\boldsymbol{C}$, leads to

$$\dot{V} = -\dot{\boldsymbol{\theta}}^\mathrm{T} \mathbf{K}_D \dot{\boldsymbol{\theta}} \leq 0 .$$

The inclusion of dissipative terms (viscous friction and spring damping) would render $\dot{V}$ even more negative-semidefinite. The analysis is completed by verifying that the maximum invariant set contained in the set of states such that $\dot{V} = 0$ (i. e., those with $\dot{\boldsymbol{\theta}} = \mathbf{0}$) collapses into the desired unique equilibrium state.

We point out that an identical control law is able to globally regulate, in the absence of gravity, robots with flexible links to a desired joint configuration. In that case, $\boldsymbol{\theta}$ in (11.28) would be the rigid coordinates at the base of the flexible links of the robot.

PD with Constant Gravity Compensation

The presence of gravity requires the addition of some form of gravity compensation to the PD action (11.28).

Moreover, this will typically need an additional structural assumption and may demand some caution in the selection of the control gains.

Before proceeding, we recall a basic property of the gravity vector $\boldsymbol{g}(\boldsymbol{q})$ (under assumption **A2**, the gravity vector in (11.7) is the same appearing in the dynamics of the equivalent rigid robot). For robots with rotational joints, elastic or not, a positive constant α exists such that

$$\left\| \frac{\partial \boldsymbol{g}(\boldsymbol{q})}{\partial \boldsymbol{q}} \right\| \leq \alpha \,, \qquad \forall \boldsymbol{q} \in \mathbb{R}^N \,. \tag{11.30}$$

The norm of a matrix $\mathbf{A}(\boldsymbol{q})$ is that induced by the Euclidean norm for vectors, i. e.,

$$\|\mathbf{A}\| = \sqrt{\lambda_{\max}(\mathbf{A}^{\mathrm{T}}\mathbf{A})} \,.$$

Inequality (11.30) implies that

$$\|\boldsymbol{g}(\boldsymbol{q}_1) - \boldsymbol{g}(\boldsymbol{q}_2)\| \leq \alpha \|\boldsymbol{q}_1 - \boldsymbol{q}_2\| \,, \qquad \forall \boldsymbol{q}_1, \boldsymbol{q}_2 \in \mathbb{R}^N \,. \tag{11.31}$$

In common practice, robot joints are never unrealistically soft. More precisely, they are stiff enough to have, under the load of the robot's own weight, a *unique* equilibrium link position $\boldsymbol{q}_{\mathrm{e}}$ associated with any assigned motor position $\boldsymbol{\theta}_{\mathrm{e}}$ – the reverse of the relationship expressed by (11.19). This situation is by no means restrictive, but will be stated for clarity as a further modeling assumption:

A5 The lowest joint stiffness is larger than the upper bound on the gradient of gravity forces acting on the robot, or

$$\min_{i=1,\dots,N} K_i > \alpha \,.$$

The simplest modification for handling the presence of gravity is to consider the addition of a constant term that compensates exactly the gravity load at the desired steady state. According to (11.19) and (11.20), the control law (11.28) is then modified to

$$\boldsymbol{\tau} = \mathbf{K}_P(\boldsymbol{\theta}_{\mathrm{d}} - \boldsymbol{\theta}) - \mathbf{K}_D\dot{\boldsymbol{\theta}} + \boldsymbol{g}(\boldsymbol{q}_{\mathrm{d}}) \,, \tag{11.32}$$

with symmetric, typically diagonal, $\mathbf{K}_P > \mathbf{0}$ (as a minimum) and $\mathbf{K}_D > \mathbf{0}$, and with the motor reference given by $\boldsymbol{\theta}_{\mathrm{d}} = \boldsymbol{q}_{\mathrm{d}} + \mathbf{K}^{-1}\boldsymbol{g}(\boldsymbol{q}_{\mathrm{d}})$.

A sufficient condition guaranteeing that $\boldsymbol{q} = \boldsymbol{q}_{\mathrm{d}}$, $\boldsymbol{\theta} = \boldsymbol{\theta}_{\mathrm{d}}$, $\dot{\boldsymbol{q}} = \dot{\boldsymbol{\theta}} = \mathbf{0}$ will be the unique globally asymptotically stable equilibrium for the system (11.7) under the control law (11.32) is that

$$\lambda_{\min}\left[\begin{pmatrix} \mathbf{K} & -\mathbf{K} \\ -\mathbf{K} & \mathbf{K}+\mathbf{K}_P \end{pmatrix}\right] > \alpha \,, \tag{11.33}$$

with α defined in (11.30). Taking into account the diagonal structure of $\mathbf{K}$ and $\mathbf{K}_P$, and thanks to assumption **A5**, it is always possible to fulfill this condition by increasing the smallest proportional gain in the controller (or, the smallest eigenvalue of $\mathbf{K}_P$ if this matrix has not been chosen diagonal).

In the following, we sketch the motivation for this condition and the associated proof of asymptotic stability. The equilibrium configurations of the closed-loop system are the solutions of

$$\begin{aligned} \mathbf{K}(\boldsymbol{q} - \boldsymbol{\theta}) + \boldsymbol{g}(\boldsymbol{q}) &= \mathbf{0} \,, \\ \mathbf{K}(\boldsymbol{\theta} - \boldsymbol{q}) - \mathbf{K}_P(\boldsymbol{\theta}_{\mathrm{d}} - \boldsymbol{\theta}) - \boldsymbol{g}(\boldsymbol{q}_{\mathrm{d}}) &= \mathbf{0} \,. \end{aligned}$$

Indeed, the pair $(\boldsymbol{q}_{\mathrm{d}}, \boldsymbol{\theta}_{\mathrm{d}})$ satisfies these equations. However, in order to obtain a global result one should guarantee that this pair is the unique solution. Therefore, recalling (11.19), the null term $\mathbf{K}(\boldsymbol{\theta}_{\mathrm{d}} - \boldsymbol{q}_{\mathrm{d}}) - \boldsymbol{g}(\boldsymbol{q}_{\mathrm{d}})$ can be added/subtracted to both equations to obtain

$$\begin{aligned} \mathbf{K}(\boldsymbol{q} - \boldsymbol{q}_{\mathrm{d}}) - \mathbf{K}(\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{d}}) &= \boldsymbol{g}(\boldsymbol{q}_{\mathrm{d}}) - \boldsymbol{g}(\boldsymbol{q}) \\ -\mathbf{K}(\boldsymbol{q} - \boldsymbol{q}_{\mathrm{d}}) + (\mathbf{K} + \mathbf{K}_P)(\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{d}}) &= \mathbf{0} \,, \end{aligned}$$

where the matrix in condition (11.33) can be clearly recognized. Taking the norms of terms on both sides and bounding gravity terms using (11.31), the introduced condition (11.33) implies that the pair $(\boldsymbol{q}_{\mathrm{d}}, \boldsymbol{\theta}_{\mathrm{d}})$ is in fact the unique equilibrium. To show asymptotic stability, a candidate Lyapunov function is built based on the one introduced in (11.29) in the absence of gravity

$$\begin{aligned} V_{\mathrm{g1}} = V &+ \mathcal{U}_{\mathrm{grav}}(\boldsymbol{q}) - \mathcal{U}_{\mathrm{grav}}(\boldsymbol{q}_{\mathrm{d}}) - (\boldsymbol{q} - \boldsymbol{q}_{\mathrm{d}})^{\mathrm{T}}\boldsymbol{g}(\boldsymbol{q}_{\mathrm{d}}) \\ &- \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}(\boldsymbol{q}_{\mathrm{d}})\mathbf{K}^{-1}\boldsymbol{g}(\boldsymbol{q}_{\mathrm{d}}) \geq 0 \,. \end{aligned} \tag{11.34}$$

The positive-definiteness of V_{g1} and the fact that its unique minimum is at the desired state are again implied by condition (11.33). (The last constant term in (11.34) is used to set to zero the minimum value of V_{g1} at the equilibrium.). By the usual computations, it follows that $\dot{V}_{\mathrm{g1}} = -\dot{\boldsymbol{\theta}}^{\mathrm{T}}\mathbf{K}_D\dot{\boldsymbol{\theta}} \leq 0$ and the result is obtained applying La Salle's theorem.

The control law (11.32) is based only on the knowledge of the gravity term $\boldsymbol{g}(\boldsymbol{q}_{\mathrm{d}})$ and of the joint stiffness $\mathbf{K}$. The latter appears in fact in the definition of the motor position reference $\boldsymbol{\theta}_{\mathrm{d}}$. Uncertainties in the gravitational term $\boldsymbol{g}(\boldsymbol{q}_{\mathrm{d}})$ and in the joint stiffness $\mathbf{K}$ affect the performance of the controller. Still, the existence of a unique closed-loop equilibrium and its asymptotic stability are preserved when the gravity bound by α is still correct and condition (11.33) holds for the true stiffness

values. Indeed, the robot will converge to an equilibrium that is different from the desired one – the better the estimates $\hat{\mathbf{K}}$ and $\hat{\boldsymbol{g}}(\boldsymbol{q}_\mathrm{d})$ used, the closer the actual equilibrium will be to the desired one.

PD with Online Gravity Compensation

Similarly to the rigid robot case, a better transient behavior is expected if gravity compensation (or, more precisely, its perfect cancellation) is performed at all configurations during motion. However, the gravity vector in (11.7) depends on the link variables $\boldsymbol{q}$, which are assumed not to be measurable at this stage. It is easy to see that using $\boldsymbol{g}(\boldsymbol{\theta})$, with the measured motor positions in place of the link positions, leads in general to an incorrect closed-loop equilibrium. Moreover, even if $\boldsymbol{q}$ were available, adding $\boldsymbol{g}(\boldsymbol{q})$ to a motor PD error feedback has no guarantee of success because this compensation, which appears in the motor equation, does not instantaneously cancel the gravity load acting on the links.

With this in mind, a PD control with *online* gravity compensation can be introduced as follows. Define the variable

$$\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta} - \mathbf{K}^{-1}\boldsymbol{g}(\boldsymbol{q}_\mathrm{d}) \,, \tag{11.35}$$

which is a *gravity-biased* modification of the measured motor position $\boldsymbol{\theta}$, and let

$$\boldsymbol{\tau} = \mathbf{K}_P\,(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta}) - \mathbf{K}_D\dot{\boldsymbol{\theta}} + \boldsymbol{g}(\tilde{\boldsymbol{\theta}}) \,, \tag{11.36}$$

where $\mathbf{K}_P > \mathbf{0}$ and $\mathbf{K}_D > \mathbf{0}$ are both symmetric (and typically diagonal) matrices. The control law (11.36) can still be implemented using only motor variables. The term $\boldsymbol{g}(\tilde{\boldsymbol{\theta}})$ provides only an approximate cancellation (though, of a large part) of gravity during motion, but leads to the correct gravity compensation at steady state. In fact, by using (11.19) and (11.35), it follows that

$$\tilde{\boldsymbol{\theta}}_\mathrm{d} := \boldsymbol{\theta}_\mathrm{d} - \mathbf{K}^{-1}\boldsymbol{g}(\boldsymbol{q}_\mathrm{d}) = \boldsymbol{q}_\mathrm{d} \,,$$

so that $\boldsymbol{g}(\tilde{\boldsymbol{\theta}}_\mathrm{d}) = \boldsymbol{g}(\boldsymbol{q}_\mathrm{d})$.

Global asymptotic stability of the desired equilibrium can be guaranteed under the same condition (11.33) used for the constant gravity compensation case. A slightly different Lyapunov candidate is defined, starting again from the one in (11.29), as

$$V_{\mathrm{g}2} = V + \mathcal{U}_\mathrm{grav}(\boldsymbol{q}) - \mathcal{U}_\mathrm{grav}(\tilde{\boldsymbol{\theta}}) - \frac{1}{2}\boldsymbol{g}^\mathrm{T}(\boldsymbol{q}_\mathrm{d})\mathbf{K}^{-1}\boldsymbol{g}(\boldsymbol{q}_\mathrm{d}) \geq 0 \,,$$

to be compared with (11.34).

The use of an online gravity compensation scheme typically provides a smoother time course and a noticeable reduction of the positional transient errors, with no additional control effort in terms of peak and average torques. We note that choices of low position gains, even when largely violating the sufficient condition (11.33) for stability, may still work, contrary to the case of constant gravity compensation. However even for increasing values of joint stiffness, in the limit for $\mathbf{K} \to \infty$, the range of feasible values of $\mathbf{K}_P$ that guarantee exact regulation does not extend down to zero.

A possibility to refine the previous online gravity compensation scheme, again based only on motor position measurement, is offered by the use of a fast iterative algorithm that elaborates the measure $\boldsymbol{\theta}$ in order to generate a quasistatic estimate $\bar{\boldsymbol{q}}(\boldsymbol{\theta})$ of the current (unmeasured) $\boldsymbol{q}$. In fact, in any steady-state configuration $(\boldsymbol{q}_\mathrm{s}, \boldsymbol{\theta}_\mathrm{s})$, a direct mapping is defined from $\boldsymbol{q}_\mathrm{s}$ to $\boldsymbol{\theta}_\mathrm{s}$

$$\boldsymbol{\theta}_\mathrm{s} = \boldsymbol{h}_\mathrm{g}(\boldsymbol{q}_\mathrm{s}) := \boldsymbol{q}_\mathrm{s} + \mathbf{K}^{-1}\boldsymbol{g}(\boldsymbol{q}_\mathrm{s}) \,.$$

Assumption **A5** is sufficient to guarantee the existence and uniqueness also of the inverse mapping $\boldsymbol{q}_\mathrm{s} = \boldsymbol{h}_\mathrm{g}^{-1}(\boldsymbol{\theta}_\mathrm{s})$. For a measured $\boldsymbol{\theta}$, the function

$$\boldsymbol{q} = \boldsymbol{T}(\boldsymbol{q}) := \boldsymbol{\theta} - \mathbf{K}^{-1}\boldsymbol{g}(\boldsymbol{q})$$

is then a *contraction mapping* and the iteration

$$\boldsymbol{q}_{i+1} = \boldsymbol{T}(\boldsymbol{q}_i) \,, \quad i = 0, 1, 2, \ldots$$

will converge to the unique fixed point of this mapping, which is exactly the value $\bar{\boldsymbol{q}}(\boldsymbol{\theta}) = \boldsymbol{h}_\mathrm{g}^{-1}(\boldsymbol{\theta})$. A suitable initialization for $\boldsymbol{q}_0$ is either the measured $\boldsymbol{\theta}$ or the value $\bar{\boldsymbol{q}}$ computed at the previous sampling instant. In this way, just two or three iterations are needed to obtain sufficient accuracy and this is fast enough to be implemented within a sensing/control sampling interval of a digital robot controller. With this iterative scheme running in the background, the regulation control law becomes

$$\boldsymbol{\tau} = \mathbf{K}_P\,(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta}) - \mathbf{K}_D\dot{\boldsymbol{\theta}} + \boldsymbol{g}\,(\bar{\boldsymbol{q}}(\boldsymbol{\theta})) \,, \tag{11.37}$$

with symmetric (diagonal) $\mathbf{K}_P > \mathbf{0}$ and $\mathbf{K}_D > \mathbf{0}$. A proof of the global asymptotic stability of this control scheme can be given by further modifying the previous Lyapunov candidates. The advantage of (11.37) is that any positive value of the feedback gain $\mathbf{K}_P$ is allowed, thus recovering in full the operative working conditions of the rigid case with exact gravity cancellation.

With reference to the reduced model (11.9), a further improvement can be obtained via a nonlinear PD-like controller that removes completely the gravity from

the actual motion of the robot links. For this, consider the control law

$$\boldsymbol{\tau} = \mathbf{K}_P[\mathbf{q}_d - \boldsymbol{\theta} + \mathbf{K}^{-1}\boldsymbol{g}(\boldsymbol{q})] - \mathbf{K}_D\left(\dot{\boldsymbol{\theta}} - \mathbf{K}^{-1}\dot{\boldsymbol{g}}(\boldsymbol{q})\right) + \boldsymbol{g}(\boldsymbol{q}) + \mathbf{B}\mathbf{K}^{-1}\ddot{\boldsymbol{g}}(\boldsymbol{q}) \tag{11.38}$$

with symmetric (diagonal) $\mathbf{K}_P > 0$ and $\mathbf{K}_D > 0$. The last two terms are those needed to cancel both static and dynamic effects due to gravity. Even when assumption **A5** does not hold, the control law (11.38) guarantees global asymptotic stability of the desired equilibrium state. This result is a byproduct of feedback linearization design (Sect.11.1.4), though much simpler than the complete one, and can be proven using again a Lyapunov analysis. Since a lower bound on the joint stiffness is not required (the joints may be arbitrarily soft), the same approach can be used also for robots using variable stiffness actuation (VSA). The price to pay is that an estimate of the link acceleration $\ddot{\boldsymbol{q}}$ is needed when computing the term $\ddot{\boldsymbol{g}}$ in (11.38).

The two online gravity compensation schemes (11.36) and (11.37) realize a *compliance control* in the joint space with only motor measurements. The same idea can also be extended to the case of Cartesian compliance control by evaluating the direct kinematics and the Jacobian (transpose) of the robot arm with $\tilde{\boldsymbol{\theta}}$ or, respectively, $\bar{\boldsymbol{q}}(\boldsymbol{\theta})$ in place of $\boldsymbol{q}$. The role of a damping action within a compliance/impedance control scheme can be clearly understood when looking at VIDEO 133 and VIDEO 134.

Full-State Feedback

When the feedback law is based on a full set of measurements of the robot state, the transient performance of regulation control laws can be improved. Taking advantage of the availability of a joint torque sensor, we present a convenient design for the reduced model (11.9) of robots with elastic joints, including spring damping as a dissipation effect. Full-state feedback can be obtained in two phases, by combining a preliminary torque feedback with the motor feedback law in (11.28).

Using $\boldsymbol{\tau}_\mathrm{J} = \mathbf{K}(\boldsymbol{\theta} - \boldsymbol{q})$, the motor equation can be rewritten as

$$\mathbf{B}\ddot{\boldsymbol{\theta}} + \boldsymbol{\tau}_\mathrm{J} + \mathbf{D}\mathbf{K}^{-1}\dot{\boldsymbol{\tau}}_\mathrm{J} = \boldsymbol{\tau} \,.$$

The joint torque feedback

$$\boldsymbol{\tau} = \mathbf{B}\mathbf{B}_\theta^{-1}\boldsymbol{u} + \left(\mathbf{I} - \mathbf{B}\mathbf{B}_\theta^{-1}\right)\left(\boldsymbol{\tau}_\mathrm{J} + \mathbf{D}\mathbf{K}^{-1}\dot{\boldsymbol{\tau}}_\mathrm{J}\right) \,, \tag{11.39}$$

where $\boldsymbol{u}$ is an auxiliary input to be designed, transforms the motor equations into

$$\mathbf{B}_\theta\ddot{\boldsymbol{\theta}} + \boldsymbol{\tau}_\mathrm{J} + \mathbf{D}\mathbf{K}^{-1}\dot{\boldsymbol{\tau}}_\mathrm{J} = \boldsymbol{u} \,.$$

In this way, the apparent motor inertia can be reduced to a desired, arbitrary small value $\mathbf{B}_\theta$, with clear benefits in terms of vibration damping. For instance in the linear and scalar case, a very small B shifts the pair of complex poles in (11.23) to a very high frequency, with the joint behaving almost as a rigid one.

Setting now

$$\boldsymbol{u} = \mathbf{K}_{P,\theta}(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta}) - \mathbf{K}_{D,\theta}\dot{\boldsymbol{\theta}} + \boldsymbol{g}(\boldsymbol{q}_\mathrm{d})$$

in (11.39) leads to the state feedback controller

$$\boldsymbol{\tau} = \mathbf{K}_P(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta}) - \mathbf{K}_D\dot{\boldsymbol{\theta}} + \mathbf{K}_T[\boldsymbol{g}(\boldsymbol{q}_\mathrm{d}) - \boldsymbol{\tau}_\mathrm{J}] - \mathbf{K}_S\dot{\boldsymbol{\tau}}_\mathrm{J} + \boldsymbol{g}(\boldsymbol{q}_\mathrm{d}) \,, \tag{11.40}$$

with gains

$$\begin{aligned}
\mathbf{K}_P &= \mathbf{B}\mathbf{B}_\theta^{-1}\mathbf{K}_{P,\theta} \,, \\
\mathbf{K}_D &= \mathbf{B}\mathbf{B}_\theta^{-1}\mathbf{K}_{D,\theta} \,, \\
\mathbf{K}_T &= \mathbf{B}\mathbf{B}_\theta^{-1} - \mathbf{I} \\
\mathbf{K}_S &= \left(\mathbf{B}\mathbf{B}_\theta^{-1} - \mathbf{I}\right)\mathbf{D}\mathbf{K}^{-1} \,.
\end{aligned}$$

Indeed, the law (11.40) can also be rewritten in terms of $(\boldsymbol{\theta}, \boldsymbol{q}, \dot{\boldsymbol{\theta}}, \dot{\boldsymbol{q}})$, if the torque sensor is not available. However, keeping this structure of the gains preserves the interesting physical interpretation of what the full-state feedback controller is able to achieve.

11.1.4 Trajectory Tracking

As for rigid robots, the problem of tracking desired time-varying trajectories for robots with elastic joints is harder than that of achieving constant regulation. In general, solving this problem requires the use of full-state feedback and knowledge of all the terms in the dynamic model.

Under these conditions, we shall focus on the *feedback linearization* approach, namely a nonlinear state feedback law that leads to a closed-loop system with decoupled and exactly linear behavior for all the N joints of the robot (in fact, the link variables $\boldsymbol{q}$). The tracking errors along the reference trajectory are forced to be *globally exponentially stable*, with a decaying rate that can be directly specified through the choice of the scalar feedback gains in the controller. This fundamental result is the direct extension of the well-known *computed torque* method for rigid robots. Because of its relevant properties, the feedback linearization approach

can be used as a reference to assess the performance of any other trajectory tracking control law, which may possibly be designed using less/approximate model information and/or only partial-state feedback.

However, in the presence of joint elasticity, the design of a feedback linearization law is not straightforward. Furthermore, as soon as $\mathbf{S} \neq \mathbf{0}$, the dynamic model (11.7) will not satisfy the necessary conditions for exact linearization (nor those for input–output decoupling) when only a *static* (or, instantaneous) feedback law from the full state is allowed. Therefore, we will restrict our attention to the more tractable case of the reduced dynamic model (11.9) and sketch only briefly the more general picture.

As a second much simpler approach to trajectory tracking problems, we also present a *linear control design* that makes use of a model-based feedforward command and a precomputed state reference trajectory, as obtained from the inverse dynamics in Sect. 11.1.2, with the addition of a linear feedback from the full state. In this case, convergence to the desired trajectory is only locally guaranteed, i. e., the tracking errors should be small enough, but the control implementation is straightforward and the real-time computational burden is considerably reduced.

Feedback Linearization

Consider the reduced model (11.9) and let the desired motion be specified by a smooth reference trajectory $\boldsymbol{q}_{\mathrm{d}}(t)$ for the robot links. The control design will proceed by system inversion, in a similar way to the inverse dynamics computations of Sect. 11.1.2 but using now the current measures of the state variables $(\boldsymbol{q}, \boldsymbol{\theta}, \dot{\boldsymbol{q}}, \dot{\boldsymbol{\theta}})$ instead of the reference state evolution $(\boldsymbol{q}_{\mathrm{d}}, \boldsymbol{\theta}_{\mathrm{d}}, \dot{\boldsymbol{q}}_{\mathrm{d}}, \dot{\boldsymbol{\theta}}_{\mathrm{d}})$. Notably, there is no need to transform the robot equations into their state-space description, which is the standard form used in control design for general nonlinear systems; we will use the robot model directly in its second-order differential form (typical of mechanical systems).

The outcome of the inversion procedure will be the definition of a torque $\boldsymbol{\tau}$, in the form of a static state feedback control law, which cancels the original robot dynamics and replaces it with a desired linear and decoupled one of suitable differential order. In this sense, the control law *stiffens* the dynamics of the robot with elastic joints, irrespective of the fact that $\mathbf{K}$ is already relatively large (stiff joint) or small (soft joint). The feasibility of inverting the system from the chosen output $\boldsymbol{q}$ without causing instability problems (related to the presence of unobservable dynamics in the closed-loop system, after cancellation) is a relevant property of robots with elastic joints. In fact, this is the direct generalization to the nonlinear multiple-input multiple-output (MIMO) case of the possibility of inverting a scalar transfer function in the absence of zeros (see also the considerations made on (11.24)).

Rewrite the link equation in the compact form

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \mathbf{K}(\boldsymbol{q} - \boldsymbol{\theta}) = \mathbf{0}\,. \tag{11.41}$$

None of the above quantities depends instantaneously on the input torque $\boldsymbol{\tau}$. Therefore, we can differentiate once w.r.t. time and obtain

$$\mathbf{M}(\boldsymbol{q})\boldsymbol{q}^{[3]} + \dot{\mathbf{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{n}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \mathbf{K}(\dot{\boldsymbol{q}} - \dot{\boldsymbol{\theta}}) = \mathbf{0}\,. \tag{11.42}$$

Proceeding one step further leads to

$$\mathbf{M}(\boldsymbol{q})\boldsymbol{q}^{[4]} + 2\dot{\mathbf{M}}(\boldsymbol{q})\boldsymbol{q}^{[3]} + \ddot{\mathbf{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \ddot{\boldsymbol{n}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \mathbf{K}(\ddot{\boldsymbol{q}} - \ddot{\boldsymbol{\theta}}) = \mathbf{0}\,, \tag{11.43}$$

where $\ddot{\boldsymbol{\theta}}$ now appears. The motor acceleration is at the same differential level of $\boldsymbol{\tau}$ in the motor equation

$$\mathbf{B}\ddot{\boldsymbol{\theta}} + \mathbf{K}(\boldsymbol{\theta} - \boldsymbol{q}) = \boldsymbol{\tau}\,, \tag{11.44}$$

and thus, replacing $\ddot{\boldsymbol{\theta}}$ from (11.44), we get

$$\mathbf{M}(\boldsymbol{q})\boldsymbol{q}^{[4]} + 2\dot{\mathbf{M}}(\boldsymbol{q})\boldsymbol{q}^{[3]} + \ddot{\mathbf{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \ddot{\boldsymbol{n}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \mathbf{K}\ddot{\boldsymbol{q}} = \mathbf{K}\mathbf{B}^{-1}\left[\boldsymbol{\tau} - \mathbf{K}(\boldsymbol{\theta} - \boldsymbol{q})\right]\,. \tag{11.45}$$

We note that, using (11.41), the last term $\mathbf{K}(\boldsymbol{\theta} - \boldsymbol{q})$ in (11.45) may also be replaced by $\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})$.

Since the matrix $\mathbf{A}(\boldsymbol{q}) = \mathbf{M}^{-1}(\boldsymbol{q})\mathbf{K}\mathbf{B}^{-1}$ is always nonsingular, an arbitrary value $\boldsymbol{v}$ can be assigned to the fourth derivative of $\boldsymbol{q}$ by a suitable choice of the input torque $\boldsymbol{\tau}$. The matrix $\mathbf{A}(\boldsymbol{q})$ is the so-called decoupling matrix of the system and its nonsingularity is a necessary and sufficient condition for imposing a decoupled input–output behavior by nonlinear static state feedback. Moreover, (11.45) indicates that each component q_i of $\boldsymbol{q}$ needs to be differentiated $r_i = 4$ times in order to be algebraically related to the input torque $\boldsymbol{\tau}$ (r_i is the *relative degree* of q_i, when this is chosen as a system output). Since there are N link variables, the total sum of the relative degrees is $4N$, equal to the dimension of the state of a robot with elastic joints. All these facts taken together lead to the conclusion that, when inverting (11.45) to determine the input $\boldsymbol{\tau}$ that imposes $\boldsymbol{q}^{[4]} = \boldsymbol{v}$, there will be no dynamics left other than the one appearing in the closed-loop input–output map.

Therefore, choose

$$\boldsymbol{\tau} = \mathbf{B}\mathbf{K}^{-1}\left[\mathbf{M}(\boldsymbol{q})\boldsymbol{v} + \boldsymbol{\alpha}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{q}^{[3]}\right)\right] + \left[\mathbf{M}(\boldsymbol{q}) + \mathbf{B}\right]\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})\,, \tag{11.46}$$

with

$$\boldsymbol{\alpha}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}},\boldsymbol{q}^{[3]}) = \ddot{\mathbf{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + 2\dot{\mathbf{M}}(\boldsymbol{q})\boldsymbol{q}^{[3]} + \ddot{\boldsymbol{n}}(\boldsymbol{q},\dot{\boldsymbol{q}}) \,,$$

where the terms in $\boldsymbol{\alpha}$ have been ordered according to the dependence on increasing orders of derivatives of $\boldsymbol{q}$. It is easy to verify that the control law (11.46) leads to a closed-loop system fully described by

$$\boldsymbol{q}^{[4]} = \boldsymbol{v} \,, \tag{11.47}$$

i. e., chains of four input–output integrators from each auxiliary input v_i to each link position output q_i, for $i = 1, \ldots, N$. Thus, the robot system has been exactly linearized and decoupled by the nonlinear feedback law (11.46). The improvement in performance can be appreciated in a comparative way by looking at VIDEO 135 and VIDEO 770.

The complete control law (11.46) is expressed as a function of the so-called *linearizing* coordinates $(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}},\boldsymbol{q}^{[3]})$ only. This has led to some misunderstandings in the past, as it seemed that the feedback linearization approach for robots with elastic joints would need direct measures of the link acceleration $\ddot{\boldsymbol{q}}$ and jerk $\boldsymbol{q}^{[3]}$, which are impossible to obtain with currently available sensors (or would require multiple numerical differentiations of position measures in real time, with critical noise problems).

When considering the latest technology in the field, it is now feasible to have a set of sensors for elastic joints measuring in a reliable and accurate way the motor position $\boldsymbol{\theta}$ (and, possibly, also its velocity $\dot{\boldsymbol{\theta}}$), the joint torque $\boldsymbol{\tau}_\mathrm{J} = \mathbf{K}(\boldsymbol{\theta} - \boldsymbol{q})$, as well as the link position $\boldsymbol{q}$. For instance, this is the arrangement of sensors available at each joint of the *LWR-III* lightweight manipulator, where expressly designed high-resolution incremental encoders of the magneto-resistive type are used for the motor position, joint torque sensing is based on a full bridge of strain gauges, and a high-end capacitive potentiometer is used for the absolute link position (Fig. 11.8). Therefore, only one numerical differentiation is needed in order to obtain a good estimate of $\dot{\boldsymbol{q}}$ and/or $\dot{\boldsymbol{\tau}}_\mathrm{J}$ as well. Note that, depending on the specific sensor resolution, it may also be convenient to evaluate $\boldsymbol{q}$ using the measures of $\boldsymbol{\theta}$ and $\boldsymbol{\tau}_\mathrm{J}$ as $\boldsymbol{\theta} - \mathbf{K}^{-1}\boldsymbol{\tau}_\mathrm{J}$.

With this in mind, it is easy to see that the following three sets of $4N$ variables

$$(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}},\boldsymbol{q}^{[3]}) \,, \qquad (\boldsymbol{q},\boldsymbol{\theta},\dot{\boldsymbol{q}},\dot{\boldsymbol{\theta}}) \,, \qquad (\boldsymbol{q},\boldsymbol{\tau}_\mathrm{J},\dot{\boldsymbol{q}},\dot{\boldsymbol{\tau}}_\mathrm{J})$$

are all equivalent state variables for a robot with elastic joints, related by globally invertible transformations. Therefore, under the assumption that the dynamic

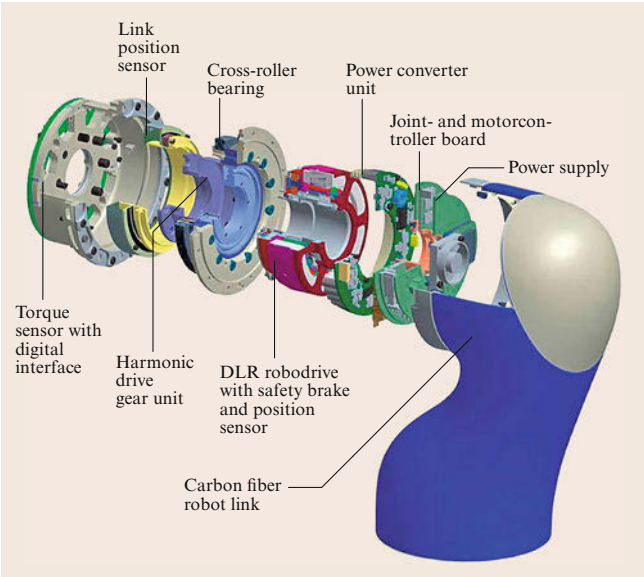


Fig. 11.8 Exploded view of a joint of the *DLR LWR-III* lightweight manipulator and its sensor suite

model is available, we can completely rewrite the feedback linearizing control law (11.46) in terms of the more *conventional* state $(\boldsymbol{q},\boldsymbol{\theta},\dot{\boldsymbol{q}},\dot{\boldsymbol{\theta}})$ or, taking advantage of a joint torque sensor, in terms of $(\boldsymbol{q},\boldsymbol{\tau}_\mathrm{J},\dot{\boldsymbol{q}},\dot{\boldsymbol{\tau}}_\mathrm{J})$. In particular, as a byproduct of (11.41) and (11.42), we have

$$\begin{aligned}\ddot{\boldsymbol{q}} &= \mathbf{M}^{-1}(\boldsymbol{q})\left[\mathbf{K}(\boldsymbol{\theta} - \boldsymbol{q}) - \boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}})\right] \\ &= \mathbf{M}^{-1}(\boldsymbol{q})\left[\boldsymbol{\tau}_\mathrm{J} - \boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}})\right] \,,\end{aligned} \tag{11.48}$$

and

$$\begin{aligned}\boldsymbol{q}^{[3]} &= \mathbf{M}^{-1}(\boldsymbol{q})\left[\mathbf{K}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}) - \dot{\mathbf{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} - \dot{\boldsymbol{n}}(\boldsymbol{q},\dot{\boldsymbol{q}})\right] \\ &= \mathbf{M}^{-1}(\boldsymbol{q})\left[\dot{\boldsymbol{\tau}}_\mathrm{J} - \dot{\mathbf{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} - \dot{\boldsymbol{n}}(\boldsymbol{q},\dot{\boldsymbol{q}})\right] \,,\end{aligned} \tag{11.49}$$

where the acceleration $\ddot{\boldsymbol{q}}$ appearing in (11.49) has already been computed through (11.48). Therefore, the exact linearizing and decoupling control law can be rewritten in terms of a static state feedback law of the form $\boldsymbol{\tau} = \boldsymbol{\tau}(\boldsymbol{q},\boldsymbol{\theta},\dot{\boldsymbol{q}},\dot{\boldsymbol{\theta}},\boldsymbol{v})$ or $\boldsymbol{\tau} = \boldsymbol{\tau}(\boldsymbol{q},\boldsymbol{\tau}_\mathrm{J},\dot{\boldsymbol{q}},\dot{\boldsymbol{\tau}}_\mathrm{J},\boldsymbol{v})$. Indeed, the evaluation of the various derivatives of the dynamic model terms that are present in these expressions should be properly organized or customized in order to optimize computations.

Based on the resulting (11.47), the trajectory-tracking problem is solved by setting

$$\begin{aligned}\boldsymbol{v} = \boldsymbol{q}_\mathrm{d}^{[4]} &+ \mathbf{K}_3\left(\boldsymbol{q}_\mathrm{d}^{[3]} - \boldsymbol{q}^{[3]}\right) + \mathbf{K}_2\left(\ddot{\boldsymbol{q}}_\mathrm{d} - \ddot{\boldsymbol{q}}\right) \\ &+ \mathbf{K}_1\left(\dot{\boldsymbol{q}}_\mathrm{d} - \dot{\boldsymbol{q}}\right) + \mathbf{K}_0\left(\boldsymbol{q}_\mathrm{d} - \boldsymbol{q}\right) \,,\end{aligned} \tag{11.50}$$

where it is assumed that the reference trajectory $\boldsymbol{q}_{\mathrm{d}}(t)$ is (at least) three times continuously differentiable (i. e., the fourth derivative $\boldsymbol{q}_{\mathrm{d}}^{[4]}$ exists), and the *diagonal* matrices $\mathbf{K}_0, \ldots, \mathbf{K}_3$ have scalar elements $\mathbf{K}_{\cdot,i}$ such that

$$s^4 + K_{3,i}s^3 + K_{2,i}s^2 + K_{1,i}s + K_{0,i}\,, \quad i = 1, \ldots, N\,,$$

are Hurwitz polynomials. In view of the obtained decoupling, the trajectory error

$$e_i(t) = q_{d,i}(t) - q_i(t)$$

for the i-th link satisfies then

$$e_i^{[4]} + K_{3,i}e_i^{[3]} + K_{2,i}\ddot{e}_i + K_{1,i}\dot{e}_i + K_{0,i}e_i = 0;\,,$$

yielding $e_i(t) \to 0$ in a global exponential way, for any initial state. A series of remarks are in order:

- When the initial state $(\boldsymbol{q}(0), \boldsymbol{\theta}(0), \dot{\boldsymbol{q}}(0), \dot{\boldsymbol{\theta}}(0))$ is matched with the reference trajectory and its first three derivatives at time $t = 0$ (for this check, (11.48) and (11.49) are to be used), exact reproduction of the reference trajectory is achieved at all times.
- In the case of a discontinuity of the reference position trajectory, or of any of its first three derivatives, at a time $t^* \in [0, T]$, a trajectory error occurs at $t = t^*$ that will again decay to zero, independently for each link and with the prescribed exponential rate.
- The choice of the gains $K_{3,i}, \ldots, K_{0,i}$ can be made by a pole placement technique (equivalent in this case to an eigenvalue assignment). Let $\lambda_1, \ldots, \lambda_4$ be four poles with negative real parts, possibly given in complex pairs and/or coincident, specifying the desired transient of the trajectory error. These closed-loop poles will be assigned by the unique choice of real and positive gains

$$\begin{aligned}
K_{3,i} &= -(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4)\,,\\
K_{2,i} &= \lambda_1(\lambda_2 + \lambda_3 + \lambda_4) + \lambda_2(\lambda_3 + \lambda_4) + \lambda_3\lambda_4\,,\\
K_{1,i} &= -\left[\lambda_1\lambda_2(\lambda_3 + \lambda_4) + \lambda_3\lambda_4(\lambda_1 + \lambda_2)\right]\,,\\
K_{0,i} &= \lambda_1\lambda_2\lambda_3\lambda_4\,.
\end{aligned}$$

  When the values of link and motor inertias are very different from each other, or when the joint stiffness is very large, the above fixed choice of gains has the drawback of generating too large control efforts. In those cases, a more tailored set of eigenvalues can be assigned scheduling their placement as a function of the physical data of robot inertias and joint stiffnesses.
- When compared to the *computed torque* method for rigid robots, the feedback linearization control for trajectory tracking given by (11.46)–(11.50) requires the inversion of the inertia matrix $\mathbf{M}(\boldsymbol{q})$ and the additional evaluation of derivatives of the inertia matrix and of other terms in the dynamic model.
- The feedback linearization approach can also be applied without any changes in the presence of viscous (or otherwise smooth) friction at the motor and link side. The inclusion of spring damping leads instead to a third-order decoupled differential relation between the auxiliary input $\boldsymbol{v}$ and $\boldsymbol{q}$, leaving thus a N-dimensional unobservable dynamics in the closed-loop system, which is, however, still asymptotically stable. In this case, only input–output (and not full-state) linearization and decoupling is achieved.

We conclude with some considerations on exact linearization/decoupling by feedback for the general model (11.7) of robots with elastic joints. Unfortunately, due to the presence of the inertial coupling matrix $\mathbf{S}(\boldsymbol{q})$ between links and rotors, the above control design can no longer be applied. Consider as an illustration the case of a constant $\mathbf{S}$ in (11.7), with the associated model simplifications. Solving for $\ddot{\boldsymbol{\theta}}$ from the motor equations, $\ddot{\boldsymbol{\theta}} = \mathbf{B}^{-1}\left(\boldsymbol{\tau} - \boldsymbol{\tau}_{\mathrm{J}} - \mathbf{S}^{\mathrm{T}}\ddot{\boldsymbol{q}}\right)$, and substituting into the link equations leads to

$$\begin{aligned}
\left[\mathbf{M}(\boldsymbol{q}) - \mathbf{S}\mathbf{B}^{-1}\mathbf{S}^{\mathrm{T}}\right]\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \left(\mathbf{I} + \mathbf{S}\mathbf{B}^{-1}\right)\boldsymbol{\tau}_{\mathrm{J}}\\
= -\mathbf{S}\mathbf{B}^{-1}\boldsymbol{\tau}\,.
\end{aligned}$$

In this case, the input torque $\boldsymbol{\tau}$ appears already in the expression for the link acceleration $\ddot{\boldsymbol{q}}$. Using (11.6), the expression of the decoupling matrix is then

$$\mathbf{A}(\boldsymbol{q}) = -\left[\mathbf{M}_{\mathrm{L}}(\boldsymbol{q}) + \mathbf{M}_{\mathrm{R}}(\boldsymbol{q})\right]^{-1}\mathbf{S}\mathbf{B}^{-1}\,,$$

which is never full rank in view of the structure (11.5) of $\mathbf{S}$. As a consequence, the necessary condition for obtaining (at least) input–output linearization and decoupling by static state feedback fails to hold.

However, by resorting to the use of a larger class of control laws, it is still possible to obtain an exact linearization and decoupling result. For this purpose, consider a *dynamic* state feedback controller of the form

$$\begin{aligned}
\boldsymbol{\tau} &= \boldsymbol{\alpha}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{\xi}) + \boldsymbol{\beta}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{\xi})\boldsymbol{v}\,,\\
\dot{\boldsymbol{\xi}} &= \boldsymbol{\gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{\xi}) + \boldsymbol{\delta}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{\xi})\boldsymbol{v}\,, \qquad (11.51)
\end{aligned}$$

where $\boldsymbol{\xi} \in \mathbb{R}^{\nu}$ is the state of the dynamic compensator, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, and $\boldsymbol{\delta}$ are suitable nonlinear vector functions, and $\boldsymbol{v} \in \mathbb{R}^N$ is (as before) an external input used for trajectory-tracking purposes. It is possible to show in general that a dynamic compensator (11.51) of order

at most $\nu = 2N(N-1)$ can be designed so as to yield a closed-loop system that is globally described by

$$\boldsymbol{q}^{[2(N+1)]} = \boldsymbol{v} \,, \tag{11.52}$$

in place of (11.47). Note the coincidence of this differential order with the one found for exact reproducibility of a desired trajectory $\boldsymbol{q}_\mathrm{d}(t)$ in Sect. 11.1.2. The tracking problem is then solved by working on (11.52), with a direct generalization of the linear stabilization design in (11.50).

It is interesting to give a physical interpretation for this control result. The structural obstruction to input–output decoupling of the model (11.7) is related to the fact that the motion of a link attached to an elastic joint is affected *too soon* (at the second differential level) by torques originating at other elastic joints. This is due to the inertial couplings present between motors and links. The addition of dynamics (i. e., of integrators) in the controller slows down these low-energy paths and allows for the high-energy effects of the elastic torque at the local joint (a fourth-order differential path) to come into play. This dynamic balancing allows both input–output decoupling and exact linearization of the extended system (having the robot and the controller states).

Linear Control Design

The feedback linearization approach to trajectory tracking gives rise to a rather complex nonlinear control law. Its main advantage of globally enforcing a linear and decoupled behavior to the trajectory error dynamics should be traded-off with a control design that achieves only local stability around the reference trajectory, but is much simpler to implement (and may run at higher sampling rates).

For this, the inverse dynamics results of Sect. 11.1.2 can be used, assuming either the reduced or the complete robot model, with and without dissipative terms. Given a sufficiently smooth desired link trajectory $\boldsymbol{q}_\mathrm{d}(t)$, we have seen that it is always possible to associate:

1. The nominal torque $\boldsymbol{\tau}_\mathrm{d}(t)$ needed for its exact reproduction.
2. The reference evolution of all other state variables (e.g., $\boldsymbol{\theta}_\mathrm{d}(t)$, as given by (11.13), or $\boldsymbol{\tau}_{\mathrm{J},d}(t)$).

These signals define a sort of steady-state (though, time-varying) operation for the system.

A simpler tracking controller combines a model-based feedforward term with a linear feedback term using the trajectory error. The linear feedback locally stabilizes the system around the reference state trajectory, whereas the feedforward torque is responsible for maintaining the robot along the desired motion as soon as the error has vanished (see some examples on the tracking of square paths in VIDEO 136).

Using full-state feedback, two possible controllers of this kind are

$$\begin{aligned}\boldsymbol{\tau} = \boldsymbol{\tau}_\mathrm{d} &+ \mathbf{K}_{P,\theta}(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta}) + \mathbf{K}_{D,\theta}(\dot{\boldsymbol{\theta}}_\mathrm{d} - \dot{\boldsymbol{\theta}}) \\ &+ \mathbf{K}_{P,q}(\boldsymbol{q}_\mathrm{d} - \boldsymbol{q}) + \mathbf{K}_{D,q}(\dot{\boldsymbol{q}}_\mathrm{d} - \dot{\boldsymbol{q}})\end{aligned} \tag{11.53}$$

and

$$\begin{aligned}\boldsymbol{\tau} = \boldsymbol{\tau}_\mathrm{d} &+ \mathbf{K}_{P,\theta}(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta}) + \mathbf{K}_{D,\theta}(\dot{\boldsymbol{\theta}}_\mathrm{d} - \dot{\boldsymbol{\theta}}) \\ &+ \mathbf{K}_{P,J}(\boldsymbol{\tau}_{\mathrm{J,d}} - \boldsymbol{\tau}_\mathrm{J}) + \mathbf{K}_{D,J}(\dot{\boldsymbol{\tau}}_{\mathrm{J,d}} - \dot{\boldsymbol{\tau}}_\mathrm{J}) \,.\end{aligned} \tag{11.54}$$

These trajectory tracking schemes are the most common in the control practice for robots with elastic joints. In the absence of full-state measurements, they can be combined with an observer of the unmeasurable quantities. An even simpler realization is

$$\boldsymbol{\tau} = \boldsymbol{\tau}_\mathrm{d} + \mathbf{K}_P(\boldsymbol{\theta}_\mathrm{d} - \boldsymbol{\theta}) + \mathbf{K}_D(\dot{\boldsymbol{\theta}}_\mathrm{d} - \dot{\boldsymbol{\theta}}) \,, \tag{11.55}$$

which uses only motor measurements and relies on the results obtained for the regulation case.

The different gain matrices used in (11.53)–(11.55) have to be tuned using a linear approximation of the robot system. This approximation may be obtained at a fixed equilibrium point or around the actual reference trajectory, leading respectively to a linear time-invariant or to a linear time-varying system. While the existence of (possibly time-varying) stabilizing feedback matrices is guaranteed by the controllability of these linear approximations, the validity of the approach is only *local* in nature and the region of convergence will depend both on the given trajectory and on the robustness of the designed linear feedback.

It should be mentioned that such a control approach to trajectory tracking problems can be used also for robots with flexible links. Once the inverse dynamics problem has been solved (for the case of link flexibility, this typically results in a noncausal solution), a controller of the form (11.55) (or (11.53), with link deflection $\boldsymbol{\delta}$ and deflection rate $\dot{\boldsymbol{\delta}}$ replacing, respectively, $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$), can be directly applied.

11.1.5 Further Reading

This section contains the main references for the part of this chapter on robots with joint flexibility. In addition, we point out to a larger bibliography for topics that have not been included here.

Early interest in problems arising from the presence of flexible transmissions in industrial robots dates back to [11.1, 2], with first experimental findings on the

GE P-50 arm. Relevant mechanical considerations involved in the design of robot arms and in the evaluation of their compliant elements can be found in [11.3].

One of the first studies on the inclusion of joint elasticity in the dynamic modeling of robot arms is due to [11.4]. The detailed analysis of the model structure presented in Sect. 11.1.1 comes from [11.5], with some updates from [11.6]. The simplifying assumption leading to the reduced model was introduced in [11.7]. The special case of motors mounted on the driven links is considered in [11.8]. The observation that joints with limited elasticity lead to a singularly perturbed dynamic model was first recognized in [11.9]. Symbolic manipulation programs for the automatic generation of dynamic models of robots with elastic joints were also developed very early [11.10].

The inverse dynamics computations in Sect. 11.1.2 can be traced back to [11.11, 12]. The use of programs such as Modelica for the numerical evaluation of inverse dynamics is highlighted in [11.6]. An efficient Newton–Euler inverse dynamics algorithm for robots with elastic joints has been proposed in [11.12].

The first specific control designs were based on decentralized linear controllers, see, e.g., [11.13] and [11.14], where a fourth-order dynamics local to each elastic joint was used. However, schemes with proved global convergence characteristics appeared only later.

In Sect. 11.1.3, the PD controller with constant gravity compensation is a contribution of [11.15]. This was also extended to the case of robots with flexible links in [11.16]. The first two versions of regulation control with online gravity compensation are due, respectively, to [11.17] and [11.18]. Both these control laws were extended to Cartesian compliance schemes in [11.19–21]. The PD-like controller (11.38), with perfect cancellation of gravity effects on link motion, is a result of [11.22]. A general regulation framework based on energy shaping has been proposed in [11.23]. In the absence of any information on gravity, a proportional–integral–derivative (PID) regulator with semiglobal stability properties has been presented in [11.24], while a global learning scheme was proposed in [11.25].

The presentation of a full-state feedback design for regulation (and tracking) follows the idea used in [11.26]. Special interest has been devoted over the years to joint torque feedback, from the placement of torque sensors on transmission shafts [11.27] or within harmonic drives [11.28] to their use for achieving robust control performance [11.29].

The fact that the reduced model of robots with elastic joints can always be feedback linearized via static state feedback was shown for the first time in [11.7]. A previous similar result for a specific robot kinematics can be found in [11.30]. The discrete-time implementation of the feedback linearization approach has been studied in [11.31], while its robustness properties were analyzed in [11.32]. Feedback linearization and input–output decoupling were also considered for the case of viscoelastic joints in [11.33], and for robots with joints of mixed type, some rigid and some elastic, in [11.34]. The same idea of inversion control, leading to input–output decoupling and linearization, has also been successfully applied to the tracking problem of joint-level trajectories in robots with flexible links [11.35].

For the general model of robots with elastic joints, the use of dynamic state feedback was first proposed in [11.36]. A comparative study of the errors induced by neglecting the motor–link inertia couplings (thus, for robots that are not linearizable by static state feedback) was carried out in [11.37]. The general algorithm for constructing the dynamic linearizing feedback is presented in [11.38, 39].

The benefit of combining a stabilizing feedback with a feedforward command in robots with flexible joints was experimentally shown for the first time in [11.39] (see also the VIDEO 770) and then formalized in [11.11].

Other control design methodologies available for trajectory tracking, which were not discussed in Sect. 11.1.4, are based on singular perturbation, backstepping, or passivity. Nonlinear controllers based on the two-time scale separation property were proposed by [11.40] and [11.41]. These corrective controllers are an outcome of the singular perturbation model form and should be preferred in the case of very stiff joints, as they do not generate high-gain laws in the limit. Backstepping is based on the idea of considering the joint elastic torque [11.42] or the motor position [11.43] as an intermediate fictitious input to be used to control the link equations, and then to design the actual torque input in the motor equations so as to follow the reference behavior for the previous intermediate input. The main advantage is that this design may be easier to be transformed into an adaptive version. Adaptive control results for robots with elastic joints include the high-gain (approximate) schemes [11.44] and [11.45], as well as the global (but very complex) solution obtained in [11.46], both analyzed using only the reduced dynamic model. Moreover, robust control schemes have been proposed in [11.47], based on sliding mode techniques, and in [11.48–52], using iterative learning on repetitive tasks.

As for optimal control results, the intrinsic fourth-order dynamics associated to flexible joints has barred so far the derivation of analytical or numerically efficient results. Finding the time-optimal velocity profile

on a constrained geometric path is a solved problem in the rigid case, but only an approximate solution is available for robots with elastic joints [11.49]. A useful generalization from the rigid to the elastic case is the dynamic trajectory scaling algorithm of [11.50], which provides a closed-form expression to the uniform time scaling that recovers feasibility of commanded torques w.r.t. their bounds. Selected optimal control problems for single viscoelastic joints, involving either the maximization of stored potential energy or the minimization of rest-to-rest motion time, have been tackled and solved in [11.51] and [11.52], respectively.

Different state observers have been presented, starting from an approximate one [11.53] up to the exact ones in [11.5, 54], in which a tracking controller based on the estimated state was also tested. In all cases, link position or link position and velocity measurements are assumed. On the other hand, the use of motor position and link acceleration has been shown to be a feasible alternative for robust state reconstruction [11.55].

Finally, the force control problem in constrained tasks for robots with elastic joints has been tackled following a singular perturbation technique [11.56, 57], an inverse dynamics approach [11.58], or an adaptive strategy [11.59].

A unified passivity-based approach has been presented in [11.60], where an inner torque feedback loop is incorporated for shaping (viz., reducing) the motor inertia. This enables approximate gravity compensation (as done in Sect. 11.1.3), as well as to assign a desired Cartesian stiffness relation for the link positions, mimicking the impedance control of the rigid case for handling interactions and contacts with the environment.

11.2 Robots with Flexible Links

Arm flexibility is a dynamic behavior in which kinetic energy interacts with elastic potential energy. Kinetic energy is stored in moving inertia, and potential energy is stored in compliant members. Flexible links provide both the inertia and compliance distributed throughout the member. Flexibility manifests itself as mechanical oscillations and static deflections, greatly complicating the motion control of a mechanical arm. If the time to settle the oscillations is significant relative to the cycle time of the overall task, flexibility will be a major consideration in the arm design.

11.2.1 Design Issues

We will deal with these issues in a modular way, separating link flexibility and joint flexibility. The first topic of link flexibility deals with the inherent spatial distribution of compliance and inertia, perhaps with both effects existing in the same element of material. The links of a manipulator arm can usually be stiffened by adding structural mass or by improving the material properties or distribution over the links. At some points, the links may be treated as rigid. The designer must realize that an arm with heavy rigid links and flexible joints may be more flexible dynamically than a lighter arm with more link flexibility and less inertia. The consequences are predicted by the natural frequency of the first flexible mode of motion, with the joint actuators locked, which can be approximated by

$$\omega_1 = \sqrt{\frac{k_{\text{eff}}}{I_{\text{eff}}}}\,, \tag{11.56}$$

where k_{eff} is the effective spring constant and I_{eff} is the effective inertia. For the elastic joint case, k_{eff} and I_{eff} are directly available from the elastic joint parameters and mass properties of the arm. The *effective* values applicable to a simple beam in bending, for example, would result from boundary conditions that are clamped at one end and free at the other and would be selected to yield a good approximation to the natural frequency. This natural frequency is readily available from handbooks on vibration in this case as $\omega_1 = 3.52\sqrt{EI/ml^4}$. (The variables are used in (11.57) and (11.58) and defined below.) More complex geometries can be evaluated by various means, including the transfer matrix method discussed in Sect. 11.2.2 on modeling. The arm should be considered flexible when this frequency is low enough to interfere with the design of controllers based on rigid assumptions. The common PD control of a single joint on a flexible link can only achieve adequate damping of a closed-loop design when the magnitude of the closed loop poles is less than about $1/3$ of ω_1 [11.61].

A distillation of the key aspects of the mechanics of materials is now in order. In some cases static compliance effects represent a member sufficiently, whereas in other cases the true distributed nature of compliance and mass and its dynamics must be modeled. A long structural member with forces f and moments M perpendicular to its long axis is subject to bending. Static bending relates the moment at the axial location x to the displacement w

$$M(x) = EI(x)\frac{\partial^2 w}{\partial x^2}\,, \tag{11.57}$$

where E is the elastic modulus of the material, and $I(x)$ is the area moment of inertia about the neutral axis of the cross section. The deflection at any point is obtained by integrating this equation from a reference point to a desired point, such as the end of the link.

This is a description of compliance that can be used when mass is isolated from elasticity. If mass is distributed throughout the beam with material mass density per unit volume ρ, the time t must be incorporated so that

$$m(x)\frac{\partial^2 w(x,t)}{\partial t^2}+\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2 w(x,t)}{\partial x^2}\right)=0\,. \tag{11.58}$$

Here $m(x)$ is the mass density per unit length, incorporating the material properties and the cross-sectional area at x. Assumptions implicit in this equation, which is called the Bernoulli–Euler equation, include the minimal impact of shear distortion and rotational inertia of the cross section of the beam, which is valid for long beams. The Timoshenko beam model relaxes these assumptions but is not included here.

Torsional deflection of an angle θ results if a twisting moment T about the long beam axis occurs, such that, in the static case

$$\theta=\frac{Tl}{JG}=\alpha_{\theta T}T\,, \tag{11.59}$$

where G is the shear modulus of the material, and J is the polar area moment of inertia about the neutral axis of the beam. Again, the addition of a distributed mass and considering the dynamics produces a partial differential equation with independent variables in space and time

$$\mu(x)\frac{\partial^2\theta}{\partial t^2}=GJ\frac{\partial^2\theta}{\partial x^2}\,, \tag{11.60}$$

where $\mu(x)$ is the rotary mass moment of inertia per unit length of the shaft. While (11.60) is potentially appropriate for links, the smaller rotational mass moment of inertia of a long slender shaft relative to the links it drives results in the static compliance effects of torsion given by (11.59) being of most importance.

Tension and compression effects should also be acknowledged, although they tend to be the least significant. Here deflection δ in the x-direction (the long axis of the member) results from axial force F_a and is calculated as $\delta=\frac{F_aL}{AE}$

$$\delta=\frac{F_{\mathrm{a}}L}{AE}=\alpha_{XF}F_{\mathrm{a}}\,, \tag{11.61}$$

where L is the element length and A is its cross sectional area. In the dynamic case, with mass density ρ and displacement ξ, the axial motion is described by

$$\rho\frac{\partial^2\xi}{\partial t^2}=E\frac{\partial^2\xi}{\partial x^2}\,. \tag{11.62}$$

Note that these effects are special cases of the more general elastic behavior of a member subject to acceleration and external loading, specialized to the member with one long axis. The same general phenomena apply to joint structures (e.g., bearings and couplings) and drive-train components (e.g., gears and cables). The functional constraints on the structural design of these members may be more severe and limit the designer's ability to make them more rigid by adding more material. Under this constraint, the combined compliance of the flexible joint design and the rigidized link design results in inferior performance of the combination. A less rigid link with less inertia will raise the lowest natural frequency. Thus the design of the link and the joint must be coordinated.

Flexibility is only one of several link design constraints. Static compliance is obviously another based on the above discussion. Buckling and strength are two more considerations that constrain some of the same design parameters. While it has been shown that flexibility is a dominant constraint for typical arm design regimes [11.62], a simple optimization of bending stiffness of a beam with a tubular geometry illustrates the fallacy of ignoring buckling. If the radius of the tube is varied to maximize bending stiffness with constrained total mass, the radius increases without bound, producing a thin shell of infinitesimal thickness and infinite radius that lacks robustness. The true constraint becomes local buckling of the tube under slight perturbations of the applied load. A thin metal can illustrates this behavior as it crushes. Strength is another applicable constraint. Various materials differ in their strength, typically represented by their elastic limit or endurance limit on the stress at a point in the member. A sufficiently stiff member may fail due to stress, particularly at points of stress concentration.

In this presentation of options to overcome flexibility, and link flexibility in particular, radically different operational and design strategies should be recognized even though they are not the core of the treatment. The elastic modulus relates stress to strain, but physically the strain rate is sometimes a relevant term, providing structural damping to augment the joint or control-based damping. Composite materials have substantially more damping inherent in their behavior. Damping can be enhanced with passive damping treatments, with constrained layer damping having particularly pronounced effects [11.63]. Smart materials can also be

configured to enhance damping [11.64]. An operational strategy known as *bracing* can be combined with redundant actuation to achieve a large workspace for a flexible arm that provides gross motion and a smaller precision workspace after docking the large arm with a stationary structure, as described in [11.65].

The implication above is that flexibility was always due to the moving parts of the arm. However, the arm base itself may be a source of significant flexibility. If the base is stationary, the analysis may be almost identical to that of the flexible links of an arm. However, if the arm is mounted on a vehicle, tires or tracks may be the source of flexibility and empirical evaluation of these new elements might be the best solution. The vehicle could be a boat, aircraft, or space vehicle. Here there is no elastic connection to an inertial reference frame and some fundamental changes in approach will be needed. These changes are not unexplored and in some cases are simple extensions of the following approaches.

11.2.2 Modeling of Flexible Link Arms

Mathematical models of link flexibility trade accuracy for ease of use. It is therefore key to determine the effects that are important to represent accurately. The techniques presented here will assume linear elasticity with light damping. Rotational motions must be of modest angular rate so that centrifugal stiffening can be ignored. Small deflections will generally be assumed. These are reasonable assumptions for most robotic devices, but can be violated in more exotic applications, requiring reevaluation of the assumptions and more tortuous modeling approaches.

Four model types will be examined:

1. Lumped-element models having compliance or inertia but not both in a given member.
2. Finite-element models, which will be only briefly addressed.
3. Assumed mode models which readily include nonlinear dynamic behavior.
4. Transfer matrix models that incorporate the true distributed intermingling of compliance and inertia and consequently the infinite-dimensional nature of flexible link arms.

With the limited space available here, the discussion will be preliminary but should enable the reader to pursue one or more alternatives knowledgably.

For lumped elements, one may build on the rigid kinematics presented in the section on fundamentals of kinematics (Chap. 2) that establishes rigid transformation matrices $\mathbf{A}_i$. The 4×4 homogenous transformations that describe position can be used to describe deflection as well. Assuming small motions and static behavior (or negligible member mass), the transformation produced by elastic bending, torsion, and compression is

$$\mathbf{E}_i = \begin{pmatrix} 1 & -\alpha_{\theta Fi}F^i_{Yi} - \alpha_{\theta Mi}M^i_{Zi} \\ \alpha_{\theta Fi}F^i_{Yi} + \alpha_{\theta Mi}M^i_{Zi} & 1 \\ \alpha_{\theta Fi}F^i_{Zi} - \alpha_{\theta Mi}M^i_{Yi} & \alpha_{Ti}M^i_{Xi} \\ 0 & 0 \\ -\alpha_{\theta Fi}F^i_{Zi} + \alpha_{\theta Mi}M^i_{Yi} & \alpha_{Ci}F^i_{Xi} \\ -\alpha_{Ti}M_{Xi} & \alpha_{XFi}F^i_{Yi} + \alpha_{XMi}M^i_{Zi} \\ 1 & \alpha_{XFi}F^i_{Zi} + \alpha_{XMi}M^i_{Yi} \\ 0 & 1 \end{pmatrix}, \tag{11.63}$$

where

α_{Ci} = coefficient in compression, displacement/force ,

α_{Ti} = coefficient in torsion, angle/moment ,

$\alpha_{\theta Fi}$ = coefficient in bending, angle/force ,

$\alpha_{\theta Mi}$ = coefficient in bending, angle/moment ,

α_{XFi} = coefficient in bending, displacement/force ,

α_{XMi} = coefficient in bending, displacement/moment ,

F^j_{Xi} = the force at the end of link i in the X direction of coordinate frame j ,

F^j_{Yi} = the force at the end of link i in the Y direction of coordinate frame j ,

F^j_{Zi} = the force at the end of link i in the Z direction of coordinate frame j ,

M^j_{Xi} = the moment at the end of link i in the X direction of coordinate frame j ,

M^j_{Yi} = the moment at the end of link i in the Y direction of coordinate frame j ,

M^j_{Zi} = the moment at the end of link i in the Z direction of coordinate frame j .

The listed coefficients depend on the construction of the element and are readily found for a slender beam from simple strength of materials. In other cases, a finite-element model or empirical measurements may be more readily used.

Alternating transformations for the undeformed and deformed aspects of N members (links or joints) produces the position vector of the end of arm (EOA)

$$\boldsymbol{p}^0 = (\mathbf{A}_1\mathbf{E}_1\mathbf{A}_2\mathbf{E}_2 \ldots \mathbf{A}_i\mathbf{E}_i\mathbf{A}_{i+1}\mathbf{E}_{i+1} \ldots \mathbf{A}_N\mathbf{E}_N)\begin{pmatrix}0\\0\\0\\1\end{pmatrix}. \tag{11.64}$$

If the massless elements connect rigid lumped masses, a linear spatial model is readily obtained, as described in [11.66]. Servo-controlled joints can also be inserted into this model. This analysis proceeds by examining the deflection of the elastic members produced by joint motion and the resulting forces and moments on the rigid inertias at the ends of the chain of links. The reference concentrates on the special case of only two inertias at each end of the chain but the technique is easily extended to inertias in the middle of the chain as well. The results are six linear second-order equations for each inertia and one first-order equation for each joint.

If the compliance and inertia properties are both to be treated as distributed throughout the same element, the partial differential equations (PDEs) introduced in the previous section for bending, torsion, and compression must be employed. Still insisting on a linear treatment, the general solution of these equations can be found in terms of each element with adjoining elements, i. e., the boundary conditions on the PDE. The convenient approach of Laplace transformation of these linear equations into the frequency domain, followed by factoring out the boundary conditions into a matrix–vector product, results in a technique known as the transfer matrix method (TMM) [11.67]. This method has been applied productively to general elastomechanics problems as well as specifically to flexible arms [11.68].

The transfer matrix relates variables at two stations along the arm with the number of variables depending on the model complexity. For the planar bending, deflection of the beam is shown in Fig. 11.9

$$z_1 = \begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_1 = \begin{pmatrix} -\text{displacement} \\ \text{angle} \\ \text{moment} \\ \text{shear force} \end{pmatrix}_{\text{at station 1}} ,$$
$$z_0 = \mathbf{T} z_1 , \qquad (11.65)$$

where T is the appropriate element transfer matrix.

If the element is a simple rotary spring with constant k and a damper with constant b, then the angle of rotation θ is related to the moment by a differential equation with the following Laplace-domain representation

$$M = -L(k\theta + b\dot{\theta}) \Rightarrow k\Theta(s) + bs\Theta(s) = M_0 = M_1 ,$$
$$W_0 = W_1 ,$$
$$V_0 = V_1 ,$$
$$\Psi_0 = \Psi_1 - \Theta(s) = \Psi_1 - \frac{1}{k+bs} M_1 .$$

Note that the only variable changed across the element is the angle related to the moment. The transfer matrix representation conventionally uses zero initial conditions and converts to the Fourier representation with $s = \imath\omega$. This leads to the transfer function

$$\mathbf{T} = \mathbf{C}(\imath\omega) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & \frac{1}{k+b(\imath\omega)} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} ,$$

which has the same form of the joint controller as shown in (11.69).

The beam model is much more complex, and the Euler–Bernoulli model gives

$$\mathbf{B} = \begin{pmatrix} c_0 & lc_1 & ac_2 & alc_3 \\ \frac{\beta^4 c_3}{l} & c_0 & \frac{ac_1}{l} & ac_2 \\ \frac{\beta^4 c_2}{a} & \frac{\beta^4 l c_3}{a} & c_0 & lc_1 \\ \frac{\beta^4 c_1}{al} & \frac{\beta^4 c_2}{a} & \frac{\beta^4 c_3}{l} & c_0 \end{pmatrix} , \qquad (11.66)$$

where

$$\beta^4 = \omega^2 l^4 \mu/(EI) ; \quad a = l^2/(EI) ,$$
$$c_0 = (\cosh\beta + \cos\beta)/2 ,$$
$$c_1 = (\sinh\beta + \sin\beta)/(2\beta) ,$$
$$c_2 = (\cosh\beta - \cos\beta)/(2\beta^2) ,$$
$$c_3 = (\sinh\beta - \sin\beta)/(2\beta^3) ,$$
$$\mu = \text{ density/unit length} ,$$
$$\omega = \text{ circular frequency of vibration} ,$$
$$E = \text{ elastic modulus} ,$$
$$I = \text{ cross-sectional area moment of inertia} .$$

The spatial variable has been transformed to the spatial Laplace variable but no longer appears explicitly. The Laplace time variable remains and is designated by s or by the frequency variable $\omega = -\imath s$, where $\imath = \sqrt{-1}$.

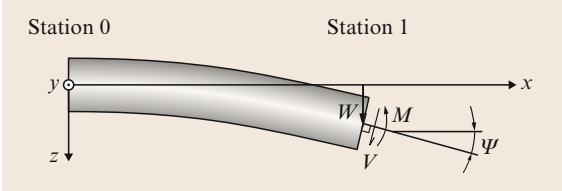


Fig. 11.9 State vector for the bending transfer matrix

The other transfer matrices needed for a simple planar model are as follows.

For a rotation in the plane of angle φ

$$\mathbf{A} = \begin{pmatrix} \frac{1}{\cos\varphi} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ m_s\omega^2 \sin\varphi \tan\varphi & 0 & 0 & \cos\varphi \end{pmatrix}, \quad (11.67)$$

where m_s is the sum of all outboard masses from the angle to the end of the arm. This is an approximation resulting from ignoring compression elasticity. These elements appear as an additional mass to the extent they are translated with beam compression.

For a rigid mass, simply applying Newton's laws and collecting term yields

$$\mathbf{R} = \begin{pmatrix} 1 & l & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{m\omega^2 l}{2} & I_m\omega^2 - \frac{m\omega^2 l^2}{2} & 1 & l \\ m\omega^2 & -\frac{m\omega^2 l}{2} & 0 & 1 \end{pmatrix}, \quad (11.68)$$

where m is the mass of the body, I_m is the mass moment of inertia about an axis through the center of mass and perpendicular to the plane of the arm, l is the length of the mass (the distance between the points of attachment at stations i and $i+1$), and $l/2$ is the distance to the center of mass from station i.

For a controlled joint with the controller transfer function (joint torque/joint angle) $= k(\imath\omega)$

$$\mathbf{C} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & \frac{1}{k(\imath\omega)} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (11.69)$$

When combined to represent the planar arm with two joints as shown in Fig. 11.10, this analysis yields a composite transfer matrix that relates the four state variables at the two ends of the arm. Further analysis results from imposing the known boundary conditions at these ends.

The transfer matrix representation of an arm presented in [11.69] and pictured in Fig. 11.10 is shown in (11.70).

$$\begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_0 = \mathbf{R}_1\mathbf{B}_2\mathbf{R}_3\mathbf{A}_4\mathbf{C}_5\mathbf{B}_6\mathbf{R}_7 \begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_7 . \quad (11.70)$$

In the example, the left end is pinned and the right end is free, yielding

$$z_0 = \begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_0 = \begin{pmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ u_{21} & u_{22} & u_{23} & u_{24} \\ u_{31} & u_{32} & u_{33} & u_{34} \\ u_{41} & u_{42} & u_{43} & u_{44} \end{pmatrix} \begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_7 ,$$

$$\begin{pmatrix} u_{11} & u_{12} \\ u_{31} & u_{32} \end{pmatrix} \begin{pmatrix} -W \\ \Psi \end{pmatrix}_7 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \quad (11.71)$$

The determinant of the 2×2 matrix must be zero when ω is a natural frequency, or a system eigenvalue if it is complex. This can be found by a numerical search. Evaluating variables (position, angle, shear, and moment) along the length of the arm with the eigenvalues used for the values of ω yields the eigenfunctions. If boundary conditions are not zero but known for given frequencies, the frequency response of the system can be obtained. Hence Bode plots are readily obtained for boundary condition forcing. Interior forcing can be handled using an extended state vector as described in [11.67], which was applied to arms in [11.61] and most recently updated with modern programming techniques in [11.68]. Inverse fast Fourier transform (FFT) will produce a time-domain response from the frequency response.

The creation of a state-space model in the time domain is attractive because it enables the use of powerful state-space design techniques and is compatible with the nonlinear behavior of arms undergoing large motion and experiencing centrifugal and Coriolis forces. The assumed modes method allows such a model to be constructed efficiently. The presentation here follows the early introduction of a recursive manner to calculate these equations introduced in [11.70]. The flexible kinematics of the links must be expressed as a sum of basis functions also known as assumed modes (shapes) $\varphi_i(x)$ with time-variable amplitudes $\delta_i(t)$

$$w(x,t) = \sum_{i=1}^{\infty} \delta_i(t)\varphi_i(x) . \quad (11.72)$$

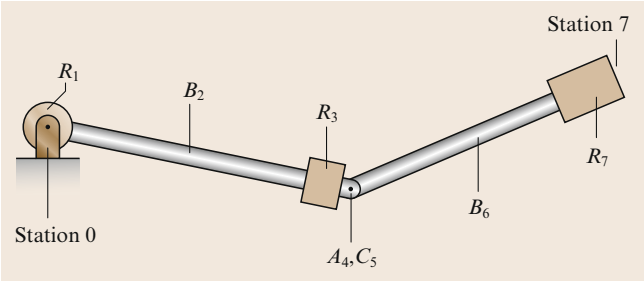


Fig. 11.10 Transfer matrix representation of an arm

The amplitudes and their derivatives become the states of the model. Compatible joint angle variables and their derivatives are also included as the rigid state variables. Joint angles tangent to the ends of the flexible links are compatible with mode shapes based on the clamped boundary conditions at one end and free boundary conditions at the other. This choice is generally chosen for simulation and the rigid coordinates are directly measured by standard joint angle transducers. If rigid coordinates are angles measured between lines connecting successive joint axes, the end of arm is known in terms of rigid coordinates alone and pinned–pinned boundary conditions are consistent. This produces advantages for inverse dynamics calculations as seen shortly.

The flexible and rigid kinematics combined describes the position and velocity of every point on the arm and can be used to express the kinetic energy $\mathcal{T}$ and the potential energy $\mathcal{V}$. These expressions are used in the conservative form of Lagrange's equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial \mathcal{T}}{\partial \dot{q}_i} - \frac{\partial \mathcal{T}}{\partial q_i} + \frac{\partial \mathcal{V}}{\partial q_i} = F_i \,, \tag{11.73}$$

where F_i is the force that does work as q_i is varied.

The kinematics of the deflection are again described by 4×4 transformation matrices, but this time the assumed mode shapes j are summed to give the position of link i

$$\boldsymbol{h}_i^i(\eta) = \begin{pmatrix}\eta\\0\\0\\1\end{pmatrix} + \sum_{j=i}^{m_i} \delta_{ij} \begin{pmatrix} x_{ij}(\eta)\\ y_{ij}(\eta)\\ z_{ij}(\eta)\\ 0\end{pmatrix}, \tag{11.74}$$

where x_{ij}, y_{ij}, and z_{ij} are the x_i, y_i, and z_i displacement components of mode j of link i's deflection, respectively, δ_{ij} is the time-varying amplitude of mode j of link i, and m_i is the number of modes used to describe the deflection of link i

$$\mathbf{E}_i = \left(\mathbf{H}_i + \sum_{j=i}^{m_i} \delta_{ij}\mathbf{M}_{ij}\right), \tag{11.75}$$

where

$$\mathbf{H}_i = \begin{pmatrix}1&0&0&l_i\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix} \tag{11.76}$$

and

$$\mathbf{M}_{ij} = \begin{pmatrix}0&-\theta_{zij}&\theta_{yij}&x_{ij}\\ \theta_{zij}&0&-\theta_{xij}&y_{ij}\\ -\theta_{yij}&\theta_{xij}&0&z_{ij}\\0&0&0&0\end{pmatrix}. \tag{11.77}$$

This description is used in the formation of the potential and kinetic energy contributions of each element of mass. The integral over the link accumulates the total energy for that link and the sum over all links gives the energy of the arm system. Interchanging the order of integration and summation allows key invariant parameters to be identified, such as the *modal mass* that multiplies the second derivative of modal amplitude and the *modal stiffness* that multiplies the amplitude itself. The intermediate steps are aimed at the evaluation of Lagrange's equations, which separate into derivatives related to the rigid variables and the flexible variables. If the rigid variables $\boldsymbol{\theta}$ are the joint variables, two groups of equations result

$$1.\quad \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathcal{T}}{\partial \dot{\theta}_j}\right) - \frac{\partial \mathcal{T}}{\partial \theta_j} + \frac{\partial \mathcal{V}_\mathrm{e}}{\partial \theta_j} + \frac{\partial \mathcal{V}_\mathrm{g}}{\partial \theta_j} = F_j \tag{11.78}$$

$$2.\quad \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathcal{T}}{\partial \dot{\delta}_{jf}}\right) - \frac{\partial \mathcal{T}}{\partial \delta_{jf}} + \frac{\partial \mathcal{V}_\mathrm{e}}{\partial \delta_{jf}} + \frac{\partial \mathcal{V}_\mathrm{g}}{\partial \delta_{jf}} = 0 \,, \tag{11.79}$$

where the subscript e indicates elasticity, subscript g indicates gravity, and $\boldsymbol{\delta}$ indicates the amplitude of a mode shape.

The simulation form of the equations is achieved by grouping all second derivatives of the rigid and flexible coordinates, which yields

$$\begin{pmatrix}\mathbf{M}_{rr}(\boldsymbol{q}) & \mathbf{M}_{rf}(\boldsymbol{q})\\ \mathbf{M}_{fr}(\boldsymbol{q}) & \mathbf{M}_{ff}\end{pmatrix}\begin{pmatrix}\ddot{\boldsymbol{\theta}}\\ \ddot{\boldsymbol{\delta}}\end{pmatrix} = -\begin{pmatrix}0&0\\0&\boldsymbol{K}_\mathrm{s}\end{pmatrix}\begin{pmatrix}\boldsymbol{\theta}\\ \boldsymbol{\delta}\end{pmatrix} + \boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{R}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{Q}) \,, \tag{11.80}$$

where $\boldsymbol{\theta}$ is a vector of rigid coordinates, usually the joint variables, $\boldsymbol{\delta}$ is the vector of flexible coordinates, $\boldsymbol{q} = [\boldsymbol{\theta}^\mathrm{T}\ \boldsymbol{\delta}^\mathrm{T}]^\mathrm{T}$, $\mathbf{M}_{ij}$ is the *mass* matrix for rigid and flexible coordinates corresponding to the rigid $(i,j=r)$ or flexible $(i,j=f)$ coordinates and equations, $\boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}})$ contains the nonlinear Coriolis and centrifugal terms, $\boldsymbol{G}(\boldsymbol{q})$ captures gravity effects, $\boldsymbol{Q}$ represents the externally applied forces, and $\boldsymbol{R}$ captures the effect of external forces and all other nonconservative forces including friction.

On the other hand, the inverse dynamics form of the equations is (assuming negligible friction)

$$\boldsymbol{Q} = \begin{pmatrix}\mathbf{M}_{rr}(\boldsymbol{q}) & \mathbf{M}_{rf}(\boldsymbol{q})\\ \mathbf{M}_{fr}(\boldsymbol{q}) & \mathbf{M}_{ff}\end{pmatrix}\begin{pmatrix}\ddot{\boldsymbol{\theta}}\\ \ddot{\boldsymbol{\delta}}\end{pmatrix} + \begin{pmatrix}0&0\\0&\boldsymbol{K}_\mathrm{s}\end{pmatrix}\begin{pmatrix}\boldsymbol{\theta}\\ \boldsymbol{\delta}\end{pmatrix} - \boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{G}(\boldsymbol{q}) \,. \tag{11.81}$$

In general, $\boldsymbol{Q}$ is dependent on the vector $\boldsymbol{T}$ of motor torques or forces at the joints, and the distribution of these effects is such that

$$\boldsymbol{Q} = \begin{pmatrix} \mathbf{B}_r \\ \mathbf{B}_f \end{pmatrix} \boldsymbol{T} \,, \tag{11.82}$$

where $\boldsymbol{T}$ has the same dimensionality as $\boldsymbol{q}_r$. By rearranging the inverse dynamics equation, we obtain

$$\begin{pmatrix} \mathbf{B}_r & -\mathbf{M}_{rf} \\ \mathbf{B}_f & -\mathbf{M}_{ff} \end{pmatrix} \begin{pmatrix} \boldsymbol{T} \\ \ddot{\boldsymbol{\delta}} \end{pmatrix} = \begin{pmatrix} \mathbf{M}_{rr} \\ \mathbf{M}_{rf} \end{pmatrix} \ddot{\boldsymbol{\theta}} + N(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{G}(\boldsymbol{q}) \,, \tag{11.83}$$

which is arranged with a vector of unknowns on the left-hand side of the equation and the prespecified rigid and solvable flexible coordinates on the right. This is at least approximately true, with the caveat that the nonlinear and gravity terms have a weak influence on the immediate flexible coordinates, which are not simultaneously known but which can be estimated for the next time step.

One consequence of the distributed flexibility is non-minimum-phase behavior. This can occur for other cases where the excitation is separated from the output point by mass and flexibility, a condition known as noncollocation. These systems are characterized by a step response in which the initial movement is opposite in sign from the final movement. This is simply characterized for linear systems as a transfer function zero in the right half-plane. As will be shown in the section on Command Generation, it may be possible to overcome this problem by redefining the output.

11.2.3 Control

Appropriate reaction to flexible dynamics requires appropriate sensors. While a Kalman filter can theoretically observe flexible link states from joint measurements, it is based on back driving the joint from the various modes of flexible dynamics. More reliable indications of flexibility include strain gauges, accelerometers, and optical sensors.

Sensors for Flexibility Control

The strain of a beam element is a direct indication of curvature, which is the second spatial derivative of beam deflection, as indicated in (11.57). Modern semiconductor strain gauges have high gauge factors producing a good signal-to-noise ratio. The second spatial derivative relationship means that a slight change in location can result in a large change in the reading, but once affixed to the arm this is not a problem. More of a problem is the relatively short lifetime of a strain gauge subject to wear and tear if not carefully protected. By suitably placing strain gauges, good measurements of multiple assumed modes are possible.

Optical measurement of deflection can be very versatile but is subject to interference between a target measurement point and the sensor, particularly if the sensor is stationary. If the sensor is mounted on the link, this difficulty is reduced but optical measurements are then usually a complex combination of the rotation of the optical axis of the sensor and the deflection of the target. Machine vision senses a wide field of view but typically with less precision. Multiple points can be detected, which is attractive in measuring multiple modes. Machine vision requires considerable processing and, even for enhanced targets or retroreflective fiducials, sample rates can be slow relative to some modes of control interest.

Accelerometers are attractive transducers in that they do not require explicit grounding to a fixed frame. Microelectromechanical systems (MEMSs) accelerometers provide the required low-frequency sensitivity and are very inexpensive. However, they are subject to orientation in a gravity field, which must be compensated for by knowing the orientation of the sensor. If one is interested in position, accelerometers require two integrations and are consequently subject to drift in position measurement. Consequently, the combined use of sensors is appealing. For example, vision sensors, which are relatively slow but give direct position information for the end of a flexible member, can be combined with accelerometers, which give information about the same point at high sample rates but with noise and drift problems. The two sensor readings can be combined with a Kalman filter or other fusion scheme.

Issues Related to Model Order

Realizing that the theoretical order of a flexible arm is infinite, one should be prepared to deal with certain issues related to model order. The first of these is possible aliasing problems arising from finite sampling rates. Antialiasing filters should be employed, and even more effective is the use of analog damping by materials, damping treatments, or the back electromotive force (emf) of actuators to provide an inherent damping of the system under digital control. A more subtle problem is that of measurement and control spill over.

Control of Special Configurations: Macro/Micro Redundancy

Several special configurations have been proposed to enable improved controlled performance of flexible link arms. The problem with noncollocation noted above is a case in point. In some cases, a large range of motion and high accuracy cannot be achieved with a single set of actuators. Redundant degrees of freedom can then

be employed with the macro degrees of freedom that provide large range of motion but with inherent lack of precision or accuracy. Vibrations are particularly difficult to eliminate with a long arm, and construction accuracy and even thermal drift can exceed the tolerance limits. If the gross motion is completed quickly, residual vibrations may remain and be very lightly damped. Actuators constructed for gross motions of large axes do not have the bandwidth capabilities to damp these vibrations actively, but small motion axes typically do. This has been illustrated in several instances for long-reach manipulators when exploring nuclear waste cleanup solutions [11.71–74]. Using the inertial forces generated at the end of the arm also has the advantage of creating forces collocated with the measurements being used for control, assuming the base of the micro arm is at the end of the macro arm and the location of sensors. *Book* and *Loper* [11.73] and *George* and *Book* [11.75] have shown that this control is possible based on accelerometers in multiple degrees of freedom. It is also possible to excite undesired modes which have not been accounted for. If the full six degrees of freedom are used it is possible to argue stability based on a simple passive analogy. Generating prescribed forces at the base of an articulated micromanipulator is a challenging task, requiring the solution of a new type of inverse dynamics problem, which solves for motions or joint torques to produce the prescribed base forces and moments.

Command Generation for Flexible Link Arms via Command Shaping

The trajectory undertaken by a flexible arm can dramatically affect the consequences of flexibility. Purely open-loop anticipation of the flexible dynamics has been used to create motion profiles wherein the motion cancels the incipient oscillation that had been created by earlier motions. This strategy is referred to as command shaping. A more complicated strategy to implement is a true inverse-dynamics-based trajectory wherein the rigid and flexible states are all orchestrated to arrive at the desired point in state space, typically at a static equilibrium. The high order of the flexible model is further complicated by the non-minimum-phase nature of the noncollocated system and noncausal inverse is necessary to produce a physically realizable trajectory (VIDEO 778).

Time delay command shaping is a variation of the deadbeat control introduced by [11.76, 77]. It can be couched in terms of a finite impulse response filter with a simple impulse response as depicted in Fig. 11.11 with a single input impulse producing a small number (typically two to four) of output impulses with appropriate time spacing and amplitudes. Extensions by *Singer* and *Seering* [11.78] and *Singhose* et al. [11.79] have produced more robust varieties under the trademark Input Shaping. The robustness is enhanced by using more output impulses and appropriate selection of the impulse spacing. The optimal arbitrary time-delay (OAT) filter [11.80, 81] reduces this selection to a simple formula for linear vibrations that depends on the natural frequency and damping ratio of those modes to be eliminated from the response. The simplicity of implementation is illustrated in Fig. 11.12, where the relationships between the filter parameters and the mode parameters are

$$\begin{aligned}
\text{coefficient 1} &= \frac{1}{M} \\
\text{coefficient 2} &= \frac{-(2\cos(\omega_\mathrm{d} T_\mathrm{d})\,\mathrm{e}^{-\zeta\omega_\mathrm{n} T_\mathrm{d}})}{M} \\
\text{coefficient 3} &= \frac{\mathrm{e}^{-2\zeta\omega_\mathrm{n} T_\mathrm{d}}}{M} \\
M &= 1 - 2\cos(\omega_\mathrm{d} T_\mathrm{d})\,\mathrm{e}^{-\zeta\omega_\mathrm{n} T_\mathrm{d}} + \mathrm{e}^{-2\zeta\omega_\mathrm{n} T_\mathrm{d}} \\
\omega_\mathrm{d} = \omega_\mathrm{n}\sqrt{1-\zeta^2} &= \text{damped natural frequency} \\
\omega_\mathrm{n} &= \text{undamped natural frequency} \\
\zeta &= \text{damping ratio} \\
T_\mathrm{d} &= \text{time delay selected, an integer number of samples .}
\end{aligned} \tag{11.84}$$

Note that the formulation here is oriented toward digital systems with fixed sampling intervals such that the time delay of the filter is adjusted to an integer

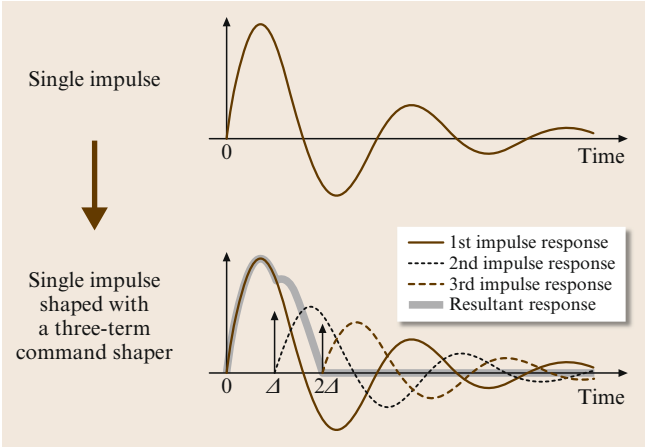


Fig. 11.11 Impulse response of a shaper and the effect on a vibratory system. Time is normalized by the intersample time T_s and T_d, chosen such that $\Delta = T_\mathrm{d}/T_\mathrm{s}$ is an integer

number of sampling intervals. Command shaping has been applied to a wide range of flexible systems including disk drives, cranes, and large robot arms, such as the example by CAMotion (VIDEO 777) as shown in Fig. 11.13. Note that this example contains a flexible drive belt, effectively a flexible joint, as well as flexible links that contribute to the compliance of the lowest mode that is quenched with an OAT filter.

Adaptive versions of command shaping have also been demonstrated. A version oriented toward repetitive motion applications is presented in [11.82].

Command Generation for Flexible Link Arms via Inverse Dynamics

In contrast to the scant system knowledge required for command shaping, the inverse dynamics of a flexible arm requires a great deal of knowledge about the system. The creation of an appropriate model was discussed above. An assumed modes model, for example, is well adapted to inverse dynamics if the rigid coordinates are not the joint angles, but the angles between lines connecting the joint axes (where it is assumed that joints are rotational). In this case, the desired end-of-arm motion can be expressed solely in terms of rigid variables, that is, the same variables $\boldsymbol{q}_{\mathrm{d}}$ used in the flexible joint case, and the desired flexible variables are found from the second-order equations driven by the known rigid positions and velocities. The remaining difficulty is the unstable zero dynamics of the system.

For the purposes of introduction, only the linear situation will be treated here in the fashion of [11.83]. First, the rigid coordinates and their derivatives are found from the desired end-of-arm motion profile. The flexible equations are separated into causal stable zeros and anticausal unstable zeros, and the stable inverse is solved forward in time. Physically the motion response responds after the torque inputs. The computational input however is the response and the torque is calculated as an output, so no physical laws are violated in solving the anticausal portion beginning at the final time when flexible coordinates are specified to be zero and proceeding to the initial time and beyond. In fact, the exact solution extends from negative infinity (where the anticausal solution decays) to positive infinity (where the causal solution decays). For realistic dynamics, the necessary time for the input is only slightly more than the motion time for the end of the arm because the zeros are often quite far from the origin in both directions. If the end-of-arm motion begins at time zero, shortly before that, the joint torques begin to *pre-shape* the flexible links without moving the end of arm. Similarly, when the end of the arm reaches its final position, the joint torques must release the arm deflection in a manner that is not seen from the

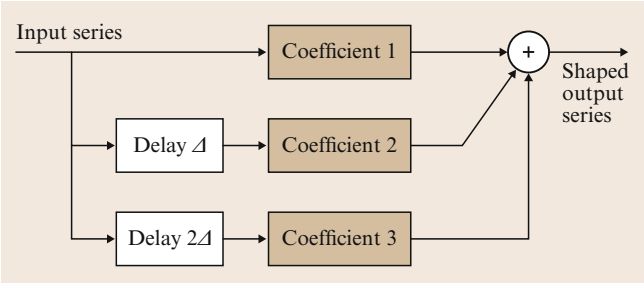


Fig. 11.12 Block diagram representation of the command shaping algorithm

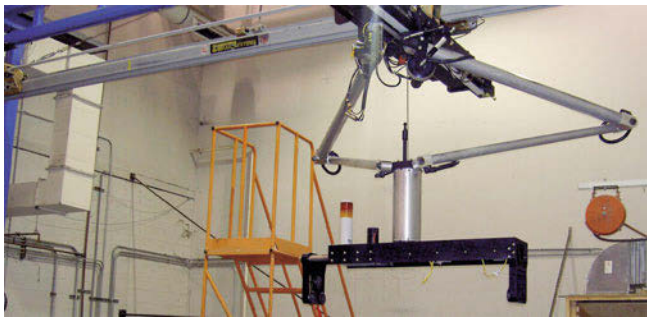


Fig. 11.13 A large flexible linking arm with command shaping

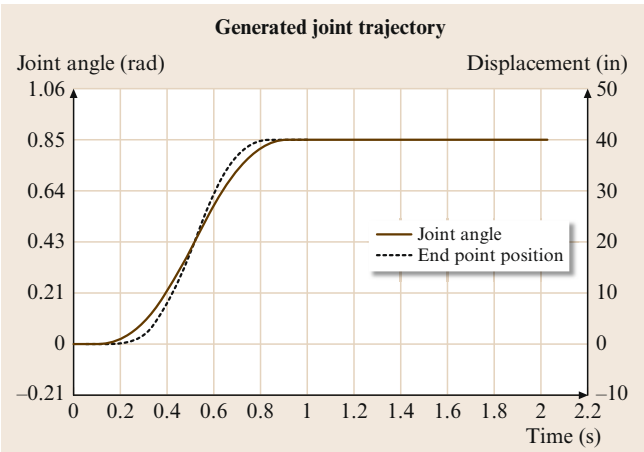


Fig. 11.14 Inverse dynamics driven motion of a single-link flexible arm

end of arm (or any other chosen output location). Figure 11.14 shows the example from *Kwon* [11.84] for a single-link arm and the VIDEO 778 shows the behavior improvement.

Since inversion of the flexible plant as described earlier containing nonminimum phase or *unstable* zeros can result in an anticausal input torque to achieve an arbitrary trajectory, it is appropriate to look for other trajectories with special characteristics that avoid these

problems. The location of zeros are dependent on where on the flexible link the trajectory is specified. *De Luca* et al. studied the one flexible link rotational arm and the migration of zeros as the point of interest is shifted from tip to joint. They utilize a point where zeros are nonexistent as the output point, a point that enables inversion to proceed without the causality problem [11.85–87]. When this point reaches the desired angle with zero deflection, the end point will also be at the desired angle. Joint velocity and deflection velocities must also be brought to zero resulting in a rest-to-rest motion completed in specified time and with no residual oscillations. The cited references show the equivalence of this development in the time or frequency domain for linear systems. For a two link arm (only the forearm is flexible), the nonlinear behavior demanded a time domain approach [11.86].

De Luca and *Di Giovanni* [11.85] address rest-to-rest motion of a single flexible link and carries through an example for a single flexible mode. The later reference [11.87] also provides experimental results for which a brief video (VIDEO 779) is available. The strength of this approach is its relative simplicity in achieving the desired final position described by a single polynomial. The degree of the single polynomial required to meet all the boundary conditions is $4n_e + 3$ where n_e is the number of flexible modes. One difficulty is the inability to efficiently constrain the peak torque required. Torque profiles developed in this simplest of strategies have high peaks but for most of the trajectory (at start and end) are relatively low. Hence the peak torque capabilities of the actuator may not be efficiently used. It is not necessary for a single polynomial to describe the entire trajectory, of course, but the complexity of developing the response without zeros becomes more challenging for piece wise polynomials, and arises from meeting all the boundary conditions on polynomial segments. The nonlinear case arising from a two link arm with the forearm flexible [11.86] provides further exploration of the inversion approach for flexible arms.

The model employed by De Luca varies slightly since deflection is measured from an axis passing through the dynamic center of mass of the flexible arm as shown in Fig. 11.15, not the tangent to the neutral axis of the beam at the rotary actuator. This modifies the boundary conditions on the shape equation and hence the mode shapes. For an introduction, a single flexible link is considered here. Variables are defined as follows:

- J_0, J_p: rotary inertia of the rotary actuator, tip payload, respectively
- m_p: mass of the tip payload
- $\theta, \theta_c, \theta_t$: angles through center of mass, tangent to actuated end, and through tip, respectively
- x: axis through the center of mass (COM)
- y: deflection perpendicular to x.

Consider the case with no damping on the modes. While modal damping is readily incorporated, its typically low values justify an abbreviated treatment here for the sake of space and simplicity. The output y specified is the sum of the angle (to the center of mass), the angular velocity, deflections and deflection rate, each times appropriate gains. These gains are found by specifying the transfer function from joint torque to y to be of maximum relative degree, i. e., it has no zeros. Equivalently, the successive derivatives of y do not explicitly involve the torque until the $(2n_e + 1)$-th derivative. Note that these characteristics ascribed to differentially flat systems may yield a more complete understanding of this approach [11.88, 89],

$$\frac{y(s)}{\tau(s)} = \frac{K}{s^2 \prod_{i=1}^{n_e} \left(s^2 + \omega_i^2\right)} ,$$

$$K = \frac{1}{J} \prod_{i=1}^{n_e} \omega_i^2 .$$

When transformed to the time domain, the resulting form of the torque based on derivatives of the desired output will be

$$\tau_d(t) = \frac{J}{\prod_{i=1}^{n_e} \omega_i^2} \left[y_d^{2(n_e+1)}(t) + \sum_{i=0}^{2n_e-1} \alpha_i y_d^{(i+2)}(t) \right] .$$

The α_i values are readily found from the convolution of the polynomial or expansion of the denominator of. $y(s)/\tau(s)$. But what must be used for y_d? With the given definition of y, the desired value of y_d for the rest-to-rest trajectory will be achieved by creating an interpolating trajectory with the desired initial and final values of y_d

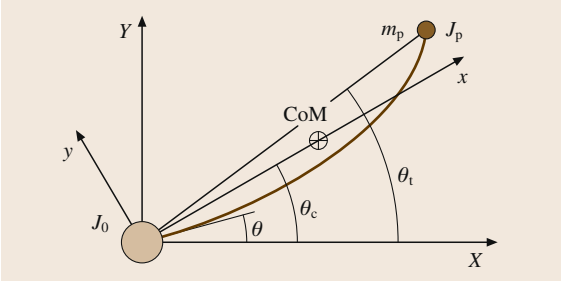


Fig. 11.15 A definition of variables for the causal inversion of De Luca

and derivatives of y_d at the initial and final times that are zero. The number of derivatives to set equal to zero are at least $2n_e + 1$, but it may be preferred to set additional derivatives equal to zero to obtain a smoother torque at the ends of the trajectory. As explained in [11.85] this is a transformation of states that results also in the desired rest-to-rest behavior of the joint angle, the deflection and their derivatives. For more robustness, the feedforward command should be supplemented with feedback control, for example a simple PD or PID control of the joint angle to follow y_d.

Shown in Fig. 11.16 are the torque and resulting bending deflection for a 2.2 s move using this combination on a single link flexible arm [11.87].

To best comprehend these results, some details of the flexible arm in the experiment are needed. The desired trajectory was defined by a polynomial of degree 19. The arm length was 0.655 m, of sheet steel 2 mm thick and 51 mm wide, directly driven by a DC motor with inertia of $1.888 \times 10^{-3}\,\mathrm{kg\,m^2}$ and no tip mass. The resulting natural frequencies were 14.4, 13.2 and 69.3 Hz with damping ratios of 0.0001, 0.001, and 0.008, respectively. Frequency errors of the model were less than 1%. Friction compensation was applied based on static and viscous models of the friction under continuous rotation.

Feedback Control of Flexible Links

Flexibility in the links is in some cases controlled in the same way as concentrated flexibility in joints. Herein the focus is on approaches that apply specifically to link flexibility; consequently the use of Kalman filtering to estimate tip position and other broadly used techniques are not reexamined. A long history of research on this topic exists. *Cannon* and *Schmitz* [11.90], *Truckenbrot* [11.91], and others showed that these techniques work for flexible link arms and that the noncollocated nature of the system aggravates the sensitivity to modeling errors.

Strain measurements provide a valuable indication of the state of a flexible link, and strain rate, obtained from filtered differentiation of the typically noisy strain signal, was shown by *Hastings* and *Book* [11.92] to be an appropriate way of overcoming the limitations imposed by purely joint feedback control. *Yuan* et al. [11.93] produced an adaptive version of strain-rate feedback.

Strain measurements readily lend themselves to calculation of the modal amplitudes, and the number of appropriately placed strain gage measurements N can be equal to the number of mode amplitudes to be evaluated. In this case, no estimation algorithm is necessary, although filtering is usually needed. If the mode shape φ_i of a beam is known, strain is proportional to the second derivative of φ_i

$$\alpha_{ij} = \left.\frac{\partial^2 \varphi_i}{\partial x^2}\right|_{x_j} . \tag{11.85}$$

By inverting the coefficient matrix thus formed, it is simple to obtain the amplitude of N modes as

$$\begin{pmatrix} \delta_1 \\ \delta_2 \\ \vdots \\ \delta_N \end{pmatrix} = \begin{pmatrix} \alpha_{11} & \alpha_{12} & \dots & \alpha_{1N} \\ \alpha_{21} & \alpha_{22} & \dots & \alpha_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{N1} & \alpha_{N2} & \dots & \alpha_{NN} \end{pmatrix}^{-1} \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{pmatrix} . \tag{11.86}$$

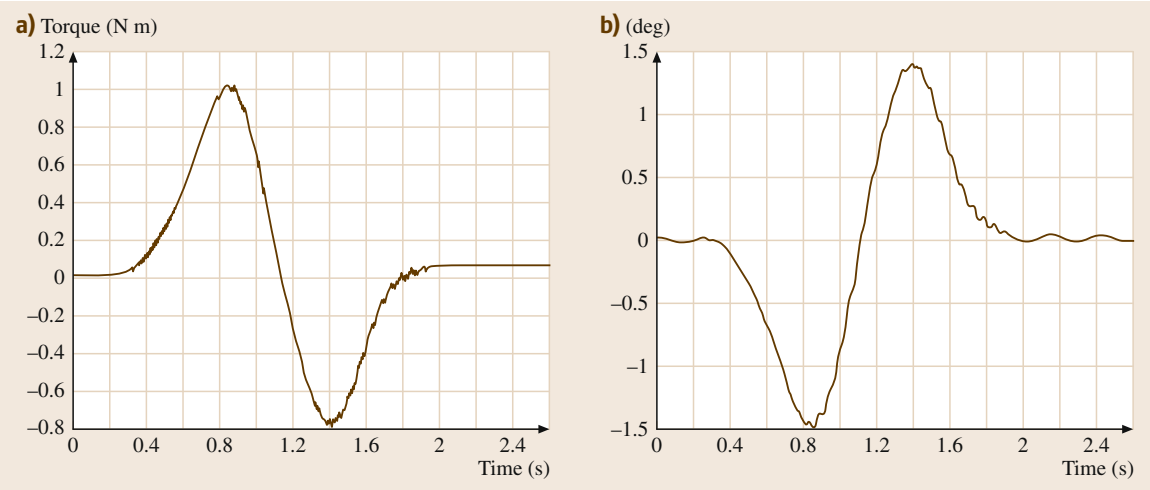


Fig. 11.16 **(a)** Applied torque for rest to rest motion with PD control. **(b)** Bending deformation at the tip

Additional attention is necessary if beam torsion is a significant part of the mode deflection, but the principle still applies. Active control of high-frequency modes may not be warranted, and feedback of the one or two lowest-frequency modes can be managed by classical or state-space design techniques to increase the damping of that mode.

The use of the measurement of the end-of-arm position has perpetual appeal and is very effective for rigid arms for the elimination of variability due to arm distortion and the change over time of arm calibration. Machine vision can eliminate uncertainty in the target position as well as the arm position. These approaches are plagued by the noncollocated and hence non-minimum-phase dynamics of link flexibility, however, and may require particular caution. *Wang* and *Vidyasagar* [11.94] proposed an alternative formulation of EOA measurements in the control formulation that he called the reflected tip position. For a rotating link of angle θ and length l, the deflected position is $l\theta + \delta$ while the reflected tip position $l\theta - \delta$ is minimum phase. This combination of joint and tip measurements produced the desired undeflected position but would not serve for EOA tracking control. *Obergfell* and *Book* [11.95] synthesized an output based on strain feedback combined with EOA position that allowed tracking control as well. They first produced a passive system with deflection feedback obtained from optical measurements of the link deflection, and then devised a traditional feedback control of the tip position measured by machine vision of the large two link arm shown in Fig. 11.17 for the experimental robot RALF (robot arm long and flexible). Figures 11.18 and 11.19 compare the path for a large square traced in a plane with only the deflection feedback and with both deflection and EOA position feedback.

Fig. 11.17 Robotic arm large and flexible (RALF) used in EOA feedback experiments

Robust Estimation for Flexible Link Systems

The state feedback algorithms readily designed for linear flexible link systems are dependent on information about states that cannot be directly measured and are theoretically infinite in number. Truncation of the states to represent the most dominant low frequency modes is a ready solution to the second problem and state estimators or observers present a viable approach to the first problem. (Estimator and observer are commonly used interchangeably, but the term observer is more common will be used in this section.) For linear systems a Luenberger observer or the stochastically based Kalman filter are commonly used. Unfortunately, observers require an accurate system model or the estimates may be poor and the observer, a dynamic system

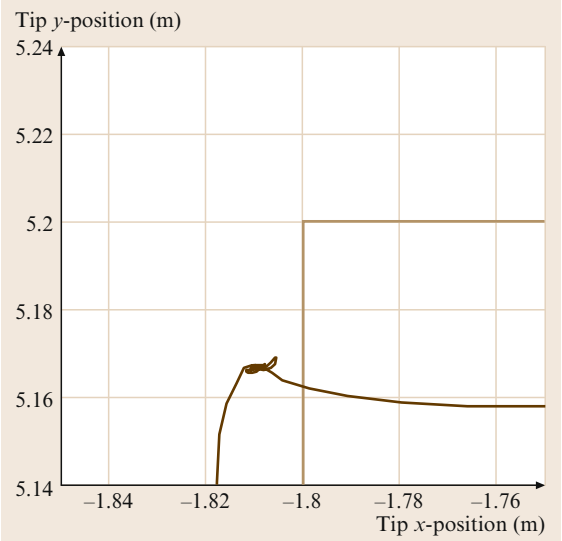


Fig. 11.18 Path followed without EOA position feedback. (Commanded trajectory is a *square* partially shown in *light brown*. Actual trajectory shown in *dark brown*)

of order as high as the order of the system itself, may even be unstable. Hence the robustness of the state observer is a critical subject for the control of flexible link systems. The quality of the control is ultimately the best measure of the quality of the estimator. Robust observer design has been the subject of considerable research and, in large part, mirrors the development of robust controllers. This is natural since an observer is a dynamic system which, like the plant itself, can be controlled to achieve a desired outcome. *Control effort* of the observer is utilized to drive convergence of the estimated outputs of the observer to the measured outputs of the real system. Hence research on estimators based on H-infinity, fuzzy logic, neural network, sliding mode, adaptive, and other design techniques abound. In this chapter, two types of observers will be discussed: sliding mode observers and multiple model observers.

For all observers the placement of sensors is crucial and for estimation of multiple modes it is desirable, although not theoretically essential, to use multiple sensors. Joint sensors (position and velocity) are typically available for general pose (joint) control, but for deflection sensing strain gages and accelerometers are currently most appropriate. The MEMS accelerometers now readily available and inexpensive will be the principle choice here. Strain gages are more fragile and generally provide more complex readings of the mode behavior. Accelerometers are complicated by the fact that acceleration is not dependent on just states z which include the mode amplitudes and their derivatives, but also on the input u creating the link motion. Acceleration measurement of a point p on the link is affected by both the states z and the input u as follows,

$$\ddot{x}_p = \left\{-\beta(p)[\Phi][\omega^2] - \beta(p)[\Phi][\hat{C}]\right\} z + \left\{\beta(p)[\Phi][\Phi]^T Q\right\} u \,,$$

where $\ddot{x}_p$ is the acceleration in the direction perpendicular to the beam axis, Φ is the matrix of eigenvectors, z is the state vector, composed of joint motion as well as beam deflections, p is the location of the sensor along the beam axis, ω^2 is the diagonal matrix of natural frequencies squared, $\hat{C}$ is the modal damping matrix, $\beta(p)$ expresses the sensor motion in terms of the modal states variables, and Q relates the input to the system states.

As *Post* [11.96] has explained, the ability of a given sensor to detect a mode depends on the amplitude of the mode at the sensor location. Overall the singular values of the observability grammian, also used for testing the system observability, is a reasonable but not perfect metric. A robust measurement requires the placement of sensors so as to avoid nodes (zero crossings) of the modes to be sensed even when parameters of the system change or are inaccurate. A single sensor (strain gage or accelerometer) is much more likely to have this problem than multiple sensors. The next sections depend on suitably placed sensors. For those modes not being controlled, one should minimize their influence by placing sensors near the node of those modes. This helps to avoid spillover of these modes that interferes with the system as approximated.

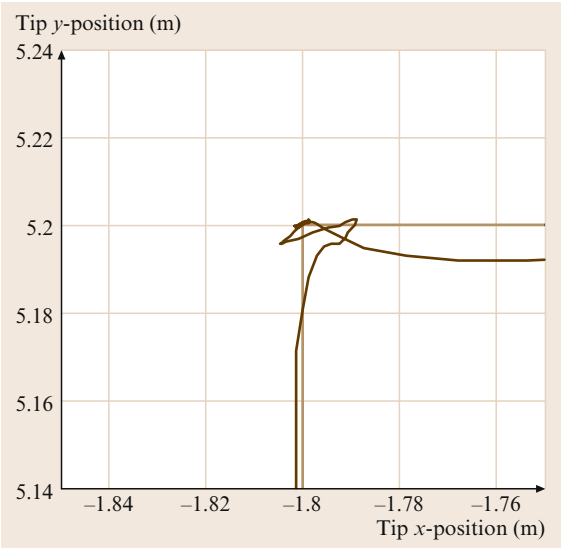


Fig. 11.19 Path followed with EOA feedback from camera. (Commanded trajectory is a *square* partially shown in *light brown*. Actual trajectory shown in *dark brown*)

Parametric Errors in System Model

Parametric errors are inaccuracies in the parameters of a model with an accurate representation of the system structure. For flexible link systems this might arise from inaccurate payload mass, stiffness values, stiffness distribution, boundary conditions, etc., all of which lead to inaccurate mode shapes, natural frequencies and input coefficients. Errors are inevitable to some degree. As a consequence of parametric errors, one of the most powerful results of linear feedback control of estimated states, the separation principle which says the design of the observer does not change the eigenvalues of the controlled system and vice versa, is compromised. The rapid convergence of observer error to zero will in general not occur with parametric error. The net effect is a major blow to the effectiveness of the classical Luenberger observer. The Kalman filter, which optimizes observer gains under conditions of Gaussian process and measurement noise, is likewise

compromised. Without good state estimates the effectiveness of the powerful state feedback approaches are in jeopardy.

One currently popular tack to improve the robustness of observers is the sliding mode observer. There are several variations of this approach, and the technique by *Wallcot* and *Zak* [11.97, 98] was followed by *Post* [11.96] in exploring the consequences of parametric error. As typical, a sliding surface is defined which is intended to guide the estimate error to zero. The reaching phase moves from the initial error state to the sliding surface and when that surface is crossed the sign on the control changes. This rapid switching may be modified by establishing a *boundary layer* near the switching surface that substitutes a linear behavior.

In this section, we will use z as the state vector, $\hat{z}$ as its estimate, L as the feedback of the observer, y as the measurement and K_S as the sliding mode gain. The *hat* on a variable will generally denote its estimate and an over dot its time derivative. The symbol Δ preceeding a symbol indicates the variation of the true system parameter from its model. The linear plant model is thus

$$\dot{z} = Az + Bu\,; \quad y = Cz + Du\,.$$

The equation approaching the switching surface is

$$\dot{\hat{z}} = A\hat{z} + Bu + L(y - \hat{y}) + K_s \mathrm{sgn}(y - \hat{y})\,.$$

When the boundary layer is included, the control is changed in a region near the switching surface where the design is essentially a conventional linear observer. For present purposes we will not employ a boundary layer but examine the pure sliding mode case, for which the error equation is

$$\dot{e} = A_0 e - K_s \mathrm{sgn}\,(Ce) + \Delta Az + \Delta Bu\,,$$

which clearly has a steady state error solution that is not zero except in the case that $\Delta Az = -\Delta Bu$ which tends to occur when the overall system comes to rest. Thus the convergence of the sliding mode observer has similar issues found with the conventional linear observers. They can be stable but can converge to the wrong steady state value leading to error, particularly after the observer's state estimates should have converged to the slower changing system state. However, when the overall plant reaches steady state and $z = 0$, $u = 0$, the observer error will also converge to zero. In contrast, the usual intent is to design the observer to quickly converge to the true state estimate in order to appropriately control the plant.

Successful Estimation with Model Parameter Uncertainty

Models are seldom, if ever, exact in their representation of a physical system. In addition to error in parameter identification, parameters can change with time and operating conditions. For example, robots acquire and release payloads and move to different poses which may change parameters. As reported before, convergence of state observers is adversely affected by these changes which can lead to poor performance and even instability. In a recent work, *Post* [11.96] described a multiple model estimation scheme that successfully overcomes these limitations for a flexible link robot in both simulation and experiment. Other authors, e.g., *Chalhoub* and *Kfoury* [11.99] have used fuzzy logic techniques to address the needs for improving sliding mode observers.

Since the true state of the system is not known, the steady state estimation error cannot be calculated, but if it is known to exist, Post has shown it to be proportional to parameter error. Hence multiple models based on a different set of sensors can be used to establish which is most accurate. The difference in estimated states based on two sensor selections $\delta_z(t) = \hat{z}_{s_1} - \hat{z}_{s_2}$ is then a metric for choosing between alternative observers in an array of observers with different assumed parameters.

The observers themselves (conventional Luenberger observers or Kalman filters) have the form

$$\dot{\hat{z}}_{s_1}(t) = A\hat{z}_{s_1}(t) + Bu(t) + L_{s_1}\left[y_{s_1}(t) - \hat{y}_{s_1}(t)\right],$$

where $\hat{z}_{s_1}$ is the estimate of the state vector z using the sensor array 1, A is the plant matrix, B is the input matrix, L_{s_1} is the gain matrix to be designed, y_{s_1} is the sensor measurement of sensor array 1, and $\hat{y}_{s_1}$ estimate of sensor measurement of sensor array 1, and similarly for sensor s_2. The difference between states and their estimates converges rapidly (relative to the states themselves) to steady state and hence $\dot{\hat{z}}_{s_1} - \dot{\hat{z}}_{s_2}$ converges to zero and hence

$$\begin{aligned}\delta_z(t) = {} & \chi_1[\Delta Az(t) + \Delta Bu] \\ & + \chi_2[\Delta C_{s_1} z(t) + \Delta D_{s_1} u(t)] \\ & + \chi_3[\Delta C_{s_2} z(t) + \Delta D_{s_2} u(t)]\end{aligned}$$

is found to be the *steady state* slowly changing (relative to the observer dynamics) difference in the two observers based on sensors s_1 and s_2. Here, the χ_i are constant matrices based on the model parameters and not the true values of the system. Δ indicates the deviation of the true matrices from the model's matrices following that symbol. Note that subscripts s_1 and s_2 indicate different sensors involved in the measurement

and feed through matrices C and D, respectively. Also note that this equation is somewhat complicated by the feedthrough matrices occurring due to the use of accelerometers to sense the motion.

Incorporating the specific representation of a linear flexible manipulator, the equations are

$$\dot{z} = \begin{pmatrix} 0_{(n\times n)} & I_{(n\times n)} \\ -\omega^2_{(n\times n)} & -\hat{C}_{(n\times n)} \end{pmatrix} z + \begin{pmatrix} 0_{(n\times m)} \\ \hat{Q}_{(n\times m)} \end{pmatrix} u \,,$$

where $\hat{C}$ denotes modal damping and $\hat{Q}$ denotes the modal forcing coefficient. The sensor's output is

$$\begin{aligned} y_{Si} = \Big\{ &- [1, \psi_1(p_{si}), \ldots, \psi_n(p_{si})]\, \Phi_{(n\times n)} \omega^2_{(n\times n)} \\ &- [1, \psi_1(p_{si}), \ldots, \psi_n(p_{si})]\, \Phi_{(n\times n)} \hat{C}_{(n\times n)} \Big\} z \\ &+ [1, \psi_1(p_{si}), \ldots, \psi_n(p_{si})]\, \Phi_{(n\times n)} \hat{Q}_{(n\times n)} u \,. \end{aligned}$$

Let $\beta_{Si} = [1, \psi_1(p_{si}), \ldots, \psi_n(p_{si})]$ and one can attribute to the sub-matrices the parametric change matrices

$$\begin{aligned} &(\Delta A z(t) + \Delta B u) = \\ &\quad \begin{pmatrix} 0_{(n\times n)} \\ I_{(n\times n)} \end{pmatrix} \left[\left(-\Delta\omega^2 - \Delta\hat{C} \right) z(t) + \Delta\hat{Q} u(t) \right], \\ &(\Delta C_{Si} z(t) + \Delta D_{Si} u) = \\ &\quad \beta_{Si} \Phi \left[\left(-\Delta\omega^2 - \Delta\hat{C} \right) z(t) + \Delta\hat{Q} u(t) \right], \\ &\delta_z(t) = \left[\chi_1 \begin{pmatrix} 0_{(n\times n)} \\ I_{(n\times n)} \end{pmatrix} + \chi_2 \beta_{s_1} \Phi + \chi_3 \beta_{s_2} \Phi \right] \\ &\quad \times \left[\left(-\Delta\omega^2 - \Delta\hat{C} \right) z(t) + \Delta\hat{Q} u \right], \end{aligned}$$

which will be coalesced to give

$$\delta_z(t) = \Lambda_{(2n\times n)} \gamma(t) \,,$$

where it can be seen that Λ consists of known model parameters and γ is an unknown metric of parameter errors that can't be solved from this equation. However, using the pseudo inverse we can produce a least squares metric $\gamma_{\mathrm{m}}(t)$ which evaluates the relative accuracy of two observers based on different models

$$(\Lambda^{\mathrm{T}}\Lambda)^{-1}\Lambda^{\mathrm{T}}\delta^*_z(t) = \gamma_{\mathrm{m}}(t) \,.$$

The smaller the norm of $\gamma_{\mathrm{m}}(t)$, the more accurate the observer based on a given model is expected to be.

While various methods of using $\gamma_{\mathrm{m}}(t)$ can be contrived, *Post* [11.96] utilized a switching approach he referred to as the multiple model switching adaptive estimator (MMSAE), visualized in Fig. 11.20. The weighting value w_i is based on $\gamma_{\mathrm{m}i}(t)$ for the observer

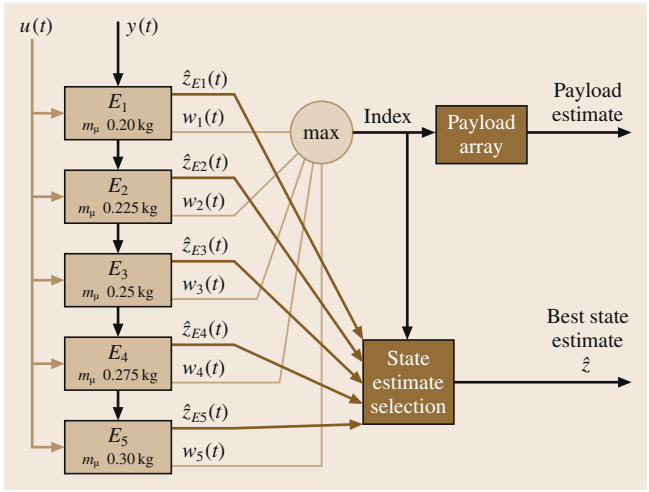


Fig. 11.20 Flow chart for the multiple model switching adaptive estimator (MMSAE)

based on a given model, then the estimate $\hat{z}_i$ from that observer was used. Figure 11.21 shows schematically the flow inside the estimator blocks E_1 through E_5. The parameter varied in Post's experiments was arm payload. Multiple parameters could be varied but would lead to a higher dimensional formulation and poten-

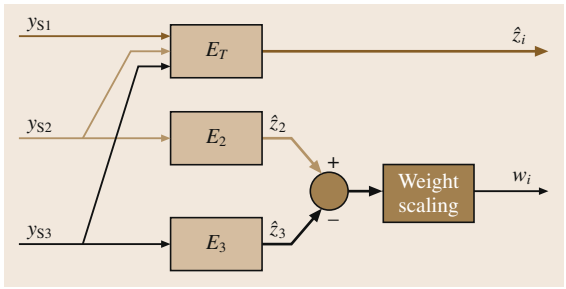


Fig. 11.21 Estimation module with weight scaling

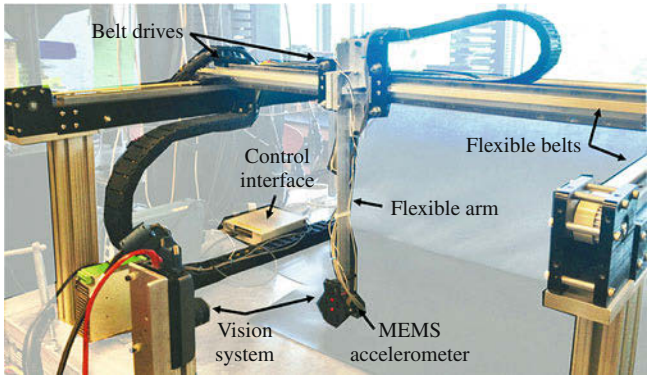


Fig. 11.22 Gantry robot used in MMSAE experiments (after *Post* [11.96])

tially many estimators numbering $N_T = N_1 N_2 \cdots N_p$ for the variation of p parameters, each with N_i values represented.

Post [11.96] conducted experiments with a linear model of the gantry robot as illustrated in Fig. 11.22 and schematically modeled in Fig. 11.23, with the parameters given in Table 11.1 and sensor placements at 0.015, 0.25, and 0.42 m from the actuated end of the beam. He compared:

- Conventional PID (VIDEO 780).
- Fixed gain linear-quadratic regulator (LQR) plus Luenberger state space designs (VIDEO 781).
- MMSAE design observers and LQR controls. Five potential payloads were chosen with spacing in even increments between 0.2 and 0.3 kg, i. e., an endpoint mass between 0.35 and 0.55.

The penalty matrices were kept the same for the LQR controls but the resulting gains changed based on the model employed by the MMSAE selection. While various measures of performance could be demonstrated involving somewhat arbitrary selection of controller types, controller gains and metrics, the following characteristics are considered most critical:

- Stability with various payloads.
- Cycle completion time, with requirements for settling of vibration amplitude at prescribed stops.
- Variation of cycle times with changing payloads.

Since reasonable work cycles often pick up and drop off payloads, the test included these operations. Cycle times are graphed in Fig. 11.24. The light bars show the cycle times for the complete cycle with segments of the task carrying the payload and others with the gripper empty. The dark bars show operation with no payload carried and only the gripper mass present for the entire cycle. In addition to the pair of bars labeled *PID* and *MMSAE* (with obvious meanings) are pairs labeled *SS 0.25* through *SS 0.40*. These latter labels refer to the *designed-for* tip mass (payload plus gripper). The most obvious message from this figure is that the state space control is a vast improvement over PID, reducing cycle time by about 90%. What is not obvious from this figure is that some means of gain change is needed to accommodate the payload variations. With a designed-for tip mass of 0.45 kg and above, the operation without payload was unstable. This is still below the actual tip mass of 0.5 kg. Hence MMSAE was essential for a successful completion of the task with these design criteria (i. e., LQR penalty matrices and resulting gains). The exact tip mass does not need to be known, only the range of masses. The near identical cycle times with and without the additional payload seen on the chart for MMSAE is also a practical advantage.

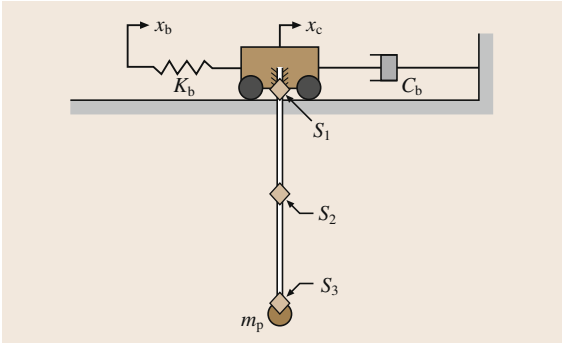


Fig. 11.23 Schematic of gantry robot model

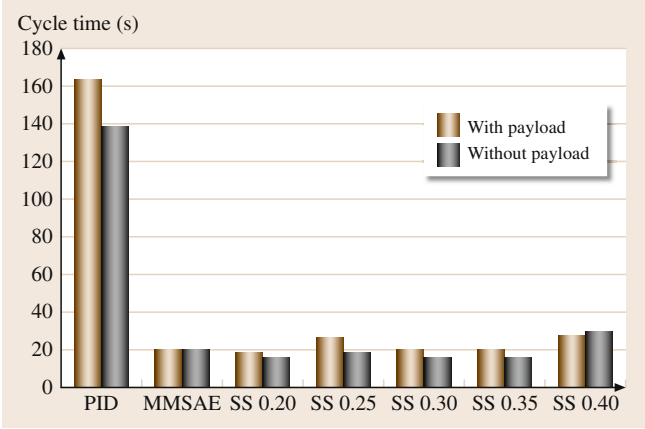


Fig. 11.24 Cycle times with implemented control methods

11.2.4 Further Reading

The topic of flexible link robot control is widely covered with papers on modeling and control. The reader wishing to probe further may want to begin with a paper giving some perspective on the nature of the problem. This was the goal of *Book* in [11.100]. Progress to date in this field from the theoretical perspective is signifi-

Table 11.1 Parameters of the gantry robot model

Parameter	Value	Unit
Payload (m_t)	0.281	kg
Cart mass (m_c)	10	kg
Arm length (L)	0.42	m
Elastic modulus (E)	7×10^{-10}	N/m
Density (ρ)	2700	kg/m^3
Area moment (I)	1.0114×10^{-10}	m^4
Belt stiffness (K_b)	2.1814×10^5	N/m
Cart damper (C_d)	100	N s/m
Structural damping coefficient (γ)	0.0005	N s/m

cant and has been collected by *Canudas de Witt* et al. in [11.101]; this reference includes a wide range of research approaches, but the examples of the practical application of this theory are more limited. The German Aerospace Center has produced lightweight space arms [11.102] and deployed them in space. The startup CAMotion, Inc. [11.103] has produced large lightweight industrial robots using the command-shaping approach to improve cycle time. High-speed arms such as the FlexPicker [11.104] achieve a light weight and avoid flexibility because of their parallel structure, but suffer the usual disadvantages of parallel actuation: restricted workspace considering the space occupied by the robot structure.

Trautt and *Bayo*'s early exploration of non-minimum-phase inverses is described in [11.105]. This is a particularly interesting variation of the work by *Kwon* [11.83, 84] in that it uses the frequency domain and deals with nonzero initial conditions. The concept of using two time scales, a fast time scale for the flexible dynamics and a slow time scale for the rigid dynamics, can be found in *Siciliano* and *Book* [11.106]. The singular perturbation approach was used earlier by Spong for elastic joint arms. *Ghorbel* and *Spong* also consider flexible links and contact with a rigid environment in [11.107]. Achieving robustness is a problem, particularly with end-point sensors. Sliding mode control [11.108] and passivity-based control [11.109] address this problem in effective but widely varying ways. While these seem to be the more promising methods of control, one will find many more approaches to this challenging control problem. H_∞, adaptive, fuzzy, and other controllers have all been applied with varying degrees of success.

Video-References

VIDEO 133 Cartesian impedance control with damping off
available from http://handbookofrobotics.org/view-chapter/11/videodetails/133

VIDEO 134 Cartesian impedance control with damping on
available from http://handbookofrobotics.org/view-chapter/11/videodetails/134

VIDEO 135 Control laws for a single-link arm with an elastic joint
available from http://handbookofrobotics.org/view-chapter/11/videodetails/135

VIDEO 136 Feedforward/feedback law for path tracking with a KUKA KR15/2 robot
available from http://handbookofrobotics.org/view-chapter/11/videodetails/136

VIDEO 770 Trajectory generation and control for a KUKA IR 161/60 robot
available from http://handbookofrobotics.org/view-chapter/11/videodetails/770

VIDEO 777 Input shaping on a lightweight gantry robot
available from http://handbookofrobotics.org/view-chapter/11/videodetails/777

VIDEO 778 Inverse dynamics control for a flexible link
available from http://handbookofrobotics.org/view-chapter/11/videodetails/778

VIDEO 779 Rest-to-rest motion for a flexible link
available from http://handbookofrobotics.org/view-chapter/11/videodetails/779

VIDEO 780 PID response to impulse in presence of link flexibility
available from http://handbookofrobotics.org/view-chapter/11/videodetails/780

VIDEO 781 State feedback response to impulse in presence of link flexibility
available from http://handbookofrobotics.org/view-chapter/11/videodetails/781

References

11.1 L.M. Sweet, M.C. Good: Redefinition of the robot motion control problem, IEEE Control Syst. Mag. **5**(3), 18–24 (1985)

11.2 M.C. Good, L.M. Sweet, K.L. Strobel: Dynamic models for control system design of integrated robot and drive systems, ASME J. Dyn. Syst. Meas. Control **107**, 53–59 (1985)

11.3 E. Rivin: *Mechanical Design of Robots* (McGraw-Hill, New York 1988)

11.4 S. Nicosia, F. Nicolò, D. Lentini: Dynamical control of industrial robots with elastic and dissipative joints, 8th IFAC World Congr., Kyoto (1981) pp. 1933–1939

11.5 P. Tomei: An observer for flexible joint robots, IEEE Trans. Autom. Control **35**(6), 739–743 (1990)

11.6 R. Höpler, M. Thümmel: Symbolic computation of the inverse dynamics of elastic joint robots, IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 4314–4319

11.7 M.W. Spong: Modeling and control of elastic joint robots, ASME J. Dyn. Syst. Meas. Control **109**, 310–319 (1987)

11.8 S.H. Murphy, J.T. Wen, G.N. Saridis: Simulation and analysis of flexibly jointed manipulators, 29th IEEE Conf. Decis. Control, Honolulu (1990) pp. 545–550

11.9 R. Marino, S. Nicosia: *On the Feedback Control of Industrial Robots with Elastic Joints: A Singular Perturbation Approach* (Dipartimento di Ingegneria Elettronica, Univ. Rome Tor Vergata, Rome 1984), Rep. R-84.01

11.10 G. Cesareo, F. Nicolò, S. Nicosia: DYMIR: A code for generating dynamic model of robots, IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1984) pp. 115–120

11.11 A. De Luca: Feedforward/feedback laws for the control of flexible robots, IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000) pp. 233–240

11.12 G. Buondonno, A. De Luca: A recursive Newton–Euler algorithm for robots with elastic joints and its application to control, IEEE/RSJ Int. Conf. Intell. Robots Syst., Hamburg (2015) pp. 5526–5532

11.13 H.B. Kuntze, A.H.K. Jacubasch: Control algorithms for stiffening an elastic industrial robot, IEEE J. Robotics Autom. **1**(2), 71–78 (1985)

11.14 S.H. Lin, S. Tosunoglu, D. Tesar: Control of a six-degree-of-freedom flexible industrial manipulator, IEEE Control Syst. Mag. **11**(2), 24–30 (1991)

11.15 P. Tomei: A simple PD controller for robots with elastic joints, IEEE Trans. Autom. Control **36**(10), 1208–1213 (1991)

11.16 A. De Luca, B. Siciliano: Regulation of flexible arms under gravity, IEEE Trans. Robotics Autom. **9**(4), 463–467 (1993)

11.17 A. De Luca, B. Siciliano, L. Zollo: PD control with on-line gravity compensation for robots with elastic joints: Theory and experiments, Automatica **41**(10), 1809–1819 (2005)

11.18 C. Ott, A. Albu-Schäffer, A. Kugi, S. Stramigioli, G. Hirzinger: A passivity based Cartesian impedance controller for flexible joint robots – Part I: Torque feedback and gravity compensation, IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 2659–2665

11.19 L. Zollo, B. Siciliano, A. De Luca, E. Guglielmelli, P. Dario: Compliance control for an anthropomorphic robot with elastic joints: Theory and experiments, ASME J. Dyn. Syst. Meas. Control **127**(3), 321–328 (2005)

11.20 A. Albu-Schäffer, C. Ott, G. Hirzinger: A passivity based Cartesian impedance controller for flexible joint robots – Part II: Full state feedback, impedance design and experiments, IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 2666–2672

11.21 C. Ott, A. Albu-Schäffer, A. Kugi, G. Hirzinger: On the passivity-based impedance control of flexible joint robots, IEEE Trans. Robotics **24**(2), 416–429 (2008)

11.22 A. De Luca, F. Flacco: A PD-type regulator with exact gravity cancellation for robots with flexible joints, IEEE Int. Conf. Robotics Autom. (ICRA), Shanghai (2011) pp. 317–323

11.23 R. Kelly, V. Santibanez: Global regulation of elastic joint robots based on energy shaping, IEEE Trans. Autom. Control **43**(10), 1451–1456 (1998)

11.24 J. Alvarez-Ramirez, I. Cervantes: PID regulation of robot manipulators with elastic joints, Asian J. Control **5**(1), 32–38 (2003)

11.25 A. De Luca, S. Panzieri: Learning gravity compensation in robots: Rigid arms, elastic joints, flexible links, Int. J. Adapt. Control Signal Process. **7**(5), 417–433 (1993)

11.26 A. Albu-Schäffer, G. Hirzinger: A globally stable state feedback controller for flexible joint robots, Adv. Robotics **15**(8), 799–814 (2001)

11.27 L.E. Pfeffer, O. Khatib, J. Hake: Joint torque sensory feedback in the control of a PUMA manipulator, IEEE Trans. Robotics Autom. **5**(4), 418–425 (1989)

11.28 M. Hashimoto, Y. Kiyosawa, R.P. Paul: A torque sensing technique for robots with harmonic drives, IEEE Trans. Robotics Autom. **9**(1), 108–116 (1993)

11.29 T. Lin, A.A. Goldenberg: Robust adaptive control of flexible joint robots with joint torque feedback, IEEE Int. Conf. Robotics Autom. (ICRA), Nagoya (1995) pp. 1229–1234

11.30 M.G. Forrest-Barlach, S.M. Babcock: Inverse dynamics position control of a compliant manipulator, IEEE J. Robotics Autom. **3**(1), 75–83 (1987)

11.31 K.P. Jankowski, H. Van Brussel: An approach to discrete inverse dynamics control of flexible-joint robots, IEEE Trans. Robotics Autom. **8**(5), 651–658 (1992)

11.32 W.M. Grimm: Robustness analysis of nonlinear decoupling for elastic-joint robots, IEEE Trans. Robotics Autom. **6**(3), 373–377 (1990)

11.33 A. De Luca, R. Farina, P. Lucibello: On the control of robots with visco-elastic joints, IEEE Int. Conf. Robotics Autom. (ICRA), Barcelona (2005) pp. 4297–4302

11.34 A. De Luca: Decoupling and feedback linearization of robots with mixed rigid/elastic joints, Int. J. Robust Nonlin. Control **8**(11), 965–977 (1998)

11.35 A. De Luca, B. Siciliano: Inversion-based nonlinear control of robot arms with flexible links, AIAA J. Guid. Control Dyn. **16**(6), 1169–1176 (1993)

11.36 A. De Luca: Dynamic control of robots with joint elasticity, IEEE Int. Conf. Robotics Autom. (ICRA), Philadelphia (1988) pp. 152–158

11.37 S. Nicosia, P. Tomei: On the feedback linearization of robots with elastic joints, 27th IEEE Conf. Decis. Control, Austin (1988) pp. 180–185

11.38 A. De Luca, P. Lucibello: A general algorithm for dynamic feedback linearization of robots with elastic joints, IEEE Int. Conf. Robotics Autom. (ICRA), Leuven (1998) pp. 504–510

11.39 J. Swevers, D. Torfs, M. Adams, J. De Schutter, H. Van Brussel: Comparison of control algorithms for flexible joint robots implemented on a KUKA IR 161/60 industrial robot, 5th Int. Conf. Adv. Robotics, Pisa (1991) pp. 120–125

11.40 K. Khorasani, P.V. Kokotovic: Feedback linearization of a flexible manipulator near its rigid body manifold, Syst. Control Lett. **6**, 187–192 (1985)

11.41 M.W. Spong, K. Khorasani, P.V. Kokotovic: An integral manifold approach to the feedback control of

flexible joint robots, IEEE J. Robotics Autom. **3**(4), 291–300 (1987)
11.42 S. Nicosia, P. Tomei: Design of global tracking controllers for flexible-joint robots, J. Robotic Syst. **10**(6), 835–846 (1993)
11.43 B. Brogliato, R. Ortega, R. Lozano: Global tracking controllers for flexible-joint manipulators: A comparative study, Automatica **31**(7), 941–956 (1995)
11.44 M.W. Spong: Adaptive control of flexible joint manipulators, Syst. Control Lett. **13**(1), 15–21 (1989)
11.45 F. Ghorbel, J.Y. Hung, M.W. Spong: Adaptive control of flexible-joint manipulators, IEEE Control Syst. Mag. **9**(7), 9–13 (1989)
11.46 R. Lozano, B. Brogliato: Adaptive control of robot manipulators with flexible joints, IEEE Trans. Autom. Control **37**(2), 174–181 (1992)
11.47 H. Sira-Ramirez, M.W. Spong: Variable structure control of flexible joint manipulators, Int. Robotics Autom. **3**(2), 57–64 (1988)
11.48 A. De Luca, G. Ulivi: Iterative learning control of robots with elastic joints, IEEE Int. Conf. Robotics Autom. (ICRA), Nice (1992) pp. 1920–1926
11.49 O. Dahl: Path constrained motion optimization for rigid and flexible joint robots, IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 223–229
11.50 A. De Luca, L. Farina: Dynamic scaling of trajectories for robots with elastic joints, IEEE Int. Conf. Robotics Autom. (ICRA), Washington (2002) pp. 2436–2442
11.51 S. Haddadin, M. Özparpucu, A. Albu-Schäffer: Optimal control for maximizing potential energy in a variable stiffness joint, IEEE 51st Conf. Decis. Control, Maui (2012) pp. 1199–1206
11.52 M. Özparpucu, S. Haddadin: Optimal control of elastic joints with variable damping, 13th European Control Conf., Strasbourg (2014) pp. 2526–2533
11.53 S. Nicosia, P. Tomei, A. Tornambè: A nonlinear observer for elastic robots, IEEE J. Robotics Autom. **4**(1), 45–52 (1988)
11.54 S. Nicosia, P. Tomei: A method for the state estimation of elastic joint robots by global position measurements, Int. J. Adapt. Control Signal Process. **4**(6), 475–486 (1990)
11.55 A. De Luca, D. Schröder, M. Thümmel: An acceleration-based state observer for robot manipulators with elastic joints, IEEE Int. Conf. Robotics Autom. (ICRA), Rome (2007) pp. 3817–3823
11.56 M.W. Spong: On the force control problem for flexible joint manipulators, IEEE Trans. Autom. Control **34**(1), 107–111 (1989)
11.57 J.K. Mills: Stability and control of elastic-joint robotic manipulators during constrained-motion tasks, IEEE Trans. Robotics Autom. **8**(1), 119–126 (1992)
11.58 K.P. Jankowski, H.A. El Maraghy: Dynamic decoupling for hybrid control of rigid-/flexible-joint robots interacting with the environment, IEEE Trans. Robotics Autom. **8**(5), 519–534 (1992)
11.59 T. Lin, A.A. Goldenberg: A unified approach to motion and force control of flexible joint robots, IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (1996) pp. 1115–1120
11.60 A. Albu-Schäffer, C. Ott, G. Hirzinger: A unified passivity-based control framework for position, torque and impedance control of flexible joint robots, Int. J. Robotics Res. **26**(1), 23–39 (2007)
11.61 W. Book, O. Maizza-Neto, D.E. Whitney: Feedback control of two beam, two joint systems with distributed flexibility, ASME J. Dyn. Syst. Meas. Control **97**(4), 424–431 (1975)
11.62 W. Book: Characterization of strength and stiffness constraints on manipulator control. In: *Theory and Practice of Robots and Manipulators*, ed. by W. Book (Elsevier, Amsterdam 1977) pp. 37–45
11.63 T.E. Alberts, W. Book, S. Dickerson: Experiments in augmenting active control of a flexible structure with passive damping, AIAA 24th Aerosp. Sci. Meet., Reno (1986)
11.64 T. Bailey, J.E. Hubbard Jr.: Distributed piezoelectric-polymer active vibration control of a cantilever beam, J. Guid. Control Dyn. **8**(5), 605–611 (1985)
11.65 W. Book, V. Sangveraphunsiri, S. Le: The Bracing Strategy for Robot Operation, Jt. IFToMM-CISM Symp. Theory Robots Manip. (RoManSy), Udine (1984)
11.66 W. Book: Analysis of massless elastic chains with servo controlled joints, ASME J. Dyn. Syst. Meas. Control **101**(3), 187–192 (1979)
11.67 E.C. Pestel, F.A. Leckie: *Matrix Methods in Elastomechanics* (McGraw-Hill, New York 1963)
11.68 R. Krauss: Transfer Matrix Modeling, Ph.D. Thesis (School of Mechanical Engineering, Georgia Institute of Technology, Atlanta 2006)
11.69 W.J. Book: Modeling, Design and Control of Flexible Manipulator Arms, Ph.D. Thesis (Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge 1974)
11.70 W. Book: Recursive lagrangian dynamics of flexible manipulators, Int. J. Robotics Res. **3**(3), 87–106 (1984)
11.71 W.J. Book, S.H. Lee: Vibration control of a large flexible manipulator by a small robotic arm, Proc. Am. Control Conf., Pittsburgh (1989) pp. 1377–1380
11.72 J. Lew, S.-M. Moon: A simple active damping control for compliant base manipulators, IEEE/ASME Trans. Mechatron. **2**, 707–714 (1995)
11.73 W.J. Book, J.C. Loper: Inverse dynamics for commanding micromanipulator inertial forces to damp macromanipulator vibration, IEEE, Robot Soc. Int. Conf. Intell. Robots Syst., Kyongju (1999)
11.74 I. Sharf: Active damping of a large flexible manipulator with a short-reach robot, Proc. Am. Control Conf., Seattle (1995) pp. 3329–3333
11.75 L. George, W.J. Book: Inertial vibration damping control of a flexible base manipulator, IEEE/ASME Trans. Mechatron. **8**(2), 268–271 (2003)
11.76 J.F. Calvert, D.J. Gimpel: Method and apparatus for control of system output in response to system input, U.S. Patent 280 1351 (1957)
11.77 O.J.M. Smith: *Feedback Control Systems* (McGraw-Hill, New York 1958)

11.78 N. Singer, W.P. Seering: Preshaping command inputs to reduce system vibration, ASME J. Dyn. Syst. Meas. Control **112**(1), 76–82 (1990)

11.79 W. Singhose, W. Seering, N. Singer: Residual vibration reduction using vector diagrams to generate shaped inputs, J. Mech. Des. **2**, 654–659 (1994)

11.80 D.P. Magee, W.J. Book: The Application of Input Shaping to a System with Varying Parameters, Proc. 1992 Japan-USA Symp. Flex. Autom., San Francisco (1992) pp. 519–526

11.81 D.P. Magee, W.J. Book: Optimal arbitrary time-delay (OAT) filter and method to minimize unwanted system dynamics, U.S. Patent 607 8844 (2000)

11.82 S. Rhim, W.J. Book: Noise effect on time-domain adaptive command shaping methods for flexible manipulator control, IEEE Trans. Control Syst. Technol. **9**(1), 84–92 (2001)

11.83 D.-S. Kwon, W.J. Book: A time-domain inverse dynamic tracking control of a single-link flexible manipulator, J. Dyn. Syst. Meas. Control **116**, 193–200 (1994)

11.84 D.S. Kwon: An Inverse Dynamic Tracking Control for a Bracing Flexible Manipulator, Ph.D. Thesis (School of Mechanical Engineering, Georgia Institute of Technology, Atlanta 1991)

11.85 A. De Luca, G. Di Giovanni: Rest-to-rest motion of a one-link flexible arm, Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron., Como (2001) pp. 923–928

11.86 A. De Luca, G. Di Giovanni: Rest-to-rest motion of a two-link robot with a flexible forearm, Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron., Como (2001) pp. 929–935

11.87 A. De Luca, V. Caiano, D. Del Vescovo: Experiments on rest-to-rest motion of a flexible arm. In: *Experimental Robotics VIII*, Springer Tracts in Advanced Robotics, Vol. 5, ed. by B. Siciliano, P. Dario (Springer, Berlin, Heidelberg 2003) pp. 338–349

11.88 R.M. Murray, M. Rathinam, W. Sluis: Differential flatness of mechanical control systems: A catalog of prototype systems, Proc. ASME Int. Mech. Engr. Congr., San Francisco (1995)

11.89 M. Fliess, P. Martin, P. Rouchon: Flatness and defect of nonlinear systems: Introductory theory and examples, Int. J. Control **61**(6), 1327–1361 (1995)

11.90 R.H. Cannon, E. Schmitz: Initial experiments on the end-point control of a flexible one-link robot, Int. J. Robotics Res. **3**(3), 62–75 (1984)

11.91 A. Truckenbrot: Modeling and control of flexible manipulator structures, Proc. 4th CISM-IFToMM Symp. Theory Robots Manip. (RoManSy), Zaborow (1981) pp. 90–101

11.92 G.G. Hastings, W.J. Book: Reconstruction and robust reduced-order observation of flexible variables, ASME Winter Ann. Meet., Anaheim (1986)

11.93 B.S. Yuan, J.D. Huggins, W.J. Book: Small motion experiments with a large flexible arm with strain feedback, Proceedings of the 1989 Am. Control Conf., Pittsburgh (1989) pp. 2091–2095

11.94 D. Wang, M. Vidyasagar: Passive control of a stiff flexible link, Int. J. Robotics Res. **11**, 572–578 (1992)

11.95 K. Obergfell, W.J. Book: Control of flexible manipulators using vision and modal feedback, Proc. Int. Conf. Risk Assess. Manag. (ICRAM), Istanbul (1995)

11.96 B. Post: Robust State Estimation for the Control of Flexible Robotic Manipulators, Ph.D. Thesis (School of Mechanical Engineering, Georgia Institute of Technology, Atlanta 2013)

11.97 B. Walcott, S. Zak: State observation of nonlinear uncertain dynamical system, IEEE Trans. Autom. Control **32**, 166–170 (1987)

11.98 B. Walcott, S. Zak: Observation of dynamical systems in the presence of bounded nonlinearities/uncertainties, Proc. IEEE Conf. Dec. Control **25**, 961–966 (1986)

11.99 N.G. Chalhoub, G.A. Kfoury: Development of a robust nonlinear observer for a single-link, flexible manipulator, Nonlin. Dyn. **39**(3), 217–233 (2005)

11.100 W.J. Book: Controlled motion in an elastic world, ASME J. Dyn. Syst. Meas. Control **2B**, 252–261 (1993)

11.101 C. Canudas-de-Wit, B. Siciliano, G. Bastin (Eds.): *Theory of Robot Control* (Springer, Berlin, Heidelberg 1996)

11.102 G. Hirzinger, N. Sporer, J. Butterfass, M. Grebenstein: Torque-controlled lightweight arms and articulated hands: Do we reach technological limits now?, Int. J. Robotics Res. **23**(4/5), 331–340 (2004)

11.103 Camotion Inc.: http://www.camotion.com (Camotion Inc., Atlanta 2007)

11.104 B. Rooks: High speed delivery and low cost from new ABB packaging robot, Ind. Robotics Int. J. **26**(4), 267–275 (1999)

11.105 T. Trautt, E. Bayo: Inverse dynamics of nonminimum phase systems with non-zero initial conditions, Dyn. Control **7**(1), 49–71 (1997)

11.106 B. Siciliano, W. Book: A singular perturbation approach to control of lightweight flexible manipulators, Int. J. Robotics Res. **7**(4), 79–90 (1988)

11.107 F. Ghorbel, M.W. Spong: Singular perturbation model of robots with elastic joints and elastic links constrained by a rigid environment, J. Intell. Robotics Syst. **22**(2), 143–152 (1998)

11.108 J. Guldner, J. Shi, V. Utkin: *Sliding Mode Control in Electromechanical Systems* (Taylor, London 1999)

11.109 J.-H. Ryu, D.-S. Kwon, B. Hannaford: Control of a flexible manipulator with noncollocated feedback: Time-domain passivity approach, IEEE Trans. Robotics **20**(4), 776–780 (2004)

Multimedia Contents

12. Robotic Systems Architectures and Programming

David Kortenkamp, Reid Simmons, Davide Brugali

Robot software systems tend to be complex. This complexity is due, in large part, to the need to control diverse sensors and actuators in real time, in the face of significant uncertainty and noise. Robot systems must work to achieve tasks while monitoring for, and reacting to, unexpected situations. Doing all this concurrently and asynchronously adds immensely to system complexity.

The use of a well-conceived architecture, together with programming tools that support the architecture, can often help to manage that complexity. Currently, there is no single architecture that is best for all applications – different architectures have different advantages and disadvantages. It is important to understand those strengths and weaknesses when choosing an architectural approach for a given application.

This chapter presents various approaches to architecting robotic systems. It starts by defining terms and setting the context, including a recounting of the historical developments in the area of robot architectures. The chapter then discusses in more depth the major types of architectural components in use today – behavioral control (Chap. 13), executives, and task planners (Chap. 14) – along with commonly used techniques for interconnecting connecting those components. Throughout, emphasis will be placed on programming tools and environments that support these architectures. A case study is then presented, followed by a brief discussion of further reading.

12.1 **Overview** 283
12.1.1 Special Needs of Robot Architectures 284
12.1.2 Modularity and Hierarchy 284
12.1.3 Software Development Tools 285
12.2 **History** 285
12.2.1 Subsumption 285
12.2.2 Layered Robot Control Architectures 286
12.3 **Architectural Components** 289
12.3.1 Connecting Components 289
12.3.2 Behavioral Control 291
12.3.3 Executive 292
12.3.4 Planning 294
12.4 **Case Study – GRACE** 296
12.5 **The Art of Robot Architectures** 298
12.6 **Implementing Robotic Systems Architectures** 299
12.6.1 Agile Robot Development Network (aRDnet) 300
12.6.2 Yet Another Robot Platform (YARP) 300
12.6.3 Open Robot Control Software (OROCOS) 300
12.6.4 Smartsoft 300
12.6.5 Robot Operating System (ROS) 301
12.6.6 GenoM/BIP 301
12.6.7 Best Practice in Robotics (BRICS) 301
12.7 **Conclusions and Further Reading** 302
Video-References 302
References 302

12.1 Overview

The term *robot architecture* is often used to refer to two related, but distinct, concepts. Architectural *structure* refers to how a system is divided into subsystems and how those subsystems interact. The structure of

a robot system is often represented informally using traditional *boxes and arrows* diagrams or more formally using techniques such as unified modeling language (UML) [12.1]. In contrast, architectural *style* refers to the computational concepts that underlie a given system. For instance, one robot system might use a publish–subscribe message passing style of communication, while another may use a more synchronous client–server approach.

All robotic systems use some architectural structure and style. However, in many existing robot systems it is difficult to pin down precisely the architecture being used. In fact, a single robot system will often use several styles together. In part, this is because the system implementation may not have clean subsystem boundaries, blurring the architectural structure. Similarly, the architecture and the domain-specific implementation are often intimately tied together, blurring the architectural style(s).

This is unfortunate, as a well-conceived, clean architecture can have significant advantages in the specification, execution, and validation of robot systems. In general, robot architectures facilitate development by providing beneficial constraints on the design and implementation of robotic systems, without being overly restrictive. For instance, separating behaviors into modular units helps to increase understandability and reusability, and can facilitate unit testing and validation.

12.1.1 Special Needs of Robot Architectures

In some sense, one may consider robot architectures as software engineering. However, robot architectures are distinguished from other software architectures because of the special needs of robot systems. The most important of these, from the architectural perspective, are that robot systems need to interact asynchronously, in real time, with an uncertain, often dynamic, environment. In addition, many robot systems need to respond at varying temporal scopes – from millisecond feedback control to minutes, or hours, for complex tasks.

To handle these requirements, many robot architectures include capabilities for acting in real time, controlling actuators and sensors, supporting concurrency, detecting and reacting to exceptional situations, dealing with uncertainty, and integrating high-level (symbolic) planning with low-level (numerical) control.

While the same capability can often be implemented using different architectural styles, there may be advantages of using one particular style over another. As an example, consider how a robot system's style of communications can impact on its reliability. Many robot systems are designed as asynchronous processes that communicate using message passing. One popular communication style is *client–server*, in which a message request from the client is paired with a response from the server. An alternate communication paradigm is *publish–subscribe*, in which messages are broadcast asynchronously and all modules that have previously indicated an interest in such messages receive a copy. With client–server-style message passing, modules typically send a request and then block, waiting for the response. If the response never comes (e.g., the server module crashes) then deadlock can occur. Even if the module does not block, the control flow still typically expects a response, which may lead to unexpected results if the response never arrives or if a response to some other request happens to arrive first. In contrast, systems that use publish–subscribe tend to be more reliable: because messages are assumed to arrive asynchronously, the control flow typically does not assume any particular order in which messages are processed, and so missing or out-of-order messages tend to have less impact.

12.1.2 Modularity and Hierarchy

One common feature of robot architectures is modular decomposition of systems into simpler, largely independent pieces. As mentioned above, robot systems are often designed as communicating processes, where the communications interface is typically small and relatively low bandwidth. This design enables the processes/modules to handle interactions with the environment asynchronously, while minimizing interactions with one another. Typically, this decreases overall system complexity and increases overall reliability.

Often, system decomposition is hierarchical – modular components are themselves built on top of other modular components. Architectures that explicitly support this type of layered decomposition reduce system complexity through abstraction. However, while hierarchical decomposition of robotic systems is generally regarded as a *desirable quality*, debate continues over the dimensions along which to decompose. Some architectures [12.2] decompose along a temporal dimension – each layer in the hierarchy operates at a characteristic frequency an order of magnitude slower than the layer below. In other architectures [12.3–6], the hierarchy is based on task abstraction – tasks at one layer are achieved by invoking a set of tasks at lower levels. In some situations, decomposition based on spatial abstraction may be more useful, such as when dealing with both local and global navigation [12.7]. The main point is that different applications need to decompose

problems in different ways, and architectural styles can often be found to accommodate those different needs.

12.1.3 Software Development Tools

While significant benefit accrues from *designing* systems using well-defined architectural styles, many architectural styles also have associated software tools that facilitate adhering to that style during *implementation*. These tools can take the form of libraries of functions calls, specialized programming languages, or graphical editors. The tools make the constraints of the architectural style explicit, while hiding the complexity of the underlying concepts.

For instance, inter-process communication libraries, such as common object request broker architecture (CORBA) [12.8] and inter-process communication (IPC) package [12.9], make it easy to implement message passing styles, such as client–server and publish–subscribe, respectively. Languages, such as Subsumption [12.10] and Skills [12.11] facilitate the development of data-driven, real-time behaviors, while languages such as the execution support language (ESL) [12.12] and the the plan execution interchange language (PLEXIL) [12.13] provide support for reliably achieving higher-level tasks. Graphical editors, such as found in ControlShell [12.14], Labview [12.15] and open robot controller computer aided design (ORCCAD) [12.6], provide constrained ways to assemble systems, and then automatically generate code that adheres to the architectural style.

In each case, the tools facilitate the development of software in the given style and, more importantly, make it impossible (or, at least, very difficult) to violate the constraints of that architectural style. The result is that systems implemented using such tools are typically easier to implement, understand, debug, verify, and maintain. They also tend to be more reliable, since the tools provide well-engineered capabilities for commonly needed control constructs, such as message passing, interfacing with actuators and sensors, and handling concurrent tasks.

12.2 History

Robot architectures and programming began in the late 1960s with the Shakey robot at Stanford University [12.16] (Fig. 12.1). Shakey had a camera, a range finder, and bump sensors, and was connected to DEC PDP-10 and PDP-15 computers via radio and video links. Shakey's architecture was decomposed into three functional elements: sensing, planning, and executing [12.17]. The sensing system translated the camera image into an internal world model. The planner took the internal world model and a goal and generated a plan (i. e., a series of actions) that would achieve the goal. The executor took the plan and sent the actions to the robot. This approach has been called the *sense–plan–act* (SPA) paradigm (Fig. 12.2). Its main architectural features are that sensing flowed into a world model, which was then used by the planner, and that plan was executed without directly using the sensors that created the model. For many years, robotic control architectures and programming focused almost exclusively on the SPA paradigm.

12.2.1 Subsumption

In the early 1980s, it became apparent that the SPA paradigm had problems. First, planning in any real-world domain took a long time, and the robot would be blocked, waiting for planning to complete. Second, and more importantly, execution of a plan without involving sensing was dangerous in a dynamic world. Several new robot control architecture paradigms be-

Fig. 12.1 Shakey (after [12.18])

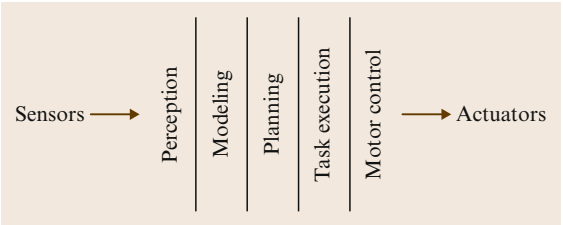


Fig. 12.2 The sense–plan–act (SPA) paradigm (after [12.3], with permission)

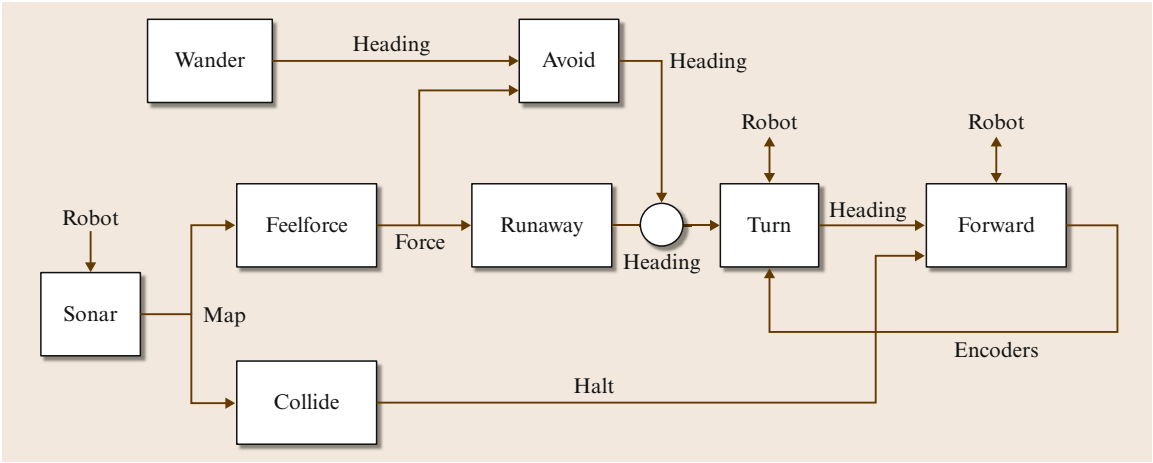


Fig. 12.3 Example of the Subsumption architecture (after [12.3])

gan to emerge, including reactive planning, in which plans were generated quickly and relied more directly on sensed information instead of internal models [12.4, 19]. The most influential work, however, was the subsumption architecture of *Brooks* [12.3]. A subsumption architecture is built from layers of interacting finite-state machines – each connecting sensors to actuators directly (Fig. 12.3). These finite-state machines were called *behaviors* (leading some to call Subsumption *behavior-based* or *behavioral* robotics [12.20]; see also Chap. 38). Since multiple behaviors could be active at any one time, Subsumption had an *arbitration* mechanism that enabled higher-level behaviors to override signals from lower-level behaviors. For example, the robot might have a behavior that simply drives the robot in random directions. This behavior is always active and the robot is always driving somewhere. A second, higher-level behavior could take sensor input, detect obstacles, and steer the robot away from them. It is also always active. In an environment with no obstacles, the higher-level behavior never generates a signal. However, if it detects an obstacle it overrides the lower-level behavior and steers the robot away. As soon as the obstacle is gone (and the higher-level behavior stops sending signals), the lower-level behavior gets control again. Multiple, interacting layers of behaviors could be built to produce more and more complex robots.

Many robots were built using the subsumption approach – most at MIT [12.21–23]. They were quite successful. Whereas SPA robots were slow and ponderous, Subsumption robots were fast and reactive. A dynamic world did not bother them because they constantly sensed the world and reacted to it. These robots scampered around like insects or small rodents. Several behavioral architectures arose in addition to Subsumption, often with different arbitration schemes for combining the outputs of behaviors [12.24, 25].

A popular example of behavior-based architectures is *Arkin*'s motor-control schemas [12.26]. In this biologically inspired approach, motor and perceptual schemas [12.27] are dynamically connected to one another. The motor schemas generate response vectors based on the outputs of the perceptual schemas, which are then combined in a manner similar to potential fields [12.28]. To achieve more complex tasks, the autonomous robot architecture (AuRA) architecture [12.29, 30] added a navigation planner and a plan sequencer, based on finite-state acceptors (FSAs), to the reactive schemas.

However, behavior-based robots soon reached limits in their capabilities. It proved very difficult to compose behaviors to achieve long-range goals and it proved almost impossible to optimize robot behavior. For example, a behavior-based robot that delivered mail in an office building could easily be built by simply wandering around the office building and having behaviors looking for rooms and then overriding the wandering and entering the office. It was much more difficult to use the behavioral style of architecture to design a system that reasoned about the day's mail to visit the offices in an optimal order to minimize delivery time. In essence, robots needed the planning capabilities of the early architectures wedded to the reactivity of the behavior-based architectures. This realization led to the development of layered, or tiered, robot control architectures.

12.2.2 Layered Robot Control Architectures

One of the first steps towards the integration of reactivity and deliberation was the reactive action packages

(RAPs) system created by *Firby*. In his thesis [12.31], we see the first outline of an integrated, three-layer architecture. The middle layer of that architecture, and the subject of the thesis, was the RAPs system. Firby also speculated on the form and function of the other two tiers, specifically with the idea of integrating classic deliberative methods with the ideas of the emerging situated reasoning community, but those layers were never implemented. Later, *Firby* would integrate RAPs with a continuous low-level control layer [12.32].

Independently and concurrently, *Bonasso* at MITRE [12.33] devised an architecture that began at the bottom layer with robot behaviors programmed in the Rex language as synchronous circuits [12.34]. These Rex machines guaranteed consistent semantics between the agent's internal states and those of the world. The middle layer was a conditional sequencer implemented in the goal as parallel programs (GAPPs) language [12.35], which would continuously activate and deactivate the Rex skills until the robot's task was complete. This sequencer based on GAPPs was appealing because it could be synthesized through more traditional planning techniques [12.36]. This work culminated in the three tiers (3T) architecture (named after its three tiers of interacting control processes – planning, sequencing, and real-time control), which has been used on many generations of robots [12.37].

Architectures similar to 3T (Fig. 12.4) have been developed subsequently. One example is a three layer architecture for navigating through intricate situations (ATLANTIS) [12.38], which leaves much more control at the sequencing tier. In ATLANTIS, the deliberative tier must be specifically called by the sequencing tier. A third example is *Saridis*' intelligent control architecture [12.39]. The architecture begins with the servo systems available on a given robot and augments them to integrate the execution algorithms of the next level, using VxWorks and the VME (Versa Module Europa) bus. The next level consists of a set of coordinating routines for each lower subsystem, e.g., vision, arm motion, and navigation. These are implemented in Petri net transducers (PNTs), a type of scheduling mechanism, and activated by a dispatcher connected to the organizational level. The organizational level is a planner implemented as a Boltzmann neural network. Essentially the neural network finds a sequence of actions that will match the required command received as text input, and then the dispatcher executes each of these steps via the network of PNT coordinators.

The LAAS architecture for autonomous systems (LAAS) is a three-layered architecture that includes software tools to support development/programming at each layer [12.40]. The lowest layer (functional) consists of a network of *modules*, which are dynamically parameterized control and perceptual algorithms. Modules are written in the generator of modules (GenoM) language, which produces standardized templates that facilitate the integration of modules with one another. Unlike most other three-layered architectures, the executive layer is fairly simple – it is purely reactive and does no task decomposition. It serves mainly as a bridge – receiving task sequences from the highest layer and selecting and parameterizing tasks to send to the functional layer. The executive is written in the Kheops language, which automatically generates decision networks that can be formally verified. At the top, the decision layer consists of a planner, implemented using the indexed time table (IxTeT) temporal planner [12.41, 42], and a supervisor, implemented using procedural reasoning system (PRS) [12.43, 44]. The supervisor is similar to the executive layer of other three-layered architectures – it decomposes tasks, chooses alternative methods for achieving tasks, and monitors execution. By combining the planner and supervisor in one layer, LAAS achieves a tighter connection between the two, enabling more flexibility in when, and how, replanning occurs. The LAAS architecture actually allows for multiple decisional layers at increasingly higher levels of abstraction, such as a high-level *mission* layer and a lower-level *task* layer.

Remote agent is an architecture for the autonomous control of spacecraft [12.45]. It actually consists of four layers – a control (behavioral) layer, an executive, a planner/scheduler, and mode identification and recovery (MIR) that combines fault detection and recovery. The control layer is the traditional spacecraft real-time control system. The executive is the core of

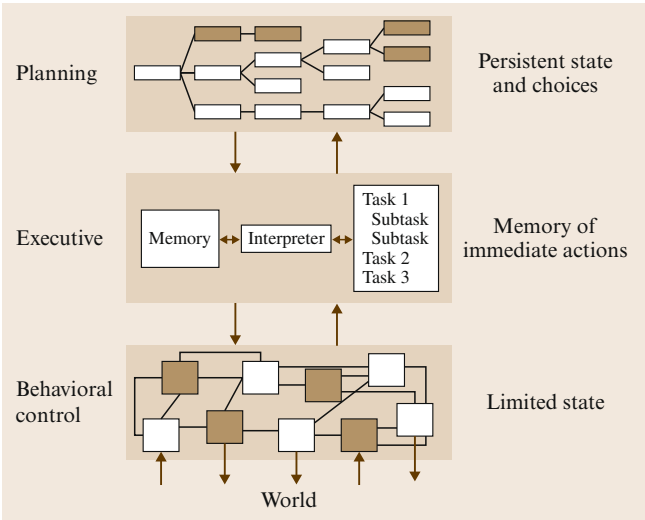


Fig. 12.4 Prototype three-tiered architecture

the architecture – it decomposes, selects, and monitors task execution, performs fault recovery, and does resource management, turning devices on and off at appropriate times to conserve limited spacecraft power. The planner/scheduler is a batch process that takes goals, an initial (projected) state, and currently scheduled activities, and produces plans that include flexible ranges on start and end times of tasks. The plan also includes a task to reinvoke the planner to produce the next plan segment. An important part of the remote agent is configuration management – configuring hardware to support tasks and monitoring that the hardware remains in known, stable states. The role of configuration management is split between the executive, which uses reactive procedures, and MIR, which uses declarative models of the spacecraft and deliberative algorithms to determine how to reconfigure the hardware in response to detected faults [12.46].

The Syndicate architecture [12.47] extends the 3T model to multirobot coordination (Chap. 51). In this architecture, each layer interfaces not only with the layers above and below, as usual, but also with the layers of the other robots at the same level (Fig. 12.5). In this way, distributed control loops can be designed at multiple levels of abstraction. The version of Syndicate in [12.48] used a distributed market-based approach for task allocation at the planning layer.

Other noteworthy multitiered architectures have appeared in the literature. The National Bureau of Standards (NBS) developed for the Aeronautics and Space Agency (NASA) the NASA/NBS standard reference model (NASREM) [12.2, 49], later named real-time control system (RCS), was an early reference model for telerobotic control (Fig. 12.6). It is a many-tiered model in which each layer has the same general structure, but operates at increasingly lower frequency as it moves from the servo level to the reasoning levels. With the exception of maintaining a global world model, NASREM, in its original inception, provided for all the data and control paths that are present in architectures such as 3T, but NASREM was a reference model, not an implementation. The subsequent implementations of NASREM followed primarily the traditional sense–plan–act approach and were mainly applied to telerobotic applications, as opposed to autonomous robots. A notable exception was the early work of *Blidberg/Chappell* [12.50].

While three-layered robot architectures are very popular, various two-layered architectures have been investigated by researchers. The coupled layered architecture for robot autonomy (CLARAty) was designed to provide reusable software for NASA's space robots, especially planetary rovers [12.52, 53]. CLARAty consists of a functional and a decision layer. The functional layer is a hierarchy of object-oriented algorithms that provide more and more abstract interfaces to the robot, such as motor control, vehicle control, sensor-based navigation, and mobile manipulation. Each object provides a generic interface that is hardware independent, so that the same algorithms can run on different hardware. The decision layer combines planning and executive capabilities. Similar to the LAAS architecture,

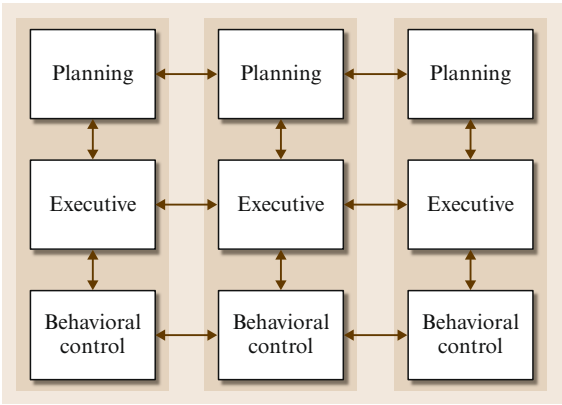


Fig. 12.5 The Syndicate multirobot architecture

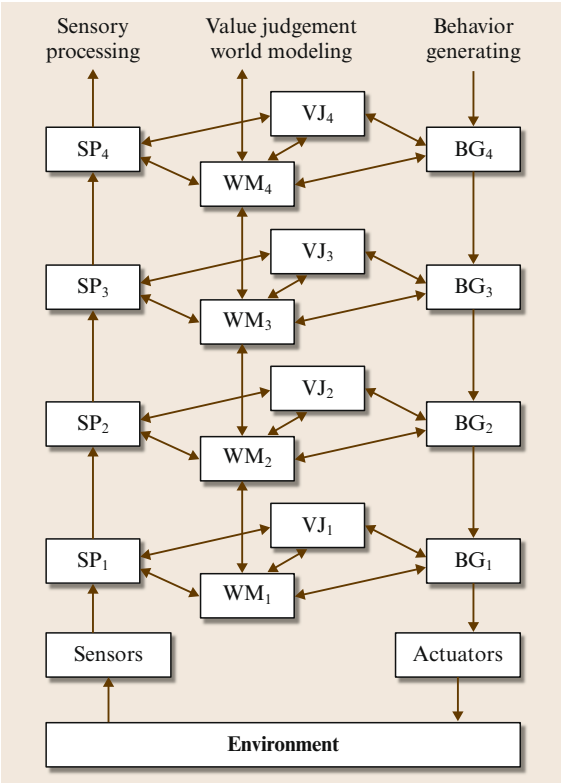


Fig. 12.6 The real-time control system (RCS) reference architecture (after [12.51])

this is done to provide for tighter coordination between planning and execution, enabling continual replanning in response to dynamic contingencies.

Closed-loop execution and recovery (CLEaR) [12.54] is one instantiation of the CLARAty decision layer. CLEaR combines the continuous activity scheduling, planning, execution and replanning (CASPER) repair-based planner [12.55] and the task description language (TDL) executive language [12.56]. CLEaR provides a tightly coupled approach to goal- and event-driven behavior. At its heart is the capability to do fast, continuous replanning, based on frequent state and resource updates from execution monitoring. This enables the planner to react to many exceptional situations, which can be important in cases where there are many tasks, few resources, and significant uncertainty. In CLEaR, both the planning and executive components are able to handle resource conflicts and exceptional situations – heuristics are used to decide which component should be involved in a given situation. The onboard autonomous science investigation system (OASIS) system [12.57] extends CLEaR to include science data analysis so that the architecture can be driven by opportunistic science-related goals (such as finding unusual rocks or formations). OASIS is planner-centric, releasing tasks to the executive component just a few seconds before their scheduled start times.

The cooperative intelligent real-time control architecture (CIRCA) is a two-layered architecture concerned with guaranteeing reliable behavior [12.58, 59]. It embodies the notion of *bounded reactivity* – an acknowledgement that the resources of the robot are not always sufficient to guarantee that all tasks can be achieved. CIRCA consists of a real-time system (RTS) and an artificial intelligence (AI) system (AIS) that are largely independent. The RTS executes a cyclic schedule of test action pairs (TAPs) that have guaranteed worst-case behavior in terms of sensing the environment and conditionally acting in response. It is the responsibility of the AIS to create a schedule that is guaranteed to prevent catastrophic failures from occurring, while running in hard real time. The AIS does this by planning over a state-transition graph that includes transitions for actions, exogenous events, and the passage of time (e.g., if the robot waits too long, bad things can happen). The AIS tests each plan (set of TAPs) to see if it can actually be scheduled. If not, it alters the planning model, either by eliminating tasks (based on goal prioritization) or by changing parameters of behaviors (e.g., reducing the robot's velocity). The AIS continues this until it finds a plan that can be successfully scheduled, in which case it downloads the new plan to the RTS in an atomic operation.

Like CIRCA, ORCCAD is a two-layered architecture that is concerned with guaranteed reliability [12.6, 60]. In the case of ORCCAD, this guarantee is achieved through formal verification techniques. Robot tasks (lower-level behaviors) and robot procedures (higher-level actions) are defined in higher-level languages that are then translated into the Esterel programming language [12.61], for logical verification, or the Timed-Argus language [12.62], for temporal verification. The verification is geared toward liveness and safety properties, as well as verifying lack of contention for resources.

12.3 Architectural Components

We will take the three-tiered architecture as the prototype for the components discussed in this chapter. Figure 12.4 shows a typical three-tiered architecture. The lowest tier (or layer) is behavioral control and is the layer tied most closely to sensors and actuators. The second tier is the executive layer and is responsible for choosing the current behaviors of the robot to achieve a task. The highest tier is the task-planning layer and it is responsible for achieving long-term goals of the robot within resource constraints. Using the example of an office delivery robot, the behavioral layer is responsible for moving the robot around rooms and hallways, for avoiding obstacles, for opening doors, etc. The executive layer coordinates the behavioral layer to achieve tasks such as leaving a room, going to an office, etc. The task-planning layer is responsible for deciding the order of deliveries to minimize time, taking into account delivery priorities, scheduling, recharging, etc. The task-planning layer sends tasks (e.g., exit the room, go to office 110) to the executive. All these tiers need to work together and exchange information. The next section deals with the problem of connecting components to each other. We then discuss each component of the three-tiered prototype architecture in detail.

12.3.1 Connecting Components

All of the architecture components that have been discussed in this chapter need to communicate with each other. They need to both exchange data and send commands. The choice of how components communicate (often called the *middleware*) is one of the most im-

portant and most constraining of the many decisions a robot architecture designer will make. From previous experience, a great deal of the problems and a majority of the debugging time in developing robot architectures have to do with communication between components. In addition, once a communication mechanism is chosen it becomes extremely difficult to change, so early decisions persist for many years. Many developers *roll their own* communication protocols, usually built on top of Unix sockets. While this allows for customization of messages, it fails to take advantage of the reliability, efficiency, and ease of use that externally available communication packages provide. There are two basic approaches to communication: client–server and publish–subscribe.

Client–Server

In a client–server (also called a point-to-point) communication protocol, components talk directly with other components. A good example of this is remote procedure call (RPC) protocols in which one component (the client) can call functions and procedures of another component (the server). A modern, and popular, variation on this is the common object request broker architecture (CORBA). CORBA allows for one component to call object methods that are implemented by another component. All method calls are defined in an interface definition language (IDL) file that is language independent. Every component uses the same IDL to generate code that compiles with component to handle communication. The advantage of this is that, when an IDL file is changed, all components that use that IDL can be recompiled automatically (by using *make* or similar code configuration tools). CORBA object request brokers (ORBs) are available for most major object-oriented languages. Although free ORBs are available, many commercial ORBs offer additional features and support. One disadvantage of CORBA is that it introduces quite a bit of additional code into applications. Some competitors have tried to address this issue, such as the internet communications engine (ICE), which has its own version of an IDL file called the specification language for ICE (SLICE). The biggest advantage of a client–server protocol is that the interfaces are very clearly defined in advance and everyone knows when the interface has changed. Another advantage is that it allows for a distributed approach to communication with no central module that must distribute data. A disadvantage of client–server protocols is that they introduce significant overhead, especially if many components need the same information. It should be noted that CORBA and ICE also have a broadcast mechanism (called an event channel, or the notification service, in CORBA).

Publish–Subscribe

In a publish–subscribe (also called a broadcast) protocol, a component publishes data and any other component can subscribe to that data. Typically, a centralized process routes data between publishers and subscribers. In a typical architecture, most components both publish information and subscribe to information published by other components. There are several existing publish–subscribe middleware solutions. A popular one for robotics is the real-time innovations' (RTI) data distribution service (DDS), formerly the network data distribution service (NDDS) [12.63]. Another popular publish–subscribe paradigm is IPC developed at Carnegie Mellon University [12.9]. Many publish-subscribe protocols are converging on using extensible markup language (XML) descriptions to define the data being published, with the added convenience of transmitting XML over HTTP, which allows for significant interoperability with Web-based applications. Publish–subscribe protocols have a large advantage in being simple to use and having low overhead. They are especially useful when it is unknown how many different components might need a piece of data (e.g., multiple user interfaces). Also, components do not get bogged down with repeated requests for information from many different sources. Publish–subscribe protocols are often more difficult to debug because the syntax of the message is often hidden in a simple string type. Thus problems are not revealed until runtime when a component tries, and fails, to parse an incoming message. Publish–subscribe protocols are also not as readable when it comes to sending commands from one module to another. Instead of calling an explicit method or function with parameters, a command is issued by publishing a message with the command and parameters in it and then having that message be parsed by a subscriber. Finally, publish–subscribe protocols often use a single central server to dispatch messages to all subscribers, providing a single point of failure and potential bottleneck.

JAUS

Recently, a standard has emerged in the defense robotics community not only for a communication protocol but also for definitions of messages that are sent via that communication protocol. The joint architecture for unmanned systems (JAUS) defines a set of reusable messages and interfaces that can be used to command autonomous systems [12.64–66]. These reusable components reduce the cost of integrating new hardware components into autonomous systems. Reuse also allows for components developed for one autonomous system to be used by another autonomous system. JAUS has two components: a *domain model* and a *reference*

architecture. The domain model is a representation of the unmanned systems' functions and information. It contains a description of the system's functional and informational capabilities. The former includes models of the system's maneuvering, navigational, sensing, payload, and manipulation capabilities. The latter includes models of the system's internal data, such as maps and system status. The reference architecture provides a well-defined set of messages. Messages cause actions to commence, information to be exchanged, and events to occur. Everything that occurs in a JAUS system is precipitated by messages. This strategy makes JAUS a component-based, message-passing architecture.

The JAUS reference architecture defines a system hierarchy, as shown in Fig. 12.7. The topology defines the *system* as the collection of vehicles, operator control units (OCUs), and infrastructure necessary to provide the full robotic capability. Subsystems are individual units (e.g., vehicles or OCUs) in the system. Nodes define a distinct processing capability within the architecture and route JAUS messages to components. Components provide the different execution capabilities and respond directly to command messages. Components might be sensors (e.g., a SICK laser or a vision sensor), actuators (a manipulator or a mobile base) or payloads (weapons or task sensors). The topology (the layout of particular system, subsystems, nodes, and components) is defined by the system implementers based on task requirements.

At the core of JAUS is a set of well-defined messages. JAUS supports the following message types.

- *Command*: Initiate mode changes or actions
- *Query*: Used to solicit information from a component
- *Inform*: Response to a query
- *Event set up*: Passes parameters to set up an event
- *Event notification*: Sent when the event happens

JAUS has about 30 predefined messages that can be used to control robots. There are messages for control of a robotic vehicle. For example, the *global vector driver* message performs closed-loop control of the desired global heading, altitude, and speed of a mobile vehicle. There are also sensor messages such as *global pose sensor*, which distributes the global position and orientation of the vehicle. There are also manipulation messages in JAUS. For example, the *set joint positions* message sets the desired joint position values. The *set tool point* message specifies the coordinates of the end-effector tool point in terms of the coordinate system attached to the end-effector.

JAUS also has user-definable messages. Messages have headers that follow a specific format and include message type, destination address (e.g., system, subsystem, node, and component), priority, etc. While JAUS is primarily point to point, JAUS messages can also be marked as *broadcast* and distributed to all components. JAUS also defines coordinate systems for navigation and manipulation to ensure all components understand any coordinates sent to them.

12.3.2 Behavioral Control

Behavioral control represents the lowest level of control in a robot architecture. It directly connects sensors and actuators. While these are typically hand-crafted functions written in C or C++, there have been specialized languages developed for behavioral control, including ALFA [12.67], Behavioral Language [12.68], and Rex [12.69]. It is at this level that traditional control theory (e.g., proportional–integral–derivative (PID) functions, Kalman filters, etc.) resides. In architectures such as 3T, the behavioral layer functions as a *Brooksian machine* – that is, the layer is composed of a small number of behaviors (also called skills) that perceive the environment and carry out the actions of the robot.

Example

Consider an office delivery robot that operates in a typical office building. The behavioral control layer con-

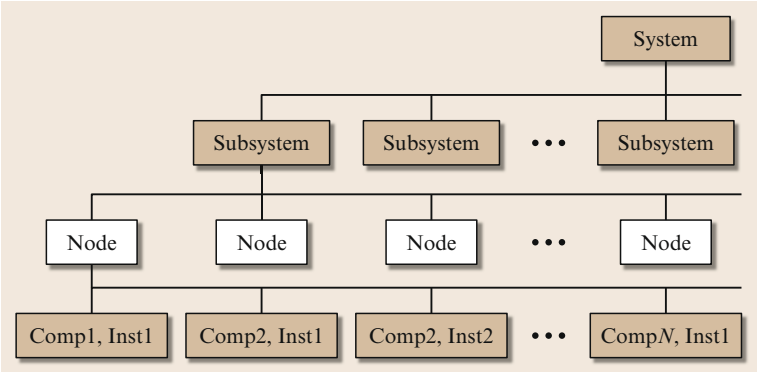


Fig. 12.7 The JAUS reference architecture topology (after JAUS Reference Architecture document [12.64])

tains the control functions necessary to move around in the building and carry out delivery tasks. Assuming the robot has an a priori map of the building, some possible behaviors for this robot include:

1. Move to location while avoiding obstacles
2. Move down hallway while avoiding obstacles
3. Find a door
4. Find a door knob
5. Grasp a door knob
6. Turn a door knob
7. Go through door
8. Determine location
9. Find office number
10. Announce delivery.

Each of these behaviors ties sensors (vision, range sensing, etc.) to actuators (wheel motors, manipulator motors, etc.) in a tight loop. In architectures such as Subsumption, all behaviors are running concurrently with a hierarchical control scheme inhibiting the outputs of certain behaviors. In AuRA [12.30], behaviors are combined using potential functions. Other architectures [12.25, 69] use explicit *arbitration* mechanisms to choose amongst potentially conflicting behaviors.

In architectures such as 3T [12.37], not all of the behaviors are active at the same time. Typically, only a few behaviors that do not conflict would be active at a time (e.g., behaviors 2 and 9 in the example above). The executive layer (Sect. 12.3.3) is responsible for activating and deactivating behaviors to achieve higher-level tasks and to avoid conflicts between two behaviors competing for the same resource (e.g., an actuator).

Situated Behaviors

An important aspect of these behaviors is that they be *situated*. This means that the behavior works only in very specific situations. For example, behavior 2 above moves down a hallway, but this is appropriate only when the robot is situated in a hallway. Similarly, behavior 5, which grasps a door knob, is appropriate only when the robot is within grasping distance of a door knob. The behavior is not responsible for putting the robot in the particular situation. However, it should recognize that the situation is not appropriate and signal as such.

Cognizant Failure

A key requirement for behaviors is that they know when they are not working. This is called *cognizant failure* [12.70]. For example, behavior 5 in our example (grasping the door knob) should not continually grasp at air if it is failing. More succinctly, the behavior should not continue to *bang its head against the wall*. A common problem with early Subsumption robots is that the behaviors did not know they were failing and continued to take actions that were not resulting in progress. It is not the job of the behavioral control layer to decide what to do in a failure situation; it is only necessary to announce that the behavior has failed and halt activity.

Implementation Constraints

The behavioral control layer is designed to bring the speed and reactivity of Subsumption to robot control. For this reason, the behaviors in the behavioral control layer need to follow the philosophies of Subsumption. In particular, the algorithms used for behaviors should be constant in state and time complexity. There should be little or no search at the behavioral control level, and little iteration. Behaviors should simply be transfer functions that take in signals (from sensors or other behaviors) and send out signals (to actuators or other behaviors), and repeat these several times a second. This will allow for reactivity to changing environments. More controversial is how much state should be allowed at the behavioral level. *Brooks* famously said several years ago to *use the world as its own best model* [12.68] – that is, instead of maintaining internal models of the world and querying those models, the robot should instead directly sense the world to get its data. State such as maps, models, etc. belong at the higher levels of the three-tiered prototype architecture, not at the behavioral control layer. Certain exceptions, such as maintaining state for data filtering calculations, could be made on a case-by-case basis. *Gat* [12.71] argues that any state kept at the behavioral layer should be ephemeral and limited.

12.3.3 Executive

The executive layer is the interface between the numerical behavioral control and the symbolic planning layers. The executive is responsible for translating high-level plans into low-level behaviors, invoking behaviors at the appropriate times, monitoring execution, and handling exceptions. Some executives also allocate and monitor resource usage, although that functionality is more commonly performed by the planning layer.

Example

Continuing the example of an office delivery robot, the main high-level task would be to deliver mail to a given office. The executive would decompose this task into a set of subtasks. It may use a geometric path planner to determine the sequence of corridors to move down and intersections at which to turn. If there are doorways along the route, a task would be inserted to open and pass through the door. At the last corridor, the executive would add a concurrent task that looks for the

office number. The final subtasks would be to announce that the person has mail and to concurrently monitor whether the mail has been picked up. If it is not picked up after some period of time, an exception would be triggered that invokes some recovery action (perhaps announcing again, perhaps checking to make sure the robot is at the correct office, perhaps notifying the planning layer to reschedule the delivery for a later time).

Capabilities

The example above illustrates many of the capabilities of the executive layer. First, the executive decomposes high-level tasks (goals) into low-level tasks (behaviors). This is typically done in a *procedural* fashion: the knowledge encoded in the executive describes *how* to achieve tasks, rather than describing *what* needs to be done and having the executive figure out the *how* by itself. Sometimes, though, the executive may also use specialized planning techniques, such as the route planner used in the example above. The decomposition is typically a hierarchical *task tree* (Fig. 12.8), with the leaves of the task tree being invocations and parameterizations of behaviors.

Besides decomposing tasks into subtasks, executives add and maintain temporal constraints between tasks (usually between sibling tasks only, but some executive languages permit temporal constraints between any pair of tasks). The most common constraints are *serial* and *concurrent*, but most executives support more expressive constraint languages, such as having one task begin 10 s after another one starts or having one task end when another ends.

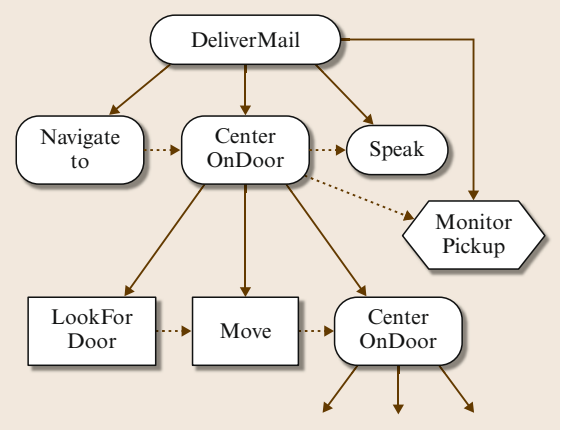


Fig. 12.8 Hierarchical task tree for mail-delivery task (*lozenge nodes* are interior; *rectangular nodes* are leaves; *hexagonal node* is an execution monitor; *solid arrows* are parent–child relationships; *dashed arrows* are sequential constraints)

The executive is responsible for dispatching tasks when their temporal constraints are satisfied. In some executives, tasks may also specify resources (e.g., the robot's motors or camera) that must be available before the task can be dispatched. As with behaviors, arbitrating between conflicting tasks can be a problem. In the case of executives, however, this arbitration is typically either programmed in explicitly (e.g., a rule that says what to do in cases where the robot's attempt to avoid obstacles takes it off the preferred route) or handled using priorities (e.g., recharging is more important than mail delivery).

The final two important executive capabilities are execution monitoring and error recovery. One may wonder why these capabilities are needed if the underlying behaviors are reliable. There are two reasons. First, as described in Sect. 12.3.2, the behaviors are situated, and the situation may change unexpectedly. For instance, a behavior may be implemented assuming that a person is available to pick up the mail, but that may not always be the case. Second, in trying to achieve some goal, the behavior may move the robot into a state that is unexpected by the executive. For instance, people may take advantage of the robot's obstacle avoidance behavior to *herd* it into a closet. While the behavior layer may, in fact, keep the robot safe in such situations, the executive needs to detect the situation in order to get the robot back on track.

Typically, execution monitoring is implemented as a concurrent task that either analyzes sensor data directly or activates a behavior that sends a signal to the executive when the monitored situation arises. These correspond to *polling* and *interrupt-driven* monitors, respectively.

Executives support various responses to monitors being triggered. A monitor may spawn subtasks that handle the situation, it may terminate already spawned subtasks, it may cause the parent task to fail, or it may raise an exception. The latter two responses involve the *error recovery* (also called *exception handling*) capability. Many executives have tasks return status values (success or failure) and allow parent tasks to execute conditionally based on the return values. Other executives use a hierarchical exception mechanism that throws named exceptions to ancestor nodes in the task tree. The closest task that has registered a handler for that exception tries to handle it; if it cannot, it rethrows the exception up the tree. This mechanism, which is inspired by the exception handling mechanisms of C++, Java, and Lisp, is strictly more expressive than the return-value mechanism, but it is also much more difficult to design systems using that approach, due to the nonlocal nature of the control flow.

Implementation Constraints

The underlying formalism for most executives is a hierarchical finite-state controller. Petri nets [12.72] are a popular choice for representing executive functions. In addition, various languages have been developed specifically to assist programmers in implementing executive-level capabilities. We briefly discuss aspects of several of these languages: reactive action packages (RAPs) [12.4, 31], the procedural reasoning system (PRS) [12.43, 44], the execution support language (ESL) [12.12], the task description language (TDL) [12.56], and the plan execution interchange language (PLEXIL) [12.13].

These languages all share features and exhibit differences. One distinction is whether the language is stand-alone (RAPs, PRS, PLEXIL) or an extension of an existing language (ESL is an extension of Common Lisp; TDL is an extension of C++). Stand-alone languages are typically easier to analyze and verify, but extensions are more flexible, especially with respect to integration with legacy software. While stand-alone executive languages all support interfaces to user-defined functions. These interfaces are usually limited in capability (such as what types of data structures can be passed around).

All of these executive languages provide support for hierarchical decomposition of tasks into subtasks. All except PLEXIL allow for recursive invocation of tasks. RAPs, TDL, and PLEXIL have syntax that distinguishes leaf nodes of the task tree/graph from interior nodes.

All these languages provide capabilities for expressing conditionals and iteration, although with RAPs and PLEXIL these are not core-language constructs, but must be expressed as combinations of other constructs. Except for TDL, the languages all provide explicit support for encoding pre- and post-conditions of the tasks and for specifying success criteria. With TDL, these concepts must be programmed in, using more primitive constructs. The stand-alone languages all enable local variables to be defined within a task description, but provide for only limited computation with those variables. Obviously, with extension languages the full capability of the base language is available for defining tasks.

All the languages support the simple serial (sequential) and concurrent (parallel) temporal constraints between tasks, as well as timeouts that can be specified to trigger after waiting a specified amount of time. In addition, TDL directly supports a wide range of temporal constraints – one can specify constraints between the start and end times of tasks (e.g., *task B starts after task A starts* or *task C ends after task D starts*) as well as metric constraints (e.g., *task B starts 10 seconds after task A ends* or *task C starts at 1pm*). ESL and PLEXIL support the signaling of events (e.g., when tasks transition to new states) that can be used to implement similarly expressive types of constraints. In addition, ESL and TDL support task termination based on the occurrence of events (e.g., *task B terminates when task A starts*).

The languages presented differ considerably in how they deal with execution monitoring and exception handling. ESL and TDL both provide explicit execution monitoring constructs and support exceptions that are *thrown* and then *caught* by registered handlers in a hierarchical fashion. This type of exception handling is similar to that used in C++, Java, and Lisp. ESL and TDL also support *clean-up* procedures that can be invoked when tasks are terminated. RAPs and PLEXIL use return values to signal failure, and do not have hierarchical exception handling. PLEXIL, though, does support clean up procedures that are run when tasks fail. PRS has support for execution monitoring, but not exception handling. ESL and PRS support the notion of *resources* that can be shared. Both provide support for automatically preventing contention amongst tasks for the resources. In the other executive languages, this must be implemented separately (although there are plans to extend PLEXIL in this area).

Finally, RAPs, PRS and ESL all include a symbolic database (*world model*) that connects either directly to sensors or to the behavior layer to maintain synchrony with the real world. Queries to the database are used to determine the truth of preconditions, to determine which methods are applicable, etc. PLEXIL has the concept of a *lookup* that performs a similar function, although it is transparent to the task how this is implemented (e.g., by a database lookup or by invoking a behavior-level function). TDL leaves it up to the programmer to specify how the tasks connect to the world.

12.3.4 Planning

The planning component of our prototype layered architecture is responsible for determining the long-range activities of the robot based on high-level goals. Where the behavioral control component is concerned with the here-and-now and the executive is concerned with what has just happened and what should happen next, the planning component looks towards the future. In our running example of an office delivery robot, the planning component would look at the day's deliveries, the resources of the robot, and a map, and determine the optimal delivery routes and schedule, including when the robot should recharge. The planning component is also responsible for replanning when the situation changes.

For example, if an office is locked, the planning component would determine a new delivery schedule that puts that office's delivery later in the day.

Types of Planning

Chapter 9 describes approaches to robot planning in detail. Here, we summarize issues with respect to different types of planners as they relate to layered architectures.

The two most common approaches used are hierarchical task net (HTN) planners and planner/schedulers. HTN planners [12.73, 74] decompose tasks into subtasks, in a manner similar to what many executives do. The main differences are that HTN planners typically operate at higher levels of abstraction, take resource utilization into account, and have methods for dealing with conflicts between tasks (e.g., tasks needing the same resources, or one task negating a precondition needed by another task). The knowledge needed by HTN planners is typically fairly easy to specify, since one indicates directly *how* tasks are to be achieved.

Planner/schedulers [12.75, 76] are useful in domains where time and resources are limited. They create high-level plans that schedule when tasks should occur, but typically leave it to the executive to determine exactly how to achieve the tasks. Planner/schedulers typically work by laying out tasks on time lines, with separate time lines for the various resources that are available on the robot (motors, power, communication, etc.). The knowledge needed by planner/schedulers includes the goals that tasks achieve, the resources they need, their duration, and any constraints between tasks.

Many architectures provide for specialized planning *experts* that are capable of solving particular problems efficiently. In particular, these include motion planners, such as path planners and trajectory planners. Sometimes, the planning layer of the architecture invokes these specialized planners directly; in other architectural styles, the motion planners are part of the lower levels of the architecture (the executive, or even the behavioral layer). Where to put these specialized planners is often a question of style and performance (Sect. 12.5).

Additionally, some architectures provide for multiple planning layers [12.40, 45, 77]. Often, there is a *mission* planning layer at the very top that plans at a very abstract level, over relatively long periods of time. This layer is responsible mainly for selecting which high-level goals are to be achieved over the next period of time (and, in some cases, determining in which order to achieve them) in order to maximize some objective function (e.g., net reward). The lower *task* planning layer is then responsible for determining exactly how and when to achieve each goal. This breakdown is usually done for efficiency reasons, since it is difficult to plan simultaneously at both a detailed level and over a long time horizon.

Integrating Planning and Execution

There are two main approaches to the integration of the planning and execution components in robotic architectures. The first approach is that the planning component is invoked as needed by the executive and returns a plan. The planning component is then dormant until called again. Architectures such as ATLANTIS [12.71] and Remote Agent [12.45] use this approach, which requires that the executive either leave enough time for planning to complete or that it *safes* the system until planning is complete. In the Remote Agent, for instance, a special *planning* task is explicitly scheduled.

The second approach is that the planning component sends high-level tasks down to the executive as required and monitors the progress of those tasks. If tasks fail, replanning is done immediately. In this approach, the planning component is always running and always planning and replanning. Signals must pass in real time between the planner and the executive to keep them synchronized. Architectures such as 3T [12.37] use this second approach. The first approach is useful when the system is relatively static, so that planning can occur infrequently, at relatively predictable times. The second approach is more suited to dynamic environments, where replanning is more frequent and less predictable.

Other decisions that need to be made when integrating planning and execution are when to stop task decomposition, where to monitor plan execution, and how to handle exceptions. By planning all the way down to primitive actions/behaviors, the planner has a very good notion of what will happen during execution, but at a price of much more computation. Also, some task decompositions are easier to describe procedurally (using an executive language) rather than declaratively (using a planning language). Similarly, monitoring at the executive level tends to be more efficient, since the monitoring happens closer to the robot sensors, but the planner may be able to use its more global knowledge to detect exceptions earlier and/or more accurately. With respect to handling exceptions, executives can handle many on their own, at the price of breaking the expectations used by the planner in scheduling tasks. On the other hand, having exceptions handled by the planner typically involves replanning, which can be computationally expensive.

For all these integration issues, however, a middle ground usually exists. For instance, one can choose to decompose some tasks more deeply than others, or han-

dle certain exceptions in the executive and others in the planner. In general, the *right* approach usually involves a compromise and is determined by analyzing the domain in detail (Sect. 12.5).

12.4 Case Study – GRACE

In this section, we present the architecture of a fairly complex autonomous mobile robot. Graduate robot attending conference (GRACE) resulted from the efforts of five research institutions (Carnegie Mellon, Naval Research Laboratory, Northwestern University, Metrica, and Swarthmore College) to tackle the American Association for Artificial Intelligence (AAAI) Robot Challenge. The Challenge was for a robot to attend the AAAI National Conference on Artificial Intelligence as a participant – the robot must find the registration desk (without knowing the layout of the convention center beforehand), register for the conference, and then, after being provided with a map, find its way to a given location in time to give a technical talk about itself.

The architectural design of the robot was particularly important given the complexity of the task and the need to integrate techniques that had been previously developed by the five institutions. These techniques included localization in a dynamic environment, safe navigation in the presence of moving people, path planning, dynamic replanning, visual tracking of people, signs and landmarks, gesture and face recognition, speech recognition and natural language understanding, speech generation, knowledge representation, and social interaction with people.

GRACE is built on top of an real world interface (RWI) B21 base and has an expressive computer-animated face projected on a flat-panel liquid-crystal display (LCD) screen (Fig. 12.9). Sensors that come with the B21 include touch, infrared, and sonar sensors.

Fig. 12.9 The robot GRACE

Near the base is a SICK scanning laser range finder that provides a 180° field of view. In addition, GRACE has several cameras, including a stereo camera head on a pan–tilt unit (PTU) built by Metrica TRACLabs and a single-color camera with pan–tilt–zoom capability, built by Canon. GRACE can speak using a high-quality speech-generation software (Festival), and receive speech responses using a wireless microphone headset (a Shure TC computer wireless transmitter/receiver pair).

The behavioral layer of the architecture consisted of individual processes that controlled particular pieces of hardware. These programs provided abstract interfaces to either control the hardware or return information from sensors. To accommodate the different coding styles of the various groups involved, both synchronous, blocking and asynchronous, nonblocking calls were supported by most of the interfaces (for the nonblocking calls, the interfaces allowed programmers to specify a callback function to be invoked when data was returned). Interfaces at the behavioral level included robot motion and localization (this interface also provided laser information), speech recognition, speech generation, facial animation, color vision, and stereo vision (Fig. 12.10).

The architecture used individual processes for each of the behavioral capabilities, mainly because the underlying code had been developed by different organizations. While having a large number of processes run concurrently is somewhat inefficient, trying to integrate everything into a monolithic process was thought to be too difficult. In addition, the use of separate processes facilitated development and debugging, since one needed to run only those aspects of the system that were being tested.

The executive layer consisted of separate programs for achieving each subtask of the challenge – finding the registration desk, riding the elevator, standing in line, interacting with the person at the desk, navigating to the talk, and giving the talk (Fig. 12.10). As is common in many implemented robotic systems, the GRACE architecture did not have a planning layer – since the high-level plan was fixed and relatively straightforward, it was coded explicitly. Several of the executive-layer programs were written using TDL (Sect. 12.3.3), which facilitated concurrent control and monitoring of the various tasks.

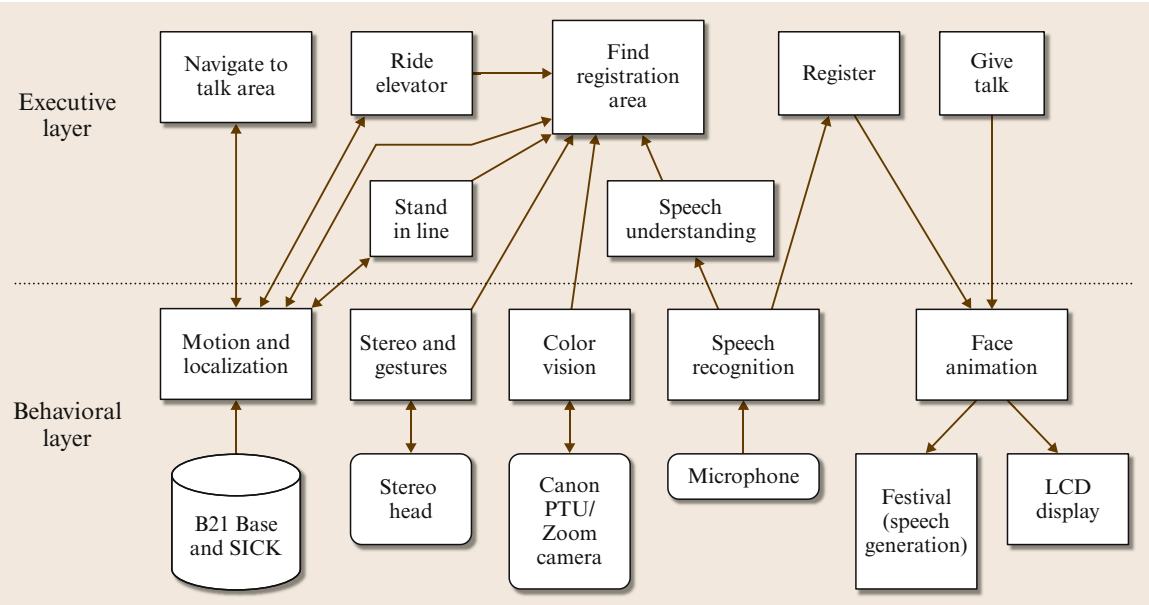


Fig. 12.10 GRACE's architectural structure

One particularly involved task was finding the registration desk (recall that GRACE had no idea where the booth was, or even what the convention center looked like). TDL was used to create a finite-state machine that allowed GRACE to maintain multiple goals, such as using an elevator to get to a particular floor and following directions to find the elevator (Fig. 12.11). The top-level goal was to find the registration desk. Intermediate subgoals were created as GRACE interacted with people to determine the directions to the desk. If there were no directions to follow, GRACE performed a random walk until a person was detected using its laser scanner. GRACE then engaged in conversation with the person to obtain directions. GRACE could handle simple commands, such as *turn left* and *go forward five meters*, as well as higher-level instructions, such as *take the elevator* and *turn left at the next intersection*. In addition, GRACE could ask questions, such as *am I at the registration desk?* and *is this the elevator?* The TDL-based finite-state machine was used to determine which interactions were appropriate at various times and to prevent the robot from getting confused.

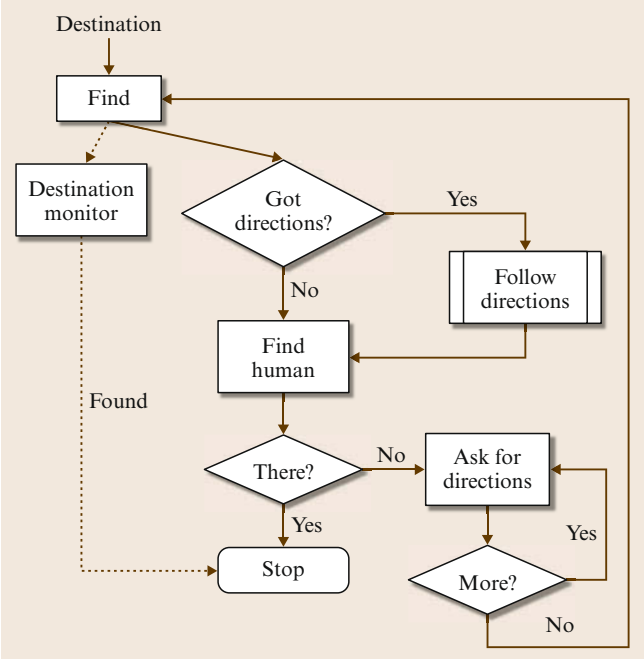


Fig. 12.11 Finite-state machine for GRACE's task for following directions to the registration booth

Communication between processes used the interprocess communication (IPC) messaging package [12.9, 78]. IPC supports both publish–subscribe and client–server messaging, and enables complex data structures to be passed transparently between processes. One side benefit of using IPC to communicate between processes was the ability to log all message traffic (both message name and data content). This proved invaluable, at times, in determining why the system failed to act as expected – did a process send out a message with invalid data? Did it fail to send out a message in a timely fashion? Was the receiving process blocked, for some reason? Was there a timing issue? While wading through the message traffic was often tedious, in

some cases it was the only way to catch intermittent bugs.

In July 2002, GRACE successfully completed the challenge at the Shaw Convention Centre in Edmonton, Canada. The processes at the behavioral level generally worked as anticipated – this was largely attributed to the fact that those modules were ported from previously developed (and hence well-tested) systems. While generally functional, the executive-level processes had more problems with off-nominal situations. This is largely attributed to problems in sensor interpretation, as well as mistaken assumptions about what the convention center was going to look like (for instance, it turned out that some of the barriers were made of glass, which is largely invisible to the laser). Overall, however, the architecture itself worked as expected, enabling a large body of complex software to be integrated rather quickly and operate together effectively.

12.5 The Art of Robot Architectures

Designing a robot architecture is much more of an art than a science. The goal of an architecture is to make programming a robot easier, safer, and more flexible. Thus, the decisions made by a developer of a robot architecture are influenced by their own prior experiences (e.g., what programming languages they are familiar with), their robot and its environment, and the tasks that need to be performed. The choice of a robot architecture should not be taken lightly, as it is the authors' experiences that early architectural decisions often persist for years. Changing robot architectures is a difficult proposition and can set back progress while a great deal of code is reimplemented.

The art of designing a robotic architecture starts with a set of questions that the designer needs to ask. These questions include:

- What are the tasks the robot will be performing? Are they long-term tasks? Short-term? User-initiated? Robot-initiated? Are the tasks repetitive or different across time?
- What actions are necessary to perform the tasks? How are those actions represented? How are those actions coordinated? How fast do actions need to be selected/changed? At what speed do each of the actions need to run in order to keep the robot safe?
- What data is necessary to do the tasks? How will the robot obtain that data from the environment or from the users? What sensors will produce the data? What representations will be used for the data? What processes will abstract the sensory data into representations internal to the architecture? How often does the data need to be updated? How often can it be updated?
- What computational capabilities will the robot have? What data will these computational capabilities produce? What data will they consume? How will the computational capabilities of a robot be divided, structured, and interconnected? What is the best decomposition/granularity of computational capabilities? How much does each computational capability have to know about the other capabilities? Are there legacy computational capabilities (from other robots, other robot projects, etc.) that will be used? Where will the different computational capabilities reside (e.g., onboard or offboard)?
- Who are the robot's users? What will they command the robot to do? What information will they want to see from the robot? What understanding do they need of the robot's computational capabilities? How will the user know what the robot is doing? Is the user interaction peer to peer, supervisory, or as a bystander?
- How will the robot be evaluated? What are the success criteria? What are the failure modes? What is the mitigation for those failure modes?
- Will the robot architecture be used for more than one set of tasks? For more than one kind of robot? By more than one team of developers?

Once designers have answers to all (or most) of these questions, they can then begin building some *use cases* for the types of operations they want the robot to perform and how they want users to interact with it. These use cases should specify the outward behavior of the robot with respect to its environment and its users. From the use cases, an initial partitioning of robot functionality can be developed. This partitioning should be accompanied by a *sequence diagram* that shows the transfer of information and control over time amongst the various components of the robot architecture [12.79]. After this, a more formal specification of the interfaces between architectural components can be developed. This may be done using a language such as the interface definition language (IDL) of CORBA or by defining the messages to be distributed in a publish–subscribe protocol. This is an important step, as once implementation begins it is very costly to change interfaces. If an interface does change, all stakeholders need

to be notified and need to agree to the change. The most common integration problems in robot architectures are mismatches between what components expect and what they are receiving in the way of data.

An advantage of tiered architectures with clear interface definitions is that the different layers can be developed in parallel. The behavioral control layer can be implemented and tested on the robot using a human as an executive. The executive can be implemented and tested using state machine *stubs* for the expected behaviors on the robot. The planning layer can be implemented and tested using *stubs* for the tasks in the executive. The *stubs* merely acknowledge that they were called and report back appropriately. Then, the tiers can be integrated to test timing and other runtime issues. This parallel approach speeds up the development of a robot architecture, but is possible only if the roles and interfaces between components are clearly defined and respected. There is still considerable real-time debugging necessary during integration. In our experiences, most of the development time in robot architectures is still spent on the behavioral control layer – that is, sensing and acting are still the hard parts of robot control, as compared to execution and planning. Having a good, robust behavioral control layer goes a long way towards having a competent robot architecture.

12.6 Implementing Robotic Systems Architectures

Robot System Architectures offer a coarse-gain view of the structure of a robot system, which is modelled according to the functional decomposition (or composition) of parts. Unfortunately, there are concerns that cannot be effectively described and modularised using the concept of composition of high cohesion functional modules. Such concerns relate to the software systems as a whole hence crosscutting their modular structure. Non-functional requirements such as real-time performance, fault tolerance, and safety, are typical examples of properties that emerge from multiple parts of a system and cannot be confined into individual modules. In order to meet these requirements, software development efforts are mainly focused on delivering highly efficient implementations of control applications that exploit at best the capabilities of specific hardware platforms to perform a specific set of tasks in specific operational environments.

The race towards performance and the pressure to develop proof-of-concept code to test a new theory [12.80] push robotic engineers to neglect other quality attributes of a software system, such as maintainability, interoperability, and reusability. As a result, a huge corpus of software applications, which implement the entire spectrum of robot functionality, algorithms, and control paradigms, is potentially available in robotic research laboratories as open source libraries. Unfortunately, they are often not reusable even in slightly different application scenarios, because the assumptions and constraints about the computational and robotics hardware, and the software and operational environments are hidden and hard coded in the software implementation.

A large variety of today's robotic systems is designed according to the relatively small number of robot control architecture paradigms discussed in the previous sections. Nevertheless, the software that implements their control applications differs significantly from system to system. These differences relate, for example, to the data structures defined to store relevant information (e.g., the map of the environment, the robot kinematic model), the application programming interfaces (APIs) to drive sensors and actuators [12.81], the information model used to represent key concepts (e.g., geometric relations and coordinate representations). These differences indicate that the most common robot functionalities have been re-implemented from scratch innumerable times.

During the last few years, many ideas from software engineering (such as component-based development and model-driven engineering) have been progressively introduced in the construction of robotic software systems to simplify their development and improve their quality (see [12.82] for a survey).

Modern robot control systems are typically designed as (logically) distributed component-based systems [12.83, 84]. Here, components are units of implementation and represent a code-based way of considering the system. As an analogy, in the electronics domain re-usable off-the-shelf electronic components have been available for many years in the form of integrated chips (ICs) which can be bought and deployed in other parts of the world. This is possible because each IC packages a clear set of functionality and provides a well-defined external interface.

In Robot System Architectures the interactions between software components are usually more complex compared to more traditional business applications. In Robotics, the software developer faces the complexity of event-based, reactive, and distributed interactions between sensors and motors and between several processing algorithms. Managing concurrent access to shared

resources by multiple (distributed) activities is one of the main issues, as thoroughly discussed in [12.85]. For this reason, robotic-specific component-based frameworks and toolkits have been developed, which offer mechanisms for real-time execution, synchronous and asynchronous communication, data flow and control flow management, and system configuration.

The following section reviews some of the most recent approaches and frameworks that have been specifically defined for robotics.

12.6.1 Agile Robot Development Network (aRDnet)

aRDnet [12.86] is a software suite developed at the German Aerospace Center (DLR) Institute of Robotics and Mechatronics that supports the development of distributed component-based systems for complex mechatronic systems with hard real-time requirements. The aRDnet suite has been used to implement computationally demanding control loops in the kHz range running over all the 41 degrees of freedom (DOF) of the upper humanoid body Justin, such as impedance control and collision avoidance. The robot control system is structured as a network of functional blocks and communication links distributed over a network of computers connected by a fast digital bus like the Gigabit-Ethernet.

An aRDnet functional block is a software module with multiple input and output ports that hide the actual transportation protocol, i. e., user datagram protocol (UDP) sockets or EtherCAT to exchange data with interconnected blocks. Blocks can be executed by individual processes or can be grouped in synchronization groups (e.g., for blocks interfacing the robot hardware) executed iteratively by a single process according to the *synchronous data flow* model of computation. Different synchronization groups can interact asynchronously through non blocking read and write operations.

12.6.2 Yet Another Robot Platform (YARP)

YARP [12.87] is an open-source project developed at the LIRA-Lab of the University of Genova in collaboration with MIT Computer Science and Artificial Intelligence Laboratory (CSAIL). It has been conceived with the same objectives as aRDnet, i. e., the development of distributed control systems for high DOF robots, such as the humanoid bodies. It consists in a lightweight software library for concurrent and distributed programming that has been used and tested on several operating systems.

Similarly to aRDnet, distributed software modules exchange data through input/output asynchronous communication ports. Differently from aRDnet, YARP ports can manage multiple connections for a given module at different data rates according to different protocols (e.g., transfer control protocol (TCP), UDP, multicast) allowing for the configuration of the quality of service (QOS) of the inter-module communication: an output port can send data to many destinations, while an input port can receive data from multiple connections.

12.6.3 Open Robot Control Software (OROCOS)

Orocos is one of the oldest open source framework in robotics, under development since 2001, and with professional industrial applications and products using it since about 2005 [12.88]. The focus of Orocos has always been to provide a hard real-time capable component framework, the so-called Real-Time Toolkit (RTT) implemented in C++ and as independent as possible from any communication middleware and operating system. Similarly to aRDnet and YARP, components interact with each other by exchanging data and events asynchronously through lock-free input/output ports according to the Data Flow communication paradigm.

The distinguish feature of OROCOS is the definition of a component model that specifies a standard behaviour for concurrent activities. Components with real-time, deterministic and cyclic behaviour get fixed and cyclic time budgets for computation and within a computation cycle they must reach stable intermediate states. OROCOS components are implemented as extension (inheritance) of the base class TaskContext, have their own thread of execution, and can be deployed as objects that share the same address space or as executables that communicate using the CORBA middleware.

12.6.4 Smartsoft

Smartsoft [12.89] is an open source framework that specifically addresses issues related to communication among software components of a robotic control system. Similarly to OROCOS, it defines a port-based component model, whose distinguished feature is a rich set of standard component interfaces (called communication patterns) with strictly defined interaction semantics:

- *Send*: Defines one-way communication with a client/server relationship
- *Query*: Two-way request communication with a client/server relationship

- *Push newest/push timed*: 1-to-n distribution (broadcast) with a publisher/subscriber relationship
- *Event*: Asynchronous conditioned notification with a client/server relationship
- *Dynamic wiring*: Dynamic component wiring with a master/slave relationship.

Smartsoft is one of the first open source frameworks that has developed an open source toolchain for robotic software development based on the Eclipse Modeling Project. The toolchain implements a workflow that guides the software developer from high level design of the robot system architecture down to the components implementation and deployment through automatic model-to-model transformations.

12.6.5 Robot Operating System (ROS)

ROS [12.90] is a message-based peer-to-peer communication infrastructure supporting the easy integration of independently developed software components, called ROS nodes. A ROS system is thus a computation graph consisting of a set of nodes communicating with one another. Nodes are blocks of functional code and are implemented as classes that wrap robotic software libraries and provide access to the communication mechanisms of the underlying infrastructure (the ROS core). Nodes are organized into packages (file system folders containing libraries, nodes, and message definitions), which are grouped into thematic stacks (i. e., the navigation stack). Messages are typed data structures that can be nested into compound messages and are exchanged between nodes according to the publish/subscribe communication paradigm in an asynchronous manner without the need for the interacting nodes to know each other and to participate to the interaction at the same time. Messages are organized by topics, which correspond to information subjects that allow subscribers to recognize the events they are interested in. When a node receives a message belonging to a subscribed topic, a message handler is invoked asynchronously, which performs some computation on the message payload data and possibly generates a new message of a given topic to publish the computation results.

The distinguish characteristics of ROS is the lack of enforced architecture in ROS libraries, i. e., ROS nodes are designed as individually reusable components. An application is built by configuring individual nodes in such a way that they can exchange messages belonging to common topics. This characteristic favours the decentralized development of many small packages by experts in the various robotics sub-domains.

12.6.6 GenoM/BIP

GenoM [12.91] is a component-oriented software package developed by the LAAS CNRS robotics group that is used for specifying and implementing the functional level of robot system architectures. GenoM components are collections of control services, which execute a finite state automaton and are implemented by a set of C functions called codels, which get appropriately called during specific state transitions (i. e., start, exec, error, etc.). GenoM components exchange data and events through posters that are regions of shared memory.

The distinguish feature of the GenoM component model is the integration with the Behavior-Interaction-Priorities (BIPs) framework. BIP is a software framework and toolset for formally modeling and verifying complex, real-time component-based systems to guarantee correctness-by-construction of robotic control systems. BIP is used to produce a formal interaction model, which can be used for system level coordination to run (using the BIP engine) the functional layer composed of all the GenoM modules. BIP allows the hierarchical construction of compound components by using connectors, which interconnect the ports of GenoM components and models two basic modes of communication, namely, caller/provider and broadcaster/listener.

12.6.7 Best Practice in Robotics (BRICS)

Complementary to the projects described above BRICS [12.92], a joint research project funded by the European Commission, has formalized the robot development process according to the principles of Model Driven Engineering [12.93] and has provided tools, models, and functional libraries, which allow reducing the development time of robot software systems.

In particular, the concepts of Component Framework and Software Product Line have been introduced in the robot development process [12.94, 95] as illustrated in VIDEO 273. This approach promotes the routine use of existing software or software knowledge to construct new software, so that similarities in requirements, architectures and design between applications can be exploited to achieve substantial benefits in software quality, productivity, and business performance.

12.7 Conclusions and Further Reading

Robot architectures are designed to facilitate the concurrent execution of task-achieving behaviors. They enable systems to control actuators, interpret sensors, plan, monitor execution, and deal with unexpected contingencies and opportunities. They provide the conceptual framework within which domain-dependent software development can take place, and they often provide programming tools that facilitate that development.

While no single architecture has proven to be best for all applications, researchers have developed a variety of approaches that can be applied in different situations. While there is not yet a specific formula for determining which architecture will be best suited for a given application, this chapter provides some guidelines to help developers in selecting the right architecture for the job. That being said, layered architectures have proven to be increasingly popular, due to their flexibility and ability to operate at multiple levels of abstraction simultaneously.

The book *Kortenkamp* et al. [12.96] provide several chapters on architectures that have influenced this chapter. Most text books in robotics [12.20, 97, 98] have sections on robot architectures. For many years in the mid 1990s, the AAAI Spring Symposia on Artificial Intelligence had sessions devoted to robot architectures, although proceedings from those symposia are not widely available. More information on GRACE can be found in [12.99–101].

Video-References

VIDEO 273 Software product line engineering for robotics
available from http://handbookofrobotics.org/view-chapter/12/videodetails/273

References

12.1 I. Jacobson, G. Booch, J. Rumbaugh: *The Unified Software Development Process* (Addison-Wesley, Reading 1998)

12.2 J.S. Albus: RCS: A reference model architecture for intelligent systems, Working Notes: AAAI 1995 Spring Symp. Lessons Learn. Implement. Software Archit. Phys. Agents (1995)

12.3 R.A. Brooks: A robust layered control system for a mobile robot, IEEE J. Robot. Autom. **2**(1), 14–23 (1986)

12.4 R.J. Firby: An investigation into reactive planning in complex domains, Proc. 5th Natl. Conf. Artif. Intel. (1987)

12.5 R. Simmons: Structured control for autonomous robots, IEEE Trans. Robot. Autom. **10**(1), 34–43 (1994)

12.6 J.J. Borrelly, E. Coste-Maniere, B. Espiau, K. Kapellos, R. Pissard-Gibollet, D. Simon, N. Turro: The ORCCAD architecture, Int. J. Robot. Res. **17**(4), 338–359 (1998)

12.7 B. Kuipers: The spatial semantic hierarchy, Artif. Intell. **119**, 191–233 (2000)

12.8 R. Orfali, D. Harkey: *Client/Server Programming with JAVA and CORBA* (Wiley, New York 1997)

12.9 R. Simmons, G. Whelan: Visualization tools for validating software of autonomous spacecraft, Proc. Int. Symp. Artif. Intel., Robot. Autom. in Space, Tokyo (1997)

12.10 R. A. Brooks: The Behavior Language: User's Guide, Tech. Rep. AIM-1227 (MIT, Artif. Intel. Lab, Cambridge 1990)

12.11 R.J. Firby, M.G. Slack: Task execution: Interfacing to reactive skill networks, Working Notes: AAAI Spring Symp. Lessons Learn. Implement. Archit. Phys. Agents, Stanford (1995)

12.12 E. Gat: ESL: A language for supporting robust plan execution in embedded autonomous agents, Proc. IEEE Aerosp. Conf. (1997)

12.13 V. Verma, T. Estlin, A. Jónsson, C. Pasareanu, R. Simmons, K. Tso: Plan execution interchange language (PLEXIL) for executable plans and command sequences, Proc. 8th Int. Symp. Artif. Intel. Robot. Autom. Space, Munich (2005)

12.14 S.A. Schneider, V.W. Chen, G. Pardo-Castellote, H.H. Wang: ControlShell: A software architecture for complex electromechanical systems, Int. J. Robot. Res. **17**(4), 360–380 (1998)

12.15 National Instruments: *LabVIEW* (National Instruments, Austin 2007) http://www.ni.com/labview/

12.16 N.J. Nilsson: A mobile automaton: an application of AI techniques, Proc. 1st Int. Joint Conf. Artif. Intel. (Morgan Kaufmann, San Francisco 1969) pp. 509–520

12.17 N.J. Nilsson: *Principles of Artificial Intelligence* (Tioga, Palo Alto 1980)

12.18 SRI International: Shakey the Robot, http://www.sri.com

12.19 P.E. Agre, D. Chapman: Pengi: An implementation of a theory of activity, Proc. 5th Natl. Conf. Artif. Intel. (1987)

12.20 R.C. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1998)

12.21 J.H. Connell: SSS: A hybrid architecture applied to robot navigation, Proc. IEEE Int. Conf. Robot. Autom. (1992) pp. 2719–2724

12.22 M. Mataric: Integration of representation into goal-driven behavior-based robots, Proc. IEEE Int. Conf. Robot. Autom. (1992)

12.23 I. Horswill: Polly: A vision-based artificial agent, Proc. Natl. Conf. Artif. Intel. (AAAI) (1993)

12.24 D.W. Payton: An architecture for reflexive autonomous vehicle control, Proc. IEEE Int. Conf. Robot. Autom. (1986)

12.25 J.K. Rosenblatt: DAMN: A Distributed Architecture for Mobile Robot Navigation, Ph.D. Thesis (Carnegie Mellon Univ., Pittsburgh 1997)

12.26 R.C. Arkin: Motor schema-based mobile robot navigation, Int. J. Robot. Res. **8**(4), 92–112 (1989)

12.27 M. Arbib: Schema Theory. In: *Encyclopedia of Artificial Intelligence*, ed. by S. Shapiro (Wiley, New York 1992) pp. 1427–1443

12.28 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Proc. IEEE Int. Conf. Robot. Autom. (1985) pp. 500–505

12.29 R.C. Arkin: Integrating behavioral, perceptual, and world knowledge in reactive navigation, Robot. Autonom. Syst. **6**, 105–122 (1990)

12.30 R.C. Arkin, T. Balch: AuRA: Principles and practice in review, J. Exp. Theor. Artif. Intell. **9**(2/3), 175–188 (1997)

12.31 R.J. Firby: Adaptive Execution in Complex Dynamic Worlds, Ph.D. Thesis (Yale Univ., New Haven 1989)

12.32 R.J. Firby: Task networks for controlling continuous processes, Proc. 2nd Int. Conf. AI Plan. Syst. (1994)

12.33 R.P. Bonasso: Integrating reaction plans and layered competences through synchronous control, Proc. Int. Joint Conf. Artif. Intel. (1991)

12.34 S.J. Rosenschein, L.P. Kaelbling: The synthesis of digital machines with provable epistemic properties, Proc. Conf. Theor. Asp. Reas. Knowl. (1998)

12.35 L.P. Kaelbling: Goals as parallel program specifications, Proc. 6th Natl. Conf. Artif. Intel. (1988)

12.36 L. P. Kaelbling: *Compiling operator descriptions into reactive strategies using goal regression*, Tech. Rep., TR90-10, (Teleos Res., Palo Alto 1990)

12.37 R.P. Bonasso, R.J. Firby, E. Gat, D. Kortenkamp, D.P. Miller, M.G. Slack: Experiences with an architecture for intelligent, reactive agents, J. Exp. Theor. Artif. Intell. **9**(2/3), 237–256 (1997)

12.38 E. Gat: Integrating Planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots, Proc. Natl. Conf. Artif. Intel. (AAAI) (1992)

12.39 G.N. Saridis: Architectures for intelligent controls. In: *Intelligent Control Systems: Theory and Applications*, ed. by S. Gupta (IEEE Press, Piscataway 1995)

12.40 R. Alami, R. Chatila, S. Fleury, M. Ghallab, F. Ingrand: An architecture for autonomy, Int. J. Robot. Res. **17**(4), 315–337 (1998)

12.41 M. Ghallab, H. Laruelle: Representation and control in IxTeT, a temporal planner, Proc. AIPS-94 (1994)

12.42 P. Laborie, M. Ghallab: Planning with sharable resource constraints, Proc. Int. Joint Conf. Artif. Intel. (1995)

12.43 M.P. Georgeff, F.F. Ingrand: Decision-making in an embedded reasoning system, Proc. Int. Joint Conf. Artif. Intel. (1989) pp. 972–978

12.44 F. Ingrand, R. Chatila, R. Alami, F. Robert: PRS: A high level supervision and control language for autonomous mobile robots, Proc. IEEE Int. Conf. Robot. Autom. (1996)

12.45 N.P. Muscettola, P. Nayak, B. Pell, B.C. Williams: Remote agent: To boldly go where no AI system has gone before, Artif. Intell. **103**(1), 5–47 (1998)

12.46 B.C. Williams, P.P. Nayak: A model-based approach to reactive self-configuring systems, Proc. AAAI (1996)

12.47 B. Sellner, F.W. Heger, L.M. Hiatt, R. Simmons, S. Singh: Coordinated multi-agent teams and sliding autonomy for large-scale assembly, Proc. IEEE **94**(7), 1425–1444 (2006), special issue on multi-agent systems

12.48 D. Goldberg, V. Cicirello, M.B. Dias, R. Simmons, S. Smith, A. Stentz: Market-based multi-robot planning in a distributed layered architecture, Proc. Int. Workshop Multi-Robot Syst (2003) pp. 27–38

12.49 J.S. Albus, R. Lumia, H.G. McCain: NASA/NBS Standard Reference model for Telerobot Control System Architecture (NASREM), Technol. #1235, (Nat. Inst. Stand, Gaithersburg 1986)

12.50 D.R. Blidberg, S.G. Chappell: Guidance and control architecture for the EAVE vehicle, IEEE J. Ocean Eng. **11**(4), 449–461 (1986)

12.51 J.S. Albus: Outline for a theory of intelligence, IEEE Trans. Syst. Man Cybern. **21**(3), 473–509 (1991)

12.52 R. Volpe, I. Nesnas, T. Estlin, D. Mutz, R. Petras, H. Das: The CLARAty architecture for robotic autonomy, Proc. IEEE Aerosp. Conf., Big Sky (2001)

12.53 I.A. Nesnas, R. Simmons, D. Gaines, C. Kunz, A. Diaz-Calderon, T. Estlin, R. Madison, J. Guineau, M. McHenry, I. Shu, D. Apfelbaum: CLARAty: Challenges and steps toward reusable robotic software, Int. J. Adv. Robot. Syst. **3**(1), 023–030 (2006)

12.54 T. Estlin, D. Gaines, C. Chouinard, F. Fisher, R. Castaño, M. Judd, R. Anderson, I. Nesnas: Enabling autonomous rover science through dynamic planning and scheduling, Proc. IEEE Aerosp. Conf., Big Sky (2005)

12.55 R. Knight, G. Rabideau, S. Chien, B. Engelhardt, R. Sherwood: CASPER: Space exploration through continuous planning, IEEE Intell. Syst. **16**(5), 70–75 (2001)

12.56 R. Simmons, D. Apfelbaum: A task description language for robot control, Proc. Conf. Intel. Robot Syst., Vancouver (1998)

12.57 T.A. Estlin, D. Gaines, C. Chouinard, R. Castaño, B. Bornstein, M. Judd, I.A.D. Nesnas, R. Anderson: Increased mars rover autonomy using AI planning, scheduling and execution, Proc. Int. Conf. Robot. Autom. (2007) pp. 4911–4918

12.58 D. Musliner, E. Durfee, K. Shin: World modeling for dynamic construction of real-time control plans, Artif. Intell. **74**(1), 83–127 (1995)

12.59 D.J. Musliner, R.P. Goldman, M.J. Pelican: Using model checking to guarantee safety in automatically-synthesized real-time controllers, Proc. Int. Conf. Robot. Autom. (2000)

12.60 B. Espiau, K. Kapellos, M. Jourdan: Formal verification in robotics: Why and how?, Proc. Int. Symp. Robot. Res., Herrsching (1995)

12.61 G. Berry, G. Gonthier: The Esterel synchronous programming language: Design, semantics, implementation, Sci. Comput. Progr. **19**(2), 87–152 (1992)

12.62 M. Jourdan, F. Maraninchi, A. Olivero: Verifying quantitative real-time properties of synchronous programs, Lect. Notes Comput. Sci. **697**, 347–358 (1993)

12.63 G. Pardo-Castellote, S.A. Schneider: The network data delivery service: Real-time data connectivity for distributed control applications, Proc. Int. Conf. Robot. Autom. (1994) pp. 2870–2876

12.64 JAUS Reference Architecture Specification, Volume II, Part 1 Version 3.2: http://www.jauswg.org/baseline/refarch.html

12.65 JAUS Tutorial Powerpoint slides: http://www.jauswg.org/

12.66 JAUS Domain Model Volume I, Version 3.2: http://www.jauswg.org/baseline/current_baseline.shtml

12.67 E. Gat: ALFA: A language for programming reactive robotic control systems, Proc. IEEE Int. Conf. Robot. Autom. (1991) pp. 116–1121

12.68 R.A. Brooks: Elephants don't play chess, J. Robot. Autonom. Syst. **6**, 3–15 (1990)

12.69 L.P. Kaelbling: Rex – A symbolic language for the design and parallel implementation of embedded systems, Proc. 6th AIAA Comput. Aerosp. Conf., Wakefield (1987)

12.70 E. Gat: Non-linear sequencing and cognizant failure, Proc. AIP Conf. (1999)

12.71 E. Gat: On the role of stored internal state in the control of autonomous mobile robots, AI Mag. **14**(1), 64–73 (1993)

12.72 J.L. Peterson: *Petri Net Theory and the Modeling of Systems* (Prentice Hall, Upper Saddle River 1981)

12.73 K. Currie, A. Tate: O-Plan: The open planning architecture, Artif. Intell. **52**(1), 49–86 (1991)

12.74 D.S. Nau, Y. Cao, A. Lotem, H. Muñoz-Avila: SHOP: Simple hierarchical ordered planner, Proc. Int. Joint Conf. Artif. Intel. (1999) pp. 968–973

12.75 S. Chien, R. Knight, A. Stechert, R. Sherwood, G. Rabideau: Using iterative repair to improve the responsiveness of planning and scheduling, Proc. Int. Conf. AI Plan. Sched. (2000) pp. 300–307

12.76 N. Muscettola: HSTS: Integrating planning and scheduling. In: *Intelligent Scheduling*, ed. by M. Fox, M. Zweben (Morgan Kaufmann, San Francisco 1994)

12.77 R. Simmons, J. Fernandez, R. Goodwin, S. Koenig, J. O'Sullivan: Lessons learned from Xavier, IEEE Robot. Autom. Mag. **7**(2), 33–39 (2000)

12.78 R. Simmons: Inter Process Communication (Carnegie Mellon Univ., Pittsburgh 2007), http://www.cs.cmu.edu/IPC

12.79 S.W. Ambler: UML 2 Sequence Diagramms (Ambisoft, Toronto 2007) http://www.agilemodeling.com/artifacts/sequenceDiagram.htm

12.80 W.D. Smart: Writing code in the field: Implications for robot software development, Springer Tract. Adv. Robot. **30**, 93–105 (2007)

12.81 I.A.D. Nesnas: The CLARAty Project: Coping with hardware and software heterogeneity, Springer Tract. Adv. Robot. **30**, 31–70 (2007)

12.82 D. Brugali, E. Prassler: Software engineering for robotics, IEEE Robot. Autom. Mag. **16**(1), 9–15 (2009)

12.83 D. Brugali, P. Scandurra: Component-based robotic engineering (Part I), IEEE Robot. Autom. Mag. **16**(4), 84–96 (2009)

12.84 D. Brugali, A. Shakhimardanov: Component-based robotic engineering (Part II), IEEE Robot. Autom. Mag. **17**(1), 100–112 (2010)

12.85 D. Calisi, A. Censi, L. Iocchi, D. Nardi: Design choices for modular and flexible robotic software development: The OpenRDK viewpoint, J. Software Eng. Robot. **3**(1), 13–27 (2012)

12.86 B. Baeuml, G. Hirzinger: When hard realtime matters: Software for complex mechatronic systems, Robot. Auton. Syst. **56**, 5–13 (2008)

12.87 G. Metta, P. Fitzpatrick, L. Natale: YARP: Yet another robot platform, Int. J. Adv. Robot. Syst **3**(1), 43–48 (2006)

12.88 H. Bruyninckx: Open robot control software: the OROCOS project, Proc. IEEE Int. Conf. Robot. Autom. (2001) pp. 2523–2528

12.89 C. Schlegel, A. Steck, D. Brugali, A. Knoll: Design abstraction and processes in robotics: From code-driven to model-driven engineering, Lect. Notes Comput. Sci. **6472**, 324–335 (2010)

12.90 S. Cousins: Welcome to ROS Topics, IEEE Robot. Autom. Mag. **17**(1), 13–14 (2010)

12.91 S. Bensalem, L. de Silva, F. Ingrand, R. Yan: A verifiable and correct-by-construction controller for robot functional levels, J. Software Eng. Robot. **2**(1), 1–19 (2011)

12.92 R. Bischoff, T. Guhl, E. Prassler, W. Nowak, G. Kraetzschmar, H. Bruyninckx, P. Soetens, M. Haegele, A. Pott, P. Breedveld, J. Broenink, D. Brugali, N. Tomatis: BRICS: Best practice in robotics, Proc. 41st Int. Symp. Robot., Munich (2010) pp. 7–9

12.93 M. Klotzbuecher, N. Hochgeschwender, L. Gherardi, H. Bruyninckx, G. Kraetzschmar, D. Brugali: The BRICS component model: A model-based development paradigm for complex robotics software systems, 28th ACM Symp. Appl. Comput., Coimbra (2013) pp. 18–22

12.94 D. Brugali, L. Gherardi, A. Luzzana, A. Zakharov: A reuse-oriented development process for component-based robotic systems, Lect. Notes Comput. Sci. **7628**, 361–374 (2012)

12.95 L. Gherardi, D. Brugali: Modeling and reusing robotic software architectures: The hyperflex

toolchain, Proc. IEEE Robot. Autom. Conf. ICRA '14, Hong Kong (2014)
12.96 D. Kortenkamp, R.P. Bonasso, R. Murphy: *Artificial Intelligence and Mobile Robots* (MIT Press, Cambridge 1998)
12.97 R. Murphy: *Introduction to AI Robotics* (MIT Press, Cambridge 2000)
12.98 R. Siegwart, I.R. Nourbakhsh: *Introduction to Autonomous Mobile Robots* (MIT Press, Cambridge 2004)
12.99 R. Simmons, D. Goldberg, A. Goode, M. Montemerlo, N. Roy, B. Sellner, C. Urmson, A. Schultz, M. Abramson, W. Adams, A. Atrash, M. Bugajska, M. Coblenz, M. MacMahon, D. Perzanowski, I. Horswill, R. Zubek, D. Kortenkamp, B. Wolfe, T. Milam, B. Maxwell: GRACE: An autonomous robot for the AAAI Robot Challenge, AAAI Mag. **24**(2), 51–72 (2003)
12.100 R. Gockley, R. Simmons, J. Wang, D. Busquets, C. DiSalvo, K. Caffrey, S. Rosenthal, J. Mink, S. Thomas, W. Adams, T. Lauducci, M. Bugajska, D. Perzanowski, A. Schultz: Grace and George: Social Robots at AAAI, Proc. AAAI Mob. Robot Comp. Workshop (2004), pp. 15–20
12.101 M.P. Michalowski, S. Sabanovic, C. DiSalvo, D. Busquets, L.M. Hiatt, N.A. Melchior, R. Simmons: Socially Distributed Perception: GRACE plays social tag at AAAI 2005, Auton. Robot. **22**(4), 385–397 (2007)

Multimedia Contents

13. Behavior-Based Systems

François Michaud, Monica Nicolescu

Nature is filled with examples of autonomous creatures capable of dealing with the diversity, unpredictability, and rapidly changing conditions of the real world. Such creatures must make decisions and take actions based on incomplete perception, time constraints, limited knowledge about the world, cognition, reasoning and physical capabilities, in uncontrolled conditions and with very limited cues about the intent of others. Consequently, one way of evaluating intelligence is based on the creature's ability to make the most of what it has available to handle the complexities of the real world. The main objective of this chapter is to explain *behavior-based systems* and their use in autonomous control problems and applications. The chapter is organized as follows. Section 13.1 overviews robot control, introducing behavior-based systems in relation to other established approaches to robot control. Section 13.2 follows by outlining the basic principles of behavior-based systems that make them distinct from other types of robot control architectures. The concept of basis behaviors, the means of modularizing behavior-based systems, is presented in Sect. 13.3. Section 13.4 describes how behaviors are used as building blocks for creating representations for use by behavior-based systems, enabling the robot to reason about the world and about itself in that world. Section 13.5 presents several different classes of learning methods for behavior-based systems, validated on single-robot and multi-robot systems. Section 13.6 provides an overview of various robotics problems and application domains that have successfully been addressed or are currently being studied with behavior-based control. Finally, Sect. 13.7 concludes the chapter.

13.1 **Robot Control Approaches** 308
13.1.1 Deliberative – Think, Then Act 308
13.1.2 Reactive – Don't Think, (Re)Act ... 308
13.1.3 Hybrid – Think and Act Concurrently 309
13.1.4 Behavior-Based Control – Think the Way You Act 309
13.1.5 When to Use What 309
13.2 **Basic Principles of Behavior-Based Systems** 310
13.2.1 Misconceptions 312
13.3 **Basis Behaviors** 313
13.4 **Representation in Behavior-Based Systems** 313
13.5 **Learning in Behavior-Based Systems** 314
13.5.1 Reinforcement Learning in Behavior-Based Systems 315
13.5.2 Learning Behavior Networks 315
13.5.3 Learning from Demonstration in Behavior-Based Systems 316
13.5.4 Learning from History of Behavior Use 317
13.6 **Applications and Continuing Work** 318
13.6.1 Adaptive Behavior Selection – A Case Study 319
13.7 **Conclusions and Further Reading** 322
Video-References 322
References 323

13.1 Robot Control Approaches

Situated robotics deals with embodied machines in complex, challenging, often dynamically changing environments. *Situatedness* thus refers to existing in a complex, challenging environment, and having one's behavior strongly affected by it. In contrast, robots that exist in static, unchanging environments are usually not thought to be situated. These include assembly robots operating in complex but highly structured, fixed, and strongly predictable environments, specifically engineered and controlled to enable the robot to accomplish very specific tasks. The predictability and stability of the environment have a direct impact on the complexity of the robot that must operate in it; situated robots therefore present a significant challenge for the designer.

Robot control, also referred to as robot decision-making or robot computational architecture, is the process of taking information about the environment through the robot's sensors, processing it as necessary in order to make decisions about how to act, and executing actions in the environment. The complexity of the environment, i. e., the level of situatedness, has a direct impact on the complexity of control, which is, in turn, directly related to the robot's task. Control architectures are covered in Chap. 12 of the Handbook.

While there are infinitely many possible ways to program a robot, there are fundamentally four classes of robot control methods, described below.

13.1.1 Deliberative – Think, Then Act

In deliberative control, the robot uses all of the available sensory information, and all of the internally stored knowledge, to reason about what actions to take next. The control system is usually organized using a functional decomposition of the decision-making processes, consisting of a sensory processing module, a modeling module, a planning module, a value judgment module, and an execution module [13.1]. Such functional decomposition allows complex operations to be performed, but implies strong sequential interdependencies between the decision-making modules.

Reasoning in deliberative systems is typically in the form of planning, requiring a search of possible state–action sequences and their outcomes. *Planning*, a major component of artificial intelligence, is known to be a computationally complex process. The process requires the robot to perform a sequence of sense–plan–act steps (e.g., *combine the sensory data into a map of the world, then use the planner to find a path in the map, then send steps of the plan to the robot's wheels*) [13.2–4]. The robot must construct and then potentially evaluate all possible plans until it finds one that enables it to reach its goal, solve the task, or decide on a trajectory to execute. Shakey, an early mobile robot that used STRIPS, a general planner, is an example of such a system applied to the problem of avoiding obstacles and navigating based on vision data [13.5].

Planning requires the existence of an internal, symbolic representation of the world, which allows the robot to look ahead into the future and predict the outcomes of possible actions in various states, so as to generate plans. The internal model, thus, must be kept accurate and up to date. When there is sufficient time to generate a plan and the world model is accurate, this approach allows the robot to act strategically, selecting the best course of action for a given situation. However, being situated in a noisy, dynamic world usually makes this impossible [13.6, 7]. Today, no situated robots are purely deliberative. The advent of alternative architectures was driven by the need for faster yet appropriate action in response to the demands of complex and dynamically changing real-world environments.

13.1.2 Reactive – Don't Think, (Re)Act

Reactive control is a technique for tightly coupling sensory inputs and effector outputs, typically involving no intervening reasoning [13.8] to allow the robot to respond very quickly to changing and unstructured environments [13.9]. Reactive control is inspired by the biological notion of *stimulus–response*; it does not require the acquisition or maintenance of world models, as it does not rely on the types of complex reasoning processes utilized in deliberative control. Rather, rule-based methods involving a minimal amount of computation, and no internal representations or knowledge of the world are typically used. Reactive systems achieve rapid real-time responses by embedding the robot's controller in a collection of preprogrammed, concurrent condition–action rules with minimal internal state (e.g., *if bumped*, *stop*; *if stopped*, *back up*) [13.8, 10]. This makes reactive control especially well suited to dynamic and unstructured worlds where having access to a world model is not a realistic option. Furthermore, the minimal amount of computation involved means that reactive systems are able to respond in a timely manner to rapidly changing environments.

Reactive control is a powerful and effective control method that abounds in nature; insects, which vastly outnumber vertebrates, are largely reactive. However, limitations to pure reactivity include the inability to store (much if any) information or have memory or internal representations of the world [13.11], and therefore the inability to learn and improve over time.

Reactive control trades off complexity of reasoning for fast reaction time. Formal analysis has shown that, for environments and tasks that can be characterized a priori, reactive controllers can be very powerful, and if properly structured, capable of optimal performance in particular classes of problems [13.12, 13]. In other types of environments and tasks where internal models, memory, and learning are required, reactive control is not sufficient.

13.1.3 Hybrid – Think and Act Concurrently

Hybrid control aims to combine the best aspects of reactive and deliberative control: the real-time response of reactivity and the rationality and optimality of deliberation. As a result, hybrid control systems contain two different components, the reactive/concurrent condition–action rules and the deliberative ones, which must interact in order to produce a coherent output. This is challenging because the reactive component deals with the robot's immediate needs, such as moving while avoiding obstacles, and thus operates on a very fast time scale and uses direct external sensory data and signals. In contrast, the deliberative component uses highly abstracted, symbolic, internal representations of the world, and operates on them on a longer time scale, for example to perform global path planning or plan for high-level decision-making. As long as the outputs of the two components are not in conflict, the system requires no further coordination. However, the two parts of the system must interact if they are to benefit from each other. Consequently, the reactive system must override the deliberative one if the world presents some unexpected and immediate challenge. Analogously, the deliberative component must inform the reactive one in order to guide the robot toward more efficient and optimal trajectories and goals. The interaction of the two parts of the system requires an intermediate component, which reconciles the different representations used by the other two and any conflicts between their outputs. The construction of this intermediate component is typically the greatest challenge of hybrid system design.

Hybrid systems are referred to as layered, or tiered, robot control architecture (Chap. 12). Many are referred to as *three-layer architectures* because of their structure, which consists of the reactive (execution) layer, intermediate (coordination) layer, and deliberative (organization/planning) layer. The layers are organized according to the principle of increasing precision of control in the lower layers with decreasing intelligence [13.14]. A great deal of research has been invested into the design these components and their interactions [13.2, 15–21].

Three-layer architectures aim to harness the best of reactive control in the form of dynamic, concurrent and time-responsive control, and the best of deliberative control, in the form of globally efficient actions over a long time scale. However, there are complex issues involved in interfacing these fundamentally differing components, and the manner in which their functionality should be partitioned is not yet well understood [13.22].

13.1.4 Behavior-Based Control – Think the Way You Act

Behavior-based control employs a set of distributed, interacting modules, called *behaviors*, that collectively achieve the desired system-level behavior. To an external observer, behaviors are patterns of the robot's activity emerging from interactions between the robot and its environment. To a programmer, behaviors are control modules that cluster sets of constraints in order to achieve and maintain a goal [13.22, 23]. Each behavior receives inputs from sensors and/or other behaviors in the system, and provides outputs to the robot's actuators or to other behaviors. Thus, a behavior-based controller is a structured network of interacting behaviors, with no centralized world representation or focus of control. Instead, individual behaviors and networks of behaviors maintain any state information and models.

Well-designed behavior-based systems take advantage of the dynamics of interaction among the behaviors themselves, and between the behaviors and the environment. The functionality of behavior-based systems can be said to emerge from those interactions and is thus neither a property of the robot or the environment in isolation, but rather a result of the interplay between them [13.22]. Unlike reactive control, which utilizes collections of reactive rules with little if any state and no representation, behavior-based control utilizes collections of behaviors, which have no such constraints: behaviors do have state and can be used to construct representations, thereby enabling reasoning, planning, and learning.

13.1.5 When to Use What

Characterizing a given robot computational architecture based on these four classes of control is often a matter of degree, as architectures attempt to combine the advantages of these paradigms, especially the responsiveness, robustness, and flexibility of the behavior-based approach with the use of abstract representational knowledge for reasoning and planning about the world [13.22] or for managing multiple

conflicting goals. For example, AuRA (autonomous robot architecture) uses a planner to select behaviors [13.22] and 3T uses behaviors in the execution layer of a three-level hierarchical architecture [13.24]; both of these architectures dynamically reconfigure behaviors according to reasoning based on available world knowledge [13.22]. *Goller* et al. [13.25] use a set of 15 navigational behaviors with 17 activating behaviors, combined with a task planner and a topological navigation module.

Each of the above approaches to robot control has its strengths and weaknesses, and all play important and successful roles in certain robot control problems and applications. Each offers interesting but different insights, and no single approach should be seen as ideal or otherwise in the absolute. Rather, the choice of robot control methodology should be based on the particular task, environment, and robot. Robot control presents fundamental tradeoffs having to do with time scale of response, system organization, and modularity: thinking allows looking ahead to avoid mistakes, but only as long as sufficient, accurate, up-to-date information is available, otherwise reacting may be the best way to handle the world. As a consequence of these inherent tradeoffs, it is important to have different methodologies at our disposal rather than having to fit all controller needs into a single methodology. Selecting an appropriate control methodology and designing an architecture within it is best determined by the situatedness properties of the problem, the nature of the task, the level of efficiency or optimality needed, and the capabilities of the robot, both in terms of hardware, world modeling, and computation.

For example, reactive control is the best choice for environments demanding immediate response, but such speed of reaction comes at the price of being myopic, not looking into the past or the future. Reactive systems are also a popular choice in highly stochastic environments, and environments that can be properly characterized so as to be encoded in a reactive input–output mapping. Deliberative systems, on the other hand, are the only choice for domains that require a great deal of strategy and optimization, and in turn search and planning. Such domains, however, are not typical of situated robotics, but more so of scheduling, game playing, and system configuration, among others. Hybrid systems are well suited for environments and tasks where internal models and planning are needed, and the real-time demands are few, or sufficiently independent of the higher-level reasoning.

Behavior-based systems, in contrast, are best suited for environments with significant dynamic changes, where fast response and adaptivity are crucial, but the ability to do some looking ahead and avoid past mistakes is required. Those capabilities are spread over the active behaviors, using representations if necessary [13.23], as discussed in the following.

13.2 Basic Principles of Behavior-Based Systems

Behavior-based robotics was developed for situated robots, allowing them to adapt to the dynamics of real-world environments without operating upon abstract representations of reality [13.11], but also giving them more computational capability and expressivity than are available to reactive robots. Behavior-based systems maintain a tight coupling of sensing and action through behaviors, and use the behavior structure for representation and learning. Therefore, it is uncommon for a behavior to perform extensive computation or reasoning relying on a traditional world model, unless such computation can be done in a timely manner in response to dynamic and fast-changing environment and task demands.

Figure 13.1 summarizes the general components of low-level behavior-based systems. Note that there is a distinction between *activation conditions*, which allow the behavior to generate actions, and *stimuli*, from which actions are generated.

The basic principles of behavior-based control can be summarized briefly as follows:

- Behaviors are implemented as control laws (sometimes similar to those used in control theory), either in software or hardware, as a processing element or as a procedure.
- Each behavior can take inputs from the robot's sensors (e.g., proximity sensors, range detectors, contact sensors, camera) and/or from other modules in the system, and can send outputs to the robot's effectors (e.g., wheels, grippers, arm, speech) and/or to other modules.
- Many different behaviors may independently receive input from the same sensors and output action commands to the same actuators.
- Behaviors are encoded to be relatively simple, and are added to the system incrementally.
- Behaviors (or subsets thereof) are executed concurrently, not sequentially, in order to exploit par-

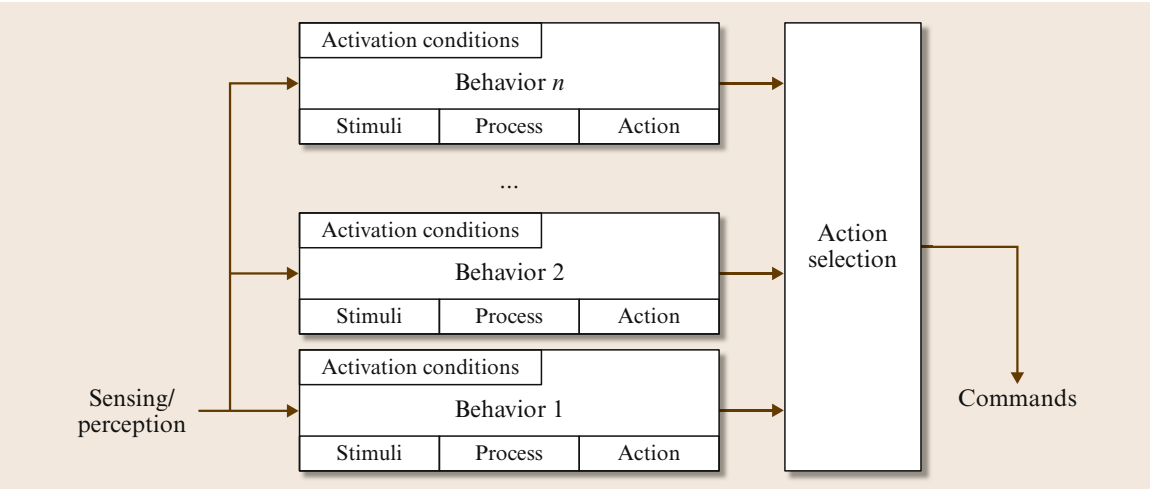


Fig. 13.1 A general schematic of one type of behavior-based systems

allelism and speed of computation, as well as the interaction dynamics among behaviors and between behaviors and the environment.

Behaviors are designed at a variety of abstraction levels, facilitating bottom-up construction of behavior-based systems. New behaviors are introduced into the system incrementally, from the simple to the more complex, until their interaction results in the desired overall capabilities of the robot. In general, behaviors encode time-extended processes, not atomic actions that are typical of feedback control (e.g., *go-forward-by-a-small-increment* or *turn-by-a-small-angle*). The interaction and integration of temporal and spatial effects are of key importance in behavior-based systems. Merely having one process controlling an actuator for predetermined intervals of time, or using as many processes as there are effectors to control them, does not suffice as the basis for behavior-based control. It is the combined effect of concurrent processes over time and driven by perception and internal states that creates the relevant behavior-based dynamics in a control system.

As a first step, survival behaviors, such as *collision-avoidance*, are implemented. These behaviors are often reactive in nature, since reactive rules can and often do form components of simple behaviors. Next, behaviors are added that provide more complex capabilities, such as *wall-following*, *target-chasing*, *homing*, *find-object*, *get-recharged*, *avoid-the-light*, *aggregate-with-group*, *pick-up-object*, *find-landmark*. Depending on the system being designed, behaviors implementing distributed representations may be added, as may be behaviors capable of learning about the world and/or the robot itself, and operating on those representations and learned information. Representation and learning are addressed in more detail in Sect. 13.4.

Behavior-based systems are typically designed so the effects of the behaviors interact largely in the environment rather than internally through the system, taking advantage of the richness of the interaction dynamics by exploiting the properties of situatedness. These dynamics are sometimes called *emergent* behaviors because they *emerge* from the interactions and are not internally specified by the robot's program. Therefore, the internal behavior structure of a behavior-based system need not necessarily mirror its externally manifested behavior. For example, a robot that flocks with other robots may not have a specific *flocking* behavior; instead, its interaction with the environment and other robots may result in flocking, although its only behaviors may be *avoid-collisions*, *stay-close-to-the-group*, and *keep-going*.

For such an approach to work, a behavior-based system must resolve the issue of choosing a particular action or behavior from multiple options, a process known as *action selection* [13.26] (or also *behavior coordination* [13.27], behavior selection [13.28], or behavior fusion [13.29]) problem. This is one of the central design challenges of behavior-based systems. One approach to action selection is the use of a predefined behavior hierarchy, in which commands from the highest-ranking active behavior are sent to the actuator and all others are ignored. Numerous approaches based on other principles as well as ad hoc methods for addressing the action selection problem have been developed and demonstrated on robotic systems. These methods aim to provide increased flexibility but, in some cases, may do so at the cost of reducing the ef-

ficiency or the analyzability of the resulting control systems. Developed methods include varieties of motor schemas [13.16], command fusion [13.30], spreading of activation through a behavior network [13.31, 32], behavior auctions [13.33, 34], and fuzzy logic [13.35, 36], among many others. *Pirjanian* [13.27] presents a survey of action selection mechanisms. Some frameworks also support the use of several action selection mechanisms, such as APOC [13.28] (allowing dynamic selection and changes) and iB2C [13.29] (along with development guidelines, analysis tools and visualization techniques).

13.2.1 Misconceptions

Because behavior-based systems are not always simple to describe or implement, they are also often misunderstood. The most common misconception equates reactive and behavior-based systems. Historically, the advent of behavior-based systems was inspired by reactive systems; both maintain real-time couplings between sensing and action [13.18, 37], and are structured and developed bottom-up, consisting of distributed modules. However, behavior-based systems are fundamentally *more powerful* than reactive systems, because they can store representations [13.38], while reactive systems cannot. Reactive systems are limited by their lack of internal state; they are incapable of using internal representations and learning. Behavior-based systems overcome this limitation because their underlying unit of representation, the *behavior*, can store state internally, in a distributed fashion.

The means by which state and representation are distributed in behavior-based systems is one of the sources of the flexibility of the control methodology. Representations in behavior-based systems are distributed, so as to best match and utilize the underlying behavior structure that causes the robot to act. This is how *thinking* can be organized in much the same way as *acting*. Thus if a robot needs to plan ahead, it does so in a network of communicating behaviors, rather than a single centralized planner. If a robot needs to store a large map, the map might be distributed over multiple behavior modules representing its components, such as a network of landmarks, as in [13.39], or a network of parameterized navigation behaviors, as in [13.40, 41], so that reasoning about the map/environment/task can be done in an active fashion, through using message passing within the behavior network. The planning and reasoning components of the behavior-based system use the same mechanisms as the sensing- and action-oriented behaviors, and as a result do not operate on a fundamentally different time scale and representation relative to one another. Various forms of distributed representations are used, ranging from static table structures and networks, to active procedural processes implemented within the behavior networks.

Another area of common misconception relates to the comparison between behavior-based systems and hybrid systems. Because the two use such different modularization strategies, it is often assumed that one approach (usually hybrid) has improved expressive capabilities. In fact, behavior-based and hybrid systems have the same expressive and computational capabilities: both can exploit representations and look ahead, but they do so in very different ways. This has resulted in different application domains being best suited to behavior-based versus hybrid systems. Specifically, hybrid systems dominate the domain of single-robot control, unless the task is so time-demanding that a reactive system must be used. Behavior-based systems dominate the domain of multi-robot control because the notion of collections of behaviors within the system scales well to collections of such robots, resulting in robust, adaptive group behavior [13.42, 43]. See Chap. 53 on multiple mobile robot systems for more details.

Like hybrid systems, behavior-based systems may be organized in layers, but unlike hybrid systems, the layers do not differ from each other drastically in terms of organization, time scale or representation used. Behavior-based systems typically do not employ the hierarchical/sequential division favored by hybrid approaches. Behavior-based systems do provide both low-level control and high-level deliberation; the latter can be performed by one or more distributed representations that compute(s) over the other behaviors or modules, often directly utilizing low-level behaviors and their outputs. The resulting systems, built from the bottom-up, do not divide into differently represented and independent components, and consist of elements directly tied in some ways to behaviors. The power, elegance, and complexity of behavior-based systems all stem from the ways in which their constituent behaviors are designed, coordinated, and used. For instance, monitoring how behaviors for low-level control contribute to the commands issued over time can be used to evaluate how appropriate the robot operates in its environment based on its set of activated behaviors. Different methods can be used to implement such a mechanism, such as temporal analyses [13.36, 44–47], pattern recognition techniques [13.48] and graph-based representations [13.49–51].

Therefore, the principles to keep in mind when extending behavior-based to higher decisional levels are:

1. Use behaviors as the building block of both decision-making and action execution processes;
2. Use distributed parallel evaluation and concurrent control over lower-level behaviors, which take real-time inputs from sensory data and send real-time commands to effectors;
3. Have no centralized components, each module carrying out its own responsibilities.

The following sections describe and illustrate in more detail how behavior-based principles can be used to control robots.

13.3 Basis Behaviors

The process of designing a set of behaviors for a robot is referred to as *behavior synthesis*, and is typically performed by hand, although some methods for automated synthesis behaviors have been developed and successfully demonstrated [13.52, 53]. In all cases, behaviors perform a particular activity, attain a goal, or maintain some state. The notion of defining an *optimal* behavior set for a given robot or task has been considered, but it is generally accepted that such a notion is not realistic as it is dependent on too many specifics of a given system and environment that cannot currently be effectively formalized.

Matarić et al. [13.43, 54] describe *basis behaviors*, also referred to as *primitive behaviors*, as a tool for structuring and thus simplifying behavior synthesis. Basis behaviors are a set of behaviors such that each is *necessary*, in the sense that each either achieves, or helps to achieve, a relevant goal that cannot be achieved without it by other members of that set. Furthermore, the basis behavior set is *sufficient* for achieving the goals mandated for the controller. The term *basis* was chosen to be indicative of the similar notion within linear algebra. The property of necessity or parsimony is analogous to the idea of linear independence; the idea of sufficiency is similar to the linear algebraic concept of span. Basis behaviors should be simple, stable, robust, and scalable.

Another organizational principle of basis behaviors is *orthogonality*. Two behaviors are orthogonal if they do not interfere with one another, each inducing no side-effects in the other. This is often achieved by having behaviors take mutually exclusive sensory inputs. Another method is to have different behaviors control separate effectors. This form of factorization is only feasible when the robot's dynamics do not inhibit their separability. In contrast, autonomous helicopter control is an example of a highly coupled system; *Saripalli* et al. [13.55] demonstrated how behavior-based control can be effectively applied to robust autonomous helicopter flight.

Basis behavior design principles have been applied to single-robot as well as multi-robot behavior-based systems in a variety of applications, ranging from navigation to foraging, coordinated group movement, box pushing, and others.

13.4 Representation in Behavior-Based Systems

Embedding representation into behavior-based systems involves the challenge of conserving the basic principles of the approach at all levels of system decision-making. Combining abstract reasoning processes with behaviors must be done in a way that exploits interaction dynamics and desirable emergent system properties.

Matarić et al. [13.38, 56] describe work with a robot named *Toto* and as shown by VIDEO 35, which introduced the use of distributed representation into behavior-based systems. Toto's capabilities included safe navigation, landmark detection, map learning, and path planning in the learned map representation, all within the behavior-based framework. To exploit the principles underlying behavior-based control, Toto's representation was not a centralized map. Instead, any newly discovered landmark in the environment was assigned to a new map representation behavior, which stored the landmark descriptor (type, estimated Cartesian location, and compass orientation). Whenever sensory inputs matched the landmark descriptor, the robot localized to that particular landmark and the behavior became active. The following is pseudo-code for each landmark behavior:

Algorithm 13.1

```
my-behavior-type: C
my-compass-direction: 0
my-approximate-location: (x,y)
my-approximate-length: 6.5
whenever received (input)
```

```
if input(behavior-type) = my-behavior-type
   and
      input(compass-direction) =
      my-compass-direction then
   active ¡- true
end if
```

As new landmarks are discovered, they are added to the map representation behavior network. In this way, the topology of the resulting map network is isomorphic to the topology of the network graph in the physical environment Toto has explored. The edges in the network graph are also communication links in the behavior network, allowing landmark behaviors to communicate through local message passing. Consequently, a currently active map behavior can send a message to its topological neighbor(s), thereby indicating expectation of it being the next recognized landmark and facilitating Toto's localization. Planning in the network takes place through the use of the same message-passing mechanism. The goal landmark (which could be selected by the user as part of the task, such as *go to this particular corridor* or *go to the nearest north-facing wall*), sends messages (i. e., spreads activation) from itself to its neighbors, which pass it on throughout the network. As messages are passed, the length of each landmark in the graph is accrued, thereby estimating the length of each path. The shortest path arriving at the currently active network behavior indicates the best direction to pursue toward the goal. This is equivalent to a distributed Dijkstra search. Importantly, because this search is an active ongoing process within a behavior map rather than a static process (as it would be in a centralized map representation), if the robot is picked up and moved to another location, as soon as it localizes, it continues on the optimal path to the goal: each landmark makes a local decision as to where to go next toward the goal, and no unique global path is stored in any central location/representation. Thus, the path is constantly refreshed and updated; if any route is blocked, the link in the graph is disconnected and the shortest path is updated dynamically.

Toto exemplifies how, in a behavior-based system, a representation can be stored in a distributed fashion, so as to best match the underlying behavior structure that produces the robot's external goal-driven activity. If Toto needs to make high-level decisions (such as planning ahead to a distant goal), it does so in a network of communicating behaviors, rather than a single centralized component. This results in scalable and efficient computation for the system as a whole, since the typically slower decision-making processes such as planning are distributed and modularized in a way that makes them more consistent with the time scale and representation of the rest of the system. Note the fundamental difference between this general attempt of behavior-based systems to homogenize their representation through the use of behaviors as universal modules, compared to hybrid systems which rely on inherently different representations and time scales at different levels of the system.

13.5 Learning in Behavior-Based Systems

The ability to improve performance over time and to reason about the world, in the context of a changing and dynamic environment, are important areas of research in situated robotics. Unlike in classical machine learning where the goal is typically to optimize performance over a long period of time, in situated learning the aim is to adapt relatively quickly toward attaining efficiency in the light of uncertainty. Models from biology are often considered, given its properties of learning directly from environmental feedback. Variations and adaptations of machine learning, and in particular reinforcement learning, have been effectively applied to behavior-based robots, which have demonstrated learning to walk [13.57], communicate [13.58], navigate and create topological maps [13.38, 59], divide tasks [13.23, 60], behave socially [13.61], and even identify opponents and score goals in robot soccer [13.62]. Methods from artificial life, evolutionary computation/genetic algorithms, fuzzy logic, vision and learning, multi-agent systems, and many other research areas continue to be actively explored and applied to behavior-based robots as their role in animal modeling and practical applications continue to develop.

When operating in unpredictable and partially observable environments, an autonomous robot must examine the evolution of its general states, and try to capture what emerges from the interaction dynamics with its environment. Temporal integration of different types of observations is an important aspect of that capability [13.63, 64]. Work on motivational systems [13.47, 65–68] has shown that a balance between planning and reactivity for goal management can be achieved using internal variables activated or inhibited by different factors [13.42, 46, 69, 70]. Motivations can be cyclic (e.g., circadian rhythms) or change in various temporally dependent ways [13.68]. In general, the

notion of motivations is used to efficiently balance the need to adapt to the contingencies of the world and to accomplish the robot's goals.

In the following subsections, we discuss four successfully validated classes of learning approaches in behavior-based systems. The approaches differ in what is learned and where learning algorithms are applied, but in all cases behaviors are used as the underlying building blocks for the learning process.

13.5.1 Reinforcement Learning in Behavior-Based Systems

Behaviors are recognized as excellent substrates for speeding up reinforcement learning (RL), which is known to suffer from the curse of dimensionality. The earliest examples of RL in behavior-based systems demonstrated hexapod walking [13.57] and box-pushing [13.71]. Both decomposed the control system into a small set of behaviors, and used generalized input states, thereby effectively reducing the size of the state space. In the box-pushing example, the learning problem was broken up into modularized policies for learning separate mutually exclusive behaviors: for getting out when stuck, for finding the box when lost and not stuck, and for pushing the box when in contact with one and not stuck. The modularization into behaviors resulted in greatly accelerated as well as more robust learning. More recently, *Kober* et al. [13.72] demonstrated the use of reinforcement learning to learn a set of meta-parameters that enable the adaptation of learned motor plans, such as a table tennis pass or a dart throwing task, to new situations.

Scaling up reinforcement learning to multi-robot behavior-based systems has also been explored [13.23, 73]. In multi-robot systems, the environment presents further challenges of nonstationarity and credit assignment, due to the presence of other agents and concurrent learners. The problem was studied in the context of a foraging task with four robots, each initially equipped with a small set of basis behaviors (*searching*, *homing*, *picking-up*, *dropping*, *following*, *avoiding*) and learning individual behavior selection policies, i. e., which behavior to execute under which conditions. Due to interference among concurrent learners, this problem could not be solved directly by standard RL. *Shaping*, a concept from psychology [13.74], was introduced; it was subsequently adopted in robot RL [13.75]. Shaping pushes the reward closer to the subgoals of the behavior, and thus encourages the learner to incrementally improve its behaviors by searching the behavior space more effectively. *Matarić* [13.73] introduced shaping through *progress estimators*, measures of progress toward the goal of a given behavior during its execution. This form of reward shaping addresses two issues associated with delayed reward: behavior termination and fortuitous reward. Behavior termination is event-driven; the duration of any given behavior is determined by the interaction dynamics with the environment, and can vary greatly. Progress estimators provide a principled means for deciding when a behavior may be terminated even if its goal is not reached and an externally generated event has not occurred. Fortuitous reward refers to reward ascribed to a particular situation–behavior (or state–action) pair which is actually a result of previous behaviors/actions. It manifests as follows: previous behaviors lead the system near the goal, but some event induced a behavior switch, and subsequent achievement of the goal is ascribed most strongly to the final behavior, rather than the previous ones. Shaped reward in the form of progress estimators effectively eliminates this effect. Because it provides feedback during behavior execution, it rewards the previous beneficial behaviors more strongly than the final one, thus more appropriately assigning credit. This case therefore illustrates how reinforcement learning has been successfully applied to behavior-based robotics, in particular at the level of behavior selection. The learning process is accelerated by the behavior structure, which provides a higher-level representation of actions and time-extended dynamics.

13.5.2 Learning Behavior Networks

The modularization of behavior-based systems as networks of behaviors allows for learning to be applied at the network level as well. *Nicolescu* and *Matarić* [13.40, 41] developed the notion of *abstract behaviors*, which separate the activation conditions of a behavior from its output actions (so-called *primitive behaviors*, which share the same principles as basis behaviors described Sect. 13.3); this allows for a more general set of activation conditions to be associated with the primitive behaviors. While this is not necessary for any single task, it is what provides generality to the representation. An abstract behavior is a pairing of a given behavior's activation conditions (i. e., preconditions) and its effects (i. e., postconditions); the result is an abstract and general operator much like those used in classical deliberative systems (Fig. 13.2). Primitive behaviors, which typically consist of a small basis set, may involve one or an entire collection of sequential or concurrently executing behaviors.

Networks of such behaviors are then used to specify strategies or general *plans* in a way that merges the advantages of both abstract representations and behavior-based systems. The nodes in the networks are abstract behaviors, and the links between them repre-

sent precondition and postcondition dependencies. The task plan or strategy is represented as a network of such behaviors. As in any behavior-based system, when the conditions of a behavior are met, the behavior is activated. Analogously, when the conditions of an abstract behavior are met, the behavior activates one or more primitive behaviors which achieve the effects specified in its postconditions. The network topology at the abstract behavior level encodes any task-specific behavior sequences, freeing up the primitive behaviors to be reused for a variety of tasks. Thus, since abstract behavior networks are computationally lightweight, solutions for multiple tasks can be encoded within a single system and dynamically switched.

Nicolescu and *Matarić* [13.40, 41] introduced a means for automatically generating such networks offline as well as at runtime. The latter enables a learning robot to acquire task descriptions dynamically, while observing its environment, which can include other robots and/or a teacher. The methodology was validated on a mobile robot following a human and acquiring a representation of the human-demonstrated task by observing the activation of its own abstract behavior pre- and postconditions, thus resulting in a new abstract behavior network representing the demonstrated task [13.76]. The robot was able to acquire novel behavior sequences and combinations (i. e., concurrently executing behaviors), resulting in successful learning of tasks involving visiting various targets in particular order as shown in VIDEO 27 and VIDEO 28, picking up, transporting, and delivering objects as shown in VIDEO 32 and VIDEO 33, dealing with barriers, and maneuvering obstacle courses in specific ways as shown in VIDEO 30 and VIDEO 31.

This approach has been extended to learning of more general task representations, in which every goal is achieved through a linear superposition (or fusion) of robot primitives, and sequencing across goals is achieved through arbitration [13.77]. In this work, controllers are built from two components: *behavior primitives* (BPs) and *fusion primitives* (FPs), which can be sequenced to form *behavior networks*. The *behavior primitives* perform a set of actions under given environmental conditions and are represented using a schema-based approach [13.78]. A *fusion primitive* encapsulates a set of multiple concurrently running primitive behaviors through linear combination of the motor commands, expressed as a vector in the robot's coordinate system. The FPs combine these vectors by weighting them and fusing them through vector addition. The method allows for learning of both the goals involved in the task (represented as the sequence of goal-achieving fusion primitives) and also of the particular ways in which the same goals can be achieved (captured in the behavior weights). Using this learning approach, a robot is capable of learning multiple navigation styles (such as staying on the center/left/right side of a corridor, preferences for turning left or right at T-junctions and particular styles of navigating around obstacles).

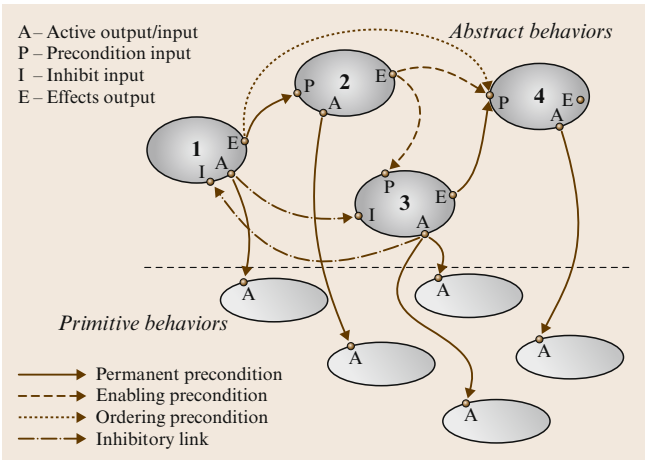


Fig. 13.2 Behavior network

13.5.3 Learning from Demonstration in Behavior-Based Systems

Learning by demonstration has also been combined with behavior-based systems to produce one-shot learning and teaching mechanisms for robot control [13.40, 79]. Furthermore, this type of behavior-based architecture has been used to learn ship navigation strategies [13.80]; during a learning phase, an instructor selects behaviors for a ship to execute in order to reach a specific goal, and a subsequent offline stage then generates dependency links between the behaviors that it witnessed during the learning phase.

Another form of learning by demonstration has used probabilistic methods to select and combine multiple behaviors [13.81]. There, the problem of learning what behaviors to execute during autonomous navigation is treated as a state estimation problem. During a learning phase, the robot observes commands used by a teacher. A particle filter then fuses the control commands that were demonstrated by the teacher to estimate behavior activation. The method produces a robust controller that is well suited for dynamic environments.

Learning by demonstration has also been used in a case-base reasoning framework [13.82], to teach a 4-legged Aibo robot different low-level navigation behaviors, such as following a ball and reaching a target. The learned behaviors encapsulate the kinematics and dynamics of the robot's control through the instruction process, which would be more difficult to achieve if de-

signed by hand, and can successfully be combined into a newer, higher-level behavior, thus enabling hierarchical representations of robot tasks.

A method based on decision trees and Support Vector Machines has been used in [13.83] to find mappings between sensory data and patterns produced by behaviors in a learned feature space. The mapping allows a robot to identify behaviors shown by a human demonstrator and to further learn a sequential representation of the demonstrated task.

13.5.4 Learning from History of Behavior Use

Most deliberative approaches derive knowledge for reasoning about the world from sensor inputs and actions taken by the robot. This results in complex state-space representations of the world and does not take into consideration the context in which these sensations/actions are taken. As already discussed, behaviors, which are readily used as low-level control blocks driven by what is experienced from the interactions with the environment, can also serve as an abstract representation to model those interactions. One approach is to use history information [13.84], i. e., to explicitly take into consideration the time sequence of observations in order to make a decision. Applying this idea to behavior-based systems, by knowing the purpose of each of the behaviors and by observing their history of use, the robot can reason and ground its intentions in what it is experiencing in its operating environment. The concept of abstract behavior is exploited here as well, to activate behaviors and as a representation on which to learn.

Learning from history has been validated in behavior-based systems capable of making the robot change its behavior selection strategy for foraging colored objects (blocks) to a homing region in nonstationary, dynamically changing environments involving multiple concurrently learning robots [13.49–51]. In that foraging task, the robot is given two subtasks: search for a block (searching task), and bring it to the home region (homing task). The robot is given behaviors to accomplish these tasks: one behavior for the searching task, called *searching-block*, and two for the homing task, *homing* and *drop-block*. A *velocity control* behavior is also used in both of these tasks to make the robot move. All these behaviors are referred to as *task behaviors*. Conditions for activating task behaviors are preprogrammed based on the presence or absence of a block in front of the robot and the proximity of the home region.

The robot also needs to navigate safely in the environment. In this approach, an *avoidance* behavior is activated unless the robot is near home and carrying a block; otherwise it is disabled to allow the robot to approach the home region. This type of behavior, used for handling harmful situations and interference while accomplishing a task, is referred to as a *maintenance behavior*. The designer determines the conditions in which maintenance behaviors should be used, but cannot indicate when they will occur during the task, as that is tied to the dynamics of the interaction between the robot and its environment.

The robot learns to use *alternative behaviors* (*follow-side*, *rest*, and *turn-randomly*), which introduce variety into its action repertoire by changing the way it accomplishes its tasks. In contrast to other types of behaviors, these have no a priori activation conditions; the objective is to allow the robot to learn when to activate these behaviors according to past experiences, for some preset periods of time, when it is experiencing interference in accomplishing its task. Figure 13.3 illustrates how the behaviors are prioritized using a fixed suppression mechanism, similar to the subsumption architecture [13.9] but with the difference that the activated behaviors, i. e., those allowed to issue outputs, change dynamically.

Following behavior-based principles, an activated behavior may or may not be used to control the robot,

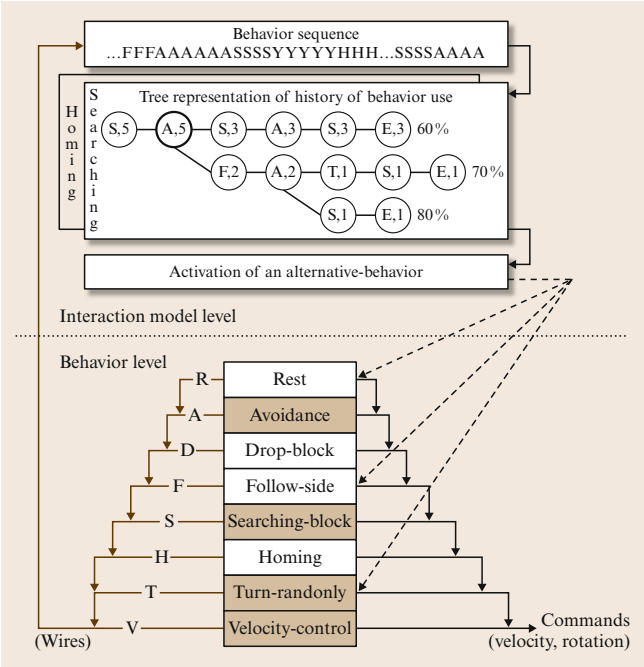


Fig. 13.3 Organization of the behavior level and the interaction model level. Behavior on *colored background* represent an example of activated behaviors for the searching task, with *Turn-randomly* as a chosen alternative behavior. For clarity, sensory inputs to behaviors are not shown

depending on the sensory conditions it monitors and the arbitration mechanism. An activated behavior is used only when it provides commands that actually control the robot. Whenever a behavior is used, its corresponding symbol is sent to the interaction model, generating the sequence of behaviors used over time. Separate learning trees are used for each task; determining which one to use is done based on the activated task behavior.

The algorithm uses a tree structure to store the history of behavior use. The upper part of Fig. 13.3 shows a typical tree with nodes storing the behavior (H for use by *homing* and *drop-block*) used for controlling the robot while accomplishing its task, and n, the number of times a transition between the node itself and its successor has been made (as observed from behavior use). Initially, the tree for a particular task is empty and is incrementally constructed as the robot goes about its task. Leaf nodes are labeled with E (for end-node) and store the total *performance* of the particular tree path. Whenever a path is completely reused, when the same sequence of behaviors is used, the E node is updated with the average of the stored and current performances for recent trials.

Learning is done through reinforcement. Depending on the domains and the tasks, a variety of factors can be used to evaluate performance, and different metrics can be used with this learning algorithm. To see how far the idea of self-evaluation and learning by observing behavior use can be taken, the evaluation function used here is not based on characteristics about the environment or the task. Instead, it is based on the amount of time behaviors are used. Comparison between the time spent using behaviors associated with the tasks and the time spent exploiting maintenance behaviors is used to derive the evaluation criterion. Consequently, behavioral selection strategy is derived from what can be learned from the experiences of the robot in its environment, without having to characterize a priori what can be considered to be optimal operating conditions in the environment.

Using the tree and the evaluation function, the algorithm has two options for using a maintenance behavior:

1. Not to make any changes in its active behavior set (the *observe* option);
2. To activate an alternative behavior.

The sequence of nodes in a tree path characterizes the interactions experienced by the robot in the world. Different selection criteria can be used by comparing the performance at the current position in the tree with the expected performance, following an exploration (to learn the effects of alternative behaviors) then exploitation (to exploit what was learned in previous trials) strategy. The expected performance of a given node is the sum of the stored end-node performances in its subpaths, multiplied by the frequency of use of the subpaths relative to the current position in the tree. Finally, since this algorithm is used in noisy and nonstationary conditions, deleting paths is necessary to keep the interaction model up to date. This is done by keeping only a fixed number of the most recent paths in the tree.

Results obtained with this approach show that the robot is able to learn unanticipated strategies (like resting in front of a static obstacle to increase the turning angle when it starts to move again, making it locate a target) and original ones (like yielding when close to other robots or following walls when the center of the pen is crowded). Developing such capabilities is important in general, because it makes it possible for robots to learn in nonstationary environments, which are common in natural settings.

13.6 Applications and Continuing Work

Behavior-based robots have demonstrated various standard capabilities, including obstacle avoidance, navigation, terrain mapping, following, chasing/pursuit, object manipulation, task division and cooperation, and learning maps [13.85], navigation and walking. They have also demonstrated some novel applications like large-scale group behaviors including flocking, foraging, and soccer playing, human–robot interaction, and modeling insect [13.86] and even human behavior [13.87–89]. Behavior-based methods span mobile robots, shopping carts [13.25], underwater vehicles, planetary rovers for space exploration, assistive [13.90], interactive and social robots [13.91, 92], as well as snake-like modular robots [13.93], robots capable of grasping, manipulation, walking, running, and many others. Consumer market products, such as the iRobot Roomba Robotic Floorvac [13.94], also use behavior-based control, demonstrating its broad applicability.

The use of behavior-based architectures for robot control has evolved from single-system implementations to approaches that combine forms of learning, state estimation, and distributed computation. Behaviors have been combined with fuzzy inference systems for indoor navigation using mobile robots [13.35, 36, 95–97], where a command fusion module acts as an arbiter that combines multiple fuzzy behavior outputs

into a single control signal. This strategy ensures the robot is capable of making inferences in the face of uncertainty.

Behavior-based methods have been employed in multi-robot systems from the outset [13.43], as illustrated by VIDEO 34, and are still used [13.98, 99]. Multi-robot researchers also consider tasks requiring tightly coupled cooperation; [13.100] for an overview and discussion. Such tasks typically require low-level sensor-sharing and/or for higher-level explicit coordination. Behavior-based controllers have been developed and extended in order to address these challenges. For example, *Parker* et al. [13.101] considered reusable behavior units that can be automatically redistributed for low-level information processing. *Werger* and *Matarić* [13.102] described broadcast of local eligibility (BLE) to enable higher-level group behaviors by allowing communications to influence each robot's local action selection mechanism. *Gerkey* and *Matarić* [13.103, 104] demonstrated scalable and efficient market-based coordination algorithms for multi-robot coordination in a variety of tasks, including tightly coupled ones (e.g., box-pushing [13.105]). A similar strategy was applied to a robot sensor network, which achieves concurrent, sequential and asynchronous multi-robot task allocation [13.106]. The market-based auction algorithm developed in this system uses a fitness function that enhances fault tolerance and the fairness of the task allocation process. *Sedat* and *Aydan* [13.107] use a fractal conductivity-based approach to generate movements of a coordinated team of mobile sensors. The team explores an environment to reach a target that is a source of a potential field chemical substance contaminated into sea or lake water.

Several researchers have shown that behavior-based controllers allow for sophisticated coordination through a utility-centered model of the collective task. Behaviors use this representation to produce actions that consider each robot's impact on the performance of the group as a whole. *Iocchi* et al. [13.108] have shown this in heterogeneous multi-robot system, while *Batalin* and *Sukhatme* [13.109] demonstrated that complex, interrelated and dynamic tasks can be performed in coordinated ways in robots with behavior-based controllers interacting with a sensor network. *Stroupe* and *Balch* [13.110] considered a mapping task and developed move value estimation for robot teams (MVERT), essentially a behavior-based method for maximizing group knowledge.

Because behavior-based methods lend itself naturally to multi-robot control problems, they have had a significant influence on that field of research. *Simmons* et al. [13.111] described a hybrid architecture designed for group-level coordination which employs behaviors as a method for organizing low-level safety-critical controller code. Behaviors have also been used to structure controllers and networked communication in minimalist systems [13.112]. Some multi-robot research with a control-theoretic flavor has used separate executable processes that can be switched on and off dynamically depending on task constraints, in a manner very similar to behavior-based control [13.113].

Behavior-based controllers have also been built using genetic algorithms [13.114], in which the controller learned through coevolution five behavioral components (waypoint prediction, force field trajectory, speed regulation, reversing and heading alignment), enabling it to outperform all its opponents in a real time car racing game held during the 2007 IEEE Congress on Evolutionary Computation (CEC).

Behavior-based architectures have also been used in complex vision systems for identification rather than control. In those contexts, each behavior represents a small unit of visual computation, such as frame differencing, motion detection, and edge detection, resulting in biologically inspired vision and attention behavior [13.115, 116].

Behavior-based architectures have also been developed for the control and learning in humanoid robots. In the past, the Cog project demonstrated behavior-based control for articulated manual and eye–hand coordination [13.117, 118]. *Edsinger* [13.119] developed a lightweight behavior architecture for organizing perception and control of Domo. The architecture allows for specification of time-contingent behaviors and distributed computation, resulting in a real-time controller that allows Domo to operate in an environment with humans. Kismet [13.120] involved several behavior-based systems that controlled perception, motivation, attention, behavior, and movement in a human–robot interaction (HRI) context. Each behavior represented Kismet's individual drives and motivations. Situated modules and episode rules were introduced as part of an HRI architecture by *Ishiguro* et al. [13.121] and *Kanda* et al. [13.122]; they employed an ordered set of general-use situated modules and behaviors that are contingent on a condition, used in sequences defined by a set of task-specific episode rules.

13.6.1 Adaptive Behavior Selection – A Case Study

Designing robots capable of safely and effectively interacting with humans in real-world settings is one of the ultimate challenges in robotics. Representation and abstract reasoning are necessary to keep up with and adapt to the inherently complex properties of such environments and tasks.

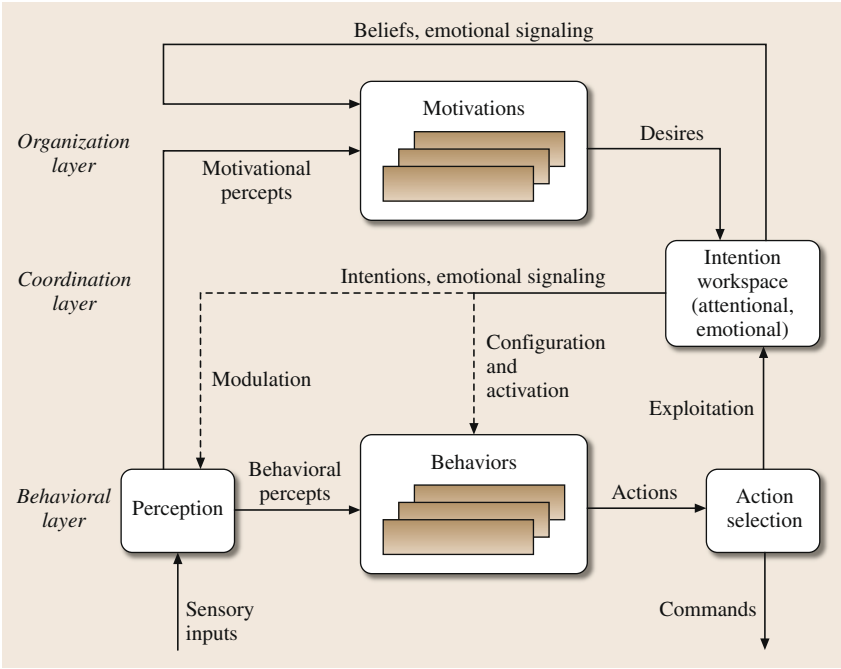


Fig. 13.4 Hybrid Behavior-Based Architecture (HBBA)

Adaptation in behavior-based systems [13.28] generally consists of modifying internal parameters of the behaviors, or modifying the set of activated behaviors. Many possible avenues to implement such mechanisms exist. One possibility consists of adding to a behavior-based architecture the idea of intentionally activating, monitoring, and configuring behaviors. An architectural framework based on this idea has evolved over the years, having been implemented on diverse robotic platforms and tested in various operational conditions:

- EMIB – Emotion, Motivation and Intentional Behaviors – has been applied to robots with different capabilities, such as a robot that uses activation variables, topological localization and mapping, and fuzzy behaviors to explore and characterize an environment [13.45, 46], and on an autonomous rolling robot that only uses simple sensors and a microcontroller to generate purposeful movements for conducting studies regarding on toddler–robot interaction [13.123, 124].
- MBA – Motivated Behavioral Architecture [13.125, 126]). In the American Association for Artificial Intelligence (AAAI) mobile robot competition and as shown in VIDEO 417, MBA was used with the Spartacus robot to demonstrate how a behavior-based system could integrate planning and sequencing tasks under temporal constraints, interleaving planning and execution [13.127] (as in [13.128] and [13.129]), and with spatial localization capabilities using a previously generated metric map. The system also used behavioral message reading [13.130], sound processing capabilities with an eight-microphone system for source localization, tracking, real-time sound separation [13.131, 132], and a touch screen interface to allow the robot to acquire information about where it is in the world, what it should do, and how it should do it [13.125].

HBBA – Hybrid Behavior-Based Architecture [13.133] – illustrated by Fig. 13.4, is the current instantiation of the architectural framework. The Behavioral Layer consists of behaviors that are activated and configured according to what are referred to as the Intentions of the system (derived by the Motivation Layer). Intentions are associated with the activation and configuration of behaviors and with the modulation of perceptual modules. The highest layer uses Motivations acting as distributed processes (just like Behaviors) and manifesting Desires for the satisfaction or inhibition of Intentions. The Intention Workspace processes these Desires to issue Intentions. The Intention Workspace can also provide Emotional Signaling (detecting situations based on models of how Behaviors are exploited over time [13.44, 134–136]), and generates Beliefs (i. e., knowledge about the interaction dynamics of the robot and the environment) which can be used to identify conflicts between Desires and their satisfaction. The objective

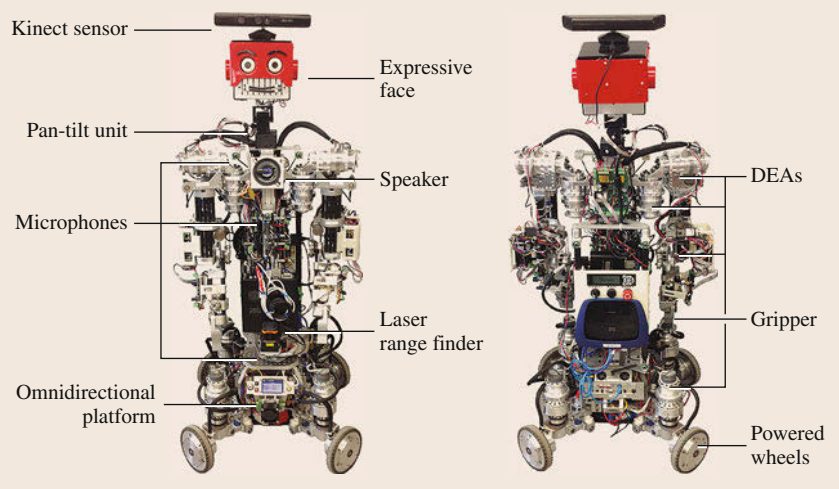


Fig. 13.5 Photographs of the front and back of the robot, IRL-1 (30 DOF and 55 sensors). DEAs stand for Differential Elastic Actuators (after [13.133])

is to have Intentions emerge from the interaction of independent motivations. Desires are communicated to the Intention Workspace, which plays a role similar to a coordination layer in a hybrid architecture, or to action selection with motivations acting as behaviors and Intentions emerging from the interaction of independent motivations. Our design requirements for the Intention Workspace module is to implement generic mechanisms (i. e., mechanisms that are independent of the robot's physical capabilities and intended usage) to generate Beliefs (i. e., data structures that represent desires, knowledge about the interaction dynamics of the robot and the environment), and identify conflicts between Desires and their satisfaction. The Exploitation link is used to derive information about the effective use of Behaviors as influenced by the events occurring in the world and the Action Selection mechanism. We believe that observing the exploitation of behaviors over time is an important source of information about the emerging functionality that comes from the Behaviors and the Motivations, because it combines both a representation of the environment (from the behavioral percepts associated with Behaviors) and of the control policy (from the internal decision-making processes).

HBBA is currently being used with IRL-1 [13.137, 138], an interactive omnidirectional platform shown in Fig. 13.5, VIDEO 418 and VIDEO 419.

IRL-1 has behaviors for safe navigation, recharge and social interaction (with gestures, voice and vision). Action selection is priority-based. Motivations are implemented to provide the intrinsic modes of operation of the robot (safe navigation while exploring the world, interact with people and ensuring energetic autonomy), to localize and plan paths to desired locations, and eventually to plan and schedule tasks [13.139]. Just like action selection at the behavior-based level, the Intention Workspace plays a key role. More specifically, in HBBA, it involves:

- A tree representation of desires. Each desire can be associated with a particular configuration and activation of one or more behaviors. The tree representation of desires provides a general representation of Desires' interdependencies issued by the independent motivations, from high-level/abstract desires (e.g., *Deliver message*) to primitive/behavior-related desires (e.g., *Avoid obstacles*), allowing for behavior configurations to arise in a distributed fashion based on the capabilities available to the robot. This allows motivations to exchange information asynchronously on how to activate, configure, and monitor behaviors. For instance, one motivation may monitor the robot's energy level to issue a recharging desire, which would be associated with a *Recharge* behavior that would make the robot opportunistically detect and dock to a charging station. Meanwhile, if the robot knows where it is and can determine a path to a nearby charging station, a path planning motivation could add a sub-desire of navigating to this position, using a *Goto* behavior. If a person is detected, a social motivation may provide an alternative desire by engaging into a conversation to ask for guidance toward the nearest charging station. The robot would then have three alternative solutions to fulfill its desire to recharge.
- A representation of the history of behavior exploitations (similar to one described in Sect. 13.5.4), indexed by intentions. For instance, for the recharging desire example, if being guided by people reveals not to be as reliable as using the internal map, a preference could be given to the second option.

- An attention selection mechanism to module computational load, by percepts selection or output complexity reduction. For instance, artificial audition [13.131, 132] can be modulated to detect sounds, localize sound sources or separate sound sources depending on the presence of people surrounding the robot and its intentions to interact. A Simultaneous Localization and Mapping (SLAM) algorithm can change resolution depending on available processing load of onboard computers.
- An emotion-based mechanism to regulate conflicts between desires and intentions. Robot's adaptability in dynamic, continuous and unpredictable environments ultimately depends on the detection of the situations for which their regular decision-making process is not appropriate. Psychologists have identified that one of the functions of human emotion is to highlight these kinds of situations, allowing other cognitive processes to address them. Therefore, an emotional process that detect such situations from the use of temporal models of intentions could be beneficial [13.44]. This process does not rely on a priori knowledge of specific environmental conditions nor of the robot's mission objectives, but only on temporal analysis of how behaviors satisfy the robot's intentions.

13.7 Conclusions and Further Reading

This chapter has described behavior-based control, a methodology for single- and multi-robot control aimed at situated robots operating in unconstrained, challenging, and dynamic conditions in the real world. While inspired by the philosophy of reactive control, behavior-based systems are fundamentally more expressive and powerful, enabling representation, planning, and learning capabilities. Distributed behaviors are used as the underlying building blocks for these capabilities, allowing behavior-based systems to take advantage of dynamic interactions with the environment rather than rely solely on explicit reasoning and planning. As the complexity of robots continues to increase, behavior-based principles and their applications in robot architectures and deployed systems will evolve as well, demonstrating increasingly higher levels of situated intelligence and autonomy.

Interested readers can find more information regarding behavior-based systems in other chapters of this Handbook, as well as in *Brooks* [13.140], *Arkin* [13.22], in artificial intelligence and robotics textbooks [13.141, 142], and in introductory textbooks on mobile robot control [13.143–145].

Video-References

VIDEO 27 Experience-based learning of high-level task representations: Demonstration
available from http://handbookofrobotics.org/view-chapter/13/videodetails/27

VIDEO 28 Experience-based learning of high-level task representations: Reproduction
available from http://handbookofrobotics.org/view-chapter/13/videodetails/28

VIDEO 30 Experience-based learning of high-level task representations: Demonstration (2)
available from http://handbookofrobotics.org/view-chapter/13/videodetails/30

VIDEO 31 Experience-based learning of high-level task representations: Reproduction (2)
available from http://handbookofrobotics.org/view-chapter/13/videodetails/31

VIDEO 32 Experience-based learning of high-level task representations: Demonstration (3)
available from http://handbookofrobotics.org/view-chapter/13/videodetails/32

VIDEO 33 Experience-based learning of high-level task representations: Reproduction (3)
available from http://handbookofrobotics.org/view-chapter/13/videodetails/33

VIDEO 34 The Nerd Herd
available from http://handbookofrobotics.org/view-chapter/13/videodetails/34

VIDEO 35 Toto
available from http://handbookofrobotics.org/view-chapter/13/videodetails/35

VIDEO 417 SpartacUS
available from http://handbookofrobotics.org/view-chapter/13/videodetails/417

VIDEO 418 Natural interaction design of a humanoid robot
available from http://handbookofrobotics.org/view-chapter/13/videodetails/418

VIDEO 419 Using ROS4iOS
available from http://handbookofrobotics.org/view-chapter/13/videodetails/419

References

13.1 J.S. Albus: Outline for a theory of intelligence, IEEE Trans. Syst. Man Cybern. **21**(3), 473–509 (1991)

13.2 G. Girald, R. Chatila, M. Vaisset: An integrated navigation and motion control system for autonomous multisensory mobile robots, Proc. 1st Int. Symp. Robotics Res. (1983)

13.3 H. Moravec, A. Elfes: High resolution maps from wide angle sonar, Proc. IEEE Int. Conf. Robotics Autom. (1995)

13.4 J. Laird, P. Rosenbloom: An investigation into reactive planning in complex domains, Proc. 9th Natl. Conf. Am. Assoc. Artif. Intell. (1990) pp. 1022–1029

13.5 N.J. Nilsson: *Shakey the Robot*, Tech. Rep. No. 325 (SRI International, Menlo Park 1984)

13.6 S.J. Rosenschein, L.P. Kaelbling: A situated view of representation and control, Artif. Intell. **73**, 149–173 (1995)

13.7 R.A. Brooks: Elephants don't play chess. In: *Designing Autonomous Agents: Theory and Practice form Biology to Engineering and Back*, ed. by P. Maes (MIT Press, Cambridge 1990) pp. 3–15

13.8 R. Brooks, J. Connell: Asynchrounous distributed control system for a mobile robot, Proc. SPIE Intell. Control Adapt. Syst. (1986) pp. 77–84

13.9 R.A. Brooks: A robust layered control system for a mobile robot, IEEE J. Robotics Autom. **RA-2**(1), 14–23 (1986)

13.10 P.E. Agre, D. Chapman: Pengi: An implementation of a theory of activity, Proc. 6th Natl. Conf. Am. Assoc. Artif. Intell. (1987) pp. 268–272

13.11 R.A. Brooks: Intelligence without representation, Artif. Intell. **47**, 139–159 (1991)

13.12 M. Schoppers: Universal plans for reactive robots in unpredictable domains, Proc. Int. Jt. Conf. Artif. Intell. (1987) pp. 1039–1046

13.13 P.E. Agre, D. Chapman: What are plans for? In: *Designing Autonomous Agents: Theory and Practice form Biology to Engineering and Back*, ed. by P. Maes (MIT Press, Cambridge 1990) pp. 17–34

13.14 G.N. Saridis: Intelligent robotic control, IEEE Trans. Autom. Control **AC-28**(5), 547–557 (1983)

13.15 R.J. Firby: An investigation into reactive planning in complex domains, Proc. AAAI Conf. (1987) pp. 202–206

13.16 R. Arkin: Towards the unification of navigational planning and reactive control, Proc. Am. Assoc. Artif. Intell., Spring Symp. Robotics Navig. (1989) pp. 1–5

13.17 C. Malcolm, T. Smithers: Symbol grounding via a hybrid architecture in an autonomous assembly system. In: *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, ed. by P. Maes (MIT Press, Cambridge 1990) pp. 123–144

13.18 J.H. Connell: SSS: A hybrid architecture applied to robot navigation, Proc. IEEE Int. Conf. Robotics Autom. (1992) pp. 2719–2724

13.19 E. Gat: Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots, Proc. Natl. Conf. Artif. Intell. (1992) pp. 809–815

13.20 M. Georgeoff, A. Lansky: Reactive reasoning and planning, Proc. 6th Natl.Conf. Am. Assoc. Artif. Intell. (1987) pp. 677–682

13.21 B. Pell, D. Bernard, S. Chien, E. Gat, N. Muscettola, P. Nayak, M. Wagner, B. Williams: An autonomous spacecraft agent prototype, Auton. Robots **1-2**(5), 1–27 (1998)

13.22 R.C. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1998)

13.23 M.J. Matarić: Reinforcement learning in the multi-robot domain, Auton. Robots **4**(1), 73–83 (1997)

13.24 P. Bonasso, R.J. Firby, E. Gat, D. Kortenkamp, D.P. Miller, M.G. Slack: Experiences with an architecture for intelligent reactive agents, Proc. Int. Jt. Conf. Artif. Intell. (1995)

13.25 M. Goller, T. Kerscher, J.M. Zollner, R. Dillmann, M. Devy, T. Germa, F. Lerasle: Setup and control architecture for an interactive shopping cart in human all day environments, Proc. Int. Conf. Adv. Robotics (2009) pp. 1–6

13.26 P. Pirjanian: Multiple objective behavior-based control, Robotics Auton. Syst. **31**(1-2), 53–60 (2000)

13.27 P. Pirjanian: *Behavior Coordination Mechanisms – State-of-the-Art*, Tech. Rep. IRIS-99-375 (Univ. of Southern California, Institute of Robotics and Intelligent Systems, Los Angeles 1999) pp. 99–375

13.28 M. Scheutz, V. Andronache: Architectural mechanisms for dynamic changes of behavior selection strategies in behavior-based systems, IEEE Trans. Syst. Man Cybern. B **34**(6), 2377–2395 (2004)

13.29 M. Proetzsch, T. Luksch, K. Berns: Development of complex robotic systems using the behavior-based control architecture iB2C, Robotics Auton. Syst. **58**(1), 46–67 (2010)

13.30 D. Payton, D. Keirsey, D. Kimble, J. Krozel, J. Rosenblatt: Do whatever works: A robust approach to fault-tolerant autonomous control, Appl. Intell. **2**(3), 225–250 (1992)

13.31 P. Maes: Situated agents can have goals. In: *Designing Autonomous Agents: Theory and Practice form Biology to Engineering and Back*, ed. by P. Maes (MIT Press, Cambridge 1990) pp. 49–70

13.32 P. Maes: The dynamics of action selection, Proc. Int. Jt. Conf. Artif. Intell. (1989) pp. 991–997

13.33 B.A. Towle, M. Nicolescu: Real-world implementation of an Auction Behavior-Based Robotic Architecture (ABBRA), Proc. IEEE Int. Conf. Technol. Pract. Robot Appl. (2012) pp. 79–85

13.34 B.A. Towle, M. Nicolescu: Fusing multiple sensors through behaviors with the distributed architecture, Proc. IEEE Int. Conf. Multisens. Fusion Integr. Intell. Syst. (2010) pp. 115–120

13.35 A. Saffiotti: The uses of fuzzy logic in autonomous robot navigation, Soft Comput. **1**, 180–197 (1997)

13.36 F. Michaud: Selecting behaviors using fuzzy logic, Proc. IEEE Int. Conf. Fuzzy Syst. (1997)

13.37 E. Gat: On three-layer architectures. In: *Artificial Intelligence and Mobile Robotics*, ed. by D. Kortenkamp, R. Bonasso, R. Murphy (MIT/AAAI Press, Cambridge 1998)

13.38 M.J. Matarić: Integration of representation into goal-driven behavior-based robots, IEEE Trans. Robotics Autom. **8**(3), 304–312 (1992)

13.39 M.J. Matarić: Navigating with a rat brain: A neurobiologically-inspired model for robot spatial representation, From animals to animats. Proc. 1st Int. Conf. Simul. Adapt. Behav. (1990) pp. 169–175

13.40 M. Nicolescu, M.J. Matarić: Experience-based representation construction: Learning from human and robot teachers, Proc. IEEE/RSJ Int. Conf. Intell. Robot Syst. (2001) pp. 740–745

13.41 M. Nicolescu, M.J. Matarić: A hierarchical architecture for behavior-based robots, Proc. Int. Jt. Conf. Auton. Agents Multiagent Syst. (2002)

13.42 L.E. Parker: ALLIANCE: An architecture for fault tolerant multirobot cooperation, IEEE Trans. Robotics Autom. **14**(2), 220–240 (1998)

13.43 M.J. Matarić: Designing and understanding adaptive group behavior, Adapt. Behav. **4**(1), 50–81 (1995)

13.44 C. Raïevsky, F. Michaud: Improving situated agents adaptability using interruption theory of emotions, From animals to animats. Proc. Int. Conf. Simul. Adapt. Behav. (2008) pp. 301–310

13.45 F. Michaud, G. Lachiver, C.T. Le Dinh: Architectural methodology based on intentional configuration of behaviors, Comput. Intell. **17**(1), 132–156 (2001)

13.46 F. Michaud: EMIB–Computational architecture based on emotion and motivation for intentional selection and configuration of behaviour-producing modules, Cogn. Sci. Q. **3-4**, 340–361 (2002)

13.47 F. Michaud, M.T. Vu: Managing robot autonomy and interactivity using motives and visual communication, Proc. Int. Conf. Auton. Agents (1999) pp. 160–167

13.48 O. Petterson, L. Karlsson, A. Saffiotti: Model-free execution monitoring in behavior-based robotics, IEEE Trans. Syst. Man Cybern. **37**(4), 890–901 (2007)

13.49 F. Michaud, M.J. Matarić: Learning from history for behavior-based mobile robots in non-stationary environments, Auton. Robots **5**(3/4), 335–354 (1998)

13.50 F. Michaud, M.J. Matarić: Learning from history for behavior-based mobile robots in non-stationary environments II, Auton. Robots **31**(3-4), 335–354 (1998)

13.51 F. Michaud, M.J. Matarić: Representation of behavioral history for learning in nonstationary conditions, Robotics Auton. Syst. **29**(2), 1–14 (1999)

13.52 O.C. Jenkins, M.J. Matarić: Deriving action and behavior primitives from human motion data, Proc. IEEE/RSJ Int. Conf. Intell. Robot Syst. (2002) pp. 2551–2556

13.53 O.C. Jenkins, M.J. Matarić: Automated derivation of behavior vocabularies for autonomous humanoid motion, Proc. 2nd Int. Jt. Conf. Auton. Agents Multiagent Syst. (2003)

13.54 M.J. Matarić: Designing emergent behaviors: From local interactions to collective intelligence, From animals to animats 2. Proc. 2nd Int. Conf. Simul. Adapt. Behav. (1992) pp. 432–441

13.55 S. Saripalli, D.J. Naffin, G.S. Sukhatme: Autonomous flying vehicle research at the University of Southern California, Proc. 1st Int. Work. Multi-Robot Syst. (2002) pp. 73–82

13.56 M.J. Matarić: Behavior-based control: Examples from navigation, learning, and group behavior, J. Exp. Theor. Artif. Intell. **9**(2–3), 323–336 (1997)

13.57 P. Maes, R.A. Brooks: Learning to coordinate behaviors, Proc. 8th Natl. Conf. Artif. Intell. AAAI (1990) pp. 796–802

13.58 H. Yanco, L.A. Stein: An adaptive communication protocol for cooperating mobile robots, From animals to animats 3. Proc 3rd Int. Conf. Simul. Adapt. Behav. (1993) pp. 478–485

13.59 J.R. del Millàn: Learning efficient reactive behavioral sequences from basic reflexes in a goal-directed autonomous robot, From animals to animats 3. Proc. 3rd Int. Conf. Simul. Adapt. Behav. (1994) pp. 266–274

13.60 L. Parker: Learning in cooperative robot teams, Proc. Int. Jt. Conf. Artif. Intell. (1993) pp. 12–23

13.61 M.J. Matarić: Learning to behave socially, From animals to animats 3. Proc. 3rd Int. Conf. Simul. Adapt. Behav. (1994) pp. 453–462

13.62 M. Asada, E. Uchibe, S. Noda, S. Tawaratsumida, K. Hosoda: Coordination of multiple behaviors acquired by a vision-based reinforcement learning, Proc. IEEE/RSJ Int. Conf. Intell. Robot Syst. (1994)

13.63 J. McCarthy: Making robots conscious of their mental states, AAAI Spring Symp. (1995)

13.64 T. Smithers: On why better robots make it harder, From animals to animats. Proc. 3rd Int. Conf. Simul. Adapt. Behav. (1994) pp. 64–72

13.65 D. McFarland, T. Bösser: *Intelligent Behavior in Animals and Robots* (MIT Press, Cambridge 1993)

13.66 P. Maes: A bottom-up mechanism for behavior selection in an artificial creature, From animals to animats. Proc. 1st Int. Conf. Simul. Adapt. Behav. (1991) pp. 238–246

13.67 B.M. Blumberg, P.M. Todd, P. Maes: No bad dogs: Ethological lessons for learning in Hamsterdam, From animals to animats. Proc. Int. Conf. Simul. Adapt. Behav., ed. by P. Maes, M.J. Matarić, J.-A. Meyer, J. Pollack, S.W. Wilson (1996) pp. 295–304

13.68 C. Breazeal, B. Scassellati: Infant-like social interactions between a robot and a human caregiver, Adapt. Behav. **8**(1), 49–74 (2000)

13.69 F. Michaud, P. Pirjanian, J. Audet, D. Létourneau: Artificial emotion and social robotics. In: *Distributed Autonomous Robotic Systems*, ed. by L.E. Parker, G. Bekey, J. Barhen (Springer, Tokyo 2000) pp. 121–130

13.70 A. Stoytchev, R. Arkin: Incorporating motivation in a hybrid robot architecture, J. Adv. Comput. Intell. Intell. Inf. **8**(3), 269–274 (2004)

13.71 S. Mahadevan, J. Connell: Automatic programming of behavior-based robots using reinforcement learning, Artif. Intell. **55**, 311–365 (1992)

13.72 J. Kober, A. Wilhelm, E. Oztop, J. Peters: Reinforcement learning to adjust parametrized motor primitives to new situations, Auton. Robots **33**, 361–379 (2012)

13.73 M.J. Matarić: Reward functions for accelerated learning, Proc. 11th Int. Conf. Mach. Learn., New Brunswick, ed. by W.W. Cohen, H. Hirsh (Morgan Kauffman, Boston 1994) pp. 181–189

13.74 H. Gleitman: *Psychology* (Norton, New York 1981)

13.75 M. Dorigo, M. Colombetti: *Robot Shaping: An Experiment in Behavior Engineering* (MIT Press, Cambridge 1997)

13.76 M. Nicolescu, M.J. Matarić: Learning and interacting in human-robot domains, IEEE Trans. Syst. Man Cybern. **31**(5), 419–430 (2001)

13.77 M. Nicolescu, O.C. Jenkins, A. Olenderski, E. Fritzinger: Learning behavior fusion from demonstration, Interact. Stud. **9**(2), 319–352 (2008)

13.78 R.C. Arkin: Motor schema based navigation for a mobile robot: An approach to programming by behavior, Proc. IEEE Int. Conf. Robotics Autom. (1987) pp. 264–271

13.79 S.B. Reed, T.R.C. Reed, M. Nicolescu, S.M. Dascalu: Recursive, hyperspherical behavioral learning for robotic control, Proc. IEEE World Autom. Congr. (2010) pp. 1–8

13.80 A. Olenderski, M. Nicolescu, S. Louis: A behavior-based architecture for realistic autonomous ship control, Proc. IEEE Symp. Comput. Intell. Games (2006)

13.81 M. Nicolescu, O.C. Jenkins, A. Olenderski: Learning behavior fusion estimation from demonstration, Proc. IEEE Int. Symp. Robot Hum. Interact. Commun. (2006) pp. 340–345

13.82 J.M. Peula, C. Urdiales, I. Herrero, I. Sánchez-Tato, F. Sandoval: Pure reactive behavior learning using case based reasoning for a vision based 4-legged robot, Robotics Auton. Syst. **57**(67), 688–699 (2009)

13.83 S. Huang, E. Aertbelien, H. Van Brussel, H. Bruyninckx: A behavior-based approach for task learning on mobile manipulators, Proc. 41st Int. Symp. Robotics 6th Ger. Conf. Robotics (2010) pp. 1–6

13.84 A.K. McCallum: Hidden state and reinforcement learning with instance-based state identification, IEEE Trans. Syst. Man Cybern. B **26**(3), 464–473 (1996)

13.85 E. Jauregi, I. Irigoien, B. Sierra, E. Lazkano, C. Arenas: Loop-closing: A typicality approach, Robotics Auton. Syst. **59**(3-4), 218–227 (2011)

13.86 D.J. Harvey, T.-F. Lu, M.A. Keller: Comparing insect-inspired chemical plume tracking algorithms using a mobile robot, IEEE Trans. Robotics **24**(2), 307–317 (2008)

13.87 A. Agha, G. Bekey: Phylogenetic and ontogenetic learning in a colony of interacting robots, Auton. Robots **4**(1), 85–100 (1997)

13.88 R.A. Brooks, L. Stein: Building brains for bodies, Auton. Robots **1**(1), 7–25 (1994)

13.89 B. Webb: Robotic experiments in cricket phonotaxis, From animals to animats 3. Proc. 3rd Int. Conf. Simul. Adapt. Behav. (1994) pp. 45–54

13.90 C. Liu, K. Conn, N. Sakar, W. Stone: Online affect detection and robot behavior adaptation for intervention of children with autism, IEEE Trans. Robotics **24**(4), 883–896 (2008)

13.91 N. Mitsunaga, C. Smith, T. Kanda, H. Ishiguro, N. Hagita: Adapting robot behavior for human–robot interaction, IEEE Trans. Robotics **24**(4), 911–916 (2008)

13.92 J.-W. Yoon, S.-B. Cho: An intelligent synthetic character for smartphone with Bayesian networks and behavior selection networks, Expert Syst. Appl. **39**(12), 11284–11292 (2012)

13.93 A. Brunete, M. Hernando, E. Gambao, J.E. Torres: A behaviour-based control architecture for heterogeneous modular, multi-configurable, chained micro-robots, Robotics Auton. Syst. **60**(12), 1607–1624 (2012)

13.94 J.L. Jones: Robots at the tipping point, IEEE Robotics Autom. Mag. **13**(1), 76–78 (2006)

13.95 M.F. Selekwa, D.D. Dunlap, D. Shi, E.G. Collins Jr.: Robot navigation in very cluttered environments by preference-based fuzzy behaviors, Robotics Auton. Syst. **56**(3), 231–246 (2008)

13.96 P. Rusu, E.M. Petriu, T.E. Whalen, A. Coronell, H.J.W. Spoelder: Behavior-based neuro-fuzzy controller for mobile robot navigation, IEEE Trans. Instrum. Meas. **52**(4), 1335–1340 (2003)

13.97 R. Huq, G.K.I. Mann, R.G. Gosine: Behaviour modulation technique in mobile robotics using fuzzy discrete event system, IEEE Trans. Robotics **22**, 903–916 (2006)

13.98 J.S. Cepeda, L. Chaimowicz, R. Soto, J.L. Gordillo, E.A. Alanís-Reyes, L.C. Carrillo-Arce: A behavior-based strategy for single and multi-robot autonomous exploration, Sensors **12**(9), 12772–12797 (2012)

13.99 A. Marino, L. Parker, G. Antonelli, F. Caccavale: A decentralized architecture for multi-robot systems based on the null-space-behavioral control with application to multi-robot border patrolling, J. Intell. Robotic Syst. **71**(3-4), 423–444 (2013)

13.100 L.E. Parker: Current research in multirobot systems, Artif. Life Robotics **7**(1–2), 1–5 (2003)

13.101 L.E. Parker, M. Chandra, F. Tang: Enabling autonomous sensor-sharing for tightly-coupled cooperative tasks, Proc. 1st Int. Work. Multi-Robot Syst. (2005) pp. 119–230

13.102 B.B. Werger, M.J. Matarić: Broadcast of Local Eligibility for Multi-Target Observation, Proc. 5th Int. Conf. Distrib. Auton. Robotics Syst. (2000) pp. 347–356

13.103 B.P. Gerkey, M.J. Matarić: Principled communication for dynamic multi-robot task allocation, Lect. Notes Control Inform. Sci. **271**, 353–362 (2001)

13.104 B.P. Gerkey, M.J. Matarić: Sold!: Auction methods for multi-robot coordination, IEEE Trans. Robotics Autom. **18**(5), 758–768 (2002)

13.105 B.P. Gerkey, M.J. Matarić: Pusher-watcher: An approach to fault-tolerant tightly-coupled robot coordination, Proc. IEEE Int. Conf. Robotics Autom. (2002) pp. 464–469

13.106 B. Akin, A.M. Aydan, I. Erkmen: A behavior based layered, hybrid, control architecture for robot/sensor networks, Proc. IEEE Int. Conf. Robotics Autom. (2006) pp. 206–211

13.107 N. Sedat, E. Aydan: A fractal conductivity-based approach to mobile sensor networks in a potential field, Int. J. Adv. Manuf. Technol. **37**(7), 732–746 (2008)

13.108 L. Iocchi, D. Nardi, M. Piaggio, A. Sgorbissa: Distributed coordination in heterogeneous multi-robot systems, Auton. Robots **15**(2), 155–168 (2004)

13.109 M. Batalin, G. Sukhatme: Coverage, exploration and deployment by a mobile robot and communication network, Telecommun. Syst. **26**(2–4), 181–196 (2004)

13.110 A.W. Stroupe, T. Balch: Value-based action selection for observation with robot teams using probabilistic techniques, Robotics Auton. Syst. **50**(2–3), 85–97 (2005)

13.111 R. Simmons, T. Smith, M.B. Dias, D. Goldberg, D. Hershberger, A. Stentz, R. Zlot: A Layered Architecture for coordination of mobile robots, Proc. NRL Work. Multi-Robot Syst. (2002)

13.112 J. Nembrini, A. Winfield, C. Melhuish: Minimalist coherent swarming of wireless networked autonomous mobile robots, Proc. 7th Int. Conf. Simul. Adapt. Behav. (2002) pp. 373–382

13.113 M. Egerstedt, X. Hu: Formation constrained multi-agent control, IEEE Trans. Robotics Autom. **17**(6), 947–951 (2001)

13.114 C.H. Tan, K.C. Tan, A. Tay: Computationally efficient behaviour based controller for real time car racing simulation, Expert Syst. Appl. **37**(7), 4850–4859 (2010)

13.115 K. Gold, B. Scassellati: Learning about the self and others through contingency, AAAI Spring Symp. Dev. Robotics (2005)

13.116 M. Baker, H.A. Yanco: Automated street crossing for assistive robots, Proc. Int. Conf. Rehabil. Robotics (2005) pp. 187–192

13.117 M. Williamson: Postural primitives: Interactive behavior for a humanoid robot arm, Proc. Int. Conf. Simul. Adapt. Behav. (1996)

13.118 M. Marjanovic, B. Scassellati, M. Williamson, R. Brooks: The Cog Project: Building a humanoid robot, Lect. Notes Artif. Intell. **1562**, 52–87 (1998)

13.119 A. Edsinger: Robot Manipulation in Human Environments, Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 2007)

13.120 C. Breazeal: Infant-like social interactions between a robot and a human caretaker, Adapt. Behav. **8**(1), 49–74 (2000)

13.121 H. Ishiguro, T. Kanda, K. Kimoto, T. Ishida: A robot architecture based on situated modules, Proc. IEEE/RSJ Int. Conf. Intell. Robot Syst. (1999) pp. 1617–1623

13.122 T. Kanda, T. Hirano, D. Eaton, H. Ishiguro: Person identification and interaction of social robots by using wireless tags, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (2003) pp. 1657–1664

13.123 F. Michaud, J.F. Laplante, H. Larouche, A. Duquette, S. Caron, D. Létourneau, P. Masson: Autonomous spherical mobile robotic to study child development, IEEE Trans. Syst. Man. Cybern. **35**(4), 1–10 (2005)

13.124 F. Michaud, S. Caron: Roball, the rolling robot, Auton. Robots **12**(2), 211–222 (2002)

13.125 F. Michaud, C. Côté, D. Létourneau, Y. Brosseau, J.-M. Valin, É. Beaudry, C. Raïevsky, A. Ponchon, P. Moisan, P. Lepage, Y. Morin, F. Gagnon, P. Giguère, M.-A. Roux, S. Caron, P. Frenette, F. Kabanza: Spartacus attending the 2005 AAAI Conference, Auton. Robots **12**(2), 211–222 (2007)

13.126 F. Michaud, Y. Brosseau, C. Côté, D. Létourneau, P. Moisan, A. Ponchon, C. Raïevsky, J.-M. Valin, E. Beaudry, F. Kabanza: Modularity and integration in the design of a socially interactive robot, Proc. IEEE Int. Work. Robot Hum. Interact. Commun. (2005) pp. 172–177

13.127 E. Beaudry, Y. Brosseau, C. Côté, C. Raïevsky, D. Létourneau, F. Kabanza, F. Michaud: Reactive planning in a motivated behavioral architecture, Proc. Am. Assoc. Artif. Intell. Conf. (2005) pp. 1242–1247

13.128 K. Haigh, M. Veloso: Planning, execution and learning in a robotic agent, Proc. 4th Int. Conf. Artif. Intell. Plan. Syst. (1998) pp. 120–127

13.129 S. Lemai, F. Ingrand: Interleaving temporeal planning and execution in robotics domains, Proc. Natl. Conf. Artif. Intell. (2004) pp. 617–622

13.130 D. Létourneau, F. Michaud, J.-M. Valin: Autonomous robot that can read, EURASIP J. Appl, Signal Process. **17**, 1–14 (2004)

13.131 J.-M. Valin, F. Michaud, J. Rouat: Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering, Robotics Auton. Syst. **55**(3), 216–228 (2007)

13.132 J.-M. Valin, S. Yamaoto, J. Rouat, F. Michaud, K. Nakadai, H.G. Okuno: Robust recognition of simultaneous speech by a mobile robot, IEEE Trans. Robotics **23**(4), 742–752 (2007)

13.133 F. Michaud, F. Ferland, D. Létourneau, M.-A. Legault, M. Lauria: Toward autonomous, compliant, omnidirectional humanoid robots for natural interaction in real-life settings, Paladyn. J. Behav. Robotics **1**, 57–65 (2010)

13.134 D.O. Hebb: *The Organization of Behavior: A Neuropsychological Theory* (Wiley, New York 1949)

13.135 G. Mandler: *Mind and Body: Psychology of Emotion and Stress* (Norton, New York 1984)

13.136 J.E. Stets: Emotions and sentiments. In: *Handbook of Social Psychology*, ed. by J.E. DeLamater (Wiley, New York 2003)

13.137 F. Ferland, F. Ferland, D. Létourneau, M.-A. Legault, M. Lauria, F. Michaud: Natural interaction design of a humanoid robot, J. Hum.-Robot Interact. **1**(2), 118–134 (2012)

13.138 F. Ferland, R. Chauvin, D. Létourneau, F. Michaud: Hello Robot, can you come here? Using ROS4iOS to provide remote perceptual capabilities for visual location, speech and speaker recognition, Proc. ACM/IEEE Conf. Hum.-Robot Interact. (1983) p. 101
13.139 E. Beaudry, D. Létourneau, F. Kabanza, F. Michaud: Reactive planning as a motivational source in a behavior-based architecture, Proc. IEEE/RSJ Int. Conf. Intell. Robot Syst. (2008) pp. 1848–1853
13.140 R.A. Brooks: *Cambrian Intelligence – The Early History of the New AI* (MIT Press, Cambridge 1999)
13.141 R. Pfeifer, C. Scheier: *Understanding Intelligence* (MIT Press, Cambridge 2001)
13.142 R.R. Murphy: *An Introduction to AI Robotics* (MIT Press, Cambridge 2000)
13.143 M.J. Matarić: *The Robotics Primer* (MIT Press, Cambridge 2007)
13.144 F. Martin: *Robotic Explorations: A Hands-On Introduction to Engineering* (Prentice Hall, Upper Saddle River 2001)
13.145 J.L. Jones, A.M. Flynn: *Mobile Robots – Inspiration to Implementation* (Peters, Wellesley 1993)

Multimedia Contents

14. AI Reasoning Methods for Robotics

Michael Beetz, Raja Chatila, Joachim Hertzberg, Federico Pecora

Artificial intelligence (AI) reasoning technology involving, e.g., inference, planning, and learning, has a track record with a healthy number of successful applications. So can it be used as a toolbox of methods for autonomous mobile robots? Not necessarily, as reasoning on a mobile robot about its dynamic, partially known environment may differ substantially from that in knowledge-based pure software systems, where most of the named successes have been registered. Moreover, recent knowledge about the robot's environment cannot be given a priori, but needs to be updated from sensor data, involving challenging problems of symbol grounding and knowledge base change.

This chapter sketches the main robotics-relevant topics of symbol-based AI reasoning. Basic methods of knowledge representation and inference are described in general, covering both logic- and probability-based approaches. The chapter first gives a motivation by example, to what extent symbolic reasoning has the potential of helping robots perform in the first place. Then (Sect. 14.2), we sketch the landscape of representation languages available for the endeavor. After that (Sect. 14.3), we present approaches and results for several types of practical, robotics-related reasoning tasks, with an emphasis on temporal and spatial reasoning. Plan-based robot control is described in some more detail in Sect. 14.4. Section 14.5 concludes.

14.1 **Why Should a Robot Use AI-Type Reasoning?** 330
14.2 **Knowledge Representation and Processing** 330
14.2.1 Logic-Based Reasoning 331
14.2.2 Probabilistic Reasoning 334
14.3 **Reasoning and Decision Making** 338
14.3.1 Using Multiple Knowledge Representation Formalisms 339
14.3.2 Reasoning About Time 339
14.3.3 Reasoning About Space 342
14.3.4 Reasoning About Multiple KR Formalisms 344
14.3.5 Knowledge Representation Systems for Robots 345
14.4 **Plan-Based Robot Control** 346
14.4.1 Models of Plan-Based Control 346
14.4.2 Representation of Plans 348
14.4.3 Generating Robot Action Plans 349
14.4.4 Revising and Improving Plans 350
14.4.5 Other Reasoning Methods 350
14.5 **Conclusions and Further Reading** 351
14.5.1 Further Reading 351
Video-References 351
References 352

Historical debates about the necessity and wisdom of employing symbolic reasoning in autonomous mobile robots notwithstanding (Chap. 13), there seems to be agreement now that some part or layer in the control system of such robots should or could include reasoning. Hybrid control architectures (Chap. 12) would be typical software structures for solving the difficult problem of amalgamating their contributions with those of other parts of the controller, yet guaranteeing sufficiently short control cycle times that the robot can act safely in a dynamic environment.

Symbolic reasoning is understood here in the classical sense of artificial intelligence (AI), i. e., deliberation based on symbols as in first-order predicate logic (FOPL) or Bayesian probability theory, but typically with restrictions and/or extensions thereof, which have to be traded off against each other to achieve a suitable combination of expressivity and inference speed.

14.1 Why Should a Robot Use AI-Type Reasoning?

If robots are to accomplish tasks that are too complicated for specifying the appropriate behavior for all possible situations beforehand, they should be able to reason themselves about what is the appropriate way to perform the task – even if the task is specified vaguely. In this setting, *reasoning* means inferring the needed knowledge from what is spelled out explicitly in the available knowledge. It further means *automated* reasoning using methods from AI including knowledge representation (KR) and inference, heuristic search, and machine learning.

To perform their course of action appropriately, robots need to reason about the actions they intend to perform, their intended effects, the unwanted side effects, whether they are executable in the respective situations, the goals the actions are to accomplish, and so on. Consider, for example, the seemingly simple task of picking up an object from a table. To do so, the robot has to decide where to go in order to pick up the object, which hand(s) to use, how to reach for the object, which type of grasp to apply, where to place the gripper, how much grasp force to apply, how much lift force, how to lift the object, where to hold it, and so on.

If programmers have to specify the decisions for every conceivable combination of object and task, the control program for picking up an object becomes very complex. But even this program would in most cases not suffice to generate competent robot behavior, because how the pickup action should be performed depends on the context, too – the state of the object, the task to be performed, the scene that the object is located in. If the object is a glass filled with juice, it has to be held upright and if it is a bottle the robot intends to fill a glass with, it should not grasp the top of the bottle. If the bottle is to be picked up in the middle of a cluttered scene, then grasping the top of the bottle might be the best choice. If the scene is cluttered and the purpose is filling a glass, then the robot might even have to re-grasp after picking up the object.

The situation gets even more complicated if the task is not a single action, but a compound activity such as cleaning the table. In this case, AI reasoning methods can be employed to enable programmers to specify the course of activity very compactly: *for every object on the table put it where it belongs*. For such a vague plan to be a sufficient activity prescription, the robot has to infer the information needed to decide on the appropriate parameterizations of the pickup actions and it has to infer where objects belong depending on whether they are to be used again, are dirty, perishable, etc. Competently cleaning the table might also require the robot to think about the order in which it wants to put away the objects. It might decide to group the carry actions according to the place where the items have to be placed, or clean up perishable items first. It might reason about whether to stack items, get a tray and use that, and whether to leave doors of cupboards open.

Thus, an important task of AI reasoning techniques is: given a vague instruction, infer what is the appropriate action, what is the appropriate object to be acted on, what is the appropriate order of action executions, and what is the appropriate way of performing every action.

To deal with these issues, autonomous robots can represent and reason about aspects including the robots' capabilities, their environments, the objects they are to act on, their actions and the effects they cause, and other agents in the environment.

Important kinds of reasoning that robots should be capable of, therefore include the following:

- *Prediction* (often called *temporal projection*): inferring what will (probably) happen if the intended course of action is executed
- *Envisioning*: inferring (all) possible events and effects that will happen if a plan gets executed in a hypothetical situation
- *Diagnosis*: inferring what caused a particular event or effect in plan execution
- *Query answering*: given some knowledge preconditions for plan execution (e.g., a robot has to know the combination of a safe in order to open it), inferring pieces of knowledge that satisfy these knowledge preconditions.

14.2 Knowledge Representation and Processing

Reasoning requires that the reasoner – in our case, a robot – has an explicit representation of parts or aspects of its environment to reason about. Two questions come up immediately: what are suitable formats for such an explicit representation, and where does the represented knowledge come from?

The second question refers to the problem of generating and maintaining in real time a symbolic description of the robot's environment, or at least of some part of it, based on, possibly, a previous symbolic description and on recent environment information as by sensors and by communication with other agents. In full

generality, this problem is currently unsolved, involving AI fundamentals such as *symbol grounding* [14.1] and *object anchoring* [14.2]. So practical symbolic reasoning in a robot is restricted to that part of its knowledge that can be kept sufficiently recent. This includes, obviously, static knowledge about the environment, such as the topological elements and their relations in a building; transient knowledge available in symbolic forms, such as in facility management data bases; and, most challenging, symbolic data that has to be distilled from sensor data. Object recognition (Chap. 33) is a crucial ingredient to solve this; these issues will not be treated here.

This section deals with answers to the first question, i. e., formalisms suitable for representing knowledge. Suitability has two aspects here that are to be treated as two sides of one coin. One is *epistemological adequacy*: Does the formalism allow the targeted aspects of the environment to be expressed compactly and precisely? The other is *computational adequacy*: Does the formalism allow typical inferences to be drawn effectively and efficiently? There is a tradeoff between the two adequacies, given that very rich, expressive, and hence epistemologically appealing formalisms are typically accompanied by intractable or even undecidable inference problems, and vice versa. KR, then [14.3, p. xiii]:

> *is the field of AI that focuses on the design of formalisms that are both epistemologically and computationally adequate for expressing knowledge about a particular domain.*

The plural in *formalisms* does not come by chance: there is no such thing as *the* KR language. The reason is twofold.

First, the archetype of KR languages, FOPL, is undecidable, i. e., any reasoner dealing with a language at least as expressive as FOPL is not guaranteed to terminate on a single query, let alone terminate fast. However, full FOPL expressivity is not always needed; for example, to represent a finite domain or to make restricted use of quantification may lead to decidable, and even tractable representation languages. Second, some applications of KR and reasoning, like quite a number of applications in Robotics, require evidence rather than truth to be represented, and FOPL, in particular, does not come handy for doing that. So, there are good reasons, both for pragmatics and for epistemological adequacy, for having several representation languages – and that has happened in AI.

Table 14.1 lists in its rows four broad types of KR languages, which have been in use in Robotics. Description logics (DLs) are in fact a specific family of relational languages, of which there are decidable and tractable members. The table gives in its columns the *Assertions* of the language types, i. e., the language elements used to represent some domain; the *Queries*, i. e., the type of problems that a reasoner in a particular language is supposed to solve; the *Subject* that a particular language of the respective type is normally used for representing; and *Examples* of such languages. The remainder of this section sketches generic KR in logic and in a probabilistic language, including the respective types of inferences. Specific robotics-related instances of reasoning on the symbol level are described in the following section, in particular, reasoning about temporal and spatial objects and relations, which are pervasive in a robot's reasoning.

14.2.1 Logic-Based Reasoning

As stated above, FOPL does not qualify for a KR formalism in the above-mentioned sense of being both epistemologically and computationally adequate for typical application domains – in many cases, it is neither of the two. Yet, it is the background for understanding conceptually and mathematically general relational languages.

In this short text, we can only introduce KR in logic by way of example, assuming familiarity with the basic notions of FOPL. For more comprehensive and formal treatments, we refer to the broad body of literature: every typical AI textbook introduces FOPL, and [14.4, 5] are no exceptions; [14.6] is a principled introduction to logics; [14.7] a practical one, and [14.8] a concise mathematical one.

Table 14.1 Abstract categorization of main relational languages as used in Robotics (see text for explanations)

	Assertions	Queries	Subject	Example
General relational languages	Sentences, theories	(Logical) consequence	Situations, contexts, relations	FOPL, prolog, situation calculus
Description logics	A-Box and T-Box contents	Subsumption, consistency	Ontologies, encyclopedias	OWL-DL
Probabilistic languages	Prior and conditional probabilities	Conditional probabilities	Evidence	Bayesian networks
Languages for temporal and spatial reasoning	Temporally and/or spatially qualified sentences	Logical consequence, satisfiability, minimal representation	Desired or observed spatial/temporal state of affairs	LTL, IA, TCSP, RCC, ARA$^+$

Representing knowledge in some variant of logic facilitates to draw inferences from that knowledge using provably sound and/or complete calculi. Automated deduction is a subfield of logic and AI that offers a healthy number of very powerful implemented deduction systems that can readily be used. Reference [14.4, Chap. 9] gives an introduction; [14.9] is a traditional handbook covering all the classical topics of logical reasoning.

Relying on logical inference, a robot could deduce a large number of facts that might otherwise be hard or impossible to get at. For example, assume a robot perceives by acquiring and interpreting sensor data, that the door D_{509} in an office building is currently closed: $Closed(D_{509})$ expressed in some ad hoc FOPL language, assuming intuitive interpretations of symbols. Let us further assume that the robot's base of static knowledge about the building contains the sentences

$$\begin{aligned}
&Connects(D_{509}, C_5, R_{509})\ ,\\
&Connects(D_{508}, C_5, R_{508})\ ,\\
&Connects(D_{508a}, R_{508}, R_{509})\ ,\\
&\forall d, l_1, l_2.[Connects(d, l_1, l_2) \leftrightarrow\\
&\quad Connects(d, l_2, l_1)]\ ,\\
&\forall d.[Closed(d) \leftrightarrow \neg Open(d)]\ ,\\
&\forall l.[At(l) \rightarrow Acessible(l)]\ ,\\
&\forall l_1, l_2.[Accessible(l_1) \rightarrow\\
&\quad (\exists d.[Connects(d, l_1, l_2) \wedge Open(d)] \rightarrow\\
&\quad Accessible(l_2))]\ . \qquad (14.1)
\end{aligned}$$

Constants D_i and variable d denote doors; R_i denote rooms; C_i corridors; variables l, l_i denote locations, i. e., rooms and corridors. Assume that the robot's localization tells it its current room or corridor location as $At(\cdot)$.

Then, assuming the robot knows it is $At(C_5)$, observing $Closed(D_{509})$ entails $\neg Open(D_{509})$, and, more interestingly, that $Accessible(R_{509})$ is true only if $Open(D_{508}) \wedge Open(D_{508a})$ is true. So, performing, for example, a delivery task to room R_{509}, the robot may replan its route through R_{508}, unless at least one of D_{508}, D_{508a} is known to be closed. If the status of one or both of them is unknown (neither $Open(\cdot)$ nor $Closed(\cdot)$ is entailed by the current knowledge base), then accessibility of R_{509} can be neither proven nor disproven. That would leave open the option of planning the route through D_{508} and D_{508a}, gathering their required statuses on site.

So, as this little example may show, FOPL is a powerful representation and reasoning tool. Moreover, its theory is very well understood. In particular, it is well-known that consequence in FOPL is in general undecidable, that means, a sound and complete deduction algorithm cannot even guarantee termination for some particular inference attempt, let alone speedy results.

However, that does not mean logic as a representation and inference tool needs to be completely disposed of. Many applications do not require the full expressivity of FOPL. Moreover, there are many interesting language subsets of FOPL that are decidable and can be handled by algorithms that are efficient in most practical cases. So considerable effort by the KR community has gone into identifying FOPL subsets and fitting inference procedures that qualify for being both epistemologically and computationally adequate for a broad number of applications. We will consider two of them in more detail: propositional logic and description logics; we will then go briefly into using logics for high-level robot control.

Propositional Theories

In quite a number of practical cases, FOPL theories (sets of formulas) represent in fact finite domains. Logically speaking, they have finite Herbrand universes or can at least be recast in a form such that they have. In this case, it may still be handy to express the domain theory in FOPL syntax, but all that goes beyond a purely propositional theory is just for notational convenience. For example, an axiom stating that a robot can only be in one location (room or corridor) at a time

$$\forall l_1, l_2.[At(l_1) \rightarrow (\neg At(l_2) \vee l_1 = l_2)] \qquad (14.2)$$

can be *flattened* for a finite building into the, clumsy but equivalent, form handling all locations explicitly, for example in a six-storey building

$$\begin{aligned}
&At(R_{001}) \rightarrow [\neg At(R_{002}) \wedge \cdots \wedge \neg At(R_{514}) \wedge\\
&\quad \neg At(C_0) \wedge \cdots \wedge \neg At(C_5)]\ ,\\
&At(R_{002}) \rightarrow [\neg At(R_{001}) \wedge \cdots \wedge \neg At(R_{514}) \wedge\\
&\quad \neg At(C_0) \wedge \cdots \wedge \neg At(C_5)]\ ,\\
&\ldots \qquad (14.3)
\end{aligned}$$

where every ground (variable-free) instance of a predicate, such as the At instances above, are to be considered as propositional variables, regarding textual identity.

The good news here is that the corresponding *flat* theory to a FOPL theory over a finite Herbrand universe is propositional, hence, it is decidable. Moreover, under some practical conditions – e.g., if the variables in the FOPL theory are sorted (that is, the information is available that the variables l_1, l_2, e.g., range over rooms and corridors) – it can even be generated mechanically from a more compact FOPL syntax.

The potentially bad news is that the now propositional theory may of course consist of a huge amount

of propositional sentences: in general, as all combinations of variable substitutions in all FOPL sentences need to be generated, the growth is multiexponential in terms of the domain sizes of the FOPL variables.

However, the technology for propositional satisfiability checking or *model checking* is making considerable progress, allowing propositional theories with thousands or even tens of thousands of variables to be handled in the order of seconds of computation time on regular hardware. The respective methods are particularly efficient if many models exist for a satisfiable propositional formula or if the truth or falsity of many ground facts is known a priori (such as $At(C_5)$ for a robot by independent localization). Both is often the case in practical KR.

References [14.4, Chap. 7.6] and [14.9, Chap. 24] give introductions to model checking. One modern family of methods is based on the classical Davis–Putnam (DPLL) algorithm [14.10]. It attempts to construct systematically a propositional model of a given theory, efficiently propagating interpretation constraints for variables. Another family of algorithms applies local search techniques, attempting to generate interpretations using random (Monte Carlo) variable assignments. Reference [14.11] is a Web page maintained for the annual satisfiability checking competitions held together with the annual SAT (*Theory and Applications of Satisfiability Testing*) conferences; it gives a continuously updated overview over current top-performing satisfiability checking systems.

Description Logics

DL have emerged as rigorous formalizations of somewhat intuitive KR forms of the AI of the 1970s, namely, semantic networks and frame systems. Logically speaking, DLs, of which there is a considerable variety, form a certain family of subsets of FOPL, some of which are decidable or even tractable.

There are two parts of representing knowledge about a particular domain using a DL language. First, the *upper ontology* of the domain is formulated. It introduces the general domain concepts as well as relations between these concepts. A particularly interesting type of relations occurring in all ontologies is the superclass-subclass relation. Given that most DLs – as part of the means for enforcing decidability – strictly disallow cyclic relations between concepts to be specified, the superclass relation imposes a hierarchical taxonomy on concepts, which serves for defining property inheritance, much like in object-oriented programming. This first part of a DL-based domain representation is, for historical reasons, often called the *T-Box* (or *terminological knowledge*).

A *concept* in the DL language corresponds to a unary predicate. To give an example, an ontology of robot navigation domains might include concepts like *Door*, *Location*, and so on. The concept hierarchy is built by defining concept equality $=$ or a subconcept property, e.g., $Room \sqsubseteq Location$, and $Corridor \sqsubseteq Location$. Concepts can be combined using concept conjunction, disjunction, and negation ($\sqcap, \sqcup, \neg$, respectively), allowing, e.g., concept definitions like $Location = Room \sqcup Corridor$, $Door = Closed \sqcup Open$, and $Open = \neg Closed$.

Roles in a DL language correspond to binary predicates, such as *leadsTo* for a door and a location. Inversity, intersection, and union of roles are defined as expected, where $leadsTo = leadsFrom^{-1}$ is an example for defining inverse roles. Roles can be composed, such as in defining $adjacent = leadsFrom \circ leadsTo$ (location l is adjacent to m if and only if some door connects them). Finally, concepts and roles can be combined for defining new concepts and roles. In particular, it is possible to quantify over *role-fillers*, i. e., the individual objects (see next) that can be consistently substituted for role arguments. For example, one could define $BlockedLoc = Location \sqcap \neg\exists leadsFrom.Open$ (assuming the intuitive binding rules of the operators). Different variants of DLs differ in what operators they make available; the set of available operators and additional constraints on definitions shape both the expressivity and the computability of the respective DL variant – the spectrum ranges from undecidable to tractable. See [14.3] for details.

As the second part of domain representation using DLs, individual objects have to be introduced into the language of concepts and roles. This part of the domain representation is called *A-Box* (or *assertional knowledge*). For example, $Room(R_{509})$, $leadsTo(D_{509}, R_{509})$, and $Closed(D_{509})$ could be asserted.

DLs have a number of reasoning services to offer, which are based on logical inference in a given T-Box and A-Box. They include consistency of the concept definition, subsumption and disjointness of concepts, consistency of the A-Box wrt. the T-Box, concept and role instances, all of which are decidable in many DLs. Given the T-Box and A-Box rudiments above, it would, e.g., be concluded that everything is consistent and that $BlockedLoc(R_{509})$ (note that only door D_{509} is known here to lead to R_{509}!). These DL inferences are theoretically intractable, but run efficiently in most of practical cases.

Reference [14.3] provides a comprehensive overview of DL. In 2004, the WWW consortium (W3C) has defined the Web Ontology Language OWL [14.12] as a technical basis for the Semantic Web, which was followed by OWL 2 in 2009 [14.13].

Part of the language is OWL-DL, a classical DL in the sense just described. OWL-DL ontologies are publicly available over the Web. Reference [14.14] gives a tutorial introduction.

Logics for High-Level Robot Control

Robot domains are dynamic by nature, including at least one physical agent, namely, the robot. Capturing that in logic-based formalisms is possible and has been undertaken, as introduced in [14.4, Chap. 12.3]. However, it poses some conceptual and technical problems, which we will describe at the end of this section on Logics.

Very briefly, one of them comes from the requirement of expressing concisely and reasoning efficiently with logical models of events in general, and actions in particular. Logically speaking, an individual action changes the truth value of a limited number of facts, leaving everything else as it was. For example, modeling on some abstract level the action of going from one location to another as atomic, it changes the robot's current location before and after; depending on the modeling, it may also change the battery state and the mileage count; but it does not change the layout of the building or the name of the president. The problem of formalizing actions concisely in a logical language so that they allow facts to be inferred efficiently that may or may not have changed after applying a sequence of actions, has been termed the *frame problem*. It has received much attention in the literature; there are now a number of practical solutions to it.

Another problem concerns knowledge-base update made necessary by independently changing facts. Consider, for example, a robot sitting – as it is told by its self-localization – in front of some door D, which is believed to be open, but the robot perceives a closed door. Logically, this is a contradiction. Now there are in theory several ways to make the knowledge and the perception consistent. One is to assume that D has been closed since learning that it was open – probably the most intuitive explanation. Logically just as good would be, e.g., that the perception is faulty, or that the robot has been teleported in front of a known closed door. Among these *explanations*, some are more intuitively plausible than others; logically, some would require less formulas of the knowledge base to be withdrawn and should therefore be preferred. Ideally, after replacing an old piece of information with a new one, one would have to make sure that the consequences of a retracted formula are no longer believed.

Theoretically, these problems are arbitrarily difficult. Practically, they can be sufficiently restricted to allow solutions within a neat logical framework. Typical solutions would introduce some notion of state or holding period for formulas, following the classical example of the situation calculus [14.15]. That allows change to be tracked. Typical solutions would also give up completeness of inference, resorting to a Prolog-like inference engine. Three examples for such solutions with reported applications in robot control are GOLOG [14.16], event calculus [14.17] and FLUX [14.18]. Another family of solutions would model the course of time in a more fine-grained way than switching between states. Temporal logics are appropriate here, and the temporal action logic (TAL) is an example, which has even been integrated in a performant planner [14.19]. It leads to a more sophisticated form of temporal reasoning that will be treated in Sect. 14.3.

14.2.2 Probabilistic Reasoning

KR formalisms based on logics are worth considering whenever factual knowledge is to be represented, from which consequences are to be queried. However, part of the knowledge that a robot might use about its environment does not really have this character.

Uncertainty is one of these traits, or rather, a whole family of them, as uncertainty is in itself an overloaded concept. *Lack of knowledge* is one of its aspects. Logics can handle this insofar as truth or falsity of some facts may remain undetermined. If too much is undetermined in a knowledge base, logics will no longer be able to make interesting deductions though, as *everything is possible* logically. Yet, the different possibilities may differ considerably in likelihood, according to intuition. The point here is to represent and reason about *evidence*.

The field of KR has adopted Bayesian probability as a means for representing and reasoning with evidence, using correlations between facts rather than strict implications. Note that this approach amalgamates different sources of lack of precision and completeness of knowledge. Some piece of knowledge may be unknown because it is in principle unknowable, or because it was judged too costly to build a precise theory or determine all information relevant for making a sound deduction. In either case, working with probabilities rather than binary truth values can serve as an approximation. Note that the notion of truth is still the same here as in classical logics: *objectively*, a fact is supposed to be either true or false; probability just models the *subjective* evidence that the fact is true.

We next describe two popular and powerful representation and processing formats for probabilistic knowledge: Bayesian networks (BNs) and Markov decision processes (MDPs). In analogy to Sect. 14.2.1, we here assume familiarity with the basic notions of probability theory. Reference [14.4, Chap. 13] gives an

excellent basic introduction; for a more comprehensive treatment, there is a large variety of introductory textbooks around, [14.20] being an example.

Bayesian Networks

Inference in Bayesian probability theory basically means to infer the probability of some event of interest, given prior and dependent other relevant probabilities. Practically, an important type of probabilistic inference is *diagnostic reasoning* from observed effects back to hidden causes, given rules that specify conditional probabilities from causes to effects. So the problem is, for a potential cause C and an observed effect E: Given prior probabilities $P(C)$ and $P(E)$ and the conditional probability $P(E|C)$, determine the posterior $P(C|E)$. The solution is of course given by the Bayes rule as

$$P(C|E) = \frac{P(E|C)P(C)}{P(E)} \,. \tag{14.4}$$

However, like in logical reasoning, the theoretically appealing principle turns out to be impractical if applied naively. Consider that not just one effect E may be observed, but $E_1, \ldots, E_n$; moreover, not all of them are conditionally independent. A generalized form of the Bayes rule to calculate the correct posterior is straightforward, but who can specify all the conditional probabilities involved – in the worst case $O(2^n)$ of them, where n may easily range in the hundreds in practical cases?

Until the late 1980s, this problem was more or less circumvented. One way to do so was to treat the E_i in bad faith as independent: simply take the n individual conditional probabilities $P(E_i|C)$ and use them for approximating straightforward the full joint probability distributions. Reference [14.4, Chap. 14] reviews this and other alternative approaches.

The solution used ever since it has appeared [14.21] is *Bayesian networks* (BN), more recently subsumed under the more general concept of *graphical models*. The idea is to represent the random variables as nodes in a directed acyclic graph, where a node is directly preceded by a set of parent nodes if and only if it is directly conditionally dependent on the corresponding parent variables. So the huge full joint probability distribution is broken down into many, typically very small, *local joint probability distributions* without loss of information, the trick being to use the typically many known conditional independences among variables to reduce the representational and inferential effort drastically.

Figure 14.1 shows a simple BN expressing that D is dependent on B and C, with probabilities given by a conditional probability table which specifies locally the joint probability distribution. In addition, the structure says that D is independent from A, given B and C (a node is independent from its ancestors, given its parents), and that B is independent from C, given A, i. e., $P(C|A,B) = P(C|A)$ and $P(B|A,C) = P(B|A)$.

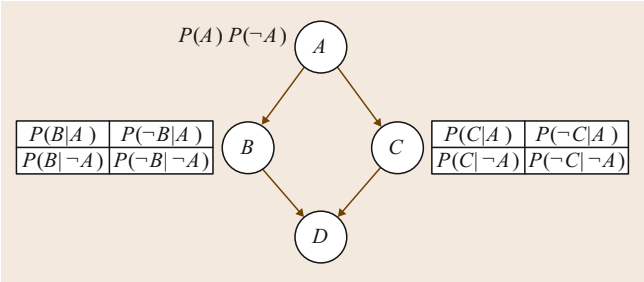


Fig. 14.1 Structure of a simple Bayesian network. It is associated with conditional probability tables. (The one for D, dependent on B and on C, is omitted)

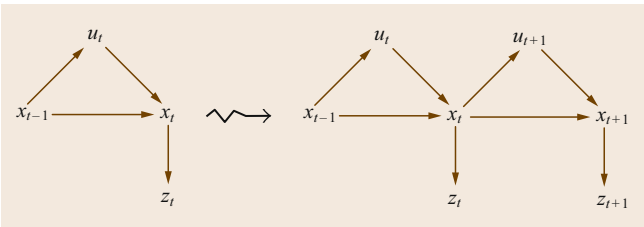


Fig. 14.2 Unfolding of a simple DBN over two time slices representing two successive states. Variables x denote states, u denote actions, and z denote measurements. The structure of each individual slice (*signaled by identical variable subscripts*) is invariant over time

Network inference can be interpreted in both directions. Bottom up, a BN enables to explain an observed phenomenon using the known conditional probabilities (diagnosis). For example, if D is observed, the likelihoods of its known causes (B or C) can be inferred. Top down, a BN enables to propagate evidence to compute the probability of a phenomenon, e.g., to compute the probability of D given A (causation).

For systems evolving over time, for which the Markov property holds, i. e., a state of a system variable in some state depends on no variables earlier than the previous state, a BN representation takes on a particular form, a *dynamic Bayesian network*, DBN. Assuming that the dependences among variables within a state remain constant over time, and assuming further that a variable x at different states is represented as a family of variables $x_1, \ldots, x_t, \ldots$, a DBN unfolded into a BN has a particular structure, as sketched by example in Fig. 14.2. The variables representing a new state correspond to a new BN *slice*, which reproduces the structure of the previous state representation and whose nodes depend only on each other and on nodes represent-

ing the previous state. This locality allows inference in DBNs to be efficient.

Markov Decision Processes

In the context of robot control, uncertainty is induced by perception, action, and incompleteness of prior knowledge about the environment, all of which are related. Perception uncertainty results from sensor noise, from occlusion, from interpretation ambiguities, and other effects. Action uncertainty results from approximate action models (e.g., wheel slippage), unsuccessful action (e.g., grasping failure), or other practically unavoidable effects.

Figure 14.3 depicts this uncertainty laden interaction between the robot and its environment.

Probabilistic representations are used to cope with these uncertainties. Planning being a sequential decision problem, MDP and, in the case of partial observability of the environment, partially observable Markov decision processes (POMDPs) [14.22] are widely used to address them. The Markov assumption, which is the hallmark of this approach, is that a state s_n of the system depends only on the previous state s_{n-1} and the action that lead to s_n, and not on any earlier states.

Under this assumption, an MDP is formally defined by four components $\langle S, A, T, R\rangle$ where S is a finite set of states, A is a finite set of actions, $T : S \times A \to S$ the transition function defining the probability of state change upon application of a given action at time t

$$T(s, a, s') = P(s_{t+1} = s' \mid s_t = s, a_t = a) \,. \qquad (14.5)$$

T is known and provided as a table of transition probabilities. Finally, $R(s, a)$, $R : S \times A \to \mathbb{R}$, is defined as the *reward* received by the system after achieving action a leading to state s. Often, the reward is associated with the state only ($R(s)$). In the sequential process, the rewards are supposed to be additive. Solving an MDP is an optimization problem in which the sum of the expected rewards, called the *utility*, is maximized. MDPs suppose that the rewards are already known. When this is not the case, the robot can acquire them through experience. One technique to do so is reinforcement learning, as described in Chap. 15. Figure 14.4 shows an example of a simple MDP.

There are two main methods to solve an MDP. In the *value iteration* (VI) method, the first step is to compute the utilities $U(s)$ for all states, and the second step is to compute the optimal *policy* $\pi^*(s)$ which provides, for each state, the optimal action. Computing the utilities iteratively is based on the Bellman equation [14.23]

$$U_{i+1}(s) = R(s) + \gamma \max_a \sum_{s'} T(s, a, s') U_i(s') \,, \qquad (14.6)$$

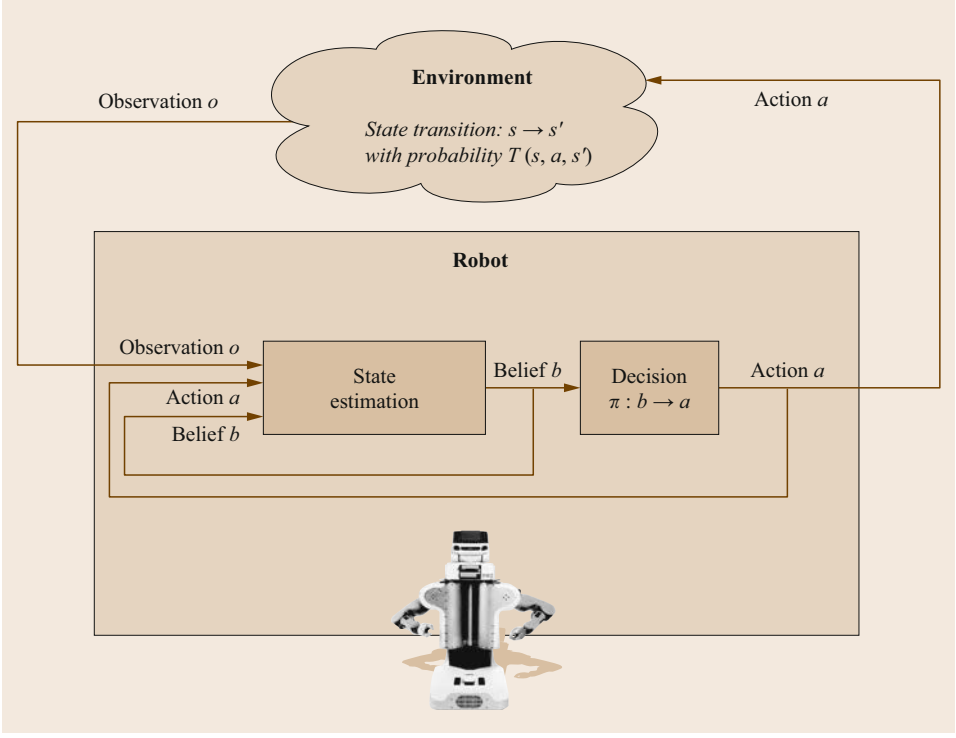


Fig. 14.3 The robot acts on its environment producing a new state with a given probability. Knowledge of the robot about state results from its observation and is also uncertain. Its decisions are taken based on its beliefs

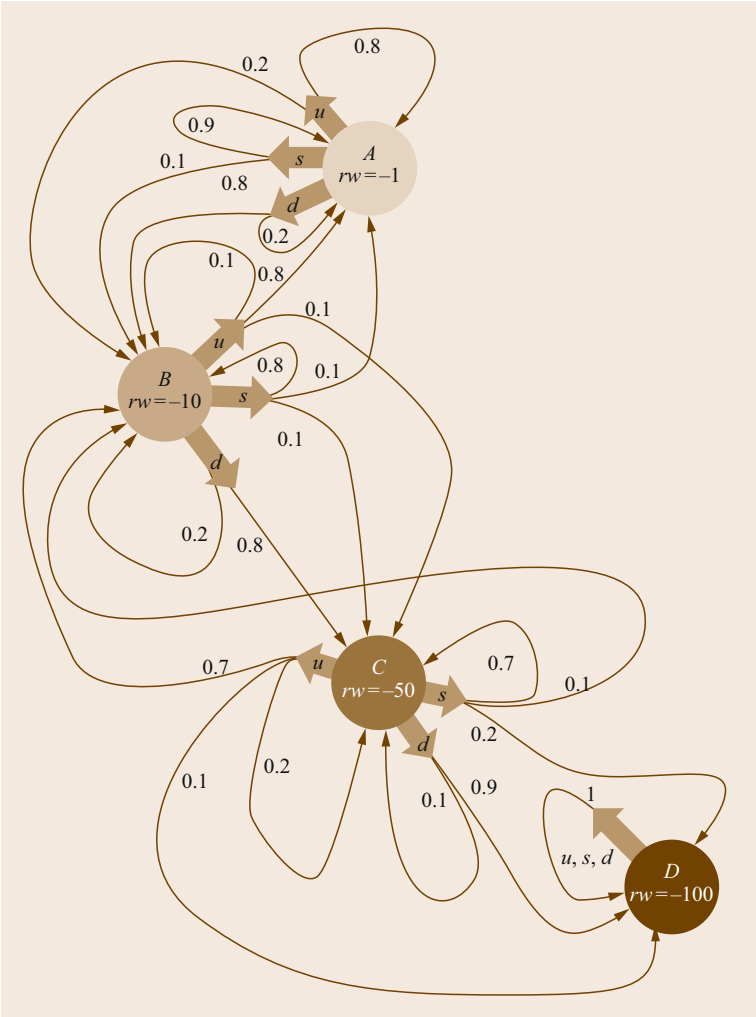


Fig. 14.4 A representation of an MDP. In this example, states A, B, C, D have associated rewards $(-1, -10, -50, -100)$. There are three possible actions in each state, u, s, d, each have a given probability (*shown on the arrows*) to lead to another state

where $0 < \gamma < 1$ is a discount factor. After computing the utilities, the optimal policy is the one that provides the best action (i. e., the one maximizing the utility) for each state. This is computed by $\pi^*(s) = \arg\max_a \sum_{s'} T(s, a, s')U(s')$. So the result of the MDP is the best local decision. A policy is an a priori computation of the best sequence of actions, which is guided by the maximization of a global reward instead of just reaching a goal with minimal cost, as in classical planning. So a policy is not a plan of actions that has to be strictly executed. Since it already takes uncertainties in action execution into account, the policy can be pursued whatever the outcome of action execution.

The second method is called *policy iteration* (PI). Here one starts with an initial (e.g., random) policy π and, through an iterative algorithm, tries to improve it gradually by looking for the actions that augment utility.

In conclusion, in a deterministic setting, the solution to a planning problem is a sequence of actions. In an observable stochastic setting, the solution is a policy defined by the best local decision.

Partially Observable Markov Decision Processes

In the (realistic) case where the robot has no full access to the environment, decision making must be made under partial observability. To that end, the MDP formalism is enriched with an observation function that provides uncertain knowledge. A POMDP is defined by six components $\langle S, A, T, R, \Omega, O \rangle$. The first four describe an MDP. Ω is a finite set of observations o. $O : S \times A \to \Omega$ is the observation model $O(s', a, o)$ (derived for the robot sensor models) that provides the probability to obtain observation o after action a which has lead to state s'. The robot *belief state* is the probabil-

ity distribution over all states, i. e., all the probabilities of being in all the states. The belief to be in a given state is usually noted by $b(s)$. The belief is updated after an action and observation by

$$b'(s') = \alpha O(s', a, o) \sum_s T(s, a, s') b(s) \,, \qquad (14.7)$$

where α is a normalization constant.

Although partial observability is a typical case for robots acting in mundane environments, and hence POMDPs are the more adequate formalism for modeling action and observation, POMDP-based models are often avoided. The reason is that calculating optimal POMDP policies is impracticably complex. Mostly, solving a POMDP is based on approximate or derived methods, such as working with the corresponding *belief MDP* of a POMDP, which results from considering the (fully observable) belief function induced by the POMDP.

A good example of using Markov processes for modelling and solving robotics decision-making problems, combining task planning, and motion planning is provided in [14.24].

14.3 Reasoning and Decision Making

The main motivation for developing reasoning capacities in robotics is to enable robots to make decisions about their future actions. This is why planning was one of the main early research topics in AI, within the SHAKEY project (VIDEO 704), resulting in the seminal STRIPS planner [14.25]. Designing a planning system requires to address three main questions:

- How to represent the world?
- How to represent actions?
- How to guide the plan search process?

According to the formalism used to answer these questions, the planning system will cope differently with the constraints of the real world, and this is an essential issue in robotics. As mentioned above, early planning techniques were based on first-order predicate logic (FOPL) for representing knowledge, and the limitation of FOPL when facing the uncertainties of the real world has lead to adopt probabilistic representations and probabilistic reasoning to cope with real situations.

World Representations. Classically, when reasoning at the symbolic level, knowledge about the world is represented as *states* which are sets of logical propositions such as Ontable(Cup). Under the closed world assumption, the facts that have the value FALSE are not represented in the states of the world. In order to deal with uncertainties, instead of having TRUE/FALSE values, the propositions have a probability distribution of being true.

Representation of Actions. In classical planning, as introduced by STRIPS, actions modify the state of the world and are defined by three sets of predicate lists. The predicates that should be true for the action to be feasible are the preconditions. The set of predicates that become TRUE after the action is executed is the ADD list. The set of predicates that becomes FALSE after action execution is the DELETE list. The planner includes a unification procedure to determine which predicates correspond to the propositions describing the world states. Taking into account uncertainties in action execution leads to including a probabilistic characterization of their outcome.

Search. Finding the most adequate action is based on a search algorithm in the state space. The state space is an implicit graph given by the initial state and a successor function that provides the states immediately adjacent to a given state. Each action, which represents the transition from one state to its successor, has a given cost. The well known A* algorithm [14.26] was proposed in 1968, again within the SHAKEY project, to address the search problem. The planning algorithm, based on this search, recursively selects the action that would produce the current goal state (or a subgoal thereof), and under certain conditions the optimal (i. e., least cost) solution is found.

This classical planning scheme is not able to cope with uncertainties because it is based on FOPL representations. The only way it can do that is through plan mending or replanning (Sect. 14.4). When probabilistic representations are used, the plan search will adopt a very different scheme as explained in Sect. 14.2.2.

However, reasoning in AI and robotics involves much more than sequential planning. Robots act in the real world in which action duration determines success or failure. They have to consider events occurring at given instants and situations that unfold over time. Temporal relations and an appropriate temporal logic are necessary to represent and reason about time. Section 14.3.2 discusses temporal reasoning.

Another central issue is reasoning about space. What does it mean exactly when we represent a piece of knowledge by the predicate Ontable(Cup), or Near(Robot,Table)? How to express spatial relationships symbolically from sensor data? This involves

specific formalisms as well which are developed in Sect. 14.3.3.

14.3.1 Using Multiple Knowledge Representation Formalisms

Automated planning carries an immediate appeal for robotic applications: the actions a robot should perform to achieve a goal are obtained as a result of reasoning in a model which can be changed to suit different environments, physical capabilities, and tasks. Yet reasoning about action per se is not a silver bullet for robot programming. As noted earlier, obtaining competent robot behavior by reasoning in a model crucially depends on the epistemological adequacy of the modeling language. As a case in point, let us elaborate on our initial example: imagine a waiter robot with two arms that has to serve meals to guests in a restaurant. The robot would have to reason about time in order to infer that cold beverages should be delivered before they get warm. Food needs to be served in front of guests, hence the robot should be capable of some form of spatial reasoning. It should be capable of reasoning about resources, e.g., infer that it cannot hold more than two parts of the meal simultaneously with its two arms, or that its tray can only accommodate a limited number of dishes. A DL ontology might inform the robot that all mugs in the cupboard are clean – hence, committing to a particular mug at planning time is not necessary – or that a cup or a mug both work well for serving coffee. Most of this knowledge is difficult or impossible to model in *classical* planning models [14.27], which are limited to describing the causal relations that exist between actions and predicates in FOPL.

The types of knowledge a robot should possess ultimately depend on the application. It should be however evident that – short of encoding this knowledge in FOPL, which is computationally inadequate – several models expressed in different formalisms are necessary in most meaningful domains. Note also that any plan containing actions that are to be executed by a robot should also be translated to actionable metric terms that are understandable by the robot. For instance, specific positions in which objects are to be placed, locations to navigate to, and action dispatch times should be indicated in the plan. Finally, the knowledge represented in different models a robot should use for reasoning (the causal model being only one of these) often presents nontrivial interdependences: the capacity of the robot's tray (resource model) determines the number of trips a robot has to perform to and from the table it is clearing (causal reasoning), the type of meal (ontological model) may affect the spatial layout of the cutlery (spatial reasoning), and the time it takes for a beverage to get cold (temporal model) will affect the order of goal achievement (causal reasoning). The development of problem solving algorithms that account for these interdependences is an active topic of research. An overview of recent results in this direction is provided in Sect. 14.3.4.

Of particular interest to this discussion are temporal and spatial KR formalisms. Several temporal and spatial logics possess properties that make them both epistemological and computationally adequate for a variety of robotic problems. Below, we outline the principal temporal and spatial KR formalisms that are relevant to robotics.

14.3.2 Reasoning About Time

Linear temporal logic (LTL) [14.28] is a decidable propositional logic which provides a means to describe how execution paths will evolve in the future. The temporal operators $\bigcirc$ (next), $\Box$ (always), $\Diamond$ (eventually), $\mathcal{U}$ (until) and $\mathcal{R}$ (release) are used to represent conditions on the state of a system. For instance, *the robot will eventually place the cup on the table* (reachability), *the robot will never serve a cold coffee* (safety), and *the robot will always reach its charging station before its battery is discharged* (liveness). A fragment of LTL (specifically, syntactically co-safe LTL formulae [14.29]) has become relevant in robotics because of its ability to predicate upon discrete temporal properties of motions. For example, an exploration robot may be required to visit locations A, B and C in a predetermined sequential order; or conditions like *avoid C unless A or B have been visited, and always avoid D*. Given such temporal goal specifications, it is possible to compute motions that satisfy these goals with probabilistic roadmaps (PRMs) [14.30, 31].

LTL formulae express qualitative temporal specifications, thus offering the domain expert a means to specify conditions on execution at a high level of abstraction. LTL does not however consider that robot actions have a temporal extent, and that the temporal extents of different actions may be required to have specific relations. The qualitative temporal logics point algebra (PA) [14.32] and interval algebra (IA) [14.33] can be used to capture (and enforce) such relations. Specifically, they allow to represent qualitative temporal relations among temporal variables. Variables in PA represent time points, which can be used to represent events, such as *the robot has reached the table*, or start/end times of actions, such as *the robot starts to place a mug on the table*. In IA, variables represent time intervals, e.g., *the robot navigates to the table*, or *the robot places the mug on the table*. The basic PA relations are the three temporal relations $\{<, >, =\}$. All

unions of these relations ($\leq$, $\geq$, $\neq$, the universal relation $\top$, and the empty relation $\emptyset$) are also valid PA constraints. In IA, constraints represent temporal relations among temporal intervals. The basic IA relations B_{IA} are the 13 possible temporal relations between intervals, namely *precedes* (p), *meets* (m), *overlaps* (o), *during* (d), *starts* (s), *finishes* (f), their inverses (e.g., p^{-1}), and *equals* ($\equiv$) (Fig. 14.5). A constraint in IA is a disjunction of basic relations $\{r_1, \ldots, r_n\} \in B_{\mathrm{IA}} \times \ldots \times B_{\mathrm{IA}}$. Going back to our example, a relevant piece of temporal knowledge for our waiter robot is that the action of picking the mug temporally overlaps the fact that it is holding the mug, i. e., the IA relation

$$\text{Pick(Mug)} \ \{\mathrm{o}\} \ \text{Holding(Mug)} \ . \tag{14.8}$$

This knowledge in fact represents the qualitative state of affairs in the presence of a successful pick-up action, whereas the contingency in which the mug slips from the robot's gripper (an all too common occurrence with today's robots!) can be expressed by the temporal relation

$$\text{Pick(Mug)} \ \{\mathrm{d}^{-1}\} \ \text{Holding(Mug)} \ . \tag{14.9}$$

A set of variables and constraints in PA or IA constitutes a constraint satisfaction problem (CSP) [14.34]. A fundamental reasoning problem in temporal CSPs is to test the *satisfiability* of the CSP – assessing whether a substitution of temporal values to variables exists that satisfies all constraints. A related problem is that of computing the *minimal* representation of a given CSP, that is, the equivalent CSP formed by the strongest implied constraints. Both problems are tractable for PA and for several fragments of IA [14.35]. A particularly useful example of tractable subalgebra of IA is the set of *convex* IA relations [14.36]. For example, {b, m, o} is a convex IA relation, whereas {b, o} is not. These good computational properties can be leveraged by robots during plan execution. Ensuring robust execution in the case of our waiter robot picking a mug need not be hard-coded as an ad-hoc status checking procedure into the implementation of the pick up action. Conversely, it can be modeled as a simple relation as above, and its verification can be cast as a CSP containing the relations

$$\begin{aligned} &\text{Pick(Mug)} \ \{\mathrm{o}\} \ \text{Holding(Mug)} \ , \\ &\text{Pick(Mug)} \ \{\mathrm{d}^{-1}\} \ \text{Holding(Mug)} \ . \end{aligned} \tag{14.10}$$

The first constraint represents the requirement we expect to hold in nominal execution, whereas the second constraint results from perception and proprioception (more on this later). The above CSP is not satisfiable: the observed and executed situations are not consistent with the modeled temporal relation representing successful plan execution. This determination can be made in polynomial time (with respect to the number of variables in the CSP) with a particular form of inference, called *path consistency* [14.37].

Although convenient for specifying high-level requirements on robot plans, PA and IA cannot capture metric information. For this, we need metric temporal constraints, which allow to predicate on durations and relative temporal distances between variables. An important metric temporal problem is the temporal constraint satisfaction problem (TCSP). As in PA, variables represent time points. Constraints represent disjunctions of bounds on the temporal distance between the pairs of time points: given two time points a and b, the constraint

$$a \xrightarrow{[l_1,u_1]\vee[l_2,u_2]\vee\ldots[l_n,u_n]} b$$

states that $l_i \leq b - a \leq u_i$, for one or more $i \in \{1 \ldots n\}$. A TCSP is satisfiable iff at least one disjunct (pair of

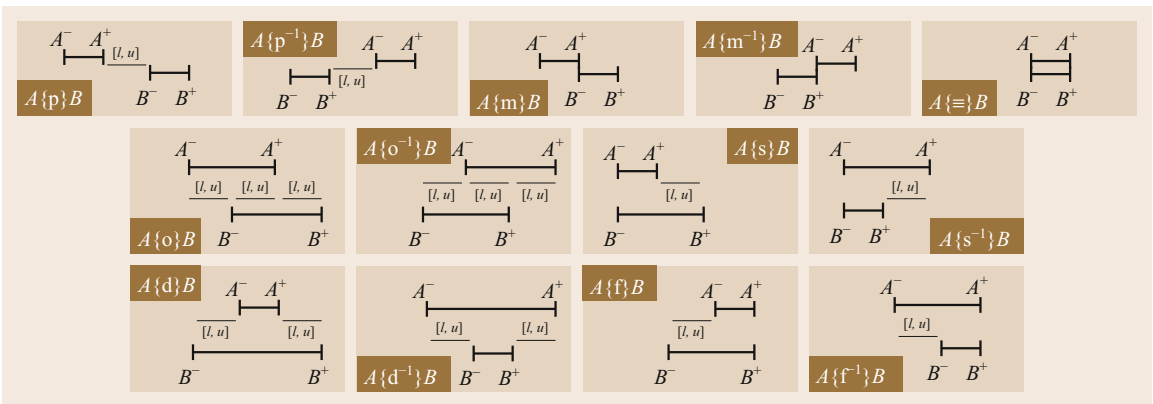


Fig. 14.5 The 13 basic qualitative relations in interval algebra. $[l, u]$ indicate metric bounds that can be attached to the basic relations to refine the metric semantics of the relations

inequalities) per constraint holds. Unlike PA and IA, variables in a TCSP are associated with an explicit set of possible assignments of time values, called *domain*. Computing the minimal representation consists of restricting the domains of time points to the smallest intervals of time containing values that satisfy all constraints. A particular restriction of the TCSP, called the simple temporal problem (STP), has the property that domains are contiguous intervals of time. The condition for remaining in the STP fragment of TCSP is that constraints have only one disjunct, i. e., are of the form

$$a \xrightarrow{[l,u]} b \,.$$

Whereas computational problems for the the general TCSP are NP-hard, the satisfiability and minimal representation problems for the STP are tractable. Both problems are solved by path consistency inference [14.38], and highly optimized algorithms have been developed for the satisfiability and minimal representation problems [14.39, 40].

The semantics of basic PA and IA can be understood through the use of metric temporal constraints. For instance, the constraint

$$\text{Pick(Mug)}\ \{\text{o}\}\ \text{Holding(Mug)} \tag{14.11}$$

is equivalent to the metric constraints

$$\begin{aligned} &\text{Pick(Mug)}^- \xrightarrow{[1,\infty)} \text{Holding(Mug)}^- \,, \\ &\text{Holding(Mug)}^- \xrightarrow{[1,\infty)} \text{Pick(Mug)}^+ \,, \end{aligned} \tag{14.12}$$

where $(\cdot)^-$ and $(\cdot)^+$ represent, respectively, the start and end times of the corresponding intervals. Moreover, metric bounds can be used to define a metric extension of IA in which we can express high-level relations between intervals with additional metric bounds (Fig. 14.5). For instance, $A\{\text{p}[5, 13]\}B$ states that interval A should end at least 5 and at most 13 time units before interval B starts. In addition to binary constraints, it is also common practice to define the two unary constraints Release$[l, u]A$ and Deadline$[l, u]A$, stating, respectively, that A starts between l and u time units after the origin of time, and that A ends between l and u time units after the origin of time.

Overall, constraint-based temporal calculi can represent both qualitative and metric time. Provided that appropriate fragments of IA are used in modeling, the reasoning tasks can be conveniently reduced to path consistency inference in an STP. This is an attractive feature for robot reasoning: it provides a means to anchor observations and proprioception in time. In the example above, the robot must infer from sensing that the relation

$$\text{Pick(Mug)}\ \{\text{d}^{-1}\}\ \text{Holding(Mug)} \tag{14.13}$$

holds. An implementation of this capability can be achieved by representing as time points both observed and planned behavior, and constraining these time points so as to reflect the precise times in which they were observed. Assuming the current time is 20, that the robot started to pick the mug at time 5, and that the gripper reports that it is holding an object between times 10 and 18, the following STP models the robot's situation

$$\begin{aligned} &\text{Pick(Mug)}\ \{\text{o}[1,\infty)[1,\infty)[1\infty)\}\ \text{Holding(Mug)}\,, \\ &\text{Release}[5,5]\ \text{Pick(Mug)}\,, \\ &\text{Deadline}[20,20]\ \text{Pick(Mug)}\,, \\ &\text{Release}[10,10]\ \text{Holding(Mug)}\,, \\ &\text{Deadline}[18,18]\ \text{Holding(Mug)}\,. \end{aligned} \tag{14.14}$$

The above STP is not satisfiable, reflecting the fact that execution has not gone according to plan. Again, this state of affairs can be ascertained in low-order polynomial time, hence providing a simple mechanism for online fault detection.

The use of temporal constraint reasoning in planning and plan execution goes beyond diagnosing failures. The fact that intervals are grounded on a metric temporal CSP (an STP) provides a means to maintain knowledge about when actions should be dispatched, and whether they should be delayed to accommodate unexpected contingencies. For instance, we can express that the robot should start moving back to the counter once the mug has been held for at least 3 s with the constraint

$$\begin{aligned} &\text{Holding(Mug)}\ \{\text{o}[3,\infty)[1,\infty)[1,\infty)\} \\ &\quad \text{Move(Table, Counter)}\,. \end{aligned} \tag{14.15}$$

A delay in the start time of the Holding(Mug) predicate will *propagate* to the start time of the Move(Table, Counter) action, progressively pushing its start time into the future until the robot starts holding the mug. Correct propagation of exogenous events is ensured by maintaining in the STP the requirements of the plan together with constraints representing when events are observed, when actions are dispatched, and when they terminate. A simple procedure for maintaining these constraints is the following:

- When an action is dispatched at time t, a Release$[t, t]$ constraint is imposed on the corresponding interval, constraining its start time to the current time.

- The fact that the action is still executing at time $t+m$ is modeled with the constraint Deadline$[t+m+1, \infty)$, modeling the fact that the action will end some time in the future.
- When the lower level executive signals at time $t+m+n$ that an action has finished, the constraint Deadline$[t+m+n, t+m+n]$ is added to the STP, thus constraining the action's end time.

Every time a constraint is added, the minimal STP is recomputed – that is, the domains of the time points (start and end times of actions and predicates) are updated. In so doing, we ensure that the lower bound of time points representing the start times of actions will always correspond to the earliest admissible time at which actions should be dispatched to the robot's executive layer. This guarantees that execution will uphold all temporal constraints modeled in the temporal CSP. These may include additional requirements on the plan, such as specifications regarding necessary orderings of actions, actions that must occur concurrently, and synchronizations with exogenous events.

The computational adequacy of temporal CSPs entails that reasoning about temporal models can be performed online during execution. Thus, reasoning about time is a way to enable fault diagnosis, guarantee timely action dispatching, and enforce temporal specifications on robot plans. Mechanisms based on temporal constraint reasoning have been used as a tool for this purpose in many settings involving execution of plans on real physical systems [14.41, 42]. Similar techniques have been used to account for uncontrollable events in plan execution [14.43], inferring context from sensor traces [14.44], and integrated context inference and planning [14.45]. A good overview of the fundamental principles of temporal constraint reasoning underlying these results is given in Dechter's book on constraint processing [14.46].

14.3.3 Reasoning About Space

Spatial KR formalisms serve the purpose of specifying desired spatial relations in a scene. As for temporal models, they can be used to make sense of sensor traces, to enforce conditions during plan execution, as well as to drive the planning process itself. Most work has focused on the use of qualitative spatial relations for scene understanding (e.g., in the context of perceptual anchoring [14.47]). Structural pattern recognition, important applications of which can be found in medical domains, employ cognitive vision techniques to match qualitative spatial knowledge (representing a specified structure) to perceived context [14.48, 49]. In many applications, qualitative relations do not belong to a well-defined calculus, rather are tailored to capture specific features (e.g., distance, orientation, and shape) which are useful for pattern specification and recognition in particular applications. As is the case for temporal knowledge, a well-defined spatial calculus is useful because of its provable formal properties – e.g., tractability of specific reasoning problems, expressiveness of the language, and so on. Well-founded spatial calculi permit logical reasoning, and, not surprisingly, they are grounded on similar principles as PA and IA. We illustrate here these principles and show simple examples of their use in robotic applications.

The main entities of interest in qualitative spatial calculi are objects, or, rather, the regions (or points) of physical space that they occupy. Spatial calculi provide a means to represent the relations among these regions. There exist several well-known and well-studied qualitative calculi. Each of these calculi focuses on one category of spatial concepts – e.g., topology, direction and distance. Region connection calculus (RCC) [14.50] is used for representing and reasoning with topological relations. Cardinal direction calculus (CDC) [14.51] is an approach based on directional relations. As for temporal calculi, these calculi use the language of constraints to represent spatial properties, and classical constraint-based reasoning techniques like path consistency [14.52] to ascertain consistency.

Of particular interest here is RCC, which is grounded on eight spatial relations describing the connectedness of regions: *disconnected* (DC), *externally connected* (EC), *tangential proper part* (TPP), *nontangential proper part* (NTPP), *partially overlapping* (PO), *equal* ($\equiv$), and the inverses TPP^{-1} and NTPP^{-1} (Fig. 14.6).

The full algebra deriving from the eight basic RCC relations is called RCC-8. An important restriction of RCC-8 is RCC-5, obtained by subsuming the relations DC and EC into one relation, and NTPP and TPP into another. The satisfiability and minimal CSP problems are NP-hard for both RCC-8 and RCC-5 – however, these problems are tractable if we restrict the language to the basic relations (as for IA), and a large number of tractable fragments of RCC-8 and RCC-5 are known [14.53]. A good example of where these properties are useful is in the problem of anchoring modeled relations to observed spatial layouts. This is exemplified in the work by *Cohn* et al. [14.54] on video sequence analysis, where the problem is to construct qualitative representations of the evolution of spatial relations in video sequences. The qualitative descriptions are obtained online, due to the computational adequacy of the qualitative spatial calculus used (a derivative of RCC called CORE-9).

Qualitative spatial reasoning can be used, much like qualitative temporal reasoning, to robustify plan execu-

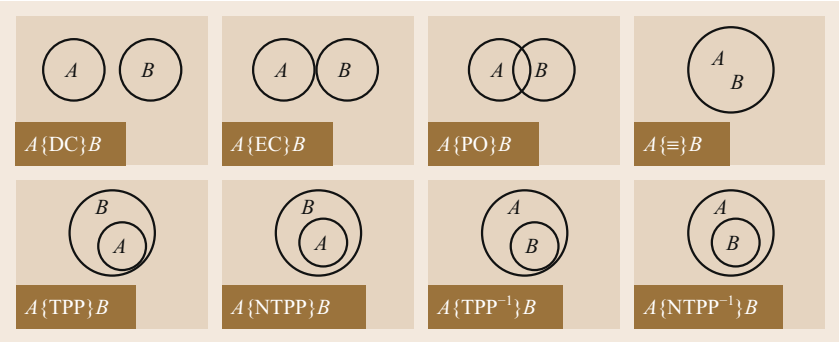


Fig. 14.6 The eight basic qualitative relations in RCC

tion. We may, for instance, require that mugs should be placed *on top of* saucers (Mug NTPP Saucer). As we have shown for temporal reasoning, the task of verifying the correct state of affairs can be cast to a CSP which is guaranteed to be feasible iff the desired spatial layout is observed.

However, note that all we can express with RCC constraints are topological relations between objects. RCC cannot express notions like *left-of*, *above*, and so on, as the only assumption made on variables is that they are convex regions of space. Conversely, rectangle algebra (RA) [14.55] expresses more than topological relations – at the cost of assuming that objects have a particular shape. Specifically, RA is an extension of IA to two dimensions. Its variables thus represent axis-parallel rectangles, and relations are pairs of IA relations, one for each axis.

RA can be augmented, like IA, with metric bounds. The resulting calculus, called ARA$^+$ [14.56], subsumes both topology and cardinal relations. For instance (Fig. 14.7), the relation

$$B\ \langle \mathrm{p}[5, 13], \mathrm{p}\rangle\ A \tag{14.16}$$

states that B precedes A in both the x and y axes, and that the distance along the x axis between A and B should be at least 5 and at most 13. The ARA$^+$ relation subsumes the qualitative relation *A is Northeast of B*, as well as the RCC relation $A\{\mathrm{DC}\}B$. Note that rectangular regions are compatible with the representation of segmented objects in most off-the-shelf perception modules. This property facilitates the process of going from sensor data to symbols. Toward this aim, ARA$^+$ provides the unary constraints $\mathrm{At}[l_1, u_1][l_2, u_2][l_3, u_3][l_4, u_4]$ and $\mathrm{Size}[l_1, u_1][l_2, u_2][l_3, u_3][l_4, u_4]$. The first constraint bounds the length of the sides of a rectangle, while the second bounds the placement of a rectangle in 2-D space. Note that the At constraint performs a similar function to the Release and Deadline constraints used in the metric extension of IA, which denote specified or perceived absolute placements in time of intervals. It is thus intuitive to see how the At constraint can be *implemented* with Release and Deadline constraints in the augmented IA CSPs used to represent the projections of rectangles on the two axes.

We can employ ARA$^+$ constraints to represent desired placements of objects, both in qualitative and metric terms. For instance, the specification of a well-set table for our waiter robot could be

$$\begin{aligned}
&\mathrm{Fork}\ \langle \mathrm{d}[5, +\infty)[5, +\infty),\\
&\qquad \mathrm{d}[5, +\infty)[5, +\infty)\rangle\ \mathrm{Table}\ ,\\
&\mathrm{Knife}\ \langle \mathrm{d}[5, +\infty)[5, +\infty),\\
&\qquad \mathrm{d}[5, +\infty)[5, +\infty)\rangle\ \mathrm{Table}\ ,\\
&\mathrm{Fork}\ \langle \mathrm{p}, \mathrm{d}\rangle\ \mathrm{Mug}\ ,\\
&\mathrm{Mug}\ \langle \mathrm{p}, \mathrm{d}\rangle\ \mathrm{Knife}\ ,\\
&\mathrm{Mug}\ \mathrm{Size}[8, 8][8, 8]\ ,\\
&\mathrm{Fork}\ \mathrm{Size}[2, 2][15, 15]\ ,\\
&\mathrm{Knife}\ \mathrm{Size}[2, 2][15, 15]\ ,
\end{aligned} \tag{14.17}$$

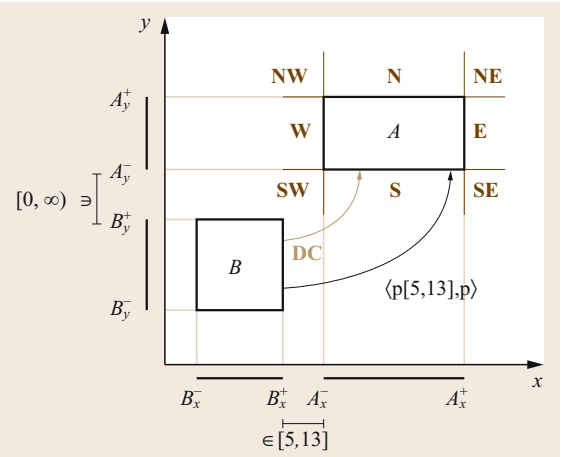


Fig. 14.7 The ARA$^+$ relation $B\langle \mathrm{p}[5, 13], \mathrm{p}\rangle A$, which subsumes RCC relation $A\{\mathrm{DC}\}B$ and CDC relation *A* Northeast of *B*

that is, forks and knives should be at least 5 cm from the edge of the table, forks should be located on the left of mugs and knives on the right, the size of forks and knives is $2 \times 15\,\text{cm}^2$, and the size of mugs is $8 \times 8\,\text{cm}^2$.

The above relations can be maintained, much like we have illustrated for temporal reasoning, in a lower-level metric constraint-based representation – in this case, consisting of two STPs, one for each axis of the reference frame. It has been shown [14.56] that these two STPs are satisfiable iff the ARA$^+$ CSP is satisfiable, and that the minimal STPs contain all the admissible placements of objects. The minimal STPs can thus be used to extract an admissible placement of the objects in the scene. Technically, this is done by *hallucinating* the objects that are to be placed in the scene, that is, representing both observed objects and those that are intended to be placed into the scene in the spatial CSP. In the example above, supposing the robot must place the mug on a table which already has a fork and a knife on it, the hallucinated object is the mug, while the fork and knife would be constrained by At constraints representing their observed locations. The bounds of all variables in the minimal STPs represent all the admissible placements of the objects. Those of the fork and knife will not have been refined in the minimal representation, as they were already fixed by the At constraints resulting from observation; conversely, the bounds of the Mug rectangle are refined in the minimal representation to contain only those points that are admissible wrt. the other constraints in the spatial CSP.

So far, we have shown how employing constraint-based temporal and spatial KR formalisms allows to cast the problem of fault detection to a tractable problem that can be solved online. Another important aspect of plan execution that is facilitated by these representations is fault identification and repair [14.57]. It is easy to see why an explicit representation formulated in terms of constraints facilitates these tasks: suppose that the fork and knife are placed on the table correctly from the qualitative point of view – i. e., the fork is left of the knife – but that they are only 5 cm apart. The resulting spatial CSP would not be satisfiable, as it would be impossible to fit the mug (whose bounding box is $8 \times 8\,\text{cm}^2$) between the two pieces of cutlery, as prescribed by constraints (Fork$\langle$p, d$\rangle$Mug) and (Mug$\langle$p, d$\rangle$Knife). The spatial CSP can be used at this point as a tool to perform *culprit analysis*: the same CSP without the At constraint modeling the observed placement of the fork represents the situation in which we assume that the fork can be moved; if this CSP is satisfiable, then we know that replacing the fork is a way to achieve a well-set table.

Spatial reasoning has not received as much attention as temporal reasoning in the context of robotic applications. The AI and Robotics research communities are, however, quickly clustering around this important topic. Spatial calculi are a well studied topic in AI, albeit with less applications in robotics than temporal reasoning. As robots become more competent in manipulation and navigation tasks, the issue of representing and enforcing high-level requirements on space begins to come to the fore. As always, it is the combination of epistemological and computational adequacy of significant fragments of spatial KR formalisms that makes them useful in robot applications. Indeed, the potential of qualitative and metric spatial reasoning goes beyond planning and plan execution, with active research directions in human–robot interaction [14.58–60] and object search [14.61]

14.3.4 Reasoning About Multiple KR Formalisms

Despite their simplicity, the examples we have used to illustrate the basic concepts of temporal and spatial reasoning point to the fact that reasoning about action, time, space, and any other knowledge possessed by the robot must occur jointly. For instance, the decision to place the mug on the table may depend on the spatial layout of other objects on the table, or on whether there is a more urgent task to be carried out before doing so. It is easy to see that even this simple task cannot ignore kinematics and geometric constraints, which may call for approaching the table from a different side, or to move another object before placing the mug. Recent work in integrated task and motion planning is a response to the need for integrating these forms of reasoning into planning. Today, this is very much seen as the next big challenge, and consistent research results in this specific area of hybrid reasoning are closing the gap between classical planning and real robotic applications [14.62–67].

We argue, however, that kinematics and geometric constraints are only two of the important types of knowledge necessary for achieving competent robot behavior. Comparatively less work exists in integrating other types of knowledge. DL reasoing has been employed to generate high-level plans [14.68], to refine planning domains [14.69], and *Galindo* et al. [14.70] show how to enhance the task planning process with spatial and ontological models. The largest volume of work in integrating diverse forms of reasoning into planning focuses on integrating planning, temporal, and resource reasoning – e.g., planning with metric temporal reasoning [14.71], qualitative temporal models [14.44], combined qualitative and metric temporal

reasoning [14.72], and resource scheduling [14.73–75].

Although these advancements contribute to widening the expressive power of models for robot reasoning, they do not suggest general methods for jointly reasoning about heterogeneous KR formalisms. As noted earlier, the types of knowledge a robot should possess ultimately depend on the application, which makes this an important basic research question. A few examples pointing in the direction of general solutions do exist, although work in the area is sparse. These include planning modulo theories [14.76], an extension of satisfiabiliy modulo theories (SMTs) [14.77], which enrich classical planning models with semantics belonging to arbitrary first order theories. SMTs themselves are an example of general hybridization scheme, and have been employed in a robotic context by *Nedunuri* et al. [14.78] to enforce metric requirements on motion and manipulation tasks. The approach known in the literature as meta-constraint reasoning is similar in structure to SMTs, but is grounded on the more general notion of CSP. Meta-constraint reasoning has been used to develop solvers for highly expressive planning domains that include qualitative and metric time, resources, and logical constraints [14.79]. In the context of robotic applications, the approach has been used for online planning with qualitative and metric time, spatial relations in ARA$^+$ and resources [14.57], as well as for online configuration planning for multirobot systems [14.80].

14.3.5 Knowledge Representation Systems for Robots

In autonomous robot control systems, the kinds of information that we have discussed in the previous subsections are not only represented and reasoned about but also used subsymbolically for generating perception-guided activity. For example, some information such as the appearance and poses of the objects on the table might be generated by the perception system of the robot through the interpretation of sensor data. Other information such as the position of the robot in the environment might be estimated using state estimation algorithms and stored in dedicated data structures. Also, symbolic action descriptions have to be translated into low-level parameterizations of control routines in order to generate behavior and cause effects.

These aspects of embodying abstract and symbolic reasoning in robots are at least partly addressed by system-level research on KR and reasoning for robots. The resulting systems serve as integrated question-answering components in a robot's control program that gather information from different sources and employ a range of inference methods for controlling and parameterizing control routines.

Examples of recent systems include the pntology-based unified robot knowledge (OUR-K) framework [14.81], mostly used for navigation actions, the ORO system [14.82] that focuses on knowledge for human–robot interaction and dialog grounding, and the KnowRob system [14.83] providing deep knowledge for robotic manipulation tasks. The semantic layer [14.84] of the PEIS Ecology [14.85] serves both autonomous robots and ambient intelligent environments. In the Ke Jia project [14.86], Answer Set Programming is used as representation for integrating KR, natural language processing, task planning and motion planning on a mobile platform. Also classical cognitive architectures such as Soar [14.87] and ACT-R also integrate modules for storing knowledge [14.88] with inference capabilities, but have only rarely been used on robots [14.89].

One field that has seen much progress in the past years is the creation and representation of environment models, often termed *semantic maps*. Advances in object recognition allowed to detect and perceptually interpret single objects in environments, which enables robots to store higher level semantic information about them in their respective maps. The term *semantic* is used for a range of different interpretation capabilities – from a mere segmentation and categorization of the environment into different region types to the segmentation of objects, their categorization, and the interpretation of their properties [14.90], or from the co-occurrence of objects in scenes [14.91] up to environment representations in description logics [14.92], from statistical relational environment models [14.93] to an embedding of spatial information into an encyclopedic and common-sense knowledge base [14.94]. Many of the reasoning tasks that we have discussed in Sect. 14.3.3 have implicitly assumed expressive semantic maps that can make symbolic place descriptions such as *on the table* or *in the cupboard* effective.

Knowledge representation aspects also become increasingly important for the instruction of robots with natural language as well as for human–robot interaction. For understanding directives involving objects in the environment [14.95] and for following spoken directions [14.96–99], a robot might have to ground the words it hears or reads into its percepts and actions. The natural-language text may be the result of direct interaction with a human, but can also be obtained from resources on the Web, for example, for mining object locality knowledge [14.100] or extracting task instructions from websites [14.101] (VIDEO 705).

Web-enabled robotics is another branch of robotics that has recently become increasingly important. Modern robot control systems use the worldwide web as an information resource [14.102], for sharing knowledge and skills [14.103], for telepresence applications [14.104], to employ web services [14.105], and to use computational resources provided through the web [14.106] (VIDEO 706).

14.4 Plan-Based Robot Control

Plan-based control is an information processing approach for controlling autonomous robots that aims at making robot control systems more general – in at least three respects. First, it enables robots to successfully carry out multiple, diverse, and possibly interfering tasks. Second, it can improve the course of action by adapting the control software for the respective execution context. Third, plan-based control can enable robots to perform novel tasks without requiring the collection of extensive experience.

In the plan-based approach, robots generate actions and behavior by synthesizing, maintaining, and executing plans, where plans are robot control programs that a robot cannot only execute but also reason about and manipulate [14.107]. Thus plan-based controllers can manage and adapt plans in order to better achieve complex and changing tasks [14.108]. The use of plans enables robots to flexibly interleave complex and interacting tasks, exploit opportunities, quickly plan their courses of action, and, if necessary, revise their intended activities.

Plans in plan-based control have two roles. They are (1) executable prescriptions that can be interpreted by the robot to generate behavior in order to accomplish its jobs and (2) representations that can be synthesized and revised by the robot to meet the robot's criterion of utility. Besides having means for representing and generating plans, plan-based controllers are also equipped with tools that enable computational processes to (1) predict what might happen when a robot controller gets executed and return the result as an execution scenario, (2) infer what might be wrong with a robot controller given an execution scenario, and (3) perform complex revisions on robot controllers.

In most general terms robot planning can be considered to be the automatic generation, refinement, revision, and optimization of robot plans [14.107]. As a computational problem it can be formulated as follows: given an abstract plan P and a description of the current situation, find an executable realization Q that maximizes some objective function V. In this problem formulation, an abstract plan might be to go shopping and to clean up the apartment, to win a robot soccer game, to monitor the traffic in a particular area using an autonomous helicopter, or, for a museum tour-guide robot, to inform and entertain the visitors of the museum.

Robot planning algorithms aim at finding appropriate plans by hypothesizing possible courses of action and predicting what could happen when the plans get executed. Based on these predictions, the robot decides whether the plan will meet its objectives.

In the remainder of this subsection we will first discuss conceptual models underlying robot planning and the representation of robot plans. We will then talk about the automated synthesis of robot plans and finally discuss mechanisms for revising plans and optimizing plans based on experience.

14.4.1 Models of Plan-Based Control

The generation of plans and the prediction of what will happen is based on models of the robot, its actions, and the world. The most commonly used models are state-transition systems (also called discrete-event systems or problem spaces) [14.27].

Using state-transition systems, the behavior and the effects of robot activity can be represented as a triple $\Sigma = \langle S, A, \gamma \rangle$, where:

- $S = \{s_1, s_2, \ldots\}$ is a set of states
- $A = \{a_1, a_2, \ldots\}$ is a set of actions
- $\gamma: S \times A \rightarrow S$ is a state-transition function.

Thus, the action repertoire of a robot can be stated as a graph where the nodes represent the states of the environment and links of the form

$$s_i \xrightarrow{A} s_j$$

represent the fact that the state s_i can be transformed into the state s_j by the robot executing action A.

Imagine a robot has to stack three different objects, e.g., a cup, a saucer, and a set. In the initial state, all objects are placed on a table surface. The robot can pick an object (provided that there is no other object on top of it) and can place it either on the surface or on top of another object (provided, again, that there is nothing already on top of it). If we further assume that each object can be placed on all other objects, and that only a single

object can be moved at a time (and not a stack of two of them), then we get the state space depicted in Fig. 14.8.

The state-transition model abstracts away aspects of robot activity such as the structure of actions and the events, behavior, and effects that are caused during action execution. Actions are considered to cause atomic state transitions.

Even with these abstractions, a number of variations of planning problems are investigated. These variations result from different trade-offs one can make with respect to the computational complexity of the planning problem, the strength of assumptions one makes about the capabilities of the robot, and the realism with which one wants to represent the behavior of the robot and the effects of its actions.

An important subclass of state-transition models are the ones that characterize action by specifying their preconditions and effects [14.25, 109, 110]. Preconditions state the conditions under which the respective action is executable and has the specified effects. The effects represent how the world changes when the action is executed.

If these models are stated as axioms in a logical representation language (Sect. 14.2) planning methods can be particularly powerful: they can compute sequences or partially ordered sets of actions that are provably working plans from the logical point of view. This means that planning algorithms can find plans that are guaranteed to transform any state satisfying a given state description into a state that satisfies a given goal, if such a plan exists. Unfortunately, in many robot applications action models that are axiomatized this way are too unrealistic.

Many extensions of the state-transition model deal with providing the expressive power needed to state action models more realistically. The first one extends the state-transition model by allowing state transitions to be caused by a combination of an action and an event ($\gamma: S \times A \times E \rightarrow S$) rather than the action alone. This modification allows us to represent action failures by stating that a failure event occurs while the action is executed. It also allows us to approximate environments that are dynamically changing. The second extension is to change the state transition function such that it maps into a subset of states rather than a single state ($\gamma: S \times A \rightarrow 2^S$). This modification allows the robot to reason through the nondeterminism that can often not be avoided when autonomous robots perform realis-

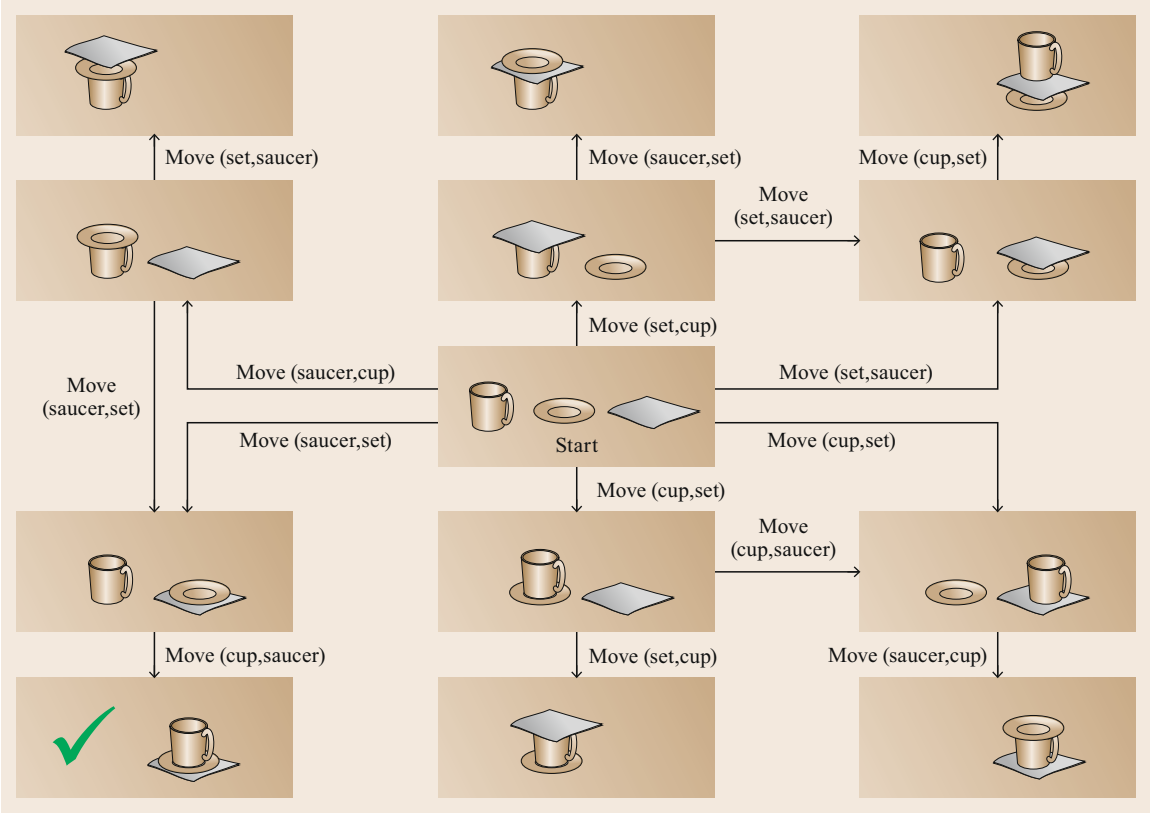


Fig. 14.8 Example: problem space for stacking three objects

tic activities. Other researchers extend models used for plan-based control by including aspects that allow the robot to reason about its resources, geometric aspects of manipulation actions [14.24, 111], sensing actions, and so on.

Besides the state-transition model researchers also use hybrid system models to reason about robot activities [14.112–114]. In the hybrid system model the action repertoires of robots are specified as continuous variable dynamical systems with a phased operation. Within each phase, called control mode, the system evolves continuously according to the dynamical law of that mode, called continuous flow. Thus, the state of the hybrid system can be thought of as a pair – the control mode and the continuous state. The control mode identifies a flow, and the continuous flow identifies a position in it. Also associated with each control mode are so-called jump conditions, specifying the conditions that the discrete state and the continuous state together must satisfy to enable a transition to another control mode. The transitions can cause abrupt changes of the discrete as well as the continuous state. The jump relation specifies the valid settings of the system variables that might occur during a jump. Then, until the next transition, the continuous state evolves according to the flow identified by the new control mode.

The advantage of a hybrid-system-based conceptualization over state-based ones is that hybrid systems can adequately model concurrent processes with interfering continuous effects. They also allow for discrete changes in process parameterization, which we need to model the activation, deactivation, and reparameterization of control processes through concurrent reactive plans (Sect. 14.4.2). In addition, hybrid system based conceptualizations can model the procedural meaning of waiting for and reacting to asynchronous events.

While the increased expressive power allows us to model the behavior that today's autonomous manipulation robots generate much more faithfully, the models are also in most cases too complex to completely synthesize action plans from primitive actions. Many of the modelling issues to be addressed in the intersection between action planning and control are discussed in [14.115, 116].

14.4.2 Representation of Plans

When the robot is performing its activity, it is under the control of a plan. The most common forms of action plans in autonomous robot control are:

- State-transition plans, totally or partially ordered sets of atomic actions.
- Policies, mappings from (perceived) environment states to actions.
- Reactive plans that specify how the robot is to respond to sensory events and execution failures in order to accomplish its jobs.

State-Transition Plans

Robot action plans that are generated according to the state-transition model are ordered sets of actions, often called plan steps. The assumptions of the state-transition model, in particular that plan steps are atomic and their execution can be abstracted away, are not satisfied in many robot applications. To execute plan steps successfully, actions have to include subactions such as detecting the object to be fetched in a scene or detecting that the object has been successfully grasped before lifting the arm. As autonomous robot activity is far from perfect, the robot might not be able to generate the necessary execution events or it can generate the events but fails to achieve the desired effects. For example, an object might slip out of the robot's hand after grasping it.

As a consequence, state-transition plans can often only serve as guidelines for generating robust and flexible robot behavior [14.117]. To close the gap to low-level robot control, researchers have proposed multilayer software architectures for autonomous robot activity, in particular 3T (3-tiered) architectures [14.118] (Chap. 12). 3T architectures run planning and execution at different software layers and different time scales where a sequencing layer synchronizes between both layers. Each layer uses a different form of plan or behavior specification language. The planning layer typically uses a problem space plan, whereas the execution layer employs feedback control routines that can be activated and deactivated. The intermediate layer typically uses a simple form of a reactive plan language (see below). Even though 3T architectures have been shown to produce competent activity in some application domains, there remain important issues to be addressed for more complex applications. One of them is that the behavior generated by the sequencing layer might deviate substantially from the action model used for planning, which might cause plans not to work as expected by the action models used by the planning component. As the planning layer abstracts away from control structures, it also lacks the expressiveness to specify robust and flexible behavior.

MDP Policies

A second category of plans are policies, which are mappings from discretized robot states into robot actions. Such policies are computed as solutions of Markov decision problems (Sect. 14.2.2). The policies computed

by MDPs aim at robustness and optimizing the average performance. They are less appropriate when the consequences of decisions are not in the near future. The decision of how to grasp an object might considerably affect the ease with which the object can be placed later in the activity episode. Extensions of MDP policies such as the combination of programming and reinforcement learning [14.119, 120] and the structuring of policies with so-called options [14.121, 122] partly address these issues.

Concurrent Reactive Plan Languages

Concurrent reactive plan languages are the result of applying programming language design methods to the specification of robot behavior. Concurrent reactive plans specify how the robot is to respond to events, including perception and failure events, in order to achieve its goals [14.123]. Such plan languages offer rich sets of control structures that support the structuring and transparent synchronization and monitoring of concurrent control processes, the detection and recovery from execution failures, the context-dependent execution of subplans, the prompt reaction to asynchronous events, and data structures for the descriptions of objects to be manipulated [14.124, 125]. Some reactive plan languages also incorporate explicit representations of the beliefs, goals, and intentions of the robot [14.126–128] and even provide powerful reasoning mechanisms to adapt the plan execution to the respective situation context [14.129].

While concurrent reactive plan languages enable robots to accomplish their tasks competently, they are too expressive for automatically generating robust, flexible high-performance plans from first principles. They typically require the use of plan libraries and transformational planning techniques to deal with the computational complexity caused by the expressiveness of the plan languages (Sect. 14.4.4).

14.4.3 Generating Robot Action Plans

The most common form of reasoning performed on plans is the synthesis of plans for given problems. In general, planning is concerned with reasoning about intended courses of action before executing them in order to avoid failures in achieving given goals, to prevent undesired side effects, or to improve the performance of the robot in accomplishing its goals.

There are many different variants of planning problems, categories of plans that can be generated, and methods that can be applied to planning problems. Comprehensive overviews can be found in textbooks [14.4, 27] and overview articles [14.130, 131]. A wide range of the planning problems is addressed within an international AI planning competition (IPC) [14.132] using standardized problem specification languages [14.109, 110]. Several implementations of planning algorithms participating in this competition are available as open-source software.

In planning for state-transition, the task of the planning system is to find an ordered set of actions that transform any state satisfying the initial state description into a state satisfying the given goals.

The first class of planning systems are systems that enumerate the states that can be reached by prefixes (or suffixes) of hypothetical action sequences until a goal state is reached. Planning systems that fall into this category include the STRIPS planning algorithm [14.25] and newer and more efficient algorithms such as the FF (fast forward) planning algorithm [14.133] as well as various variants of these algorithms.

Another class of algorithms enumerates abstract plans that represent sets of plans. The idea underlying this approach is that by reasoning about an abstract plan, the planning algorithm can compactly reason about different plan instances at the same time by reasoning about their common causal structure. For example, a partial order plan represents all possible completions of the plan. The POP (partial-order planning) algorithms is a family of algorithms that is primarily used for teaching action planning [14.130] while GraphPlan [14.134] is an algorithm that is designed for planning process efficiency [14.131].

The algorithms above are domain independent: they compute plans by analyzing whether preconditions of plan steps are achieved through the effects of prior actions using general algorithms. In contrast, domain-dependent planners use domain knowledge to constrain and prune the search space to be explored in a planning process. An example of such domain knowledge is that stacks of objects are built bottom up. Domain-dependent planners are often much more efficient than their domain-independent counterparts. Examples of domain-dependent planners are the TL- (temporal logic) [14.135] and the TAL planners [14.19].

Another category of planners that are designed to deal with more complex planning problems are HTN (hierarchical task network) planners. HTN planners are equipped with libraries of plan schemata that specify how abstract plan steps can be decomposed into more complex subplans. Examples of such planners are the SHOP planners [14.136] and extensions thereof that are combined with description logics and applied to autonomous robot control [14.69].

The planning approaches above generate plans that have similar expressiveness in terms of the behavior they can specify. The planners mainly differ with re-

spect to the size of the planning problems they can handle and their efficiency in solving planning problems.

14.4.4 Revising and Improving Plans

Another dimension that robot action planning methods can be scaled along is the expressiveness of the plans they generate in terms of specifying competent robot behavior. A subclass of planning methods that is suitable for generating concurrent reactive plans is *transformational planning*.

The underlying idea of transformational planning is to paste together plans from libraries of plan schemata and revise them for the specified task [14.137] rather than creating completely new plans from scratch. Transformational planning has also been referred to as *plan repair* or *plan debugging* [14.138, 139] and is often performed within a *generate-test-debug* control strategy [14.140]. Plan transformation is a promising approach in applications where default plans that have a substantial chance of being successful can be constructed easily, tested by projecting them, analyzed with respect to their expected execution failures, and where failures can be used to index the appropriate plan revisions.

Transformational planning subsumes classical planning methods such as partial-order planning in the sense that the operations of the POP algorithms can be simply expressed as transformations. In addition to the transformations that correspond to classical planning, transformational planning typically employs collections of transformations including domain-specific ones, heuristic transformations, and transformations for plan optimization [14.141]. In cases where the planners employ transformations that cannot be guaranteed to work, they rely on powerful plan projection methods for testing the generated plan hypotheses [14.142].

Transformational planning is a promising approach for autonomous robots performing complex manipulation tasks because it makes it easy for programmers to specify how plans should be modified to avoid certain expected behavior flaws. For example, a transformation could tell the robot to stack items in order to transport them more efficiently subject to additional constraints such as that it should not stack too many items because it might drop them. Or, it might stack one kind of object on another one but not vice versa – typically it is safe to stack cups on plates but not plates on cups.

Transformational planning can also effectively facilitate the learning not only of plans [14.139] but also of planning capabilities by learning for a particular robot and environment what the robot can stack and carry and what not.

14.4.5 Other Reasoning Methods

There are a number of other reasoning methods that plan-based robot control needs to employ in order to produce competent object manipulation capabilities [14.143]. One of the main reasons is that the tasks and subtasks that a robot is to perform are typically underspecified: the robot might be asked to fetch the screwdriver from the table but when it gets there are several different screwdrivers. Or, the robot might be asked to crack an egg and has to decide where to crack it, where to put the yolk and egg white, what to do with the shell, etc. When fetching objects, it has to reason about abstract states such as visibility and reachability of objects. We can only give a couple of examples here.

The first reasoning problem is that of *object identity resolution*, the inference task of deciding whether two internal representations refer to the same object in the real world. When receiving the command to pick up the red cup from the table, the robot must make at some point the decision that a data structure returned by the robot's perception system and the language description refer to the same object. Only by making this inference, the robot is able to infer that it can achieve the given task by picking up a particular object that it sees. In logic-based representations this problem is even harder because symbols used in the planning problem specification *denote* objects in the outside world. In this setting the problem of object identity resolution problem becomes the *symbol grounding problem* [14.1], i. e., the problem to determine *the right* or, at least, *the most likely* object in a given environment for each constant and each definite description in the logical representation. The most powerful inference methods to reason about object identity resolution are first-order probabilistic reasoning mechanisms as described in [14.144].

Other important inference methods are needed to model some predicates used in the activity models more realistically. Predicates that deserve such special treatment include *visibility* and *reachability*. For example, the Asymov planner [14.111] reasons about action preconditions that include reachability conditions and tests reachability through the existence of motion plans. Other planning systems reason about the next best views that a robot should take in order to perform complex perception tasks. Sophisticated action planners that include action and motion planning employing more general geometric reasoning capabilities can be found in [14.66]. Cognition-enabled plans employ such reasoning methods in order to parameterize actions with respect to the effects that should be achieved and avoided [14.62].

14.5 Conclusions and Further Reading

The aim of this chapter was, first, to motivate the purpose and benefit of employing general reasoning methods in robot control; second, to sketch some of the generic reasoning methods from AI that are well-understood, available, and most likely of use on a robot; and third, to go into plan-based robot control in some more detail, as it is a particularly obvious form of employing reasoning to the end of improving robot action.

Yet, the chapter has only sketched the tip of an iceberg. Employing symbolic reasoning in an autonomous mobile robot requires to have and update on the robot a symbolic model of the world, or at least of that part of the world which is relevant for the current task – but who knows beforehand exactly what is and what isn't relevant for some task? It therefore requires to be able to interpret the current sensor data stream on a semantic level to transform it into a symbolic description in terms of categories that are meaningful in the KR. Assuming that it is not practical to equip a robot with all knowledge that may likely be relevant for each and every situation it may find itself in, it requires some form of learning, generalization, and/or the ability to look up unknown knowledge from available sources. These and some more requirements that are needed for practically using AI reasoning methods on a robot cannot be fulfilled completely based on current state-of-the-art AI methods and technology; as a consequence, currently existing robot prototypes employing AI reasoning methods in their control systems are limited in terms of generality and adaptability.

It is no recent insight that employing AI reasoning methods on robots is both a chance and a challenge. In fact some of the early influential work in AI was driven by the idea of designing integrated *reasoning robots*, such as SHAKEY [14.145] (VIDEO 704); as a consequence, early AI textbooks like [14.146] would naturally include robotics as a target field for using AI reasoning methods. The field of AI has since focused on applications other than Robotics. Recent efforts in integrating reasoning systems with autonomous robots have done much to bring the original robot-inspired vision back into focus, as witnessed by series of workshops on *AI-Based Robotics* (e.g., IROS 2013) or *AI and Robotics* (AAAI 2014). It is exactly the family of challenges and chances that this chapter is about that is discussed at these events.

14.5.1 Further Reading

For a more complete coverage of the generic methods described here, as well as the rest of AI, we refer to AI textbooks, of which the one by *Russell* and *Norvig* [14.4] is comprehensive and currently most widely used. Among the sources for first-hand material, let us mention the two journals *Artificial Intelligence* and *Journal of Artificial Intelligence Research* (JAIR) [14.147], which both cover well the topics addressed here. The main international conference in the field is IJCAI (*International Joint Conference on Artificial Intelligence*); ECAI (*European Conference on Artificial Intelligence* [14.148]) and AAAI (*Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence* [14.149]) are other major international symposia on Artificial Intelligence in general. Regarding planning in particular, ICAPS (*International Conference on Automated Planning and Scheduling* [14.150]) is the main international conference.

Reference [14.151] reviews about the same topic as this chapter, but structures it differently along the notion of *deliberation functions*.

Video-References

VIDEO 704 SHAKEY: Experimentation in robot learning and planning (1969)
available from http://handbookofrobotics.org/view-chapter/14/videodetails/704

VIDEO 705 From knowledge grounding to dialogue processing
available from http://handbookofrobotics.org/view-chapter/14/videodetails/705

VIDEO 706 RoboEarth final demonstrator
available from http://handbookofrobotics.org/view-chapter/14/videodetails/706

References

14.1 S. Harnad: The symbol grounding problem, Physica D **42**, 335–346 (1990)

14.2 S. Coradeschi, A. Saffiotti: An introduction to the anchoring problem, Robotics Auton. Syst. **43**(2/3), 85–96 (2003)

14.3 F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider (Eds.): *The Description Logic Handbook* (Cambridge Univ. Press, Cambridge 2003)

14.4 S.J. Russell, P. Norvig: *Artificial Intelligence: A Modern Approach*, 3rd edn. (Pearson Education, Upper Saddle River 2010)

14.5 R.J. Brachman, H.J. Levesque: *Knowledge Representation and Reasoning* (Morgan Kaufmann, San Francisco 2004)

14.6 W.O. van Quine: *Methods of Logic*, 4th edn. (Harvard Univ. Press, Cambridge 1982)

14.7 Z. Manna, R. Waldinger: *The Deductive Foundations of Computer Programming: A One-Volume Version of "The Logical Basis for Computer Programming"* (Addison-Wesley, Reading 1993)

14.8 W. Hodges: Elementary predicate logic. In: *Handbook of Philosophical Logic*, Vol. 1, ed. by D. Gabbay, F. Guenthner (D. Reidel, Dordrecht 1983)

14.9 A. Robinson, A. Voronkov (Eds.): *Handbook of Automated Reasoning* (Elsevier, Amsterdam 2001)

14.10 M. Davis, G. Logemann, D. Loveland: A machine program for theorem proving, Communications ACM **5**(7), 394–397 (1962)

14.11 The international SAT Competitions web page: http://www.satcompetition.org/

14.12 The Web Ontology Language OWL: http://www.w3.org/TR/owl-features/

14.13 OWL 2 Web Ontology Language Document Overview (Second Edition): http://www.w3.org/TR/owl2-overview/

14.14 P. Hitzler, M. Krötzsch, S. Rudolph: *Foundations of Semantic Web Technologies* (Chapman Hall/CRC, Boca Raton 2009)

14.15 J. McCarthy, P. Hayes: Some philosophical problems from the standpoint of artificial intelligence, Mach. Intell. **4**, 463–507 (1969)

14.16 H. Levesque, R. Reiter, Y. Lespérance, F. Lin, R. Scherl: Golog: A logic programming language for dynamic domains, J. Log. Program. **31**, 59–83 (1997)

14.17 M. Shanahan, M. Witkowski: High-level robot control through logic, ATAL '00: Proc. 7th Int. Workshop Intell. Agents VII. Agent Theor. Archit. Lang. (2001) pp. 104–121

14.18 M. Thielscher: *Reasoning Robots. The Art and Science of Programming Robotic Agents* (Springer, Berlin 2005)

14.19 P. Doherty, J. Kvarnström: TALplanner: A temporal logic based planner, AI Magazine **22**(3), 95–102 (2001)

14.20 K.L. Chung, F. AitSahila: *Elementary Probability Theory*, 4th edn. (Springer, Berlin, Heidelberg 2003)

14.21 J. Pearl: *Probabilistic Reasoning in Intelligent Systems* (Morgan Kaufmann, San Mateo 1988)

14.22 A.R. Cassandra, L.P. Kaelbling, M.L. Littman: *Acting Optimally in Partially Observable Stochastic Domains*, Tech. Rep. AAAI-94 (Department of Computer Science, Brown University, Providence 1994) pp. 1023–1028

14.23 R.E. Bellman: *Dynamic Programming* (Princeton Univ. Press, Princeton 1957)

14.24 L.P. Kaelbling, T. Lozano-Pérez: Integrated task and motion planning in belief space, Int. J. Robotics Res. **32**(9/10), 1194–1227 (2013)

14.25 R.E. Fikes, N.J. Nilsson: Strips: A new approach to theorem proving in problem solving, J. Artif. Intell. **2**, 189–208 (1971)

14.26 P.E. Hart, N.J. Nilsson, B. Raphael: A formal basis for the heuristic determination of minimum cost paths, IEEE Trans. Syst. Sci. Cybern. **4**(2), 100–107 (1968)

14.27 M. Ghallab, D.S. Nau, P. Traverso: *Automated Planning – Theory and Practice* (Elsevier, Amsterdam 2004)

14.28 A. Pnueli: The temporal logic of programs, Proc. 18th Annu. Symp. Found. Comput. Sci., Providence (1977) pp. 46–57

14.29 O. Kupferman, M.Y. Vardi: Model checking of safety properties, Form. Methods Syst. Des. **19**(3), 291–314 (2001)

14.30 E. Plaku, L.E. Kavraki, M.Y. Vardi: Hybrid systems: From verification to falsification by combining motion planning and discrete search, Form. Methods Syst. Des. **34**, 157–182 (2009)

14.31 A. Bhatia, M.R. Maly, L.E. Kavraki, M.Y. Vardi: Motion planning with complex goals, IEEE Robotics Autom. Mag. **18**(3), 55–64 (2011)

14.32 M. Vilain, H. Kautz, P. van Beek: Constraint propagation algorithms for temporal reasoning: A revised report. In: *Readings in Qualitative Reasoning About Physical Systems*, ed. by D.S. Weld, J. de Kleer (Morgan Kaufmann, San Francisco 1990) pp. 373–381

14.33 J. Allen: Towards a general theory of action and time, Artif. Intell. **23**(2), 123–154 (1984)

14.34 E.P.K. Tsang: *Foundations of Constraint Satisfaction* (Academic Press, London, San Diego 1993)

14.35 P. Jonsson, A. Krokhin: Complexity classification in qualitative temporal constraint reasoning, Artif. Intell. **160**(1/2), 35–51 (2004)

14.36 G. Ligozat: A new proof of tractability for ORD-horn relations, AAAI Workshop Spat. Temp. Reason., Portland (1996)

14.37 U. Montanari: Networks of constraints: Fundamental properties and applications to picture processing, Inf. Sci. **7**, 95–132 (1974)

14.38 R.W. Floyd: Algorithm 97: Shortest path, Communications ACM **5**(6), 345 (1962)

14.39 L. Xu, B.Y. Choueiry: A new efficient algorithm for solving the simple temporal problem, Proc. 4th Int. Conf. Temp. Log., Cairns (2003)

14.40 L.R. Planken, M.M. De Weerdt, R.P.J. van der Krogt: P3C: A new algorithm for the simple temporal problem, Proc. Int. Conf. Autom. Plan. Sched. (ICAPS), Sydney (2008) pp. 256–263

14.41 C. McGann, F. Py, K. Rajan, J. Ryan, R. Henthorn: Adaptive control for autonomous underwater vehicles, Proc. 23rd Natl. Conf. Artif. Intell., Chicago (2008) pp. 1319–1324

14.42 B.C. Williams, M.D. Ingham, S.H. Chung, P.H. Elliott: Model-based programming of intelligent embedded systems and robotic space explorers, Proceedings IEEE **91**(1), 212–237 (2003)

14.43 T. Vidal, H. Fargier: Handling contingency in temporal constraint networks: From consistency to controllabilities, J. Exp. Theor. Artif. Intell. **11**, 23–45 (1999)

14.44 P. Doherty, J. Kvarnström, F. Heintz: A temporal logic-based planning and execution monitoring framework for unmanned aircraft systems, J. Auton. Agents Multi-Agent Syst. **2**(2), 332–377 (2009)

14.45 F. Pecora, M. Cirillo, F. Dell'Osa, J. Ullberg, A. Saffiotti: A constraint-based approach for proactive, context-aware human support, J. Ambient Intell. Smart Environ. **4**(4), 347–367 (2012)

14.46 R. Dechter: *Constraint Processing*, The Morgan Kaufmann Series in Artificial Intelligence (Morgan Kaufmann, San Francisco 2003) pp. 155–165

14.47 A. Loutfi, S. Coradeschi, M. Daoutis, J. Melchert: Using knowledge representation for perceptual anchoring in a robotic system, Int. J. Artif. Intell. Tools **17**(5), 925–944 (2008)

14.48 O. Colliot, O. Camara, I. Bloch: Integration of fuzzy spatial relations in deformable models – Application to brain MRI segmentation, Pattern Recogn. **39**(8), 1401–1414 (2006)

14.49 X. Wang, J.M. Keller, P. Gader: Using spatial relationships as features in object recognition, Annu. Meet. North Am. Fuzzy Inf. Proces. Soc., Syracuse (1997)

14.50 D.A. Randell, Z. Cui, A.G. Cohn: A Spatial Logic based on Regions and Connection, Proc. Int. Conf. Princ. Knowl. Represent. Reason., Cambridge (1992)

14.51 S. Skiadopoulos, M. Koubarakis: Composing cardinal direction relations, Artif. Intell. **152**(2), 143–171 (2004)

14.52 J. Renz, B. Nebel: Qualitative spatial reasoning using constraint calculi. In: *Handbook of Spatial Logics*, ed. by M. Aiello, I. Pratt-Hartmann, J.F.A.K. van Benthem (Springer, Berlin, Heidelberg 2007) pp. 161–215

14.53 T. Drakengren, P. Jonsson: A complete classification of tractability in Allen's algebra relative to subsets of basic relations, Artif. Intell. **106**(2), 205–219 (1998)

14.54 A.G. Cohn, J. Renz, M. Sridhar: Thinking inside the box: A comprehensive spatial representation for video analysis, Proc. 13th Int. Conf. Princ. Knowl. Represent. Reason., Rome (2012) pp. 588–592

14.55 P. Balbiani, J.-F. Condotta, L. Farinas Del Cerro: A new tractable subclass of the rectangle algebra, Proc. 16th Int. Jt. Conf. Artif. Intell., Stockholm (1999) pp. 442–447

14.56 M. Mansouri, F. Pecora: A representation for spatial reasoning in robotic planning, Proc. IROS Workshop AI-Based Robotics, Tokyo (2013)

14.57 M. Mansouri, F. Pecora: More knowledge on the table: Planning with space, time and resources for robots, IEEE Int. Conf. Robotics Autom. (ICRA), Hong Kong (2014) pp. 647–654

14.58 M. Skubic, D. Perzanowski, S. Blisard, A. Schultz, W. Adams, M. Bugajska, D. Brock: Spatial language for human-robot dialogs, IEEE Trans. Syst. Man Cybern. C **34**(2), 154–167 (2004)

14.59 R. Moratz, T. Tenbrink: Spatial reference in linguistic human-robot interaction: Iterative, empirically supported development of a model of projective relations, Spat. Cogn. Comput. **6**(1), 63–107 (2006)

14.60 S. Guadarrama, L. Riano, D. Golland, D. Gouhring, Y. Jia, D. Klein, P. Abbeel, T. Darrell: Grounding spatial relations for human-robot interaction, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Tokyo (2013) pp. 1640–1647

14.61 L. Kunze, K.K. Doreswamy, N. Hawes: Using qualitative spatial relations for indirect object search, IEEE Int. Conf. Robotics Autom. (ICRA), Hong Kong (2014) pp. 163–168

14.62 L. Mosenlechner, M. Beetz: Parameterizing actions to have the appropriate effects, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., San Francisco (2011) pp. 4141–4147

14.63 A. Gaschler, R.P.A. Petrick, M. Giuliani, M. Rickert, A. Knoll: KVP: A knowledge of volumes approach to robot task planning, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Tokyo (2013) pp. 202–208

14.64 G. Havur, K. Haspalamutgil, C. Palaz, E. Erdem, V. Patoglu: A case study on the Tower of Hanoi challenge: Representation, reasoning and execution, IEEE Int. Conf. Robotics Autom. (ICRA), Tokyo (2013) pp. 4552–4559

14.65 L. de Silva, A.K. Pandey, R. Alami: An interface for interleaved symbolic-geometric planning and backtracking, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Tokyo (2013) pp. 232–239

14.66 L.P. Kaelbling, T. Lozano-Pérez: Hierarchical task and motion planning in the now, IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 1470–1477

14.67 F. Lagriffoul, D. Dimitrov, A. Saffiotti, L. Karlsson: Constraint propagation on interval bounds for dealing with geometric backtracking, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Vilamoura (2012) pp. 957–964

14.68 G. de Giacomo, L. Iocchi, D. Nardi, R. Rosati: Moving a robot: the KR&R approach at work, Proc. 5th Int. Conf. Princ. Knowl. Represent. Reason. (KR'96), Cambridge (1996) pp. 198–209

14.69 R. Hartanto, J. Hertzberg: Fusing DL reasoning with HTN planning, Lect. Notes Comput. Sci. **5243**, 62–69 (2008)

14.70 C. Galindo, J.A. Fernandez-Madrigal, J. Gonzalez, A. Saffiotti: Using semantic information for improving efficiency of robot task planning, Proc.

ICRA-07 Workshop Semant. Inf. Robotics, Rome (2007) pp. 27–32
14.71 W. Cushing, S. Kambhampati, Mausam, D.S. Weld: When is temporal planning really temporal?, Proc. 20th Int. Jt. Conf. Artif. Intell., Hyderabad (2007)
14.72 J.L. Bresina, A.K. Jónsson, P.H. Morris, K. Rajan: Activity planning for the Mars exploration rovers, Proc. 15th Int. Conf. Autom. Plan. Sched. (ICAPS), Monterey (2005) pp. 1852–1859
14.73 M. Cirillo, F. Pecora, H. Andreasson, T. Uras, S. Koenig: Integrated motion planning and coordination for industrial vehicles, Proc. 24th Int. Conf. Autom. Plan. Sched. (ICAPS), Portsmouth (2014)
14.74 S. Fratini, F. Pecora, A. Cesta: Unifying planning and scheduling as timelines in a component-based perspective, Arch. Control Sci. **18**(2), 231–271 (2008)
14.75 M. Ghallab, H. Laruelle: Representation and control in IxTeT, a temporal planner, Proc. 2nd Int. Conf. Artif. Intell. Plan. Syst. (AIPS-94), Chicago (1994) pp. 61–67
14.76 P. Gregory, D. Long, M. Fox, J.C. Beck: Planning modulo theories: Extending the planning paradigm, Proc. 15th Int. Conf. Autom. Plan. Sched. (ICAPS), São Paulo (2012)
14.77 R. Nieuwenhuis, A. Oliveras, C. Tinelli: Solving SAT and SAT modulo theories: From an abstract Davis–Putnam–Logemann–Loveland procedure to DPLL(T), Journal ACM **53**, 937–977 (2006)
14.78 S. Nedunuri, S. Prabhu, M. Moll, S. Chaudhuri, L.E. Kavraki: SMT-based synthesis of integrated task and motion plans for mobile manipulation, IEEE Int. Conf. Robotics Autom. (ICRA), Hong Kong (2014)
14.79 U. Köckemann, L. Karlsson, F. Pecora: Grandpa hates robots – Interaction constraints for planning in inhabited environments, Proc. 28th Conf. Artif. Intell., Quebéc City (2014)
14.80 M. Di Rocco, F. Pecora, A. Saffiotti: When robots are late: Configuration planning for multiple robots with dynamic goals, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Tokyo (2013)
14.81 G.H. Lim, I.H. Suh, H. Suh: Ontology-based unified robot knowledge for service robots in indoor environments, IEEE Trans. Syst. Man Cybern. A **41**(3), 492–509 (2011)
14.82 S. Lemaignan, R. Ros, L. Mösenlechner, R. Alami, M. Beetz: ORO, a knowledge management module for cognitive architectures in robotics, Proc. 2010 IEEE/RSJ Int. Conf. Intell. Robots Syst., Taipei (2010) pp. 3548–3553
14.83 M. Tenorth, M. Beetz: KnowRob – A knowledge processing infrastructure for cognition-enabled robots, Int. J. Robotics Res. **32**(5), 566–590 (2013)
14.84 M. Daoutis, S. Coradeschi, A. Loutfi: Grounding commonsense knowledge in intelligent systems, J. Ambient Intell. Smart Environ. **1**(4), 311–321 (2009)
14.85 A. Saffiotti, M. Broxvall, M. Gritti, K. LeBlanc, R. Lundh, J. Rashid, B.S. Seo, Y.J. Cho: The PEIS-Ecology project: Vision and results, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Nice (2008) pp. 2329–2335
14.86 X. Chen, J. Ji, J. Jiang, G. Jin, F. Wang, J. Xie: Developing high-level cognitive functions for service robots, Proc. 9th Int. Conf. Auton. Agents Multiagent Syst., Toronto (2010) pp. 989–996
14.87 J.F. Lehman, J.E. Laird, P. Rosenbloom: A gentle introduction to Soar, an architecture for human cognition, Invit. Cogn. Sci. **4**, 212–249 (1996)
14.88 N. Derbinsky, J.E. Laird: Extending soar with dissociated symbolic memories, Symp. Human Mem. Artif. Agents, AISB (2010) pp. 31–37, http://ai.eecs.umich.edu/soar/sitemaker/docs/pubs/aisb2010_rwwa_soar9.pdf
14.89 W.G. Kennedy, M. Rouleau, J.K. Bassett: Multiple levels of cognitive modeling within agent-based modeling, Proc. 18th Conf. Behav. Represent. Model. Simul., Sundance (2009) pp. 143–144
14.90 R.B. Rusu, Z.C. Marton, N. Blodow, M. Dolha, M. Beetz: Towards 3D point cloud based object maps for household environments, Robotics Auton. Syst. J. Semant. Knowl. Robotics **56**(11), 927–941 (2008)
14.91 S. Vasudevan, R. Siegwart: Bayesian space conceptualization and place classification for semantic maps in mobile robotics, Robotics Auton. Syst. **56**(6), 522–537 (2008)
14.92 H. Zender, O. Martinez Mozos, P. Jensfelt, G.J.M. Kruijff, W. Burgard: Conceptual spatial representations for indoor mobile robots, Robotics Auton. Syst. **56**(6), 493–502 (2008)
14.93 B. Limketkai, L. Liao, D. Fox: Relational object maps for mobile robots, Proc. Int. Jt. Conf. Artif. Intell. (IJCAI) (2005) pp. 1471–1476
14.94 M. Tenorth, L. Kunze, D. Jain, M. Beetz: KNOWROB-MAP – Knowledge-linked semantic object maps, 10th IEEE-RAS Int. Conf. Humanoid Robots, Nashville (2010) pp. 430–435
14.95 N. Mavridis, D. Roy: Grounded situation models for robots: Where words and percepts meet, Proc. 2006 IEEE/RSJ Int. Conf. Intell. Robots Syst., Beijing (2006) pp. 4690–4697
14.96 D.K. Misra, J. Sung, K. Lee, A. Saxena: Tell me Dave: Context-sensitive grounding of natural language to mobile manipulation instructions, Proc. Robotics Sci. Syst. (RSS) (2014)
14.97 T. Kollar, S. Tellex, D. Roy, N. Roy: Toward understanding natural language directions, Proc. 5th AMC/IEEE Int. Conf. Hum.-Robot Interact. (HRI), Osaka (2010) pp. 259–266
14.98 C. Matuszek, E. Herbst, L. Zettlemoyer, D. Fox: Learning to parse natural language commands to a robot control system, Proc. 13th Int. Symp. Exp. Robotics (ISER) Québec City (2012) pp. 403–415
14.99 F. Duvallet, T. Kollar, A. Stentz: Imitation learning for natural language direction following through unknown environments, 2013 IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 1047–1053
14.100 K. Zhou, M. Zillich, H. Zender, M. Vincze: Web mining driven object locality knowledge acquisition for efficient robot behavior, IEEE/RSJ Int.

Conf. Intell. Robots Syst. (IROS), Vilamoura (2012) pp. 3962–3969
14.101 M. Tenorth, D. Nyga, M. Beetz: Understanding and executing instructions for everyday manipulation tasks from the World Wide Web, IEEE Int. Conf. Robotics Autom. (ICRA), Anchorage (2010) pp. 1486–1491
14.102 M. Tenorth, U. Klank, D. Pangercic, M. Beetz: Web-enabled robots – Robots that use the web as an information resource, Robotics Autom. Mag. **18**(2), 58–68 (2011)
14.103 M. Waibel, M. Beetz, R. D'Andrea, R. Janssen, M. Tenorth, J. Civera, J. Elfring, D. Gálvez-López, K. Häussermann, J.M.M. Montiel, A. Perzylo, B. Schießle, O. Zweigle, R. van de Molengraft: RoboEarth – A world wide web for robots, Robotics Autom. Mag. **18**(2), 69–82 (2011)
14.104 S. Osentoski, B. Pitzer, C. Crick, G. Jay, S. Dong, D.H. Grollman, H.B. Suay, O.C. Jenkins: Remote robotic laboratories for learning from demonstration – Enabling user interaction and shared experimentation, Int. J. Soc. Robotics **4**(4), 449–461 (2012)
14.105 M.B. Blake, S.L. Remy, Y. Wei, A.M. Howard: Robots on the web, IEEE Robotics Autom. Mag. **18**, 33–43 (2011)
14.106 D. Hunziker, M. Gajamohan, M. Waibel, R. D'Andrea: Rapyuta: The RoboEarth cloud engine, IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 438–444
14.107 D. McDermott: Robot planning, AI Magazine **13**(2), 55–79 (1992)
14.108 M.E. Pollack, J.F. Horty: There's more to life than making plans: Plan management in dynamic, multiagent environments, AI Magazine **20**(4), 71–83 (1999)
14.109 D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, D. Wilkins: *PDDL – The Planning Domain Definition Language*, Tech. Rep. CVC TR-98-003/DCS TR-1165 (Yale Center for Computational Vision and Control, New Haven 1998)
14.110 M. Fox, D. Long: PDDL2.1: An extension of PDDL for expressing temporal planning domains, J. Artif. Intell. Res. **20**, 61–124 (2003)
14.111 F. Gravot, S. Cambon, R. Alami: aSyMov: A planner that deals with intricate symbolic and geometric problems, Springer Tracts Adv. Robotics **15**, 100–110 (2005)
14.112 R. Alur, T. Henzinger, H. Wong-Toi: Symbolic analysis of hybrid systems, Proc. 37th IEEE Conf. Decis. Control, Tampa (1997) pp. 702–707
14.113 R. Alur, T. Henzinger, P. Ho: Automatic symbolic verification of embedded systems, IEEE Trans. Softw. Eng. **22**, 181–201 (1996)
14.114 M. Beetz, H. Grosskreutz: Probabilistic hybrid action models for predicting concurrent percept-driven robot behavior, J. Artif. Intell. Res. **24**, 799–849 (2005)
14.115 K. Passino, P. Antsaklis: A system and control-theoretic perspective on artificial intelligence planning systems, Appl. Artif. Intell. **3**, 1–32 (1989)
14.116 T. Dean, M. Wellmann: *Planning and Control* (Morgan Kaufmann Publishers, San Mateo 1991)
14.117 R. Alami, R. Chatila, S. Fleury, M. Ghallab, F. Ingrand: An architecture for autonomy, Int. J. Robotics Res. **17**(4), 315–337 (1998)
14.118 R.P. Bonasso, R.J. Firby, E. Gat, D. Kortenkamp, D.P. Miller, M.G. Slack: Experiences with an architecture for intelligent, reactive agents, J. Exp. Theor. Artif. Intell. **9**(2/3), 237–256 (1997)
14.119 D. Andre, S. Russell: Programmable reinforcement learning agents, Proc. 13th Conf. Neural Inf. Process. Syst. (2001) pp. 1019–1025
14.120 D. Andre, S.J. Russell: State abstraction for programmable reinforcement learning agents, 18th Natl. Conf. Artif. Intell., Edmonton (2002) pp. 119–125
14.121 R.S. Sutton, D. Precup, S.P. Singh: Between MDPs and Semi-MDPs: A framework for temporal abstraction in reinforcement learning, Artif. Intell. **112**(1/2), 181–211 (1999)
14.122 D. Precup: Temporal Abstraction in Reinforcement Learning, Ph.D. Thesis (University of Massachusetts, Amherst 2000)
14.123 M. Beetz: Structured reactive controllers, J. Auton. Agents Multi-Agent Syst. **4**(1/2), 25–55 (2001)
14.124 D. McDermott: *A Reactive Plan Language* (Yale University, New Haven 1991)
14.125 M. Ingham, R. Ragno, B. Williams: A reactive model-based programming language for robotic space explorers, Proc. 6th Int. Symp. Artif. Intell. Robotics Autom. Space (ISAIRAS) (2001)
14.126 M. Bratman: *Intention, Plans, and Practical Reason* (Harvard Univ. Press, Cambridge 1987)
14.127 M. Bratman, D. Israel, M. Pollack: Plan and resource-bounded practical reasoning, Comput. Intell. **4**, 349–355 (1988)
14.128 M. Georgeff, F. Ingrand: Decision making in an embedded reasing system, Proc. 11th Int. Jt. Conf. Artif. Intell. (1989) pp. 972–978
14.129 M. Beetz, D. Jain, L. Mosenlechner, M. Tenorth, L. Kunze, N. Blodow, D. Pangercic: Cognition-enabled autonomous robot control for the realization of home chore task intelligence, Proceedings IEEE **100**(8), 2454–2471 (2012)
14.130 D.S. Weld: An introduction to least commitment planning, AI Magazine **15**(4), 27–61 (1994)
14.131 D.S. Weld: Recent advances in AI planning, AI Magazine **20**(2), 93–123 (1999)
14.132 D.V. McDermott: The 1998 AI planning systems competition, AI Magazine **21**(2), 35–55 (2000)
14.133 J. Hoffmann, B. Nebel: The FF planning system: Fast plan generation through heuristic search, J. Artif. Intell. Res. **14**, 253–302 (2001)
14.134 A.L. Blum, M.L. Furst: Fast planning through plan graph analysis, J. Artif. Intell. **90**, 281–300 (1997)
14.135 F. Bacchus, F. Kabanza: Planning for temporally extended goals, Ann. Math. Artif. Intell. **22**(1/2), 5–27 (1998)
14.136 D. Nau, O. Ilghami, U. Kuter, J.W. Murdock, D. Wu, F. Yaman: SHOP2: An HTN planning system, J. Artif. Intell. Res. **20**, 379–404 (2003)

14.137 D. McDermott: *Transformational planning of reactive behavior*, Tech. Rep. (Yale University, New Haven 1992)
14.138 P. H. Winston: *Learning Structural Descriptions from Examples*, AI Tech. Rep. 231 (MIT, Cambridge 1970)
14.139 K.J. Hammond: *Case-Based Planning: Viewing Planning as a Memory Task* (Academic Press, Waltham 1989)
14.140 R.G. Simmons: A theory of debugging plans and interpretations, Proc. 7th Natl. Conf. Artif. Intell. (1988) pp. 94–99
14.141 M. Beetz: *Concurrent Reactive Plans: Anticipating and Forestalling Execution Failures*, Lecture Notes in Artificial Intelligence, Vol. 1772 (Springer, Berlin, Heidelberg, 2000)
14.142 H. Grosskreutz: Probabilistic projection and belief update in the pGOLOG framework. In: *Informatik 2000*, Informatik Aktuell, ed. by K. Mehlhorn, G. Snelting (Springer, Berlin, Heidelberg 2000) pp. 233–249
14.143 L. Morgenstern: Mid-sized axiomatizations of commonsense problems: A case study in egg cracking, Studia Log. **67**(3), 333–384 (2001)
14.144 N. Blodow, D. Jain, Z.-C. Marton, M. Beetz: Perception and probabilistic anchoring for dynamic world state logging, Proc. 10th IEEE-RAS Int. Conf. Humanoid Robots (Humanoids) (2010) pp. 160–166
14.145 N.J. Nilsson: *Shakey the Robot*, Tech. Note , Vol. TN 323 (SRI International, Stanford 1984) http://www.ai.sri.com/shakey/
14.146 B. Raphael: *The Thinking Computer: Mind Inside Matter* (W.H. Freeman, San Francisco 1976)
14.147 Journal of Artificial Intelligence Research: http://www.jair.org/
14.148 European Conference on Artificial Intelligence: http://www.eccai.org/ecai.shtml
14.149 AAAI Conference on Artificial Intelligence: http://www.aaai.org/Conferences/AAAI/aaai.php
14.150 International Conference on Automated Planning and Scheduling: http://www.icaps-conference.org/
14.151 F. Ingrand, M. Ghallab: Robotics and artificial intelligence: A perspective on deliberation functions, AI Communications **27**(1), 63–80 (2014)

Multimedia Contents

15. Robot Learning

Jan Peters, Daniel D. Lee, Jens Kober, Duy Nguyen-Tuong, J. Andrew Bagnell, Stefan Schaal

Machine learning offers to robotics a framework and set of tools for the design of sophisticated and hard-to-engineer behaviors; conversely, the challenges of robotic problems provide both inspiration, impact, and validation for developments in robot learning. The relationship between disciplines has sufficient promise to be likened to that between physics and mathematics. In this chapter, we attempt to strengthen the links between the two research communities by providing a survey of work in robot learning for learning control and behavior generation in robots. We highlight both key challenges in robot learning as well as notable successes. We discuss how contributions tamed the complexity of the domain and study the role of algorithms, representations, and prior knowledge in achieving these successes. As a result, a particular focus of our chapter lies on model learning for control and robot reinforcement learning. We demonstrate how machine learning approaches may be profitably applied, and we note throughout open questions and the tremendous potential for future research.

15.1 **What Is Robot Learning** 358
15.1.1 Robot Learning Approaches 358
15.1.2 Robot Learning Settings 359
15.1.3 Notation 360
15.2 **Model Learning** 360
15.2.1 Model Types 361
15.2.2 Model Learning Architectures 363
15.2.3 Model Learning Challenges and Constraints 365
15.2.4 Model Learning Methods 366
15.2.5 Applications of Model Learning 369
15.2.6 Future Directions 371
15.3 **Reinforcement Learning** 372
15.3.1 Relation to Classical Control 372
15.3.2 Reinforcement Learning Methods 373
15.3.3 Optimal Control with Learnt Models 375
15.3.4 Model-Free Value Function Approaches 376
15.3.5 Policy Search 377
15.3.6 Challenges in Robot Reinforcement Learning 380
15.3.7 Tractability Through Representation 382
15.3.8 Tractability Through Prior Knowledge 383
15.3.9 Tractability Through Models 384
15.4 **Conclusions** 385
Video-References 386
References 386

In the future, robots will be part of daily life in human society, providing assistance in many areas ranging from clinical applications, education and care giving, to normal household environments, disaster scenarios, and space exploration [15.1]. Robots will no longer be used to only execute the same task thousands of times, but rather they will be faced with thousands of different tasks that rarely repeat in an ever changing environment [15.2]. It will be difficult to preprogram all possible tasks and scenarios in such robots, and robots will need to be able to learn by themselves or with the help of humans. They will need to automatically adjust to stochastic and dynamic environments, as well as to changes due to wear and tear. Machine learning will be necessary to achieve this high degree of autonomy and enabling future robots to perceive and act in the world.

15.1 What Is Robot Learning

Robot learning consists of a multitude of machine learning approaches in the context of robotics. The type of the learning problem is usually characterized by the type of feedback, the process of data generation, and the type of data. At the same time, the type of data will determine the robot learning approach which actually can be employed. While the field of robot learning includes a huge number of topics ranging from learning in perception, state abstraction, decision making, and probabilistic techniques in robotics [15.3–5], this chapter focuses on issues of learning action generation and control. As indicated in Fig. 15.1, we explain robot learning using the two key ingredients: (i) the setting in terms of data and feedback, and (ii) the learning approach.

15.1.1 Robot Learning Approaches

In this chapter, we will focus on the core robot learning approaches capable of learning action generation and control. Particularly, we will be focusing on techniques of *model learning* and *reinforcement learning* (RL). (Note that we omit imitation learning by behavioral cloning as Chap. 75 deals with it extensively.) These methods have enabled real-time learning in complex robot systems such as helicopters, legged robots, anthropomorphic arms, and humanoid robots. We first discuss some broader issues of learning control before reviewing technical details and the related literature.

Learning Models for Control

Models are among the most essential tools in robotics, and it is widely believed that intelligent mammals also rely on internal models in order to generate their actions. A model describes essential predictive information about the behavior of the environment and the influence of an agent on this environment. While classical robotics relies on manually generated models based on human insights into physics, future autonomous

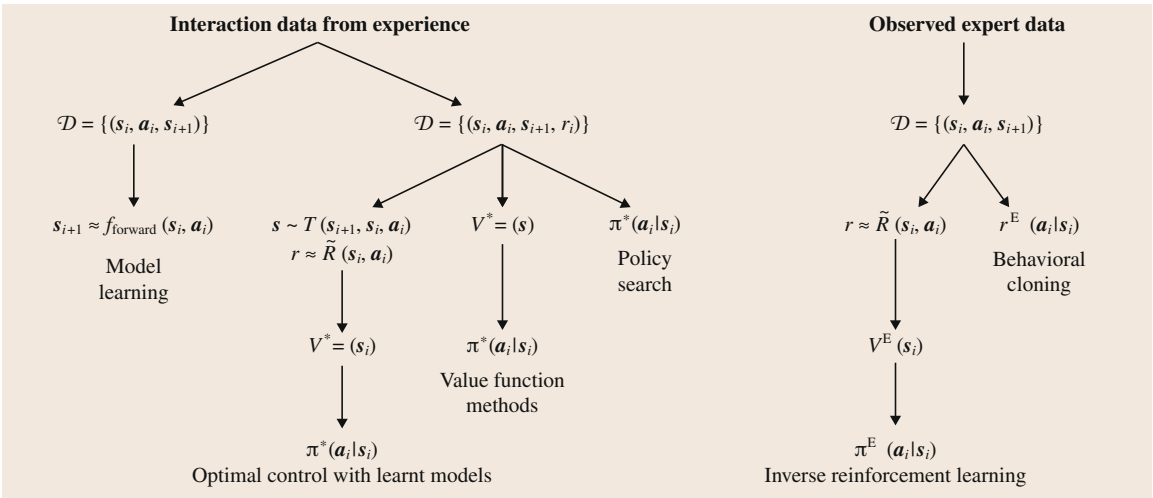


Fig. 15.1 The main branches of robot learning for action generation and control can be characterized both in terms of the available data $\mathcal{D}$ and the employed approach that learns from the data. A key distinction for data sets is whether they were generated from observed expert's demonstration or the robot's embodied experience. Data sets generically include samples i that consist at least of states $\boldsymbol{s}_i$, actions $\boldsymbol{a}_i$, and next states $\boldsymbol{s}_{i+1}$. However, in many cases, a quality assessment in form of a reward r_i is available as feedback. The branches in the tree illustrate the core robot learning problems considered here. Model learning generate a model $T(\boldsymbol{s}_{i+1}, \boldsymbol{s}_i, \boldsymbol{a}_i)$ from data. Reinforcement learning approaches require a reward in order to obtain an optimal policy π^*. Model-based methods usually learn a model from which they subsequently construct an optimal value function and policy. Model-free methods either learn value functions from data without model knowledge to construct a policy while policy search methods optimize the policy directly based on the interaction data. Expert data allows for reproducing observed behavior π^{E}. Two branches have had a particular importance in robot learning: imitation learning by behavioral cloning can treated by supervised learning and is discussed in detail in Chap. 74. Imitation learning by inverse RL reconstructs the reward function of the expert and uses reinforcement learning approaches to re-construct the expert's policy (This figure was inspired by [15.6])

robots need to be able to automatically generate models from information which is extracted from the data streams accessible to the robot. Machine learning approaches in generating these models will also need to be adapted to the control architectures for which they are intended. Such model learning for control will be discussed in Sect. 15.2.

Planning with Reinforcement Learning

Planning motor behaviors and optimizing them with respect to some criteria is part of every motor skill. Reinforcement learning offers one of the most general frameworks to address these issues. In essence, it assumes that the goal or intent of a movement is represented in terms of an optimization or reward criterion, such that learning algorithms can fill in the often nontrivial details of the plan. While RL has wide area of application in many domains ranging from computer games to elevator scheduling [15.7], its application to high-dimensional physical systems with continuous state–action spaces requires often dedicated approaches, algorithms, representations, and prior knowledge. An important issue will also be the choice between model-based and model-free algorithms, as well as between value function-based and policy search methods. Reinforcement learning's different branches are indicated in Fig. 15.1 and will be discussed in Sect. 15.3.

Learning from Demonstration with Inverse Reinforcement Learning

Learning a skill starting from scratch with a complex robot system is often infeasible or extremely time consuming. Taking inspiration from child teaching techniques, learning from demonstration has become a very popular tool to endow a robot with some initial reasonable performance, from which further learning can progress more rapidly. As another chapter in this handbook will address learning from demonstration in more detail, we also discuss a particular subclass of learning from demonstration, i. e., using inverse RL (the two most relevant classes of imitation learning are indicated in Fig. 15.1). Inverse RL addresses how to extract a suitable cost function from observed behavior, an idea going back to *Kalman* [15.8]. Algorithms for inverse RL are tightly coupled to RL which we can only briefly discuss in Sect. 15.3.6, *Curse of Goal Specification*, but they are also discussed in Sect. 75.4.3.

As learning has found many applications in robotics, this chapter can only scratch the surface and extensions of several sections can be found in [15.9, 10]. By focusing on the key approaches for teaching a robot new abilities, we nevertheless hope to give a concise overview.

15.1.2 Robot Learning Settings

The setting of robot learning methods is determined by the data. Data can either be obtained from an expert or generated by interaction with the system. Often data needs to be reproduced accurately (as e.g., for model learning or imitation learning by behavioral cloning) while in other cases a feedback signal can determine what data is actually relevant for the system (directly in RL and somewhat more indirectly in imitation learning by inverse RL). Also, the type of data generation will determine what robot learning methods are applicable as indicated in Fig. 15.1. For example, while expert data is crucial for imitation, it may simply be the wrong data to learn from in model learning.

Data for Robot Learning

In comparison to many other machine learning domains, robot learning suffers from a variety of complex real-world data problems. Real-world training time is limited – hence, only relatively few complete executions of a task can ever be generated. These episodes are frequently perceived noisily, have a fair amount of variability in the outcome, do not cover all possible scenarios and most often do not include all reactions to external stimuli. While episodes are scarce, the samples executed during a trial are not (but are often not helpful as they contain less information for some types of learning). Instead, high-dimensional data is obtained at a fast rate (e.g., proprioceptive information at 500 Hz to 5 kHz, vision at 30–200 Hz, tactile at 50 Hz to 5 kHz depending on the measurement mode). The data the robot can learn from is usually generated by the imperfect controller that needs to be improved. Thus, generating *interesting* data that has sufficiently useful information is a recurring problem in robot learning, discussed under various names, like exploration/exploitation tradeoff, persistent excitation, etc. Last not least, for high-dimensional robot systems (like humanoids), there is no *representative training data set* to learn from. A small change in posture of a robot can radically change in which part of the workspace the robot operates. Rapidly learning on the fly is frequently more important than squeezing the most of information from a few data points. Sharing data across different robots would be desirable, envisioning a *cloud robotics* approach. However, so far, even various instantiations of the same robot model can create quite different data depending on their wear and tear, calibration, and maintenance. Sharing data across different robot models has not seen a lot of success, as kinematic and dynamic differences can lead to rather different control and planning realizations. Thus, machine learning techniques for robotics

become quickly rather specialized and domain/robotic specific.

What Feedback Is Available for Learning?

A straightforward way to categorize robot learning approaches is given by the type of feedback [15.11]. A scalar performance score such as a reward or cost will often result in a *RL* approach. A presented desired action or predicted behavior allow supervised learning approaches such as model learning or direct imitation learning. Feedback in terms of an explanation has become most prominent in inverse RL. These methods will be explained in more detail below. Note that unsupervised learning problems where no feedback is required also can be found in robotics [15.5, 12].

15.1.3 Notation

In the next section and in Fig. 15.1, we will use the following notation. In general, we follow the tradition in RL [15.13]. When it comes to specific control problems, we use $\boldsymbol{q}$ for a vector of states in configuration space (often called joint-space), $\boldsymbol{u}$ for motor commands in configuration space, and $\boldsymbol{x}$ for a state in task- or effector-space.

15.2 Model Learning

In recent years, methods to learn models from data have become increasingly interesting tools for robotics. There are a variety of reasons for this development. First, robot control – as needed in less structured dynamic and more stochastic environments – requires model-based control to realize high compliance, since stiff control without good models exposes the robot and its environment to more risk of damage. Second, due to the complexity of modern robot systems, standard models like rigid body dynamics are only rough approximations which do not accurately model unknown sources of nonlinearities from actuation (hydraulics and motor saturation), passive elements like wires, hydraulic hoses, cable stretch, and/or sources of friction and stiction. Finally, interaction with an unstructured, uncertain, and human-inhabited environment may require to learn models on the fly [15.14, 15]. In addition, wear and tear of a robot in continual use will create drifting models. All these issues can be addressed by learning models directly from measured data, ideally in a continual and incremental on-line fashion, using appropriate nonlinear function approximation techniques.

We discuss different types of models (forward models, inverse models, mixed models, and operator models) in Sect. 15.2.1, along with difference architectures for model learning in Sect. 15.2.2. Some of the challenges that arise in these models in the domain of robotics are discussed in Sect. 15.2.3. How models can be learned using machine learning techniques are described in Sect. 15.2.4, and examples where model learning has proven to be helpful for the action generation in complex robot systems [15.14, 16–21] are highlighted in Sect. 15.2.5.

While this chapter attempts to give a fairly exhaustive overview on model learning in robotics, it is of course not complete due to the space constraints of a book chapter. An extended version of this section can be found in [15.9].

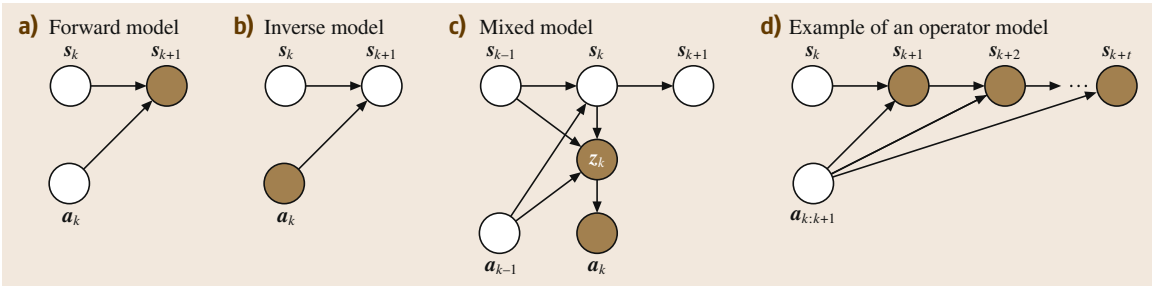


Fig.15.2a–d Graphical models for different types of models. The *white nodes* denote the *observed* quantities, while the *brown nodes* represent the quantities to be *inferred*. **(a)** The forward model allows inferring the next state given current state and action. **(b)** The inverse model determines the action required to move the system from the current state to the next state. **(c)** The mixed model approach combines forward and inverse models in problems where a unique inverse does not exist. Here, the forward and inverse models are linked by a latent variable z_k. **(d)** The operator model is needed when dealing with finite sequences of future states

15.2.1 Model Types

Models represent the functional relationship between input and output variables. If we can observe the current state $\boldsymbol{s}_k$ of a system and the action $\boldsymbol{a}_k$ currently applied to the system, we can attempt to predict the next state $\boldsymbol{s}_{k+1}$, which is called a forward model with mapping $(\boldsymbol{s}_k, \boldsymbol{a}_k) \rightarrow \boldsymbol{s}_{k+1}$. We can also use the inverse model to infer the current action, i. e., the relation $(\boldsymbol{s}_k, \boldsymbol{s}_{k+1}) \rightarrow \boldsymbol{a}_k$, if we know the current state and the desired or expected future state. There are also approaches combining forward and inverse models for prediction which we will refer to as mixed model approaches. However, for many applications the system behavior has to be predicted for the next t-steps rather than for the next single step; models that predict a series of states will be called operator models. Graphical depictions of these different models are shown in Fig. 15.2.

Forward Models

Forward models predict the next state of a dynamic system given the current action and current state. The forward model directly corresponds to the state transfer

$$\boldsymbol{s}_{k+1} = f_{\text{forward}}(\boldsymbol{s}_k, \boldsymbol{a}_k) \,. \tag{15.1}$$

This function expresses the causal physical properties of the system, and the learning of such causal mappings is a well-defined problem using standard regression techniques. While forward models are unique mappings in classical physics, there are several cases where forward models alone do not provide sufficient information to uniquely determine the next system's state [15.22]. For instance, when a pendulum is located at its unstable equilibrium point, it will tend to go either to the left or right rather than remain in the center. Nevertheless, the prediction of the forward model would be for the pendulum to remain centered. In this case, the state transfer function is not sufficient to capture the interesting dynamics of the system [15.22, 23]. For such cases, a probabilistic model $T(\boldsymbol{s}_{k+1}, \boldsymbol{a}_k, \boldsymbol{s}_k) = P(\boldsymbol{s}_{k+1}|\boldsymbol{s}_k, \boldsymbol{a}_k)$ may be more appropriate as it captures both likely modes with a conditional density instead of just the mean behavior.

An early application of forward models in classical control is the Smith predictor, where the forward model is employed to cancel out delays imposed by the feedback loop [15.24]. Subsequently, forward models have been applied in the context of model reference adaptive control (MRAC) [15.25]. MRAC is a control system in which the performance of an action is predicted using a forward reference model. The controller adjusts the action based on the resulting error between the desired and current state. Hence, the policy π for MRAC can be written as

$$\pi(\boldsymbol{s}) = \underset{\boldsymbol{a}}{\text{argmin}} \left(f_{\text{forward}}(\boldsymbol{s}, \boldsymbol{a}) - \boldsymbol{s}_{\text{des}}\right) , \tag{15.2}$$

where $\boldsymbol{s}_{\text{des}}$ denotes a desired trajectory and $\boldsymbol{s}$ represents the observed state. MRAC was originally developed for continuous-time system and extended later for discrete and stochastic systems [15.25]. Applications of MRAC can be widely found for robot control, such as adaptive manipulator control [15.26].

Further applications of forward models can be found in the wide class of model predictive control (MPC) [15.27]. MPC computes optimal actions by minimizing a given cost function over a certain prediction horizon N. The MPC control policy can be described by

$$\pi(\boldsymbol{s}) = \underset{\boldsymbol{a}_{t:t+N}}{\text{argmin}} \sum_{k=t}^{t+N} F_{\text{cost}} \left(f_{\text{forward}}(\boldsymbol{s}_k, \boldsymbol{a}_k), \boldsymbol{s}_{k,\text{des}}, \boldsymbol{a}_k\right) , \tag{15.3}$$

where $F_{\text{cost}}(\boldsymbol{s}, \boldsymbol{s}_{\text{des}})$ denotes the cost of an action $\boldsymbol{a}_k$ resulting in a state $\boldsymbol{s}$ away from the desired state $\boldsymbol{s}_{\text{des}}$. MPC was first developed for linear system models and, subsequently, extended to more complex nonlinear models with constraints [15.27]. Forward models have also been essential in model-based RL approaches, which relate to the problem of optimal control [15.28–30]. Here, the forward models describe the transition dynamics determining the probability of reaching the next state given current state and action. In contrast to previous applications, the forward models incorporate a probabilistic description of the system dynamics in this case [15.31, 32]. Example applications of forward models for optimal control are given in the case studies in Sect. 15.2.5, *Simulation-Based Optimization*.

Inverse Models

Inverse models predict the action required to move the system from the current state to a desired future state. In contrast to forward models, inverse models represent an anticausal system. Thus, inverse models may not exist or be well defined. However, for several cases, such as for inverse dynamics of nonoveractuated robots, the inverse relationship is well-defined. Ill-posed inverse modeling problems can be solved by introducing additional constraints, as discussed in Sect. 15.2.1, *Mixed Models*.

For control, applications of inverse models can be traditionally found in computed torque robot control, where the inverse dynamics model predicts the required torques, i. e., the action, to move the robot along a desired joint space trajectory. Thus, the computed torque

control policy can be described by

$$\pi(\boldsymbol{s}) = f_{\text{inverse}}(\boldsymbol{s}, \boldsymbol{s}_{\text{des}}) + k(\boldsymbol{s} - \boldsymbol{s}_{\text{des}}) \,, \tag{15.4}$$

where the function $k(\boldsymbol{s} - \boldsymbol{s}_{\text{des}})$ is an error correction term (e.g., a proportional–derivative (PD)-controller since both positions and velocities are normally part of the state in a second order dynamical system) needed for stabilization of the robot. If an accurate inverse dynamics model is given, the predicted torques are sufficient to obtain a precise tracking performance. The inverse dynamics control approach is closely related to the computed torque control method.

Here, the feedback term acts through the inverse model of the system giving a control policy,

$$\pi(\boldsymbol{s}) = f_{\text{inverse}}\left[\boldsymbol{s}, \boldsymbol{s}_{\text{des}} + k(\boldsymbol{s} - \boldsymbol{s}_{\text{des}})\right] . \tag{15.5}$$

If the inverse model perfectly describes the inverse dynamics, inverse dynamics control will perfectly compensate for all nonlinearities occurring in the system. Control approaches based on inverse models are well-known in the robotics community, and have gained increasing popularity since the late 1980s, as the increase in computational power and the existence of torque-controlled robots have allowed computing more complex models for real-time control. The concept of feedback linearization is another, more general way to derive inverse dynamics control laws, offering possibly more applications for learned models [15.33, 34].

Mixed Models

In addition to forward and inverse models, there are also methods combining both types of models. Since modeling the forward relationship is well defined while modeling the inverse relation may be an ill-posed problem, the basic idea behind the combined mixed model is that the information encoded in the forward model can help to resolve the nonuniqueness of the inverse model. A typical ill-posed inverse modeling problem is the inverse kinematics of redundant robots. Given a joint configuration $\boldsymbol{q}$, the task space position $\boldsymbol{x} = f(\boldsymbol{q})$ can be determined exactly (i. e., the forward model is well defined), while there may be many possible joint configurations $\boldsymbol{q}$ for a given task space position $\boldsymbol{x}$, i. e., the inverse model could have infinitely many solutions and deriving an average suitable solution is not straightforward. The problem is that the set of inverse solutions may not form a convex set. Thus, when learning such inverse mappings, the learning algorithm will potentially average over the nonconvex set, giving rise to an invalid solution.

The ill-posedness of the inverse model can be resolved if it is combined with the prediction of a forward model, which checks whether the proposed inverse solution results in the desired state [15.35]. If not, the remaining error can be used to adjust the learned inverse model such that it chooses a valid solution.

The mixed model approach, i. e., the composite of forward and inverse models, was first proposed in conjunction with the distal teacher learning approach [15.35], and discussed in Sect. 15.2.2, *Distal Teacher Learning*. The proposed mixed models approach has subsequently evoked significant interest and has been extensively studied in the field of motor control [15.36, 37]. Furthermore, the mixed model approach is supported by some evidence that the human cerebellum can be modeled using forward–inverse composite models, such as MOSAIC [15.38, 39]. While mixed models have become well known in the neuroscience community, the application of such models in robot control is not yet widespread. Pioneering work on mixed models in the control community can be found in [15.40, 41], where mixed models are used for model reference control of an unknown Markov jump system. Despite that mixed model approaches are not widely used in control, with the appearance of increasingly more humanoid robots, biologically inspired robot controllers are gaining more popularity and the ones based on mixed models may present a promising approach [15.17, 42, 43].

Operator Models

The models from the preceding sections are mainly used to predict a single future state or action. However, in many problems, such as open-loop control, one would like to have information of the system for the next t-steps in the future. This problem is the multi-step ahead prediction problem, where the task is to predict a sequence of future values without the availability of output measurements in the horizon of interest. We will term the models which are employed to solve this problem as operator models. It turns out that such operator models are difficult to develop because of the lack of measurements in the prediction horizon. A straightforward idea would be to apply single-step prediction models t times in sequence, in order to obtain a series of future predictions. However, this approach is susceptible to error accumulation, i. e., errors made in the past are propagated into future predictions. An alternative is to apply autoregressive models which are extensively investigated in time series prediction [15.44], in order to employ past predicted values to predict future outcomes.

Combining operator models with control was first motivated by the need of extension of forward models for multi-step predictions [15.45]. In more recent work, variations of traditional ARX and ARMAX models for

Table 15.1 Overview on model types associated with applicable learning architectures and example applications

Model type	Learning architecture	Example applications
Forward model	Direct modeling	Prediction, filtering, learning simulations, optimization
Inverse model	Direct modeling (if invertible), indirect modeling	Inverse dynamics control, computed torque control, feedback linearization control
Mixed model	Direct modeling (if invertible), indirect modeling, distal-teacher	Inverse kinematics, operational space control, multiple model control
Operator model	Direct modeling	Planning, optimization, model predictive control, delay compensation

nonlinear cases have been proposed for control [15.46, 47]. However, operator models based on some parametric structures, such as ARX or ARMAX, have shown to have difficulties in more complex systems. The situation is even worse in the presence of noise or complex nonlinear dynamics. These difficulties give reasons to employ nonparametric operator models for multi-step predictions [15.48, 49].

15.2.2 Model Learning Architectures

In the previous section, we have presented different prediction problems that require different types of models. A central question is how to learn and adapt these different models while they are being used. We will distinguish between direct modeling, indirect modeling, and the distal teacher approach. Table 15.1 shows an overview of model types associated with applicable learning architectures. These architectures can be observed in Fig. 15.3.

Direct Modeling

Direct learning is probably the most straightforward and most frequently applied way to obtain a model despite that it is not always applicable. In this learning paradigm, the model is directly learned by observing the inputs and outputs, creating a supervised learning problem. Direct model learning can be implemented using most regression techniques, such as linear regression, local linear regression, recursive regression [15.50, 51], neural networks [15.52, 53], or other statistical approximation techniques, currently mostly based on kernel methods [15.54, 55].

An early example for direct learning control was the self-tuning regulator that generates a forward model and adapts it online [15.56]. Assuming the estimated forward model is correct (also called the certainty equivalence principle), the self-tuning regulator will estimate an appropriate control law online. However, the forward model in the traditional self-tuning regulator has a fixed parametric structure and, hence, it cannot deal automatically with unknown nonlinearities [15.47, 57]. The main reason why parametric models need to be used in direct modeling techniques is that such model parameterization is necessary for a convenient formulation of the control law and, more importantly, for the rigorous stability analysis. As the parametric model structure is often too limited for complex robot systems, learning methods with more degrees of freedom

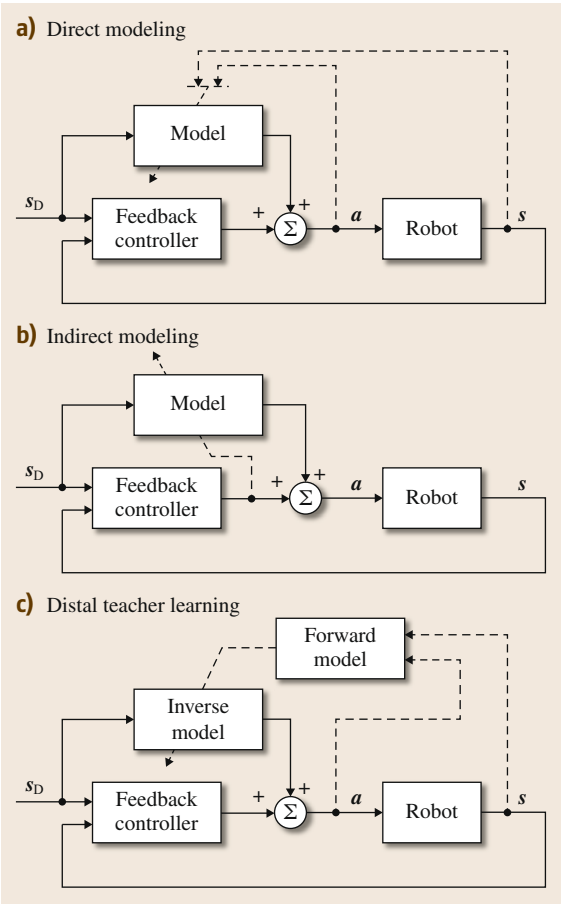


Fig.15.3a–c Learning architectures in model learning applied to control. **(a)** In the direct modeling approach, the model is learned directly from the observations – the given example shows learning and inverse model. **(b)** Indirect modeling approximates the model using the output of the feedback controller as error signal. **(c)** In the distal teacher learning approach, the inverse model's error is determined using the forward model. The resulting models will converge to an identity transformation

are needed, such as neural networks or other machine learning techniques [15.58, 59]. However, including sophisticated learning algorithms in control makes the analysis more complex if not impossible. Some examples of nonparameteric learning for control have been applied to unmanned airplane control [15.60], where the learning system models can prevent the airplane from crashing in case of changes in the airplane dynamics, e.g., due to failure of the mechanics.

Inverse models can also be learned in a direct manner if the inverse mapping is well defined. A well-known example is the inverse dynamics model required by computed torque and inverse dynamics control [15.61, 62]. If direct modeling is applicable, learning becomes straightforward and can be achieved using standard regression techniques [15.14, 63, 64]. Early work in learning inverse models for control attempted to adapt a parametric form of the rigid body dynamics model which is linear in its parameters and, hence, can be estimated from data straightforwardly using linear regression [15.65–67]. In practice, the estimation of dynamics parameters is not always straightforward. It is hard to create sufficiently rich data sets so that physically plausible parameters can be identified [15.15], and when identified online, additional persistent excitation issues occur [15.68]. Models with fixed parametric structures are not capable of capturing certain nonlinearities; hence, more sophisticated models have been introduced for learning inverse dynamics, such as neural networks [15.64, 69] or statistical nonparametric models [15.14, 63, 70]. There have also been attempts to combine parametric rigid body dynamics model with nonparametric model learning for approximating the inverse dynamics [15.71]. Similar as in inverse dynamics models, feedback linearization can also be used in conjunction with direct model learning. Again, the nonlinear dynamics can now be approximated using neural networks or other nonparametric learning methods [15.60, 72]. Stability analysis of feedback linearization control with learned models is possible extending the cases where the nonlinear dynamics could not be canceled perfectly [15.60]. It should be noted that learning an inverse dynamics model $(\boldsymbol{s}, \boldsymbol{s}_{\text{des}}) \rightarrow \boldsymbol{a}$ can have very slow learning performance when the initially learned model is rather far from reality. In this case, the predicted action $\boldsymbol{a}$ will create outcomes that are very far away from $\boldsymbol{s}_{\text{des}}$, such that the learning system, when trying out the bad action $\boldsymbol{a}$, can hardly extract any useful information from this experience how to actually realize $\boldsymbol{s}_{\text{des}}$.

While direct learning is mostly associated with learning a single type of model, it can also be applied to mixed models and operator models. The mixed model approach (e.g., combining inverse and forward models) finds its application in learning control for multiple-module systems. The basic idea is to decompose a complex system into many simpler subsystems which can be controlled individually [15.41]. The problem is how to choose an appropriate architecture for the multiple controllers, and how to switch between the multiple modules. Following the multiple-module idea, each controller module consists of a pair of inverse and forward models in the mixed model approach. The controller can be considered as an inverse model, while the forward model is essentially used to switch between the different modules [15.36]. Such multiple pairs of forward and inverse models can be learned directly using gradient-descend methods or expectation-maximization [15.42]. Direct learning of operator models has been done with neural networks [15.73]. In more recent work, probabilistic methods are employed to learn such operator models [15.48, 49].

Indirect Modeling

Direct learning works well when the functional relationship is well defined. However, there can be situations where the inverse relationship is not well defined, such as in the differential inverse kinematics problem. In such cases, inverse models can often still be learned, but in an indirect way. One such indirect modeling technique is feedback error model learning [15.74]. Feedback error learning relies on the output of a feedback controller that is used to generate the error signals employed to learn the feedforward controller. In problems, such as feedforward inverse dynamics control [15.61], this approach can be understood particularly well: if the inverse dynamics model is perfect, the feedback controller is not needed and its output will be zero. If it is nonzero, it corresponds to the error of the feedforward model [15.61]. The intuition behind feedback error learning is that by minimizing the feedback errors for learning the inverse model, the feedback control term will be decreasing as the model converges.

Compared to the direct model learning, feedback error learning is a goal-directed model learning approach resulting from the minimization of feedback error – in this error can only be zero if the goal was achieved. Another important difference between feedback error learning and direct learning is that feedback error learning has to perform online, while direct model learning can be done both online and offline. Additionally, feedback error learning does not create supervised learning data in the classical sense: the feedback error is not computed from an output target, but just from an error that represents the gradient toward better performance. This issue, and additionally a complex interaction between PD gains and learning stability [15.75] can make

it more complex to set up a successful feedback error learning system.

Feedback error learning is biologically motivated due to its inspiration from cerebellar motor control [15.37]. It has been further developed for control with robotics applications, originally employing neural networks [15.76, 77]. Feedback error learning can also be used with various nonparametric learning methods [15.75]. Conditions for the stability of feedback error learning control in combination with nonparametric approaches have also been investigated [15.75].

Indirect model learning can also be used in the mixed model approach [15.78]. Here, the attempt has been made to combine the feedback error learning with the mixture of experts' architecture to learn multiple inverse models for different manipulated objects, where the inverse models are learned indirectly using the feedback error learning approach [15.78]. The forward model is used for training a gating network, as it is well defined. The gating network subsequently generates a weighted prediction of the multiple inverse models, where the predictors determine the locally responsible models.

Distal Teacher Learning

The distal teacher learning approach was motivated by the necessity to learn general inverse models which suffer from the ill-posedness problem [15.35]. Here, the nonuniqueness of the inverse model is resolved by the distal teacher which determines the error of the inverse model using a forward model. Unlike the feedback error learning approach, this insight allows directly aiming at a globally consistent inverse model instead of local on-policy optimization. The distal teacher employs two interacting learning process: one process where the forward model is learned, and the other process where the learned forward model is used for determining the error of the inverse model. In the original distal learning approach, the inverse model's output is validated by the forward model, as the composite of these models yields an identity mapping if perfectly learned [15.35]. The intuition behind this approach is that the inverse model will learn a correct solution by minimizing the error between the output of the forward model and the input to the inverse model. Thus, the inverse model will result in solutions that are consistent with the forward model. The distal teacher approach has successfully learned particular solutions for multi-valued mappings, such as inverse kinematics of redundant robots [15.35]. Similar to feedback error model learning [15.74], distal teacher learning is also a goal-directed learning method applicable for various robot control scenarios.

The distal learning approach is particularly suitable for control when combining with the mixed models, as it naturally incorporates the mixed model principle. The distal teacher learning approach with mixed models has motivated a number of follow-up projects with several robot control applications [15.17, 43, 79].

15.2.3 Model Learning Challenges and Constraints

Given different types of models and their associated learning architectures, it may not be straightforward to apply machine learning methods [15.54, 55, 80] to these models in robotics. Several important problems arise that need to be addressed before learning methods can be successfully applied, and are summarized in Table 15.2.

Data Challenge

In order to learn a *good* model for applications in robotics, the sampled data has to cover a preferably large region of the model state space, and ideally cover it densely. Of course, in high-dimensional systems, one can never cover the complete state space – indeed, in complex robots like humanoids with state spaces of over 30 dimensions, only a very tiny fraction of the possible space can be explored in a lifetime. Thus, one can only hope to sample data in a task specific way, which is usually just in a very small part of the entire data space. For the purpose of generalization and accuracy, the data has to be sufficiently rich, i. e., it should contain as much information about the system as possible. Thus, generating large and rich data sets for model learning is an essential step. This step often requires additional excitation of the robot system during data generation, analogous to persistent excitation in classical system identification [15.68].

Table 15.2 Some challenges for model learning in robotics

Data challenges	Algorithmic constraints	Real-world challenges
High-dimensionality	Incremental updates	Safety
Smoothness	Real-time	Robustness
Richness of data	Online learning	Generalization
Noise	Efficiency	Interaction
Outliers	Large data sets	Stability
Redundant data	Prior knowledge	Uncertainty
Missing data	Sparse data	In the environment

Many machine learning methods for regression assume that the model to be learned is sufficiently smooth. However, there are many applications in robotics where the approximated functions are known to be nonsmooth – one example is a stiction-friction model. Such nonsmooth functions are sometimes approximated using kernel methods, where special kernels need to be defined in order to take the expected nonsmoothness in account [15.54, 55]. Discontinuities in a function can also be approximated by learning how to switch discontinuously between local models [15.81].

Difficulties arising from the large number of degrees of freedom in robotics can also be mitigated using dimensionality reduction methods [15.82, 83]. Dimensionality reduction is based on the insight that the useful information in the data may be represented by a low-dimensional manifold of the original input space. Dimensionality reduction methods have proven to be a powerful method for model learning in high-dimensional robot systems [15.63, 84].

As the data is sampled over a possibly long period of time in many robotics applications [15.85], problems of redundant and irrelevant data can occur. Redundant and irrelevant data can bias the model which severely degenerates the generalization performance. To overcome these issues, data needs to be filtered that is informative for the learning process [15.70, 86]. Dealing with noise and outliers is always a challenging problem for robot learning. Recent work in the machine learning community has addressed this problem by developing regularization frameworks based on statistical learning theory. The basic idea is to constrain the model to be learned in order to alleviate the contributions made by the noisy components in the data. Examples of regularized model learning methods include support vector regression and Gaussian process regression [15.87, 88]. Methods that explicitly account for noise in the data can significantly improve their generalization performance.

Algorithmic Constraints

In many scenarios, robot learning algorithms have to deal with massive amounts of data coming from their sensors. Thus, algorithms need to be efficient in terms of computation without sacrificing learning accuracy [15.89]. Fast and real-time computation presents a challenging problem. Standard model learning approaches, such as Gaussian process regression, for example, scale cubically in the number of training data points, preventing a straightforward usage in robotics. Sparse and reduced set methods smartly reduce the size of training data and, thus, decrease the computational effort for the learning and the prediction step [15.90]. Efficient implementations using parallel computation may also be used to speed up machine learning algorithms [15.91].

Online learning is also needed for continuous adaptation to a changing world which is essential to make robots more autonomous [15.85]. Most machine learning methods are developed for learning in batch mode, while online learning requires incremental approximation of the model [15.14, 92]. An advantage of online model learning is the ability to uncover interesting state-space regions during task execution.

For some robots, it may be difficult or expensive to obtain a lot of data for learning. In these cases, we need algorithms which allow us to incorporate additional prior knowledge to improve the learning performance in the presence of sparse data [15.93, 94]. In kernel methods, prior knowledge can be specified by feature vectors which can be used to define appropriate kernels [15.93]. In contrast, probabilistic frameworks allow specifying a priori information [15.54]. Furthermore, prior knowledge, if given as a parametric model, can be inserted into nonparametric models yielding semiparametric learning approaches [15.71, 95]. Semiparametric models have shown to be capable in learning competitive models, when little data is available [15.71].

Real-World Constraints

Ensuring safe interaction of robots with human beings in everyday life, machine learning algorithms developed for robotics applications have to minimize the risk of injury and damage. For critical applications, such as medical or service robotics, robustness and reliability are among the most important criteria which have to be fulfilled by model learning. Learning becomes more robust by employing feature selection as a preprocessing step, and has been used in several robotics application in vision and control [15.70, 96].

Robustness also requires the ability of the learning algorithm to deal with missing data. Missing data due to measurement errors poses a difficult problem for model learning in robotics. Recently, probabilistic learning methods have been used to infer the missing components in the training data [15.97]. An advantage of probabilistic methods comes from the ability to assign uncertainty to the predicted values, making it easier to deal with missing and imprecise data.

15.2.4 Model Learning Methods

We consider model learning in the context of supervised learning, where the inputs $\boldsymbol{x}$ and outputs $\boldsymbol{y}$ are given. The true output data is corrupted by noise $\boldsymbol{\epsilon}$, i. e.,

$$\boldsymbol{y} = f(\boldsymbol{x}) + \boldsymbol{\epsilon} \,. \tag{15.6}$$

Table 15.3 A large variety of machine learning methods have been applied in model learning for robotics. We distinguish between global and local methods, as well as semilocal methods which combine both approaches. The methods differ in the training mode and their online capabilities. For computational complexity, n denotes the total number of training points, m is the number of data points in a local model, and M is the number to local models

Method	Type	Mode	Online	Complexity	Learning applications
Locally weighted projection regression [15.80]	Local	Incremental	Yes	$\mathcal{O}(n)$	Inverse dynamics [15.63], Foothold quality model [15.21]
Local Gaussian process regression [15.14]	Local	Incremental	Yes	$\mathcal{O}(m^2)$	Inverse dynamics [15.14]
Gaussian mixture model [15.98]	Semilocal	Batch	No	$\mathcal{O}(Mn)$	Human motion model [15.99]
Bayesian committee machine [15.100]	Semilocal	Batch	No	$\mathcal{O}(m^2n)$	Inverse dynamics [15.54]
Sparse Gaussian process regression [15.101]	Global	Incremental	Yes	$\mathcal{O}(n^2)$	Transition dynamics [15.32], Task model [15.102]
Gaussian process regression [15.103]	Global	Batch	No	$\mathcal{O}(n^3)$	Terrain model [15.104], State estimation model [15.105]
Support vector regression [15.55]	Global	Batch	No	$\mathcal{O}(n^2)$	ZMP control model [15.106], Grasp stability model [15.107]
Incremental support vector machine [15.108]	Global	Incremental	Yes	$\mathcal{O}(n^2)$	Inverse dynamics [15.109]

Note that the noise is additive to $f(\boldsymbol{x})$; models that address noise in the inputs can be developed as well [15.110]. The performance of a model is given by its *generalization* ability, i. e., the learned model should be able to provide precise predictions of the output on new input data. Different learning techniques make different assumptions on how to model the function f given the data. From a function approximation point of view, we will distinguish between *global* and *local* techniques. Beyond the local and global types of model learning, there are also approaches that combine both ideas. An example of such a hybrid approach is the mixtures of experts [15.98, 111]. Here, the data is partitioned into smaller local models by a learned gating network, and prediction is accomplished by fusing local model predictions. Mixture of experts have been further embedded into a Bayesian learning framework, giving rise to a number of Bayesian hybrid approaches such as committee machines [15.100], mixtures of Gaussian models [15.112], or infinite mixtures of experts [15.113].

Global Regression

Global regression techniques model the underlying function f using *all* observed data [15.114], i. e., every data point is potentially adjusting every parameter of the learning system. A straightforward way to model the function f is to assume a parametric structure, such as linear or polynomial models or multilayer perceptron neural networks, and to fit the model parameters using training data [15.52, 114, 115]. However, fixing the model with a parametric structure beforehand may not suffice to explain the sampled data with sufficient accuracy, which motivates nonparametric model learning frameworks [15.54, 55, 115]. Nonparametric learning does not mean that there are no parameters, just that the correct number of relevant parameters is unknown, and that it is unlikely that a finite set of parameters would suffice to model the entire regression problem.

In the last two decades, parametric and nonparametric regression methods have been focussing on functions f modeled as

$$f(\boldsymbol{x}) = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}) \,, \tag{15.7}$$

where $\boldsymbol{\theta}$ is a weight vector and $\boldsymbol{\phi}$ is a vector of nonlinear functions, also called *features* or *basis-functions*. $\boldsymbol{\phi}$ projects $\boldsymbol{x}$ into some high-dimensional space, in which the learning problem is assumed to become linear in the parameters. The basic idea behind nonparametric regression is that the optimal model structure should be obtained from the training data. Hence, the size of the weight vector $\boldsymbol{\theta}$ is not fixed but can increase with the number of training data points. In contrast to nonparametric statistical approaches, other popular function approximation methods, such as neural networks, fix the model structure beforehand [15.52], like the number of nodes and their connections. However, there is a connection between artificial neural networks and probabilistic approaches [15.116, 117] using the formalization in (15.7).

Learning systems as formalized in (15.7) are mostly discussed as kernel methods [15.55]. Learning a model is equivalent to finding a tradeoff between the model's complexity and the best fit of the model to the observed data. Thus, it is desirable to have a model which is simple but at the same time can explain the data well. The number of features (i. e., the length of feature vector $\boldsymbol{\phi}$) has a direct correlation with the complexity of the regression model, as the more features one allows, usually the more powerful the learning system

becomes, but it also takes more data to constrain the open parameters of the model. One of the key insights of kernel methods [15.54, 55] was that one can actually work with *infinitely* many features using the so-called *kernel trick*, where the formulation of the regression problem only involves inner products of the feature vectors $\boldsymbol{\phi}(\boldsymbol{x}_i)^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}_j)$ of any two inputs $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and that this inner product can often be computed in closed form by a kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. This approach introduces a dual form of the regression learning problem where the number of open parameters is identical to the number of n training data, and not the number of features. In support vector regression (SVR) [15.55], a sparsification process will choose the minimal number of parameters needed to represent the training data. In Gaussian process regression [15.54], a probabilistic approach is taken that places a prior distribution over $\boldsymbol{\theta}$. The posterior parameters can subsequently be obtained by optimizing the corresponding marginal likelihood.

Kernel and probabilistic methods have proven to be successful tools for model learning over the last decade, resulting in a number of widely applied regression methods [15.87, 88, 118]. These methods can be applied to high-dimensional data. They can also deal well with noisy data, as the noise is taken in account indirectly by regularizing the model complexity. Furthermore, they are relatively easy to use, as several black-box implementations are available. However, the major drawback of these methods are the computational complexity. For instance, for Gaussian process regression the complexity scales cubically in terms of the number of training data points. Thus, one active research line in machine learning is to reduce the computational cost of those approaches. Due to customized techniques for robotics, kernel and probabilistic regression techniques found their ways into several robotics applications, such as robot control [15.70], sensor modeling [15.119], or state estimation [15.105].

Local Learning Methods

The basic idea of local regression techniques is to estimate the underlying function f *within a local neighborhood* around a query input point $\boldsymbol{x}_{\mathrm{q}}$. The data points in this neighborhood can then be used to predict the outcome for the query point. Generally, local regression models can be obtained by minimizing the following cost function J using n training data points

$$J = \sum_{k=1}^{n} w\left[(\boldsymbol{x}_k - \boldsymbol{x}_{\mathrm{q}})^{\mathrm{T}}\boldsymbol{D}(\boldsymbol{x}_k - \boldsymbol{x}_{\mathrm{q}})\right] \|\boldsymbol{y}_k - \hat{\boldsymbol{f}}(\boldsymbol{x}_k)\|^2 . \tag{15.8}$$

As indicated by (15.8), the essential ingredients for a local regression model are the neighborhood function w and the local model $\hat{f}$. The neighborhood function w, which is controlled by a distance metric $\boldsymbol{D}$, often simplified as a scalar width parameter $\boldsymbol{D} = 1/h\boldsymbol{I}$, measures the distance between a query point $\boldsymbol{x}_{\mathrm{q}}$ to the points in the training data. The local model $\hat{f}$ describes the function structure used to approximate f within the neighborhood around $\boldsymbol{x}_{\mathrm{q}}$ [15.120, 121]. Depending on the complexity of the data, different function structures can be assumed for the local model $\hat{f}$, such as a linear or a polynomial model. The open-parameters of $\hat{f}$ can be estimated straightforwardly by minimizing J with respect to these parameters.

The choice of the neighborhood function and its distance metric parameter is more involved. Several techniques have been suggested for estimating $\boldsymbol{D}$ for a given w, including the minimization of the leave-one-out cross-validation error and adaptive bandwidth selection [15.122, 123].

Due to their simplicity and computational efficiency, local regression techniques have become widespread in model learning for robotics [15.124–126]. In the last decade, novel local regression approaches were further developed in order to cope with the demands in real-time robotics applications, such as locally weighted projection regression [15.92, 127]. Inspired by local regression techniques, these methods first employ a partitioning of the input space into smaller local regions, for which locally linear models are approximated. A useful metaphor is to imagine that the input space is partitioned into disjoint boxes, and for the data points falling into each of these boxes, an independent local model is fit, usually of very simple complexity, like locally linear models.

In addition to the computational efficiency, local methods can deal with less smooth functions and do not require the same smoothness and regularity conditions as global regression methods. However, it has been shown in practice that local methods suffer from problems induced by high-dimensional data, as notions of locality break down for sparse, high-dimensional data. Furthermore, the learning performance of local methods may be sensitive to noise and heavily depends on the way how the input space is partitioned, i. e., the configuration of the neighborhood function w. These problems still present an active research topic [15.17, 128].

Several attempts have been made to scale local regression models to higher dimensional problems as required for many modern robotics systems. For example, locally weighted projection regression combines local regression with dimensionality reduction by projecting the input data into a lower dimensional space, where local regression is employed afterward [15.92]. Other methods combine nonparametric

probabilistic regression, such as Gaussian process regression, with the local approaches while exploiting the strength of probabilistic methods for model learning in high-dimension [15.14, 17]. Local learning methods are very well suited to parallelization on modern computer hardware, as each local model can be fit independently.

15.2.5 Applications of Model Learning

Several representative examples of model learning are illustrated in the following sections.

Simulation-Based Optimization

As forward models directly describe the causal dynamic behavior of a system, learning such models has received much attention in model-based control, particularly for optimal control problems. In optimal control, a forward model is needed to predict the behavior of the control system over an extended time horizon in order to optimize the control policy. If an analytical model is inaccurate, learning the model offers an interesting alternative.

Atkeson and *Morimoto* [15.29] have been among the first to explore this approach using differential dynamic programming [15.129] for optimizing open-loop control policies. The basic idea was to use receptive field-weighted regression (a type of locally weighted regression) to learn the models of the cost function and the state transition. Differential dynamic programming locally linearizes the state transition model and generates a local quadratic approximation of the cost. These approximations are used to improve an open-loop policy where the linearizations are also updated after every policy update [15.29]. Atkeson and Morimoto used the method to learn the underactuated pendulum swing up task, where a pole is attached to with a passive joint to the endeffector of a robot arm. The goal of the robot is to bring the pole from a hanging to an upright position. Hence, the system needs to *pump* energy into the pendulum in order to swing it up (Fig. 15.4a). Subsequently, the pole needs to be stabilized at the upright position [15.130]. Starting from an unconstrained human demonstration, the robot was able to successfully learn the swing up and balance task after three trials [15.130]. The local trajectory optimization technique has been further extended to biped robot walking [15.131]. More recently, a related approach with parametric function approximation has been applied by *Abbeel* et al. [15.132] to learn autonomous helicopter flight (VIDEO 353).

While *Atkeson* and *Morimoto* [15.29] and *Abbeel* et al. [15.132] used the forward model as an implicit simulator, *Ng* et al. [15.30] used it as an explicit simulator (as originally suggested by *Sutton* [15.28] in form of the DYNA model). Here, with the help of the forward model, complete trajectories (or roll-outs) were simulated. The predictions of the forward model were perturbed by Gaussian noise with a repeating, fixed noise history. This injection of noise requires the learned controller to be robust to noise and modeling errors, while the re-use of the noise history reduces the variance of policy updates and results in much faster learning [15.133, 134]. In this way, roll-outs from a similar set of start-states are generated. The resulting performance of different control policy can be compared, which allows policy updates both by pairwise comparison or by gradient-based optimization. The approach has been able to successfully stabilize an helicopter in an inverted flight (Fig. 15.4b and VIDEO 352). A few similar examples in model predictive control (Sect. 15.2.1) exist, which employ a variety of different learning approaches such as statistical learning methods [15.48], neural networks [15.135], both for robot navigation [15.136] and helicopter control [15.137].

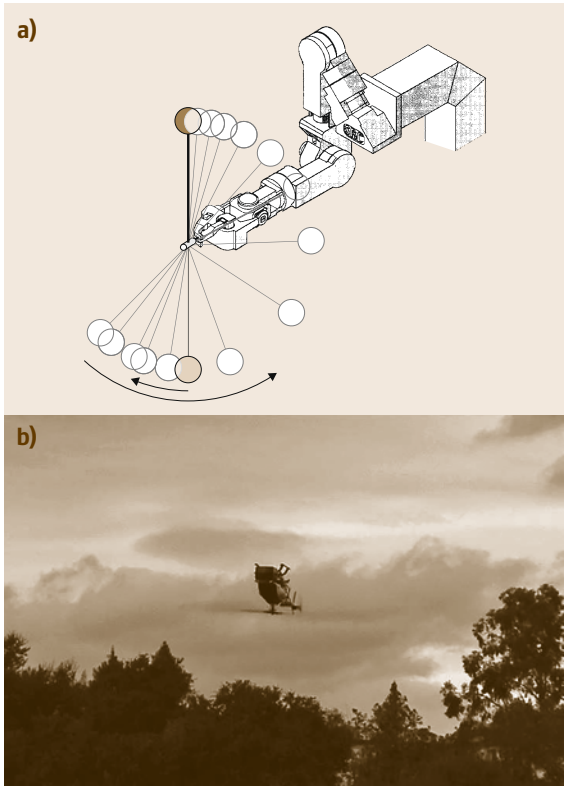


Fig. 15.4 (a) Sketch of learning a pendulum swing up task, and (b) inverse helicopter flight (courtesy of Andrew Y. Ng). In both the cases, the forward model is used to learn the dynamics of the system for policy optimization

Approximation-Based Inverse Dynamics Control

Inverse models, such as inverse dynamics models, are frequently used in robotics for feedforward control [15.62]. Inverse dynamics models characterize the required joint torques $\boldsymbol{\tau} = f(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ to achieve a desired robot acceleration $\ddot{\boldsymbol{q}}$ for the state $\boldsymbol{q}, \dot{\boldsymbol{q}}$, i. e., the vectors of joint positions and velocities. In classical robot control, analytical dynamics models are derived in the framework of rigid body dynamics

$$\boldsymbol{\tau}\,(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \mathbf{M}\,(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + \boldsymbol{F}\,(\boldsymbol{q}, \dot{\boldsymbol{q}})\,, \tag{15.9}$$

where $\mathbf{M}(\boldsymbol{q})$ is the generalized inertia matrix of the robot, $\boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is a vector defined by forces/torques from Coriolis effects, centripetal effects, and gravity. This model relies on many assumptions, such as rigidity of the robot's links, negligible nonlinearities from other sources (e.g., actuators), second-order dynamics, etc. [15.62]. However, modern robotics systems with highly nonlinear components, such as hydraulic tubes or elastic cable drives, can no longer be accurately modeled with rigid body dynamics. An inaccurate dynamics model can lead to severe losses in control performance and, in the worst case, instability. Instead of modeling the inverse dynamics manually based on physics and human insight, an inverse dynamics model can be learned from sampled data. Such a data-driven approach has the advantage that all nonlinearities encoded in the data will be approximated by the model [15.14].

Fig. 15.5 The 7 degree of freedom anthropomorphic SARCOS arm used in learning inverse dynamics, inverse kinematics, and operational space control

In most robots, the inverse dynamics model is a unique mapping from joint space into torque space, and learning inverse dynamics models is a standard regression problem. In order to generalize the learned models for a larger state space and to adapt the models for time-dependent changes in the dynamics, real-time online learning becomes necessary. However, online learning poses difficult challenges for any regression method. These problems have been addressed by real-time learning methods such as locally weighted projection regression [15.80]. *Nguyen-Tuong* and *Peters* [15.14, 70] combined the basic ideas behind the locally weighted projection regression method with Gaussian process regression [15.54], attempting to get as close as possible to the speed of local learning while having a comparable accuracy to Gaussian process regression. The resulting method has shown to be capable of real-time online learning of the robot's inverse dynamics. Instead of using local models, data sparsification methods can be employed to speed up kernel regression approaches for real-time scenarios [15.70].

It is worth noting that inverse dynamics model learning approaches can also be motivated from a biological point of view. *Kawato* [15.37] suggested that the cerebellum may act as an inverse dynamics model. Motivated by this insight, *Shibata* and *Schaal* [15.76] proposed a biologically inspired vestibulo-oculomotor control approach based on feedback-error learning of the inverse dynamics model. The problem is to stabilize the gaze of an oculomotor system that is mounted on a moving body, which creates permanent perturbations to eye (or camera) system. The cerebellum is known to predict the forces required to keep the image stabilized on the retina, based on efferent motor signals and inputs from the vestibular system. *Vijayakumar* and *Schaal* employ the locally weighted projection regression approach to learn the inverse model of the eye dynamics online [15.80]. The same locally weighted projection regression technique has also been used to learn a complete inverse dynamics model for a humanoid robot [15.63] (Fig. 15.5).

Learning Operational Space Control

Operational space control (OSC) allows a robot to accomplish a task space controller, e.g., for following a desired trajectory in task space or for impedance control [15.15, 138]. The relationship between the task space and joint space of the robot is well defined by the classical forward kinematics models $\boldsymbol{x} = f(\boldsymbol{q})$, where $\boldsymbol{q}$ denotes a joint space configuration and $\boldsymbol{x}$ represents the corresponding task space position. The task space velocity and acceleration are given by $\dot{\boldsymbol{x}} = \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{x}} =$

$\dot{\mathbf{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \mathbf{J}(\boldsymbol{q})\ddot{\boldsymbol{q}}$, respectively, where $\mathbf{J}(\boldsymbol{q}) = \partial f / \partial \boldsymbol{q}$ is the Jacobian. To obtain the joint torques required for task space control, the dynamics model (as given in (15.9)) is needed. The direct combination of dynamics and kinematics model yields one possible operational space control law

$$\boldsymbol{a} = \mathbf{M}\mathbf{J}_{\mathrm{W}}^{+}(\ddot{\boldsymbol{x}} - \dot{\mathbf{J}}\dot{\boldsymbol{q}}) + \boldsymbol{F}\,, \tag{15.10}$$

where $\mathbf{J}_{\mathrm{W}}^{+}$ denotes the weighted pseudoinverse of $\mathbf{J}$ [15.139] and $\boldsymbol{a}$ represents the desired joint torques. Equation (15.10) can be employed to generate the joint torques necessary for tracking a task space trajectory determined by a reference task-space acceleration [15.15]. Note that in practical applications, such control laws often require further terms, such as a null-space control law for joint-space stabilization [15.15].

As discussed before, dynamics models can be hard to obtain and, thus, learning can be a promising alternative. Learning an operational space control law corresponds to learning an inverse model such as $(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{x}}) \rightarrow \boldsymbol{a}$ [15.43]. However, learning such OSC models is an ill-posed problem, as there are infinitely many possible inverse models, due to the redundancy of joint space to task space mappings. For example, infinitely many solutions for a redundant robot can be derived analytically by varying the metric W of the weighted pseudoinverse in (15.10). As the space of possible solutions is not convex (Sect. 15.2.1, *Mixed Models*), such OSC models cannot be straightforwardly learned using regression models (unless the system has no redundant degrees of freedom). Similar problems appear in the limited case of differential inverse kinematics.

Both *D'Souza* et al. [15.79] and *Peters* and *Schaal* [15.43] noticed that local linearizations of the mapping in (15.10) will always form a locally convex space. Hence, data sets generated by such systems will also be locally convex. They furthermore realized that the predictive abilities of forward models allows determining local regions, where a locally consistent inverse model can be learned. However, depending on how the learning data was collected, i. e., the local data distribution, learning may result in rather different solutions to the local inverse model. Thus, two neighboring local inverse models may have quite different policies how to resolve redundancy, resulting in a discontinuous jump when transitioning from one local model to the next during control. As a result, a globally consistent solution to OSC can no longer be ensured. This insight leads to two significantly different approaches. *D'Souza* et al. [15.79] created a differential inverse kinematics learning system (i. e., a limited special case of an operational space control law) and chose to bias the learning system by selectively generating data. However, such an approach will generically be limited by the tradeoff between this intentional bias and the inverse model's accuracy. *Peters* and *Schaal* [15.43] treated learning of complete operational space control laws. They realized that a re-weighting of the data using an additional reward function allows regularizing these inverse models toward a globally consistent solution. Inspired by a result in analytical OSC [15.139], they suggest appropriate reward functions both for learning full OSC and differential inverse kinematics. The resulting mapping was shown to work on several robot systems. *Ting* et al. [15.17] presented an implementation of *Peters* and *Schaal* [15.43] approach with modern Bayesian machine learning, which sped up the performance significantly.

Instead of learning a direct OSC control law as done by *Peters* and *Schaal* [15.43], *Salaun* et al. [15.140] attempted to learn the well-defined differential forward kinematics as a first step (i. e., learning the Jacobian) using locally weighted projection regression. The corresponding weighted pseudoinverse of the Jacobian is subsequently computed using singular value decomposition (SVD). The obtained differential inverse kinematics model is combined with an inverse dynamics model to generate the joint space control torques [15.140]. Approximating inverse kinematics models has also been investigated in neural network learning and reservoir computing architectures [15.35, 141].

15.2.6 Future Directions

A key question in robotics is how to deal with uncertainty in the environment. As probabilistic machine learning techniques (e.g., Bayesian inference) have reached a high level of maturity [15.54], it has become clear how beneficial Bayesian machine learning can be for robot learning, especially, for model learning in the presence of uncertainty. However, machine learning techniques based on Bayesian inference are known to suffer from high computational complexity. Thus, special approximations will be needed as illustrated by the problems discussed in Sect. 15.2.3. Recently developed approximate inference methods such as in [15.90, 142] may become interesting new tools for robotics.

In order for robots to enter everyday life, they need to continuously learn and adapt to new tasks. Recent research on learning robot control has predominantly focused on learning single tasks that were studied in isolation. However, there is an opportunity to transfer knowledge between tasks, which is known as transfer learning in the field of machine learning [15.143]. To achieve this goal, robots need to learn the invariants of the individual tasks and environments and, sub-

sequently, exploit them when learning new tasks. In this context, similarities between tasks also need to be investigated and how they can be exploited for generalization [15.144].

In most of the model learning approaches, supervised learning methods are used. However, for many robot applications, target outputs are not always available. In these scenarios, semisupervised learning techniques can be useful to learn such models [15.145]. Semisupervised learning employs labeled as well as unlabeled data for model learning and can help to overcome the sparse labeling problem. It would also be beneficial to develop online versions of semisupervised approaches for real-time adaptation and learning.

Approximation-based control often still suffers from a lack of proper analysis of stability and convergence properties. Modern statistical learning theory might offer new tools to derive appropriate error bounds [15.55]. For example, generalization bounds can be used to estimate the learning performance of the controller and, from this insight, further statements and conditions about stability could be made.

Learning nonunique and ill-posed mappings is a key problem for several robot applications. This problem can be considered as a nonunique labeling problem, and statistical machine learning techniques, such as conditional random fields [15.146], may help to solve the problem. A conditional model specifies the probabilities of possible target outputs given an input observation sequence. As the target outputs are conditioned on the current input observations, nonuniqueness in the mappings can be resolved. It would be beneficial to investigate how such models could be incorporated into learning control and hierarchical control frameworks.

15.3 Reinforcement Learning

Probably the most general frameworks for learning control is RL which enables a robot to autonomously discover optimal behaviors through trial and error interactions with its environment. No explicit teaching signal is provided, as in supervised learning, but rather learning is based on a scalar objective function that measures the overall performance of the robot.

Consider the example of training a robot to return a table tennis ball ([15.147], VIDEO 354). In this case, the robot makes observations of dynamic variables specifying ball position and velocity as well as its internal dynamics including joint positions and velocities. These measurements capture the relevant *state* $\boldsymbol{s}$ of the system. The *actions* $\boldsymbol{a}$ available to the robot are the torque commands sent to the motors. A function π that computes the motor commands (i. e., the actions) based on the current state is the *policy*. The RL problem is to find a policy that optimizes the long term sum of *rewards* $R(\boldsymbol{s}, \boldsymbol{a})$, as already indicated in Sect. 15.1.3. The reward function in this example could be based on the number of successful hits, or on secondary criteria like energy consumption; RL algorithms aim to find (near-)optimal policies that maximize the reward function over a chosen time horizon. An extended discussion of robot RL techniques, on which this chapter is based, can be found in [15.10].

15.3.1 Relation to Classical Control

RL is closely related to the theory of classical optimal control, as well as dynamic programming, stochastic programming, simulation optimization, stochastic search, and optimal stopping [15.152]. Both RL and optimal control address the problem of finding an optimal policy (often also called the *controller* or *control policy*) that optimizes an objective function over a time horizon, and both rely on the notion of a system being described by an underlying set of states, controls and a plant or model that describes transitions between states. However, optimal control assumes perfect knowledge of the system's description in the form of a model (e.g., a model as in Sect. 15.2 that describes what the next state of the robot will be given the current state and action). For such models, optimal control ensures strong guarantees which, nevertheless, often break down due to model and computational approximations.

In contrast, RL operates directly on measured data and rewards from interaction with the environment. Reinforcement learning research has placed great focus on addressing cases which are analytically intractable by using approximations and data-driven techniques. An important class of approaches to RL within robotics centers on the use of classical optimal control techniques (e.g., linear-quadratic regulation and differential dynamic programming) to system models learned via repeated interaction with the environment [15.150, 153, 154]. A concise discussion of viewing RL as *adaptive optimal control* is presented in [15.155]. This path is indicated in Fig. 15.1 as *optimal control with learnt models*.

Reinforcement learning applied to robotics (Fig.15.6) differs considerably from most well-studied RL benchmark problems. Problems in robotics are

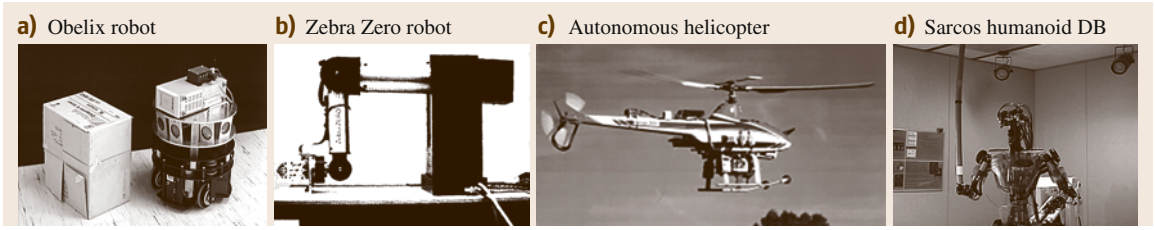


Fig.15.6a–d A small sample of robots with behaviors that were reinforcement learned. These cover the whole range of aerial vehicles, robotic arms, autonomous vehicles, and humanoid robots. **(a)** The OBELIX robot is a wheeled mobile robot that learned to push boxes (after [15.148]) with a value function-based approach (courtesy of Sridhar Mahadevan). **(b)** A Zebra Zero robot arm learned a peg-in-hole insertion task (after [15.149]) with a model-free policy gradient approach (courtesy of Rod Grupen). **(c)** Carnegie Mellon's autonomous helicopter leveraged a model-based policy search approach to learn a robust flight controller (after [15.150]). **(d)** The Sarcos humanoid DB learned a pole-balancing task (after [15.151]) using forward models

often involve high-dimensional, continuous states and actions (note that the 10–30-dimensional continuous actions common in robot RL are considered large [15.152]). In robotics, it is often unrealistic to assume that the true state is completely observable and noise free. The learning system will not be able to know precisely in which state it is and even vastly different states might look very similar. Thus, RL in robotics is often modeled as partially observed, a point we take up in detail in our formal model description below. The learning system must hence use filters to estimate the true state. It is often essential to maintain the information state of the environment that not only contains the raw observations but also a notion of uncertainty on its estimates (e.g., both the mean and the variance of a Kalman filter tracking the ball in the robot table tennis example).

Experience on a real physical system is tedious to obtain, expensive, and often hard to reproduce. Even getting to the same initial state is impossible for the robot table tennis system. Every single trial run, often called a roll-out or episode, is costly and, as a result, such applications force us to focus on difficulties that do not arise as frequently as in classical RL benchmark examples that are mostly simulation studies. In order to learn within a reasonable time frame, suitable approximations of state, policy, value function, and/or system dynamics need to be introduced. However, while real-world experience is costly, it usually cannot be replaced by learning in simulations alone. In analytical or learned models of the system, even small modeling errors can accumulate to a substantially different behavior. Hence, algorithms need to be robust with respect to models that do not capture all the details of the real system, also referred to as *under-modeling*, and to model uncertainty. Another challenge commonly faced in robot RL is the generation of appropriate reward functions. Rewards that guide the learning system quickly to success are needed to cope with the cost of real-world experience. This problem is called *reward shaping* [15.156] and represents a substantial manual contribution. Specifying good reward functions in robotics requires a fair amount of domain knowledge and may often be hard in practice.

Not every RL method is equally suitable for the robotics domain. In fact, many of the methods thus far demonstrated on difficult problems have been model-based [15.50, 132, 157] and robot learning systems often employ policy search methods rather than value function-based approaches [15.149, 150, 158–164]. Such design choices stand in contrast to possibly the bulk of the early research in the machine learning community [15.13, 165].

15.3.2 Reinforcement Learning Methods

In RL, an agent tries to optimize the accumulated reward over a time horizon, e.g., a manipulation task or even its entire lifetime. In an *episodic* setting, where the task is restarted at the end of each episode, the objective is to maximize the total reward per episode. If the task is on-going, a discounted accumulated reward which weighs earlier rewards with less influence is typically optimized. The agent is modeled as being in an environment with state $\boldsymbol{s} \in S$ and can perform actions $\boldsymbol{a} \in A$. The states and actions can be either discrete or continuous and are normally multi-dimensional. A state $\boldsymbol{s}$ contains all relevant information about the agent and environment needed to model future states or observables. An action $\boldsymbol{a}$ is used to control the time evolution of the system. For example, in a navigation task, the state could be the position and orientation of the robot, whereas the action corresponds to the torques applied to the wheels.

At every time step, the agent gets a reward R, a scalar value which typically depends on its state and

action. For instance, in a navigation task, rewards could be given for reaching the target, and penalties (negative reward) would arise for expending energy. The goal of RL is to find a policy π that chooses action $\boldsymbol{a}$ in $\boldsymbol{s}$ such as to maximize the cumulative reward over time. The policy π can be either deterministic, which always chooses the same action for each state $\boldsymbol{a} = \pi(\boldsymbol{s})$, or probabilistic which draws a sample action from a distribution: $\boldsymbol{a} \sim \pi(\boldsymbol{s}, \boldsymbol{a}) = P(\boldsymbol{a}|\boldsymbol{s})$. Reinforcement learning algorithms need to discover the relations between states, actions, and rewards. Hence exploration is required to sufficiently sample these functional relationships, which can be directly embedded in the policy or performed separately as part of the learning process.

Classical RL approaches are based upon the assumption of a *Markov decision process* or MDP which models the dynamics of the environment through transition probabilities T. Given the current state $\boldsymbol{s}$ and action $\boldsymbol{a}$, the conditional transition probability $T(\boldsymbol{s}', \boldsymbol{a}, \boldsymbol{s}) = P(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a})$ describes the distribution over the next state $\boldsymbol{s}'$ and generalize the notion of deterministic dynamics. The Markov property requires that the next state $\boldsymbol{s}'$ and reward depend only on the previous state $\boldsymbol{s}$ and action $\boldsymbol{a}$, and not on any other information [15.13]. In robotics, we are able to find an approximate notion of a state in a MDP for many situations.

Goals of Reinforcement Learning

The goal of RL is to discover an optimal policy π^* that maps states (or observations) to actions so as to maximize the expected return J, which corresponds to the cumulative expected reward. There are different models of optimal behavior [15.165] which result in different definitions of the expected return. A finite-horizon model only attempts to maximize the expected reward for the horizon H, i. e., the next H (time-)steps h

$$J = E\left\{\sum_{h=0}^{H} R_h\right\} .$$

This setting can also be applied to model problems where it is known how many steps are remaining.

Alternatively, future rewards can be discounted by a discount factor γ (with $0 \leq \gamma < 1$) yielding a metric

$$J = (1-\gamma)E\left\{\sum_{h=0}^{\infty} \gamma^h R_h\right\} .$$

This setting is most frequently discussed in classical RL texts. The parameter γ affects how much the future is taken into account and needs to be tuned manually. As illustrated in [15.165], this parameter often qualitatively changes the form of the optimal solution. Policies designed by optimizing with small γ are myopic and greedy, and may lead to poor performance if we actually care about longer term rewards. It is straightforward to show that the optimal control law can be unstable if the discount factor is too low (e.g., it is not difficult to show this destabilization even for discounted linear quadratic regulation problems). Hence, discounted formulations are occasionally inadmissible in robot control despite that they are somewhat more common in the machine learning community.

In the limit when the discount factor γ approaches 1, the metric approaches what is known as the average-reward criterion [15.7]

$$J = \lim_{H\to\infty} E\left\{\frac{1}{H}\sum_{h=0}^{H} R_h\right\} .$$

This setting has the problem that it cannot distinguish between policies that initially gain a transient of large rewards and those that do not. This transient phase, also called prefix, is dominated by the rewards obtained in the long run. If a policy accomplishes both an optimal prefix as well as an optimal long-term behavior, it is called bias optimal [15.166]. An example in robotics would be the transient phase during the start of a rhythmic movement, where many policies will accomplish the same long-term reward but differ substantially in the transient (e.g., there are many ways of starting the same gait in dynamic legged locomotion) allowing for room for improvement in practical application.

In real-world domains, the shortcomings of the discounted formulation are often more critical than those of the average reward setting as stable behavior is often more important than a good transient [15.167]. We also often encounter an episodic control task, where the task runs only for H time-steps and then reset (potentially by human intervention) and started over. This horizon H may be arbitrarily large, as long as the expected reward over the episode can be guaranteed to converge. As such episodic tasks are probably the most frequent ones, finite-horizon models are often the most relevant.

Two natural goals arise for the learner. In the first, we attempt to find an optimal strategy at the end of a phase of training or interaction. In the second, the goal is to maximize the reward over the whole time the robot is interacting with the world.

In contrast to supervised learning, the learner must first discover its environment and is not told the optimal action it needs to take. To gain information about the rewards and the behavior of the system, the agent needs to explore by considering previously unused actions or actions it is uncertain about. It needs to decide whether to play it safe and stick to well-known actions

with (moderately) high rewards or to dare trying new things in order to discover new strategies with an even higher reward. This problem is commonly known as the *exploration–exploitation tradeoff*. However, as state spaces in robotics problems are often tremendously large due to continuous and high-dimensional state–action spaces, exploring the entire state–action space is mostly infeasible. This problem is often referred to as the *curse of dimensionality* [15.168].

Off-policy methods learn independently of the employed policy, i. e., an explorative strategy that is different from the desired final policy can be employed during the learning process. *On-policy* methods collect sample information about the environment using the current policy. As a result, exploration must be built into the policy and determines the speed of the policy improvements. Such exploration and the performance of the policy can result in an exploration–exploitation tradeoff between long- and short-term improvement of the policy. Modeling exploration models with probability distributions has surprising implications, e.g., stochastic policies have been shown to be the optimal stationary policies for selected problems [15.169, 170] and can even break the curse of dimensionality [15.171]. Furthermore, stochastic policies often allow the derivation of new policy update steps with surprising ease.

The agent needs to determine a correlation between actions and reward signals. An action taken does not have to have an immediate effect on the reward but can also influence a reward in the distant future. The difficulty in assigning credit for rewards is directly related to the horizon or mixing time of the problem. It also increases with the dimensionality of the actions as not all parts of the action may contribute equally.

15.3.3 Optimal Control with Learnt Models

The classical RL setup is a MDP where, additionally, to the states $\boldsymbol{s}$, actions $\boldsymbol{s}$, and rewards R we also have transition probabilities $T(\boldsymbol{s}', \boldsymbol{a}, \boldsymbol{s})$. Here, the reward is modeled as a reward function $R(\boldsymbol{s}, \boldsymbol{a})$. If both the transition probabilities and reward function are known, this can be seen as an optimal control problem [15.152]. If these functions are acquired from measured data through model learning methods as described in Sect. 15.2, we are in the scenario of *optimal control with learnt models* indicated in Fig. 15.1.

The value function $V^{\pi}(\boldsymbol{s})$ provides an assessment of much reward will be accumulated starting from state $\boldsymbol{s}$ and following a given policy π. If a policy is optimal, denoted as π^*, the corresponding optimal value function will be denoted a V^{π^*}, often abbreviated as V^*. For each state there can be one or several optimal actions $\boldsymbol{a}^*$, resulting in the *same* value $V^*(\boldsymbol{s})$. The consistency condition for optimality is

$$V^*(\boldsymbol{s}) = \max_{\boldsymbol{a}} \left[R(\boldsymbol{s}, \boldsymbol{a}) + \gamma \sum_{\boldsymbol{s}'} V^*(\boldsymbol{s}') T(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}') \right] \tag{15.11}$$

in the discounted case. This statement is equivalent to the *Bellman principle of optimality* [15.168] that states:

> *An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.*

Thus, we have to perform an optimal action $\boldsymbol{a}^*$, and, subsequently, follow the optimal policy π^* in order to achieve a global optimum. This principle of optimality has also been crucial in enabling the field of optimal control [15.172]. Note that for constructing the optimal policy, the max operator in (15.11) needs to be evaluated, which requires knowledge of the state transition model $T(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}')$, i. e., this approach is generically a model-based requiring either a predefined model or a learnt one using insights from Sect. 15.2.

Dynamic programming-based methods are the classical approach for solving optimal control problems in RL. These require, as mentioned before, a model of the transition probabilities $T(\boldsymbol{s}', \boldsymbol{a}, \boldsymbol{s})$ and the reward function $R(\boldsymbol{s}, \boldsymbol{a})$ to calculate the value function. These functions do not necessarily need to be predetermined but can also be learned from data, potentially incrementally. Such methods are called *model-based* RL methods or optimal control with learnt models. Typical methods include policy iteration and value iteration.

Policy iteration alternates between the two phases of policy evaluation and policy improvement. The approach is initialized with an arbitrary policy. Policy evaluation determines the value function for the current policy. Each state is visited and its value is updated based on the current value estimates of its successor states, the associated transition probabilities, as well as the policy. This procedure is repeated until the value function converges to a fixed point, which approximates the true value function. Policy improvement greedily selects the best action in every state according to the value function, thus creating a policy update. The two steps of policy evaluation and policy improvement are iterated until the policy does not change any longer.

Policy iteration only updates the policy once the policy evaluation step has converged. In contrast, value iteration combines the steps of policy evaluation and policy improvement by directly updating the value function based on (15.11) every time a state is updated.

15.3.4 Model-Free Value Function Approaches

Many traditional RL approaches are based on identifying (possibly approximate) solutions to this equation, and are known as *value function methods*. Instead of deriving the value functions from a transition model and a reward function, they first approximate the value function, and use it to construct the optimal policy.

Instead of the value function $V^{\pi}(\boldsymbol{s})$, these algorithms usually rely on the state–action value function $Q^{\pi}(\boldsymbol{s},\boldsymbol{a})$, which has advantages of determining the optimal policy, as shown below, but also enables the use of model-free methods. This state–action value function is defined as

$$Q^{\pi}(\boldsymbol{s},\boldsymbol{a}) = R(\boldsymbol{s},\boldsymbol{a}) + \gamma \sum_{\boldsymbol{s}'} V^{\pi}(\boldsymbol{s}')T(\boldsymbol{s},\boldsymbol{a},\boldsymbol{s}') \,.$$

In contrast to the value function $V^{\pi}(\boldsymbol{s})$, the state–action value function $Q^{\pi}(\boldsymbol{s},\boldsymbol{a})$ explicitly contains the information about the effects of a particular action. The optimal state–action value function is

$$Q^{*}(\boldsymbol{s},\boldsymbol{a}) = R(\boldsymbol{s},\boldsymbol{a}) + \gamma \sum_{\boldsymbol{s}'} \left(\max_{\boldsymbol{a}'} Q^{*}(\boldsymbol{s}',\boldsymbol{a}') \right) T(\boldsymbol{s},\boldsymbol{a},\boldsymbol{s}') \,.$$

If Q^{*} is known, finding the optimal action just requires finding the maximal value of Q at a given state $\boldsymbol{s}$ – this does not require a model, but is just a standard (although not necessarily trivial) root-finding problem of functions. Importantly, using state–action value function allows a model-free learning approach, at the cost of learning a value function over the joint space of $\boldsymbol{s}$ and $\boldsymbol{a}$.

If the optimal value function $V^{*}(\boldsymbol{s}')$ and the transition probabilities $T(\boldsymbol{s},\boldsymbol{a},\boldsymbol{s}')$ for the following states are known, determining the optimal policy is straightforward in a setting with discrete actions, as an exhaustive search is possible. For continuous spaces, determining the optimal action $\boldsymbol{a}^{*}$ is an optimization problem in itself. If both states and actions are discrete, the value function and the policy may, in principle, be represented by tables and picking the appropriate action is reduced to a look-up. For large or continuous spaces, representing the value function as a table becomes intractable. Function approximation is employed to find a lower dimensional representation that matches the real value function as closely as possible, as discussed later.

A wide variety of methods of model-free value function based RL algorithms that attempt to estimate $V^{*}(\boldsymbol{s})$ or $Q^{*}(\boldsymbol{s},\boldsymbol{a})$ have been developed and can be split mainly into two classes: (i) rollout-based Monte Carlo methods and (ii) temporal difference methods such as TD(λ), Q-learning, and SARSA (state-action-reward-state-action).

Monte Carlo methods use sampling in order to estimate the value function. This procedure can be used to replace the policy evaluation step of the dynamic programming-based methods above. Monte Carlo methods are *model-free*, i. e., they do not need an explicit transition function. They perform roll-outs by executing the current policy on the system, hence operating on-policy. The frequencies of transitions and rewards are kept track of and are used to form estimates of the value function. For example, in an episodic setting the state–action value of a given state–action pair can be estimated by averaging all the returns that were received when starting from them.

Temporal difference methods, unlike Monte Carlo methods, do not have to wait until an estimate of the return is available (i. e., at the end of an episode) to update the value function. Rather, they use temporal errors and update at every time step. The temporal error is the difference between the old estimate and a new estimate of the value function, taking into account the reward received in the current sample. These updates are done iterativley and, in contrast to dynamic programming methods, only take into account the sampled successor states rather than the complete distributions over successor states. Like the Monte Carlo methods, these methods are model-free, as they do not use a model of the transition function to determine the value function. For example, the value function could be updated iteratively by

$$V'(\boldsymbol{s}) = V(\boldsymbol{s}) + \alpha \left[R(\boldsymbol{s},\boldsymbol{a}) + \gamma V(\boldsymbol{s}') - V(\boldsymbol{s}) \right] ,$$

where $V(\boldsymbol{s})$ is the old estimate of the value function, $V'(\boldsymbol{s})$ the updated one, and α is a learning rate. This update step is called the TD(0)-algorithm in the discounted reward case. However, in order to select the optimal action, a model of the transition function would still be required, i. e., this method can only be used for policy evaluation in a policy iteration scenario.

The equivalent temporal difference learning algorithm for state–action value functions is SARSA with

$$Q'(\boldsymbol{s},\boldsymbol{a}) = Q(\boldsymbol{s},\boldsymbol{a}) + \alpha \left[R(\boldsymbol{s},\boldsymbol{a}) + \gamma Q(\boldsymbol{s}',\boldsymbol{a}') - Q(\boldsymbol{s},\boldsymbol{a}) \right] ,$$

where $Q(\boldsymbol{s},\boldsymbol{a})$ is the old estimate of the state–action value function and $Q'(\boldsymbol{s},\boldsymbol{a})$ the updated one. This algorithm is on-policy as both the current action $\boldsymbol{a}$ as well as the subsequent action $\boldsymbol{a}'$ are chosen according to the current policy π. The average-reward variant is called R-learning [15.173], which is closely related to

Q-learning, with the updates

$$Q'(\boldsymbol{s},\boldsymbol{a}) = Q(\boldsymbol{s},\boldsymbol{a}) + \alpha\left[R(\boldsymbol{s},\boldsymbol{a}) + \gamma \max_{\boldsymbol{a}'} Q(\boldsymbol{s}',\boldsymbol{a}') - Q(\boldsymbol{s},\boldsymbol{a})\right].$$

These methods do not require a model of the transition function for determining the deterministic optimal policy $\pi^*(\boldsymbol{s})$.

15.3.5 Policy Search

Value function approaches infer the optimal policy indirectly by first finding a value function, and then computing the corresponding optimal actions. Instead, the policy can be learned directly, without a detour through a value function. For robotics, this approach has several advantages. It allows for a natural integration of expert knowledge, e.g., through both structure and initializations of the policy. It allows domain-appropriate prestructuring of the policy in an approximate form without changing the original problem. Optimal policies often have many fewer parameters than optimal value functions. For example, in linear quadratic control, the value function has quadratically many parameters in the dimensionality of the state-variables, while the policy requires only linearly many parameters. Local search in policy space can directly lead to good results as exhibited by early hill-climbing approaches [15.172], as well as more recent successes, [15.149, 150, 157, 159–163, 174–182]. Additional constraints can be incorporated naturally, e.g., regularizing the change in the path distribution. As a result, policy search often appears more natural to robotics.

Policy search has been considered the harder problem for a long time as the optimal solution most often cannot directly be determined from equations as in *Bellman principle of optimality* [15.168] and dynamic programming. Nevertheless, in robotics, policy search has recently become an important alternative to value function based methods due to better scalability as well as the convergence problems of approximate value function methods. Most policy search methods optimize locally around existing policies π, parameterized by a set of policy parameters θ_i, by computing changes in the policy parameters $\Delta\theta_i$ that will increase the expected return and results in iterative updates of the form

$$\theta_{i+1} = \theta_i + \Delta\theta_i\,.$$

The computation of the policy update is the key step here and a variety of updates have been proposed, including pairwise comparisons [15.183, 184], gradient estimation using finite policy differences [15.159, 160, 180, 185–187], general stochastic optimization methods (such as Nelder-Mead [15.150], cross entropy [15.188] and population-based methods [15.189]), and approaches coming from optimal control such as differential dynamic programming or DDP [15.153] and multiple shooting approaches [15.190].

Let us briefly take a closer look at gradient-based approaches first. The updates of the policy parameters are based on a hill-climbing approach, which is following the gradient of the expected return J for a defined step-size α

$$\theta_{i+1} = \theta_i + \alpha\nabla_\theta J\,.$$

Different methods exist for estimating the gradient $\nabla_\theta J$ and many algorithms require tuning of the step-size α.

Finite difference gradient estimators evaluate P policy parameter perturbations to obtain an estimate of the policy gradient. Here, we have $\Delta\hat{J}_p \approx J(\theta_i + \Delta\theta_p) - J_{\text{ref}}$, where $p = [1,\ldots,P]$ are the individual perturbations, $\Delta\hat{J}_p$ the estimate of their influence on the return, and J_{ref} is a reference return, e.g., the return of the unperturbed parameters. The gradient can now be estimated by linear regression

$$\nabla_\theta J \approx \left(\Delta\boldsymbol{\Theta}^{\mathrm{T}}\Delta\boldsymbol{\Theta}\right)^{-1}\Delta\boldsymbol{\Theta}^{\mathrm{T}}\Delta\hat{\mathbf{J}}\,,$$

where the matrix $\Delta\boldsymbol{\Theta}$ contains all the stacked samples of the perturbations $\Delta\theta_p$ and $\Delta\hat{\mathbf{J}}$ contains the corresponding $\Delta\hat{J}_p$. In order to estimate the gradient the number of perturbations needs to be at least as large as the number of parameters. The approach is very straightforward and even applicable to policies that are not strictly differentiable. However, it is usually considered to be very noisy and inefficient. For the finite difference approach tuning the step-size α for the update, the number of perturbations P, and the type and magnitude of perturbations are all critical tuning factors.

Likelihood ratio methods rely on the insight that in an episodic setting where the episodes τ are generated according to the distribution $P^\theta(\tau) = P(\tau|\theta)$ with the return of an episode $J^\tau = \sum_{h=1}^{H} R_h$ and number of steps H in each episode, the expected return for a set of policy parameter θ can be expressed as

$$J^\theta = \sum_\tau P^\theta(\tau)J^\tau\,. \tag{15.12}$$

The gradient of the episode distribution can be written as

$$\nabla_\theta P^\theta(\tau) = P^\theta(\tau)\nabla_\theta \log P^\theta(\tau)\,, \tag{15.13}$$

which is commonly known as the likelihood ratio or *REINFORCE* (reward increment = nonnegative factor × offset reinforcement × characteristic eligibility) [15.191] trick. From multivariate calculus, we have $\nabla_\theta \log P^\theta(\tau) = \nabla_\theta P^\theta(\tau)/P^\theta(\tau)$. Combining (15.12) and (15.13) we get the gradient of the expected return in the form

$$\begin{aligned}\nabla_\theta J^\theta &= \sum_\tau \nabla_\theta P^\theta(\tau) J^\tau \\ &= \sum_\tau P^\theta(\tau) \nabla_\theta \log P^\theta(\tau) J^\tau \\ &= E\left\{\nabla_\theta \log P^\theta(\tau) J^\tau\right\}.\end{aligned}$$

If we have a stochastic policy $\pi^\theta(\boldsymbol{a}|\boldsymbol{s})$ that generates the episodes τ, we do not need to keep track of the probabilities of the episodes but can directly express the gradient in terms of the policy as $\nabla_\theta \log P^\theta(\tau) = \sum_{h=1}^H \nabla_\theta \log \pi^\theta(\boldsymbol{a}_h|\boldsymbol{s}_h)$. Finally the gradient of the expected return with respect to the policy parameters can be estimated as

$$\nabla_\theta J^\theta = E\left\{\left(\sum_{h=1}^H \nabla_\theta \log \pi^\theta(\boldsymbol{a}_h|\boldsymbol{s}_h)\right) J^\tau\right\}.$$

If we now take into account that rewards at the beginning of an episode cannot be caused by actions taken at the end of an episode, we can replace the return of the episode J^τ by the state–action value function $Q^\pi(\boldsymbol{s}, \boldsymbol{a})$ and get [15.161]

$$\begin{aligned}\nabla_\theta J^\theta = (1-\gamma) \\ \times E\left\{\sum_{h=1}^H \gamma^h \nabla_\theta \log \pi^\theta(\boldsymbol{a}_h|\boldsymbol{s}_h) Q^\pi(\boldsymbol{s}_h, \boldsymbol{a}_h)\right\},\end{aligned}$$

which is equivalent to the *policy gradient theorem* [15.169]. In practice, it is often advisable to subtract a reference J_{ref}, also called baseline, from the return of the episode J^τ or the state–action value function $Q^\pi(s, a)$, respectively, to get better estimates, similar to the finite difference approach. In these settings, the exploration is automatically taken care of by the stochastic policy.

Initial gradient-based approaches such as finite differences gradients or REINFORCE [15.191] have been rather slow. The weight perturbation algorithm is related to REINFORCE but can deal with non-Gaussian distributions which significantly improves the signal-to-noise ratio of the gradient [15.180]. Recent natural policy gradient approaches [15.161, 162] have allowed for faster convergence which may be advantageous for robotics as it reduces the learning time and required real-world interactions.

Alternatives to gradient methods are currently a hot research topic. For example, a different class of safe and fast policy search methods, inspired by expectation-maximization from machine learning, can be derived when the reward is treated as an improper probability distribution [15.192]. Some of these approaches have proven successful in robotics, e.g., reward-weighted regression [15.161], policy learning by weighting exploration with the returns [15.163], Monte Carlo expectation maximization [15.193], and cost-regularized kernel regression [15.194]. Algorithms with closely related update rules can also be derived from different perspectives including policy improvements with path integrals [15.195] and relative entropy policy search [15.175].

Finally, the policy search by dynamic programming [15.196] method is a general strategy that combines policy search with the principle of optimality. The approach learns a nonstationary policy backward in time like dynamic programming methods, but does not attempt to enforce the Bellman equation and the resulting approximation instabilities. The resulting approach provides some of the strongest guarantees that are currently known under function approximation and limited observability. It has been demonstrated in learning walking controllers and in finding near-optimal trajectories for map exploration [15.197].

One of the key open issues in the field is determining when it is appropriate to use each of these methods. Some approaches leverage significant structure specific to the RL problem [15.195], including reward structure, Markovanity, causality of reward signals [15.191], and value-function estimates when available [15.161]. Others embed policy search as a generic, black-box, problem of stochastic optimization [15.150, 198–201]. Significant open questions remain regarding which methods are best in which circumstances and further, at an even more basic level, how effective leveraging the kinds of problem structures mentioned above are in practice. Examples of resulting success can be observed in VIDEO 354 and VIDEO 355.

Value Function Approaches Versus Policy Search

The discussion whether value function approaches or policy search methods are more suitable for RL has been going on for a long time, and, so far, the answer is really problem specific. If a complete optimal value function is known, a globally optimal solution follows by greedily choosing actions that receive the highest value. However, value-function-based approaches have up to now been difficult to scale to high-dimensional

robotics as they require function approximation for the value function – in low dimensions, discretizations and lookup tables work best and have no theoretical issues. Unfortunately, theoretical guarantees from discrete state–action representations no longer hold when employing function approximation. Even finding the optimal action can be a hard problem due to the sensitivity of selecting the optimal action in face of even tiny inaccuracies in the approximated value function, and the cost of solving an optimization problem in a continuous unknown function for action selection. For example, the computational cost of updating a policy in policy search can easily be cheaper than finding just *one* optimal action for one state by searching the state–action value function.

In principle, a value function requires total coverage over the state space and the *largest local error* determines the quality of the resulting policy. A particularly significant problem is the error propagation in value functions. A small change in the policy may cause a large change in the value function, which again causes a large change in the policy. While this may lead more quickly to good, possibly globally optimal solutions, such learning processes often prove unstable under function approximation [15.196, 202, 203] and are considerably more dangerous when applied to real robotic systems where overly large policy deviations may lead to dangerous motor commands.

In contrast, policy search methods usually only consider the current policy and its neighborhood in order to gradually improve performance. The result is that usually only local optima, and not the global one, can be found. However, these methods work well in conjunction with continuous states and actions, and they scale to high-dimensional robots, e.g., humanoid robots.

Policy search methods are sometimes called *actor*-only methods; value function methods are sometimes called *critic*-only methods. The idea of a critic is to first observe and estimate the performance of choosing controls on the system (i. e., the value function), then derive a policy based on the gained knowledge. In contrast, the actor directly tries to deduce the optimal policy. A set of algorithms called *actor-critic* methods attempt to incorporate the advantages of each: a policy is explicitly maintained, as is a value function for the current policy. The value function (i. e., the critic) is not employed for action selection. Instead, it observes the performance of the actor and decides when the policy needs to be updated and which action should be preferred. The resulting update step features the local convergence properties of policy gradient algorithms while reducing update variance [15.204]. There is a tradeoff between the benefit of reducing the variance of the updates and having to learn a value function as the samples required to estimate the value function could also be employed to obtain better gradient estimates for the update step. *Rosenstein* and *Barto* [15.205] propose an actor-critic method that additionally features a supervisor in the form of a stable policy.

Function Approximation

The topic of *function approximation* occurs repeatedly in RL and deserves some attention. For discrete state–action problems with not too many states and actions, look-up tables are a reasonable representation for value functions, rewards functions, and state-transitions (i. e., models). But look-up tables have no generalization to neighboring states, which makes them computationally inefficient. Moreover, for continuous state–action problems with many dimensions, discretization would create an exponential growth of the number of discrete states and actions, and is thus infeasible (*curse of dimensionality*). Thus, replacing discretized representations by continuous function approximators is the only possibility.

Function approximation can be employed to represent policies, value functions, and state-transition models. Thus, the techniques discussed in Sect. 15.2.4 also apply to these function approximation problems. However, a fundamental problem in using supervised learning methods developed in the literature for function approximation is that most such methods are designed for independently and identically distributed sample data. However, the data generated by the RL process is usually neither independent nor identically distributed. Usually, the function approximator itself plays some role in the data collection process (e.g., by serving to define a policy that we execute on a robot.)

Linear basis function approximators form one of the most widely used approximate value function techniques in continuous (and discrete) state spaces. This popularity is largely due to the simplicity of their representation as well as a convergence theory, albeit limited, for the approximation of value functions based on samples [15.206]. Let us briefly take a closer look at a radial basis function network to illustrate this approach. The value function maps states to a scalar value. The state space can be covered by a grid of points, each of which correspond to the center of a Gaussian-shaped basis function. The value of the approximated function is the weighted sum of the values of all basis functions at the query point. As the influence of the Gaussian basis functions drops rapidly with the distance of its center, the value of the query points will be predominantly influenced by the neighboring basis functions. The weights are set in a way to minimize the error between the observed samples and the reconstruction. For the mean squared error, these weights can be de-

termined by linear regression. *Kolter* and *Ng* [15.207] discuss the benefits of regularization of such linear function approximators to avoid over-fitting.

Other possible function approximators for value functions include wire fitting, which *Baird* and *Klopf* [15.208] suggested as an approach that makes continuous action selection feasible. Using Fourier basis has been suggested by *Konidaris* et al. [15.209]. Even discretizing the state space can be seen as a form of function approximation where coarse values serve as estimates for a smooth continuous function. One example is tile coding [15.13], where the space is subdivided into (potentially irregularly shaped) regions, called tiling. The number of different tilings determines the resolution of the final approximation. Policy search also benefits from compact representations of the policy.

Models of the system dynamics can be represented using a wide variety of techniques. In this case, it is often important to model the uncertainty in the model (e.g., by a stochastic model or Bayesian estimates of model parameters) to ensure that the learning algorithm does not exploit model inaccuracies. See Sect. 15.3.9 for a more detailed discussion.

15.3.6 Challenges in Robot Reinforcement Learning

Reinforcement learning is generally a difficult problem and many of its challenges are particularly apparent in the robotics setting. As the states and actions of most robots are inherently continuous, we are forced to consider the resolution at which they are represented. We must decide how fine grained the control is that we require over the robot, whether we employ discretization or function approximation, and what time step we establish. Additionally, as the dimensionality of both states and actions can be high, we face the *Curse of Dimensionality* [15.168] as discussed in Sect. 15.3.6, *Curse of Dimensionality*. As robotics deals with complex physical systems, samples can be expensive due to the long execution time of complete tasks, required manual interventions, and the need for maintenance and repair. In these real-world measurements, we must cope with the uncertainty inherent in complex physical systems. A robot requires that the algorithm runs in real-time. The algorithm must be capable of dealing with delays in sensing and execution that are inherent in physical systems (Sect. 15.3.6). A simulation might alleviate many problems but these approaches need to be robust with respect to model errors as discussed later. An often underestimated problem is the goal specification, which is achieved by designing a good reward function. As noted in Sect. 15.3.6, this choice can make the difference between feasibility and an unreasonable amount of exploration.

Curse of Dimensionality

When *Bellmann* [15.168] explored optimal control in discrete high-dimensional spaces, he faced an exponential explosion of states and actions for which he coined the term *Curse of Dimensionality*. As the number of dimensions grows, exponentially more data and computation is needed to cover the complete state–action space. Evaluating every state quickly becomes infeasible with growing dimensionality, even for discrete states.

Robotic systems often have to deal with these high-dimensional states and actions due to the many degrees of freedom of modern anthropomorphic robots. For example, in a ball-paddling task (VIDEO 355) the proper representation of a robot's state would consist of its joint angles and velocities for each of its seven degrees of freedom as well as the Cartesian position and velocity of the ball (Fig. 15.7). The robot's actions would be the generated motor commands, which often are torques or accelerations. In this example, we have $2 \times (7+3) = 20$ state dimensions and seven-dimensional continuous actions.

In robotics, such tasks are often rendered tractable to the robot engineer by a hierarchical task decomposition that shifts some complexity to a lower layer of functionality. Classical RL approaches often consider a grid-based representation with discrete states and actions, often referred to as a *grid-world*. A navigational task for mobile robots could be projected into

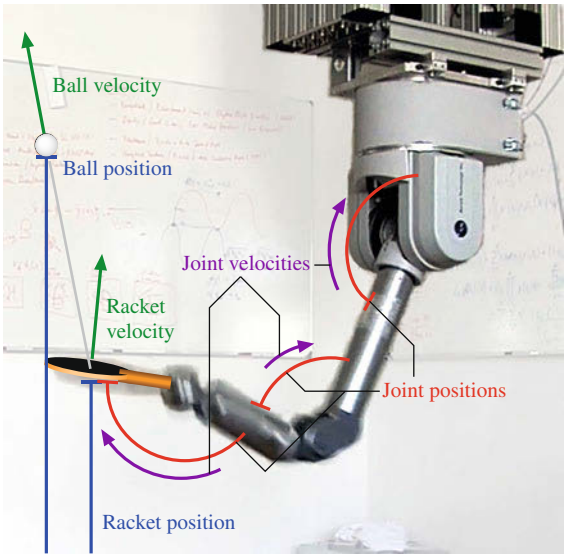


Fig. 15.7 State–space used in the modeling of a robot RL task of paddling a ball. This example can also be observed in VIDEO 355

this representation by employing a number of actions like *move to the cell to the left* that use a lower level controller that takes care of accelerating, moving, and stopping while ensuring precision. In the ball-paddling example, we may simplify by controlling the robot in racket space (which is lower dimensional as the racket is orientation-invariant around the string's mounting point) with an operational space control law [15.15]. Many commercial robot systems also encapsulate some of the state and action components in an embedded control system (e.g., trajectory fragments are frequently used as actions for industrial robots). However, this form of a state dimensionality reduction severely limits the dynamic capabilities of the robot according to our experience [15.63, 210].

The RL community has a long history of dealing with dimensionality using computational abstractions. It offers a larger set of applicable tools ranging from adaptive discretizations [15.211] and function approximation approaches [15.13] to macro-actions or options [15.212, 213]. Options allow a task to be decomposed into elementary components and quite naturally translate to robotics. Such options can autonomously achieve a subtask, such as opening a door, which reduces the planning horizon [15.212]. The automatic generation of such sets of options is a key issue in order to enable such approaches. We will discuss approaches that have been successful in robot RL in Sect. 15.3.7.

Curse of Real-World Samples

Robots inherently interact with the physical world and, thus, create their own learning data from physical sensors. Real physical systems create a variety of problems that are worthwhile to keep in mind:

- Robot hardware is usually expensive, suffers from wear and tear, and requires careful maintenance. Repairing a robot system is a nonnegligible effort associated with cost, physical labor, and long waiting periods. To apply RL in robotics, safe exploration becomes a key issue of the learning process [15.157, 214–216].
- The robot and its environment may continually change slowly, e.g., due to wear and tear, or light conditions. Thus, the learning process should be able to track these changing conditions, i. e., learn continually.
- Experimentation with real robots is time consuming, e.g., putting the pole back on the robot's end-effector during pole balancing after a failure, and running many trials for learning. Algorithms that are data efficient are often more important than memory and computations needed to process the data.
- Real-time constraints on an actual robot impose constraints for how much computation can be performed for action generation and learning updates. These constraints are less severe in an episodic setting where the time intensive part of the learning can be postponed to the period between episodes. *Hester* et al. [15.217] have proposed a real-time architecture for model-based value function RL methods taking into account these challenges.
- Time-discretization, inevitable in computer implementations, can generate undesirable artifacts, e.g., the distortion of distance between states, or smoothing effect by temporal aliasing.
- Delays in information processing with physical sensors and actuators can negatively affect learning. This effect can be addressed by putting some number of recent actions into the state, at the cost of significantly increasing the dimensionality of the problem.

Curse of Under-Modeling and Model Uncertainty

One way to offset the cost of real-world interaction is to use accurate models as simulators as suggested in Sect. 15.2.5. In an ideal setting, this approach would render it possible to learn the behavior in simulation and subsequently transfer it to the real robot. Unfortunately, creating a sufficiently accurate model of the robot and its environment is often impossible. As small modeling errors accumulate over time, the simulated robot can quickly diverge from the real-world system, such that the policy will not transfer without significant modifications as experienced by [15.218]. For tasks where the system is self-stabilizing (i. e., where the robot does not require active control to remain in a safe state or return to it), transferring policies often works much better [15.219]. Learning in simulation, however, becomes quickly useless when contact dynamics and friction are too hard to model. For example, in the ball-paddling task the elastic string that attaches the ball to the racket, and the contact dynamics of the bouncing ball have such modeling problems.

Curse of Goal Specification

In RL, the desired behavior is implicitly specified by the reward function. While often dramatically simpler than specifying the behavior itself, in practice, it can be surprisingly difficult to define a good reward function, such that manual design and tuning become very important. In many domains, it seems natural to provide rewards only upon task achievement – for example, when a table tennis robot wins a match. This view results in an apparently simple, binary reward specification. However, a robot may receive such a reward so rarely that it is un-

likely to ever succeed in its lifetime. Instead of relying on simpler binary rewards, we frequently need to include intermediate rewards in the scalar reward function to guide the learning process to a reasonable solution, a process known as *reward shaping* [15.156]. Additionally, physical constraints of the robot in terms of range of motion and torque saturation need to be taken into account. Often, a change in coordinates, i. e., the question in which space (e.g., task space, joint space) the cost function should be specified, can determine success or failure. In general, RL algorithms are also notorious for exploiting the reward function in ways that are not anticipated by the designer.

Inverse RL, also known as inverse optimal control [15.220], is a promising alternative to specifying the reward function manually. It assumes that a reward function can be reconstructed from a set of expert demonstrations as indicated in Fig. 15.1. This reward function does not necessarily correspond to the true reward function, but provides guarantees on the resulting performance of learned behaviors [15.221, 222]. Inverse RL was initially studied in the control community [15.8] and in the field of economics [15.223]. The initial results were only applicable to limited domains (linear quadratic regulator problems) and required closed form access to plant and controller, hence samples from human demonstrations could not be used. *Russel* [15.220] brought the field to the attention of the machine learning community. *Abbeel* and *Ng* [15.221] defined an important constraint on the solution to the inverse RL problem when reward functions are linear in a set of features: a policy that is extracted by observing demonstrations has to earn the same reward as the policy that is being demonstrated. *Ratliff* et al. [15.222] demonstrated that inverse optimal control can be understood as a generalization of ideas in machine learning of *structured prediction* and introduced efficient subgradient-based algorithms with regret bounds that enabled large scale application of the technique within robotics. *Ziebart* et al. [15.2] extended the technique developed by *Abbeel* and *Ng* [15.221] by rendering the idea robust and probabilistic, enabling its effective use for both learning policies and predicting the behavior of suboptimal agents. These techniques, and many variants, have been recently successfully applied to outdoor robot navigation [15.224–226], manipulation [15.227], and quadruped locomotion [15.224, 227, 228]. An example of a resulting success can be observed in VIDEO 353.

More recently, the notion that complex policies can be built on top of simple, easily solved optimal control problems by exploiting rich, parametrized reward functions has been exploited within RL more directly. In [15.229, 230], complex policies are derived by adapting a reward function for simple optimal control problems using policy search techniques. *Zucker* and *Bagnell* [15.230] demonstrate that this technique can enable efficient solutions to robotic marble-maze problems that effectively transfer between mazes of varying design and complexity. These works highlight the natural tradeoff between the complexity of the reward function and the complexity of the underlying RL problem for achieving a desired behavior.

15.3.7 Tractability Through Representation

Much of the success of RL methods has been due to the clever use of approximate representations. The need of such approximations is particularly pronounced in robotics, where table-based representations (as discussed in Sect. 15.3.4) are rarely scalable. Common themes are to reduce the dimensionality of states and actions, to choose representations that are robust in face of the data generation process of RL, or to find compact parameterizations of policies and value functions. The following outlines a list of items that are relevant:

- A variety of authors have manually developed discretizations so that basic tasks can be learned on real robots. For low-dimensional tasks, we can generate discretizations straightforwardly by splitting each dimension into a number of regions. The main challenge is to find the right number of regions for each dimension that allows the system to achieve a good final performance while still learning quickly. Example applications include balancing a ball on a beam [15.231], one degree of freedom ball-in-a-cup [15.232], two degree of freedom crawling motions [15.233], and gait patterns for four-legged walking [15.234]. Complex tasks need much more hand crafting, e.g., as in navigation [15.235], vision-based processing [15.236], or RoboCup scenarios [15.237].
- Adaptive representations can be very useful, e.g., as in cooperative task achievement [15.238], or computer vision [15.239].
- Automatic construction of meta-actions (and the closely related concept of options) has fascinated RL researchers. The idea is to have more intelligent actions that are composed of a sequence of commands and that in themselves achieve a simple task. A simple example would be to have a meta-action *move forward* 5 m. For example, in [15.240], the state and action sets are constructed in such a way that repeated action primitives lead to a change in the state to overcome problems associated with the discretization. *Huber* and *Grupen* [15.241] use a set of controllers with associated predicate states as

a basis for learning turning gates with a quadruped. *Fidelman* and *Stone* [15.242] use a policy search approach to learn a small set of parameters that controls the transition between a walking and a capturing meta-action in a RoboCup scenario. A task of transporting a ball with a dog robot [15.243] can be learned with semiautomatically discovered options. Using only the subgoals of primitive motions, a humanoid robot can learn a pouring task [15.244]. Other examples include foraging [15.245] and cooperative tasks [15.246] with multiple robots, grasping with restricted search spaces [15.247], and mobile robot navigation [15.248]. If the meta-actions are not fixed in advance, but rather learned at the same time, these approaches are hierarchical RL approaches. *Konidaris* et al. [15.249, 250] propose an approach that constructs a skill tree from human demonstrations. Here, the skills correspond to options and are chained to learn a mobile manipulation skill.

- In a relational representation, the states, actions, and transitions are not represented individually. Entities of the same predefined type are grouped and their relationships are considered. This representation may be preferable for highly geometric tasks (which frequently appear in robotics) and has been employed to learn to navigate buildings with a real robot in a supervised setting [15.251] and to manipulate articulated objects in simulation [15.252].
- Function approximation has always been the key component that allowed value function methods to scale into interesting domains. Unfortunately, the max-operator used within the Bellman equation and temporal-difference updates can theoretically make most linear or nonlinear approximation schemes unstable for either value iteration or policy iteration. Quite frequently such an unstable behavior is also exhibited in practice. Linear function approximators are stable for policy evaluation, while nonlinear function approximation (e.g., neural networks) can even diverge if just used for policy evaluation [15.206]. A large number of approaches have been suggested to facilitate value function approximation. For instance, creating physics-inspired features as nonlinear basis functions for linear approximator [15.151, 253] can be promising. Generic neural networks can be successful [15.254, 255], in particular the CMAC (cerebellar model articulation controller) neural network [15.256]. Local model function approximators often provide more stable learning behavior [15.151, 257, 258]. Probabilistic nonparametric function approximators like Gaussian process regression has recently shown interesting successes in robot RL [15.259–262].
- Similar to choosing a good structure in value function approximation, choosing a good structure and parameterization for the policy can make a big different for success or failure or RL. Finding sparse parameterization essentially reduces the dimensionality of the learning problem, as only a few parameters need to be learned, and not an action for every time step. Among the most popular have been spline-based parameterizations [15.158, 180, 199], linear parameterizations based on physics-inspired features [15.263], parameterized motor primitives [15.161, 162, 264, 265], Gaussian Mixture Models [15.157, 177, 266], generic neural networks [15.149, 176, 185], and nonparametric approaches [15.175, 194].

15.3.8 Tractability Through Prior Knowledge

Prior knowledge can dramatically help guide the learning process. It can be included in the form of initial policies, demonstrations, initial models, a predefined task structure, or constraints on the policy such as torque limits or ordering constraints of the policy parameters. These approaches significantly reduce the search space and, thus, speed up the learning process. Providing a (partially) successful initial policy allows a RL method to focus on promising regions in the value function or in policy space. Prestructuring a complex task such that it can be broken down into several more tractable ones can significantly reduce the complexity of the learning task. Constraints may also limit the search space, but often pose new, additional problems for the learning methods. For example, policy search limits often do not handle hard limits on the policy well. Relaxing such constraints (a trick often applied in machine learning) is not feasible if they were introduced to protect the robot in the first place.

As indicated in Fig. 15.1, observed expert data can be used for constructing useful policies. Using demonstrations to initialize RL has become rather popular and provides multiple benefits. Perhaps the most obvious benefit is that it provides supervised training data of what actions to perform in states that are encountered. Such data may be helpful when used to bias policy action selection. The most dramatic benefit, however, is that demonstration – or a hand-crafted initial policy – removes the need for *global exploration* of the policy or state-space of the RL problem. The student can improve by locally optimizing a policy knowing what states are important, making local optimization methods feasible. Of course, if the demonstration was not close to the globally optimal behavior, only local optima can be discovered.

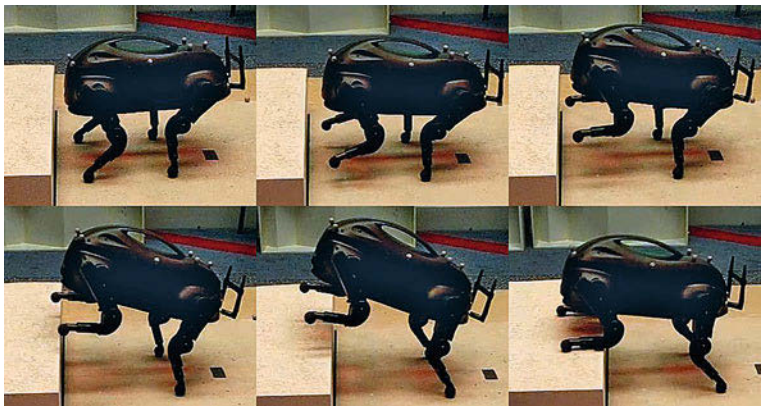


Fig. 15.8 Boston Dynamics LittleDog jumping (after [15.178], courtesy of Zico Kolter)

Demonstrations by a teacher can be obtained in two different scenarios. In the first, the teacher demonstrates the task using his or her own body; in the second, the teacher controls the robot to do the task. The first scenario is limited by the embodiment issue, as the movement of a human teacher usually cannot be mapped directly to the robot due to different physical constraints and capabilities. For example, joint angles of a human demonstrator need to be adapted to account for the kinematic differences between the teacher and the robot. Often it is more advisable to only consider task-relevant information, such as the Cartesian positions and velocities of the end-effector and the object. Demonstrations obtained by motion capture have been used to learn a pendulum swingup [15.174], ball-in-a-cup [15.267], and grasping [15.260].

The second scenario obtains demonstrations by a human teacher directly controlling the robot. Here the human teacher first has to learn how to achieve a task with the particular robot's hardware, adding valuable prior knowledge. If the robot is back-drivable, kinesthetic teach-in (i. e., by taking it by the hand and moving it) can be employed, which enables the teacher to interact more closely with the robot. This method has resulted in applications including T-ball batting [15.161, 162], reaching tasks [15.177, 268], ball-in-a-cup [15.163] and VIDEO 355, flipping a light switch [15.269], playing pool and manipulating a box [15.270], and opening a door and picking up objects [15.271]. A marble maze task can be learned using demonstrations by a human player [15.272].

Often a task can be decomposed hierarchically into basic components or into a sequence of increasingly difficult tasks. In both cases the complexity of the learning task is significantly reduced. For example, hierarchical Q-learning has been used to learn different behavioral levels for a six-legged robot: moving single legs, locally moving the complete body, and globally moving the robot toward a goal [15.273]. A stand-up behavior considered as a hierarchical RL task has been learned using Q-learning in the upper-level and a continuous actor-critic method in the lower level [15.274]. Navigation in a maze can be learned using an actor-critic architecture by tuning the influence of different control modules and learning these modules [15.275]. *Huber* and *Grupen* [15.241] combine discrete event system and RL techniques to learn turning gates for a quadruped. *Hart* and *Grupen* [15.213] learned to bi-manual manipulation tasks by assembling policies hierarchically. *Daniel* et al. [15.276] learn options in a tetherball scenario and *Muelling* et al. [15.147] learn different strokes in a table tennis scenario (VIDEO 354). *Whitman* and *Atkeson* [15.277] show that the optimal policy for some global systems (like a walking controller) can be constructed by finding the optimal controllers for simpler subsystems and coordinating these.

As discussed in Sect. 15.3.2, balancing exploration and exploitation is an important consideration. Task knowledge can be employed to guide to robots curiosity to focus on regions that are novel and promising at the same time. For example, a mobile robot learns to direct attention by employing a modified Q-learning approach using novelty [15.278]. Using *corrected truncated returns* and taking into account the estimator variance, a six-legged robot employed with stepping reflexes can learn to walk [15.279]. Offline search can be used to guide Q-learning during a grasping task [15.280]. Using upper confidence bounds [15.281] to direct exploration into regions with potentially high rewards, grasping can be learned efficiently [15.262].

15.3.9 Tractability Through Models

In Sect. 15.3.2, we discussed robot RL from a model-free perspective where the system simply served as a data generating process. Such model-free reinforce-

ment algorithms try to directly learn the value function or the policy without any explicit modeling of the transition dynamics. In contrast, many robot RL problems can be made tractable by learning forward models, i. e., approximations of the transition dynamics based on data. Such model-based RL approaches jointly learn a model of the system with the value function or the policy and often allow for training with less interaction with the the real environment. Reduced learning on the real robot is highly desirable as simulations are frequently faster than real-time while safer for both the robot and its environment. The idea of combining learning in simulation and in the real environment was popularized by the Dyna-architecture [15.282], prioritized sweeping [15.283], and incremental multi-step Q-learning [15.284] in RL. In robot RL, the learning step on the simulated system is often called *mental rehearsal*.

A few remarks concerning model-based RL are important. It is essentially impossible to obtain a forward model that is accurate enough to simulate a complex real-world robot system without error. Reinforcement learning approaches exploit such model inaccuracies if they are beneficial for the reward received in simulation [15.174]. The resulting policies may work well with the forward model (i. e., the simulator) but poorly on the real system. This effect is known as simulation bias.

Nevertheless, simulation biases can be addressed by introducing stochastic models or distributions over models even if the system is very close to deterministic. Artificially adding a little noise will smooth model errors and avoid policy over-fitting [15.153, 285]. Model learning methods that maintain probabilistic uncertainty about true system dynamics allow the RL algorithm to generate distributions over the performance of a policy. Such methods explicitly model the uncertainty associated with the dynamics at each state and action. For example, when using a Gaussian process model of the transition dynamics, a policy can be evaluated by propagating the state and associated uncertainty forward in time. Such evaluations in the model can be used by a policy search approach to identify where to collect more data to improve a policy, and may be exploited to ensure that control is safe and robust to model uncertainty [15.150, 214]. When the new policy is evaluated on the real system, the novel observations can subsequently be incorporated into the forward model. *Bagnell* and *Schneider* [15.150] showed that maintaining model uncertainty and using it in the inner-loop of a policy search method enabled effective flight control using only minutes of collected data, while performance was compromised by considering a best-fit model. This approach uses explicit Monte Carlo simulation in the sample estimates. By treating model uncertainty as if it were noise [15.214] as well as employing analytic approximations of forward simulation, a cart-pole task can be solved with less than 20 s of interaction with the physical system [15.157]; a visually driven block-stacking task has also been learned data-efficiently [15.164].

15.4 Conclusions

We focussed on the two key branches of robot learning, i. e., supervised learning of internal models with regression methods, and trial-and-error with reinforcement learning. While both topics are traditional topics of machine learning, inserting these learning topics into *robot learning* adds a variety of complexities and intricacies, which are not immediately obvious – indeed many issues have been revealed only through experiments with physical robots. A large number of issues are control theoretic, i. e., due to dealing with a closed-loop physical system that needs to be stable. Other issues are due to the inherent constraints that a physical system imposes, e.g., actuator saturation and the limits of the work space. High-dimensional continuous state–action space causes yet another set of problems, and particularly affects feasibly and efficiency of robot learning. Model-errors and senor noise can have adverse outcome on the learning process. And the fact that a robot needs to generate its own learning data, i. e., the exploration–exploitation tradeoff and ever changing sampling distributions, creates a large number of problems.

We note that there is no clear receipe for robot learning, but rather a large number of ingredients that, when used appropriately, can achieve excellent learning results. As a caveat, even minor incorrect choices can completely prevent any success of learning. That robot learning has yet not reached maturity is obvious from the fact that learning algorithms are rarely used on robots on a daily basis – most results are feasibility studies and not robust enough for daily use. Many results are also overtuned to a particular physical robot, and do not generalize to others easily.

Some of the open questions for the future of robot learning include:

- How to choose appropriate representations automatically for model learning, value function learning, and policy learning? Perhaps, there is one particular choice that robustly works for many different robots, or maybe one could find a small set of possible approaches that can easily be compared.
- How to create useful reward functions? This topic connects to inverse RL, and on a higher level to how to understand the intentions of observed behavior.
- How much can prior knowledge help? How much is needed? How should it be provided?
- How to integrate more tightly with perception? Robot learning is largely *action-centric* and assumes that *perception* is provided. In reality, there is a perception-action-learning loop where the different components interact significantly, and need to be jointly developed.
- How to reduce parameter sensitivity? Manual tuning of hyperparameters such as gradient rates, forgetting rates, exploration rates, etc., are a common curse for the robot learning practitioner, and often, a small change in a parameter determines success or failure.
- How to robustly deal with modeling errors? Models are great when they work, but disastrous if they are not accurate enough. Probabilistic and robust control methods may help, but can also degrade performance due to very conservative learning results.

This list is by no means exhaustive, but captures some of the key issues. In the end, there is strong need for researchers and scientists who are willing to tackle a complex mixture of theoretical problems and experimental problems. Often, just setting up an experimental robot environment with accurate and efficient debugging and visualization tools is a formidable effort. Then, finding the right experiment and the right data trace that allows for error diagnosis in robot learning is still a bit of an art and requires fairly deep insights into physics, algorithms, software architectures, and technology.

Video-References

VIDEO 352 Inverted helicopter hovering
available from http://handbookofrobotics.org/view-chapter/15/videodetails/352

VIDEO 353 Inverse reinforcement
available from http://handbookofrobotics.org/view-chapter/15/videodetails/353

VIDEO 354 Machine learning table tennis
available from http://handbookofrobotics.org/view-chapter/15/videodetails/354

VIDEO 355 Learning motor primitives
available from http://handbookofrobotics.org/view-chapter/15/videodetails/355

References

15.1 S. Schaal: The new robotics – Towards human-centered machines, HFSP J. Front. Interdiscip. Res, Life Sci. **1**(2), 115–126 (2007)

15.2 B.D. Ziebart, A. Maas, J.A. Bagnell, A.K. Dey: Maximum entropy inverse reinforcement learning, AAAI Conf. Artif. Intell. (2008)

15.3 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT, Cambridge 2005)

15.4 B. Apolloni, A. Ghosh, F. Alpaslan, L.C. Jain, S. Patnaik (Eds.): *Machine Learning and Robot Perception*, Stud. Comput. Intell., Vol. 7 (Springer, Berlin, Heidelberg 2005)

15.5 O. Jenkins, R. Bodenheimer, R. Peters: Manipulation manifolds: Explorations into uncovering manifolds in sensory-motor spaces, Int. Conf. Dev. Learn. (2006)

15.6 M. Toussaint: Machine learning and robotics, Tutor. Conf. Mach. Learn. (2011)

15.7 D.P. Bertsekas: *Dynamic Programming and Optimal Control* (Athena Scientific, Nashua 1995)

15.8 R.E. Kalman: When is a linear control system optimal?, J. Basic Eng. **86**(1), 51–60 (1964)

15.9 D. Nguyen-Tuong, J. Peters: Model learning in robotics: A survey, Cogn. Process. **12**(4), 319–340 (2011)

15.10 J. Kober, D. Bagnell, J. Peters: Reinforcement learning in robotics: A survey, Int. J. Robotics Res. **32**(11), 1238–1274 (2013)

15.11 J.H. Connell, S. Mahadevan: *Robot Learning* (Kluwer Academic, Dordrecht 1993)

15.12 J. Ham, Y. Lin, D.D. Lee: Learning nonlinear appearance manifolds for robot localization, Int. Conf. Intell. Robots Syst. (2005)

15.13 R.S. Sutton, A.G. Barto: *Reinforcement Learning* (MIT, Cambridge 1998)

15.14 D. Nguyen-Tuong, J. Peters: Model learning with local Gaussian process regression, Adv. Robotics **23**(15), 2015–2034 (2009)

15.15 J. Nakanishi, R. Cory, M. Mistry, J. Peters, S. Schaal: Operational space control: A theoretical and em-

prical comparison, Int. J. Robotics Res. **27**(6), 737–757 (2008)
15.16 F.R. Reinhart, J.J. Steil: Attractor-based computation with reservoirs for online learning of inverse kinematics, Proc. Eur. Symp. Artif. Neural Netw. (2009)
15.17 J. Ting, M. Kalakrishnan, S. Vijayakumar, S. Schaal: Bayesian kernel shaping for learning control, Adv. Neural Inform. Process. Syst., Vol. 21 (2008) pp. 1673–1680
15.18 J. Steffen, S. Klanke, S. Vijayakumar, H.J. Ritter: Realising dextrous manipulation with structured manifolds using unsupervised kernel regression with structural hints, ICRA 2009 Workshop: Approaches Sens. Learn. Humanoid Robots, Kobe (2009)
15.19 S. Klanke, D. Lebedev, R. Haschke, J.J. Steil, H. Ritter: Dynamic path planning for a 7-dof robot arm, Proc. 2009 IEEE Int. Conf. Intell. Robots Syst. (2006)
15.20 A. Angelova, L. Matthies, D. Helmick, P. Perona: Slip prediction using visual information, Proc. Robotics Sci. Syst., Philadelphia (2006)
15.21 M. Kalakrishnan, J. Buchli, P. Pastor, S. Schaal: Learning locomotion over rough terrain using terrain templates, IEEE Int. Conf. Intell. Robots Syst. (2009)
15.22 N. Hawes, J.L. Wyatt, M. Sridharan, M. Kopicki, S. Hongeng, I. Calvert, A. Sloman, G.-J. Kruijff, H. Jacobsson, M. Brenner, D. Skočaj, A. Vrečko, N. Majer, M. Zillich: The playmate system, Cognit. Syst. **8**, 367–393 (2010)
15.23 D. Skočaj, M. Kristan, A. Vrečko, A. Leonardis, M. Fritz, M. Stark, B. Schiele, S. Hongeng, J.L. Wyatt: Multi-modal learning, Cogn. Syst. **8**, 265–309 (2010)
15.24 O.J. Smith: A controller to overcome dead-time, Instrum. Soc. Am. J. **6**, 28–33 (1959)
15.25 K.S. Narendra, A.M. Annaswamy: *Stable Adaptive Systems* (Prentice Hall, New Jersey 1989)
15.26 S. Nicosia, P. Tomei: Model reference adaptive control algorithms for industrial robots, Automatica **20**, 635–644 (1984)
15.27 J.M. Maciejowski: *Predictive Control with Constraints* (Prentice Hall, New Jersey 2002)
15.28 R.S. Sutton: Dyna, an integrated architecture for learning, planning, and reacting, SIGART Bulletin **2**(4), 160–163 (1991)
15.29 C.G. Atkeson, J. Morimoto: Nonparametric representation of policies and value functions: A trajectory-based approach, Adv. Neural Inform. Process. Syst., Vol. 15 (2002)
15.30 A.Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, E. Liang: Autonomous inverted helicopter flight via reinforcement learning, Proc. 11th Int. Symp. Exp. Robotics (2004)
15.31 C.E. Rasmussen, M. Kuss: Gaussian processes in reinforcement learning, Adv. Neural Inform. Process. Syst., Vol. 16 (2003) pp. 751–758
15.32 A. Rottmann, W. Burgard: Adaptive autonomous control using online value iteration with Gaussian processes, Proc. IEEE Int. Conf. Robotics Autom. (2009)
15.33 J.-J.E. Slotine, W. Li: *Applied Nonlinear Control* (Prentice Hall, Upper Saddle River 1991)
15.34 A. De Luca, P. Lucibello: A general algorithm for dynamic feedback linearization of robots with elastic joints, Proc. IEEE Int. Conf. Robotics Autom. (1998)
15.35 I. Jordan, D. Rumelhart: Forward models: Supervised learning with a distal teacher, Cognit. Sci. **16**, 307–354 (1992)
15.36 D.M. Wolpert, M. Kawato: Multiple paired forward and inverse models for motor control, Neural Netw. **11**, 1317–1329 (1998)
15.37 M. Kawato: Internal models for motor control and trajectory planning, Curr. Opin. Neurobiol. **9**(6), 718–727 (1999)
15.38 D.M. Wolpert, R.C. Miall, M. Kawato: Internal models in the cerebellum, Trends Cogn. Sci. **2**(9), 338–347 (1998)
15.39 N. Bhushan, R. Shadmehr: Evidence for a forward dynamics model in human adaptive motor control, Adv. Neural Inform. Process. Syst., Vol. 11 (1999) pp. 3–9
15.40 K. Narendra, J. Balakrishnan, M. Ciliz: Adaptation and learning using multiple models, switching and tuning, IEEE Control Syst, Mag. **15**(3), 37–51 (1995)
15.41 K. Narendra, J. Balakrishnan: Adaptive control using multiple models, IEEE Trans. Autom. Control **42**(2), 171–187 (1997)
15.42 M. Haruno, D.M. Wolpert, M. Kawato: Mosaic model for sensorimotor learning and control, Neural Comput. **13**(10), 2201–2220 (2001)
15.43 J. Peters, S. Schaal: Learning to control in operational space, Int. J. Robotics Res. **27**(2), 197–212 (2008)
15.44 H. Akaike: Autoregressive model fitting for control, Ann. Inst. Stat. Math. **23**, 163–180 (1970)
15.45 R.M.C. De Keyser, A.R.V. Cauwenberghe: A self-tuning multistep predictor application, Automatica **17**, 167–174 (1980)
15.46 S.S. Billings, S. Chen, G. Korenberg: Identification of mimo nonlinear systems using a forward-regression orthogonal estimator, Int. J. Control **49**, 2157–2189 (1989)
15.47 E. Mosca, G. Zappa, J.M. Lemos: Robustness of multipredictor adaptive regulators: MUSMAR, Automatica **25**, 521–529 (1989)
15.48 J. Kocijan, R. Murray-Smith, C. Rasmussen, A. Girard: Gaussian process model based predictive control, Proc. Am. Control Conf. (2004)
15.49 A. Girard, C.E. Rasmussen, J.Q. Candela, R.M. Smith: Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting, Adv. Neural Inform. Process. Syst., Vol. 15 (2002) pp. 545–552
15.50 C.G. Atkeson, A. Moore, S. Stefan: Locally weighted learning for control, AI Review **11**, 75–113 (1997)
15.51 L. Ljung: *System Identification – Theory for the User* (Prentice-Hall, New Jersey 2004)

15.52 S. Haykin: *Neural Networks: A Comprehensive Foundation* (Prentice Hall, New Jersey 1999)

15.53 J.J. Steil: Backpropagation-decorrelation: Online recurrent learning with O(N) complexity, Proc. Int. Jt. Conf. Neural Netw. (2004)

15.54 C.E. Rasmussen, C.K. Williams: *Gaussian Processes for Machine Learning* (MIT, Cambridge 2006)

15.55 B. Schölkopf, A. Smola: *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond* (MIT, Cambridge 2002)

15.56 K.J. Aström, B. Wittenmark: *Adaptive Control* (Addison Wesley, Boston 1995)

15.57 F.J. Coito, J.M. Lemos: A long-range adaptive controller for robot manipulators, Int. J. Robotics Res. **10**, 684–707 (1991)

15.58 P. Vempaty, K. Cheok, R. Loh: Model reference adaptive control for actuators of a biped robot locomotion, Proc. World Congr. Eng. Comput. Sci. (2009)

15.59 J.R. Layne, K.M. Passino: Fuzzy model reference learning control, J. Intell. Fuzzy Syst. **4**, 33–47 (1996)

15.60 J. Nakanishi, J.A. Farrell, S. Schaal: Composite adaptive control with locally weighted statistical learning, Neural Netw. **18**(1), 71–90 (2005)

15.61 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Prentice Hall, Upper Saddle River 2004)

15.62 M.W. Spong, S. Hutchinson, M. Vidyasagar: *Robot Dynamics and Control* (Wiley, New York 2006)

15.63 S. Schaal, C.G. Atkeson, S. Vijayakumar: Scalable techniques from nonparametric statistics for real-time robot learning, Appl. Intell. **17**(1), 49–60 (2002)

15.64 H. Cao, Y. Yin, D. Du, L. Lin, W. Gu, Z. Yang: Neural network inverse dynamic online learning control on physical exoskeleton, 13th Int. Conf. Neural Inform. Process. (2006)

15.65 C.G. Atkeson, C.H. An, J.M. Hollerbach: Estimation of inertial parameters of manipulator loads and links, Int. J. Robotics Res. **5**(3), 101–119 (1986)

15.66 E. Burdet, B. Sprenger, A. Codourey: Experiments in nonlinear adaptive control, Int. Conf. Robotics Autom. **1**, 537–542 (1997)

15.67 E. Burdet, A. Codourey: Evaluation of parametric and nonparametric nonlinear adaptive controllers, Robotica **16**(1), 59–73 (1998)

15.68 K.S. Narendra, A.M. Annaswamy: Persistent excitation in adaptive systems, Int. J. Control **45**, 127–160 (1987)

15.69 H.D. Patino, R. Carelli, B.R. Kuchen: Neural networks for advanced control of robot manipulators, IEEE Trans. Neural Netw. **13**(2), 343–354 (2002)

15.70 D. Nguyen-Tuong, J. Peters: Incremental sparsification for real-time online model learning, Neurocomputing **74**(11), 1859–1867 (2011)

15.71 D. Nguyen-Tuong, J. Peters: Using model knowledge for learning inverse dynamics, Proc. IEEE Int. Conf. Robotics Autom. (2010)

15.72 S.S. Ge, T.H. Lee, E.G. Tan: Adaptive neural network control of flexible joint robots based on feedback linearization, Int. J. Syst. Sci. **29**(6), 623–635 (1998)

15.73 C.M. Chow, A.G. Kuznetsov, D.W. Clarke: Successive one-step-ahead predictions in multiple model predictive control, Int. J. Control **29**, 971–979 (1998)

15.74 M. Kawato: Feedback error learning neural network for supervised motor learning. In: *Advanced Neural Computers*, ed. by R. Eckmiller (Elsevier, North-Holland, Amsterdam 1990) pp. 365–372

15.75 J. Nakanishi, S. Schaal: Feedback error learning and nonlinear adaptive control, Neural Netw. **17**(10), 1453–1465 (2004)

15.76 T. Shibata, C. Schaal: Biomimetic gaze stabilization based on feedback-error learning with nonparametric regression networks, Neural Netw. **14**(2), 201–216 (2001)

15.77 H. Miyamoto, M. Kawato, T. Setoyama, R. Suzuki: Feedback-error-learning neural network for trajectory control of a robotic manipulator, Neural Netw. **1**(3), 251–265 (1988)

15.78 H. Gomi, M. Kawato: Recognition of manipulated objects by motor learning with modular architecture networks, Neural Netw. **6**(4), 485–497 (1993)

15.79 A. D'Souza, S. Vijayakumar, S. Schaal: Learning inverse kinematics, IEEE Int. Conf. Intell. Robots Syst. (2001)

15.80 S. Vijayakumar, S. Schaal: Locally weighted projection regression: An O(N) algorithm for incremental real time learning in high dimensional space, Proc. 16th Int. Conf. Mach. Learn. (2000)

15.81 M. Toussaint, S. Vijayakumar: Learning discontinuities with products-of-sigmoids for switching between local models, Proc. 22nd Int. Conf. Mach. Learn. (2005)

15.82 J. Tenenbaum, V. de Silva, J. Langford: A global geometric framework for nonlinear dimensionality reduction, Science **290**, 2319–2323 (2000)

15.83 S. Roweis, L. Saul: Nonlinear dimensionality reduction by locally linear embedding, Science **290**, 2323 (2000)

15.84 H. Hoffman, S. Schaal, S. Vijayakumar: Local dimensionality reduction for non-parametric regression, Neural Process. Lett. **29**(2), 109–131 (2009)

15.85 S. Thrun, T. Mitchell: Lifelong robot learning, Robotics Auton. Syst. **15**, 25–46 (1995)

15.86 Y. Engel, S. Mannor, R. Meir: Sparse online greedy support vector regression, Eur. Conf. Mach. Learn. (2002)

15.87 A.J. Smola, B. Schölkopf: A tutorial on support vector regression, Stat. Comput. **14**(3), 199–222 (2004)

15.88 C.E. Rasmussen: Evaluation of Gaussian Processes and Other Methods for Non-Linear Regression (University of Toronto, Toronto 1996)

15.89 L. Bottou, O. Chapelle, D. DeCoste, J. Weston: *Large-Scale Kernel Machines* (MIT, Cambridge 2007)

15.90 J.Q. Candela, C.E. Rasmussen: A unifying view of sparse approximate Gaussian process regression, J. Mach. Learn. Res. **6**, 1939–1959 (2005)

15.91 R. Genov, S. Chakrabartty, G. Cauwenberghs: Silicon support vector machine with online learning, Int. J. Pattern Recognit. Articial Intell. **17**, 385–404 (2003)

15.92 S. Vijayakumar, A. D'Souza, S. Schaal: Incremental online learning in high dimensions, Neural Comput. **12**(11), 2602–2634 (2005)

15.93 B. Schölkopf, P. Simard, A. Smola, V. Vapnik: Prior knowledge in support vector kernel, Adv. Neural Inform. Process. Syst., Vol. 10 (1998) pp. 640–646

15.94 E. Krupka, N. Tishby: Incorporating prior knowledge on features into learning, Int. Conf. Artif. Intell. Stat. (San Juan, Puerto Rico 2007)

15.95 A. Smola, T. Friess, B. Schoelkopf: Semiparametric support vector and linear programming machines, Adv. Neural Inform. Process. Syst., Vol. 11 (1999) pp. 585–591

15.96 B.J. Kröse, N. Vlassis, R. Bunschoten, Y. Motomura: A probabilistic model for appearance-based robot localization, Image Vis. Comput. **19**, 381–391 (2001)

15.97 M.K. Titsias, N.D. Lawrence: Bayesian Gaussian process latent variable model, Proc. 13th Int. Conf. Artif. Intell. Stat. (2010)

15.98 R. Jacobs, M. Jordan, S. Nowlan, G.E. Hinton: Adaptive mixtures of local experts, Neural Comput. **3**, 79–87 (1991)

15.99 S. Calinon, F. D'halluin, E. Sauser, D. Caldwell, A. Billard: A probabilistic approach based on dynamical systems to learn and reproduce gestures by imitation, IEEE Robotics Autom. Mag. **17**, 44–54 (2010)

15.100 V. Treps: A bayesian committee machine, Neural Comput. **12**(11), 2719–2741 (2000)

15.101 L. Csato, M. Opper: Sparse online Gaussian processes, Neural Comput. **14**(3), 641–668 (2002)

15.102 D.H. Grollman, O.C. Jenkins: Sparse incremental learning for interactive robot control policy estimation, IEEE Int. Conf. Robotics Autom., Pasadena (2008)

15.103 M. Seeger: Gaussian processes for machine learning, Int. J. Neural Syst. **14**(2), 69–106 (2004)

15.104 C. Plagemann, S. Mischke, S. Prentice, K. Kersting, N. Roy, W. Burgard: Learning predictive terrain models for legged robot locomotion, Proc. IEEE Int. Conf. Intell. Robots Syst. (2008)

15.105 J. Ko, D. Fox: GP-bayesfilters: Bayesian filtering using Gaussian process prediction and observation models, Auton. Robots **27**(1), 75–90 (2009)

15.106 J.P. Ferreira, M. Crisostomo, A.P. Coimbra, B. Ribeiro: Simulation control of a biped robot with support vector regression, IEEE Int. Symp. Intell. Signal Process. (2007)

15.107 R. Pelossof, A. Miller, P. Allen, T. Jebara: An SVM learning approach to robotic grasping, IEEE Int. Conf. Robotics Autom. (2004)

15.108 J. Ma, J. Theiler, S. Perkins: Accurate on-line support vector regression, Neural Comput. **15**, 2683–2703 (2005)

15.109 Y. Choi, S.Y. Cheong, N. Schweighofer: Local online support vector regression for learning control, Proc. IEEE Int. Symp. Comput. Intell. Robotics Autom. (2007)

15.110 J.-A. Ting, A. D'Souza, S. Schaal: Bayesian robot system identification with input and output noise, Neural Netw. **24**(1), 99–108 (2011)

15.111 S. Nowlan, G.E. Hinton: Evaluation of adaptive mixtures of competing experts, Adv. Neural Inform. Process. Syst., Vol. 3 (1991) pp. 774–780

15.112 V. Treps: Mixtures of Gaussian processes, Adv. Neural Inform. Process. Syst., Vol. 13 (2001) pp. 654–660

15.113 C.E. Rasmussen, Z. Ghahramani: Infinite mixtures of Gaussian process experts, Adv. Neural Inform. Process. Syst., Vol. 14 (2002) pp. 881–888

15.114 T. Hastie, R. Tibshirani, J. Friedman: *The Elements of Statistical Learning* (Springer, New York, 2001)

15.115 W.K. Haerdle, M. Mueller, S. Sperlich, A. Werwatz: *Nonparametric and Semiparametric Models* (Springer, New York 2004)

15.116 D.J. MacKay: A practical Bayesian framework for back-propagation networks, Computation **4**(3), 448–472 (1992)

15.117 R.M. Neal: *Bayesian Learning for Neural Networks*, Lecture Notes in Statistics, Vol. 118 (Springer, New York 1996)

15.118 B. Schölkopf, A.J. Smola, R. Williamson, P.L. Bartlett: New support vector algorithms, Neural Comput. **12**(5), 1207–1245 (2000)

15.119 C. Plagemann, K. Kersting, P. Pfaff, W. Burgard: Heteroscedastic Gaussian process regression for modeling range sensors in mobile robotics, Snowbird Learn. Workshop (2007)

15.120 W.S. Cleveland, C.L. Loader: Smoothing by local regression: Principles and methods. In: *Statistical Theory and Computational Aspects of Smoothing*, ed. by W. Härdle, M.G. Schimele (Physica, Heidelberg 1996)

15.121 J. Fan, I. Gijbels: *Local Polynomial Modelling and Its Applications* (Chapman Hall, New York 1996)

15.122 J. Fan, I. Gijbels: Data driven bandwidth selection in local polynomial fitting, J. R. Stat. Soc. **57**(2), 371–394 (1995)

15.123 A. Moore, M.S. Lee: Efficient algorithms for minimizing cross validation error, Proc. 11th Int. Conf. Mach. Learn. (1994)

15.124 A. Moore: Fast, robust adaptive control by learning only forward models, Adv. Neural Inform. Process. Syst., Vol. 4 (1992) pp. 571–578

15.125 C.G. Atkeson, A.W. Moore, S. Schaal: Locally weighted learning for control, Artif. Intell. Rev. **11**, 75–113 (1997)

15.126 G. Tevatia, S. Schaal: Efficient Inverse Kinematics Algorithms for High-Dimensional Movement Systems (University of Southern California, Los Angeles 2008)

15.127 C.G. Atkeson, A.W. Moore, S. Schaal: Locally weighted learning, Artif. Intell. Rev. **11**(1–5), 11–73 (1997)

15.128 N.U. Edakunni, S. Schaal, S. Vijayakumar: Kernel carpentry for online regression using randomly varying coefficient model, Proc. 20th Int. Jt. Conf. Artif. Intell. (2007)

15.129 D.H. Jacobson, D.Q. Mayne: *Differential Dynamic Programming* (American Elsevier, New York 1973)
15.130 C.G. Atkeson, S. Schaal: Robot learning from demonstration, Proc. 14th Int. Conf. Mach. Learn. (1997)
15.131 J. Morimoto, G. Zeglin, C.G. Atkeson: Minimax differential dynamic programming: Application to a biped walking robot, Proc. 2009 IEEE Int. Conf. Intell. Robots Syst. (2003)
15.132 P. Abbeel, A. Coates, M. Quigley, A.Y. Ng: An application of reinforcement learning to aerobatic helicopter flight, Adv. Neural Inform. Process. Syst., Vol. 19 (2007) pp. 1–8
15.133 P.W. Glynn: Likelihood ratio gradient estimation: An overview, Proc. Winter Simul. Conf. (1987)
15.134 A.Y. Ng, M. Jordan: Pegasus: A policy search method for large MDPs and POMDPs, Proc. 16th Conf. Uncertain. Artif. Intell. (2000)
15.135 B.M. Akesson, H.T. Toivonen: A neural network model predictive controller, J. Process Control **16**(9), 937–946 (2006)
15.136 D. Gu, H. Hu: Predictive control for a car-like mobile robot, Robotics Auton. Syst. **39**, 73–86 (2002)
15.137 E.A. Wan, A.A. Bogdanov: Model predictive neural control with applications to a 6 DOF helicopter model, Proc. Am. Control Conf. (2001)
15.138 O. Khatib: A unified approach for motion and force control of robot manipulators: The operational space formulation, J. Robotics Autom. **3**(1), 43–53 (1987)
15.139 J. Peters, M. Mistry, F.E. Udwadia, J. Nakanishi, S. Schaal: A unifying methodology for robot control with redundant dofs, Auton. Robots **24**(1), 1–12 (2008)
15.140 C. Salaun, V. Padois, O. Sigaud: Control of redundant robots using learned models: An operational space control approach, Proc. IEEE Int. Conf. Intell. Robots Syst. (2009)
15.141 F.R. Reinhart, J.J. Steil: Recurrent neural associative learning of forward and inverse kinematics for movement generation of the redundant PA-10 robot, Symp. Learn. Adapt. Behav. Robotics Syst. (2008)
15.142 J.Q. Candela, C.E. Rasmussen, C.K. Williams: *Large Scale Kernel Machines* (MIT, Cambridge 2007)
15.143 S. Ben-David, R. Schuller: Exploiting task relatedness for multiple task learning, Proc. Conf. Learn. Theory (2003)
15.144 I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun: Large margin methods for structured and interdependent output variables, J. Mach. Learn. Res. **6**, 1453–1484 (2005)
15.145 O. Chapelle, B. Schölkopf, A. Zien: *Semi-Supervised Learning* (MIT, Cambridge 2006)
15.146 J.D. Lafferty, A. McCallum, F.C.N. Pereira: Conditional random fields: Probabilistic models for segmenting and labeling sequence data, Proc. 18th Int. Conf. Mach. Learn. (2001)
15.147 K. Muelling, J. Kober, O. Kroemer, J. Peters: Learning to select and generalize striking movements in robot table tennis, Int. J. Robotics Res. **32**(3), 263–279 (2012)
15.148 S. Mahadevan, J. Connell: Automatic programming of behavior-based robots using reinforcement learning, Artif. Intell. **55**(2/3), 311–365 (1992)
15.149 V. Gullapalli, J.A. Franklin, H. Benbrahim: Acquiring robot skills via reinforcement learning, IEEE Control Syst. Mag. **14**(1), 13–24 (1994)
15.150 J.A. Bagnell, J.C. Schneider: Autonomous helicopter control using reinforcement learning policy search methods, IEEE Int. Conf. Robotics Autom. (2001)
15.151 S. Schaal: Learning from demonstration, Adv. Neural Inform. Process. Syst., Vol. 9 (1996) pp. 1040–1046
15.152 W. B. Powell: AI, OR and Control Theory: A Rosetta Stone for Stochastic Optimization, Tech. Rep. (Princeton University, Princeton 2012)
15.153 C.G. Atkeson: Nonparametric model-based reinforcement learning, Adv. Neural Inform. Process. Syst., Vol. 10 (1998) pp. 1008–1014
15.154 A. Coates, P. Abbeel, A.Y. Ng: Apprenticeship learning for helicopter control, Communication ACM **52**(7), 97–105 (2009)
15.155 R.S. Sutton, A.G. Barto, R.J. Williams: Reinforcement learning is direct adaptive optimal control, Am. Control Conf. (1991)
15.156 A.D. Laud: Theory and Application of Reward Shaping in Reinforcement Learning (University of Illinois, Urbana-Champaign 2004)
15.157 M.P. Deisenrot, C.E. Rasmussen: PILCO: A model-based and data-efficient approach to policy search, 28th Int. Conf. Mach. Learn. (2011)
15.158 H. Miyamoto, S. Schaal, F. Gandolfo, H. Gomi, Y. Koike, R. Osu, E. Nakano, Y. Wada, M. Kawato: A Kendama learning robot based on bidirectional theory, Neural Netw. **9**(8), 1281–1302 (1996)
15.159 N. Kohl, P. Stone: Policy gradient reinforcement learning for fast quadrupedal locomotion, IEEE Int. Conf. Robotics Autom. (2004)
15.160 R. Tedrake, T.W. Zhang, H.S. Seung: Learning to walk in 20 minutes, Yale Workshop Adapt. Learn. Syst. (2005)
15.161 J. Peters, S. Schaal: Reinforcement learning of motor skills with policy gradients, Neural Netw. **21**(4), 682–697 (2008)
15.162 J. Peters, S. Schaal: Natural actor-critic, Neurocomputing **71**(7–9), 1180–1190 (2008)
15.163 J. Kober, J. Peters: Policy search for motor primitives in robotics, Adv. Neural Inform. Process. Syst., Vol. 21 (2009) pp. 849–856
15.164 M.P. Deisenroth, C.E. Rasmussen, D. Fox: Learning to control a low-cost manipulator using data-efficient reinforcement learning. In: *Robotics: Science and Systems VII*, ed. by H. Durrand-Whyte, N. Roy, P. Abbeel (MIT, Cambridge 2011)
15.165 L.P. Kaelbling, M.L. Littman, A.W. Moore: Reinforcement learning: A survey, J. Artif. Intell. Res. **4**, 237–285 (1996)
15.166 M.E. Lewis, M.L. Puterman: *The Handbook of Markov Decision Processes: Methods and Applications* (Kluwer, Dordrecht 2001) pp. 89–111
15.167 J. Peters, S. Vijayakumar, S. Schaal: *Linear Quadratic Regulation as Benchmark for Policy*

Gradient Methods, Technical Report (University of Southern California, Los Angeles 2004)
15.168 R.E. Bellman: *Dynamic Programming* (Princeton Univ. Press, Princeton 1957)
15.169 R.S. Sutton, D. McAllester, S.P. Singh, Y. Mansour: Policy gradient methods for reinforcement learning with function approximation, Adv. Neural Inform. Process. Syst., Vol. 12 (1999) pp. 1057–1063
15.170 T. Jaakkola, M.I. Jordan, S.P. Singh: Convergence of stochastic iterative dynamic programming algorithms, Adv. Neural Inform. Process. Syst., Vol. 6 (1993) pp. 703–710
15.171 J. Rust: Using randomization to break the curse of dimensionality, Econometrica **65**(3), 487–516 (1997)
15.172 D.E. Kirk: *Optimal Control Theory* (Prentice-Hall, Englewood Cliffs 1970)
15.173 A. Schwartz: A reinforcement learning method for maximizing undiscounted rewards, Int. Conf. Mach. Learn. (1993)
15.174 C.G. Atkeson, S. Schaal: Robot learning from demonstration, Int. Conf. Mach. Learn. (1997)
15.175 J. Peters, K. Muelling, Y. Altun: Relative entropy policy search, Natl. Conf. Artif. Intell. (2010)
15.176 G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, G. Cheng: Learning CPG-based biped locomotion with a policy gradient method: Application to a humanoid robot, Int. J. Robotics Res. **27**(2), 213–228 (2008)
15.177 F. Guenter, M. Hersch, S. Calinon, A. Billard: Reinforcement learning for imitating constrained reaching movements, Adv. Robotics **21**(13), 1521–1544 (2007)
15.178 J.Z. Kolter, A.Y. Ng: Policy search via the signed derivative, Robotics Sci. Syst. V, Seattle (2009)
15.179 A.Y. Ng, H.J. Kim, M.I. Jordan, S. Sastry: Autonomous helicopter flight via reinforcement learning, Adv. Neural Inform. Process. Syst., Vol. 16 (2004) pp. 799–806
15.180 J.W. Roberts, L. Moret, J. Zhang, R. Tedrake: From motor to interaction learning in robots, Stud. Comput. Intell. **264**, 293–309 (2010)
15.181 R. Tedrake: Stochastic policy gradient reinforcement learning on a simple 3D biped, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2004)
15.182 F. Stulp, E. Theodorou, M. Kalakrishnan, P. Pastor, L. Righetti, S. Schaal: Learning motion primitive goals for robust manipulation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2011)
15.183 M. Strens, A. Moore: Direct policy search using paired statistical tests, Int. Conf. Mach. Learn. (2001)
15.184 A.Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, E. Liang: Autonomous inverted helicopter flight via reinforcement learning, Int. Symp. Exp. Robotics (2004)
15.185 T. Geng, B. Porr, F. Wörgötter: Fast biped walking with a reflexive controller and real-time policy searching, Adv. Neural Inform. Process. Syst., Vol. 18 (2006) pp. 427–434
15.186 N. Mitsunaga, C. Smith, T. Kanda, H. Ishiguro, N. Hagita: Robot behavior adaptation for human-robot interaction based on policy gradient reinforcement learning, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2005)
15.187 M. Sato, Y. Nakamura, S. Ishii: Reinforcement learning for biped locomotion, Int. Conf. Artif. Neural Netw. (2002)
15.188 R.Y. Rubinstein, D.P. Kroese: *The Cross Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation* (Springer, New York 2004)
15.189 D.E. Goldberg: *Genetic Algorithms* (Addision Wesley, New York 1989)
15.190 J.T. Betts: *Practical Methods for Optimal Control Using Nonlinear Programming*, Adv. Design Control, Vol. 3 (SIAM, Philadelphia 2001)
15.191 R.J. Williams: Simple statistical gradient-following algorithms for connectionist reinforcement learning, Mach. Learn. **8**, 229–256 (1992)
15.192 P. Dayan, G.E. Hinton: Using expectation-maximization for reinforcement learning, Neural Comput. **9**(2), 271–278 (1997)
15.193 N. Vlassis, M. Toussaint, G. Kontes, S. Piperidis: Learning model-free robot control by a Monte Carlo EM algorithm, Auton. Robots **27**(2), 123–130 (2009)
15.194 J. Kober, E. Oztop, J. Peters: Reinforcement learning to adjust robot movements to new situations, Proc. Robotics Sci. Syst. Conf. (2010)
15.195 E.A. Theodorou, J. Buchli, S. Schaal: Reinforcement learning of motor skills in high dimensions: A path integral approach, IEEE Int. Conf. Robotics Autom. (2010)
15.196 J.A. Bagnell, A.Y. Ng, S. Kakade, J. Schneider: Policy search by dynamic programming, Adv. Neural Inform. Process. Syst., Vol. 16 (2003) pp. 831–838
15.197 T. Kollar, N. Roy: Trajectory optimization using reinforcement learning for map exploration, Int. J. Robotics Res. **27**(2), 175–197 (2008)
15.198 D. Lizotte, T. Wang, M. Bowling, D. Schuurmans: Automatic gait optimization with Gaussian process regression, Int. Jt. Conf. Artif. Intell. (2007)
15.199 S. Kuindersma, R. Grupen, A.G. Barto: Learning dynamic arm motions for postural recovery, IEEE-RAS Int. Conf. Humanoid Robots (2011)
15.200 M. Tesch, J.G. Schneider, H. Choset: Using response surfaces and expected improvement to optimize snake robot gait parameters, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2011)
15.201 S.-J. Yi, B.-T. Zhang, D. Hong, D.D. Lee: Learning full body push recovery control for small humanoid robots, IEEE Proc. Int. Conf. Robotics Autom. (2011)
15.202 J.A. Boyan, A.W. Moore: Generalization in reinforcement learning: Safely approximating the value function, Adv. Neural Inform. Process. Syst., Vol. 7 (1995) pp. 369–376
15.203 S. Kakade, J. Langford: Approximately optimal approximate reinforcement learning, Int. Conf. Mach. Learn. (2002)
15.204 E. Greensmith, P.L. Bartlett, J. Baxter: Variance reduction techniques for gradient estimates in

reinforcement learning, J. Mach. Learn. Res. **5**, 1471–1530 (2004)
15.205 M.T. Rosenstein, A.G. Barto: Reinforcement learning with supervision by a stable controller, Am. Control Conf. (2004)
15.206 J.N. Tsitsiklis, B. Van Roy: An analysis of temporal-difference learning with function approximation, IEEE Trans. Autom. Control **42**(5), 674–690 (1997)
15.207 J.Z. Kolter, A.Y. Ng: Regularization and feature selection in least-squares temporal difference learning, Int. Conf. Mach. Learn. (2009)
15.208 L.C. Baird, H. Klopf: *Reinforcement Learning with High-Dimensional Continuous Actions*, Technical Report WL-TR-93-1147 (Wright-Patterson Air Force Base, Dayton 1993)
15.209 G.D. Konidaris, S. Osentoski, P. Thomas: Value function approximation in reinforcement learning using the Fourier basis, AAAI Conf. Artif. Intell. (2011)
15.210 J. Peters, K. Muelling, J. Kober, D. Nguyen-Tuong, O. Kroemer: Towards motor skill learning for robotics, Int. Symp. Robotics Res. (2010)
15.211 L. Buşoniu, R. Babuška, B. de Schutter, D. Ernst: *Reinforcement Learning and Dynamic Programming Using Function Approximators* (CRC, Boca Raton 2010)
15.212 A.G. Barto, S. Mahadevan: Recent advances in hierarchical reinforcement learning, Discret. Event Dyn. Syst. **13**(4), 341–379 (2003)
15.213 S. Hart, R. Grupen: Learning generalizable control programs, IEEE Trans. Auton. Mental Dev. **3**(3), 216–231 (2011)
15.214 J.G. Schneider: Exploiting model uncertainty estimates for safe dynamic control learning, Adv. Neural Inform. Process. Syst., Vol. 9 (1997) pp. 1047–1053
15.215 J.A. Bagnell: Learning Decisions: Robustness, Uncertainty, and Approximation. Dissertation (Robotics Institute, Carnegie Mellon University, Pittsburgh 2004)
15.216 T.M. Moldovan, P. Abbeel: Safe exploration in markov decision processes, 29th Int. Conf. Mach. Learn. (2012)
15.217 T. Hester, M. Quinlan, P. Stone: RTMBA: A real-time model-based reinforcement learning architecture for robot control, IEEE Int. Conf. Robotics Autom. (2012)
15.218 C.G. Atkeson: Using local trajectory optimizers to speed up global optimization in dynamic programming, Adv. Neural Inform. Process. Syst., Vol. 6 (1994) pp. 663–670
15.219 J. Kober, J. Peters: Policy search for motor primitives in robotics, Mach. Learn. **84**(1/2), 171–203 (2010)
15.220 S. Russell: Learning agents for uncertain environments (extended abstract), Conf. Comput. Learn. Theory (1989)
15.221 P. Abbeel, A.Y. Ng: Apprenticeship learning via inverse reinforcement learning, Int. Conf. Mach. Learn. (2004)
15.222 N.D. Ratliff, J.A. Bagnell, M.A. Zinkevich: Maximum margin planning, Int. Conf. Mach. Learn. (2006)
15.223 R.L. Keeney, H. Raiffa: *Decisions with Multiple Objectives: Preferences and Value Tradeoffs* (Wiley, New York 1976)
15.224 N. Ratliff, D. Bradley, J.A. Bagnell, J. Chestnutt: Boosting structured prediction for imitation learning, Adv. Neural Inform. Process. Syst., Vol. 19 (2006) pp. 1153–1160
15.225 D. Silver, J.A. Bagnell, A. Stentz: High performance outdoor navigation from overhead data using imitation learning. In: *Robotics: Science and Systems*, Vol. IV, ed. by O. Brock, J. Trinkle, F. Ramos (MIT, Cambridge 2008)
15.226 D. Silver, J.A. Bagnell, A. Stentz: Learning from demonstration for autonomous navigation in complex unstructured terrain, Int. J. Robotics Res. **29**(12), 1565–1592 (2010)
15.227 N. Ratliff, J.A. Bagnell, S. Srinivasa: Imitation learning for locomotion and manipulation, IEEE-RAS Int. Conf. Humanoid Robots (2007)
15.228 J.Z. Kolter, P. Abbeel, A.Y. Ng: Hierarchical apprenticeship learning with application to quadruped locomotion, Adv. Neural Inform. Process. Syst., Vol. 20 (2007) pp. 769–776
15.229 J. Sorg, S.P. Singh, R.L. Lewis: Reward design via online gradient ascent, Adv. Neural Inform. Process. Syst., Vol. 23 (2010) pp. 2190–2198
15.230 M. Zucker, J.A. Bagnell: Reinforcement planning: RL for optimal planners, IEEE Proc. Int. Conf. Robotics Autom. (2012)
15.231 H. Benbrahim, J.S. Doleac, J.A. Franklin, O.G. Selfridge: Real-time learning: A ball on a beam, Int. Jt. Conf. Neural Netw. (1992)
15.232 B. Nemec, M. Zorko, L. Zlajpah: Learning of a ball-in-a-cup playing robot, Int. Workshop Robotics, Alpe-Adria-Danube Region (2010)
15.233 M. Tokic, W. Ertel, J. Fessler: The crawler, a class room demonstrator for reinforcement learning, Int. Fla. Artif. Intell. Res. Soc. Conf. (2009)
15.234 H. Kimura, T. Yamashita, S. Kobayashi: Reinforcement learning of walking behavior for a four-legged robot, IEEE Conf. Decis. Control (2001)
15.235 R.A. Willgoss, J. Iqbal: Reinforcement learning of behaviors in mobile robots using noisy infrared sensing, Aust. Conf. Robotics Autom. (1999)
15.236 L. Paletta, G. Fritz, F. Kintzler, J. Irran, G. Dorffner: Perception and developmental learning of affordances in autonomous robots, Lect. Notes Comput. Sci. **4667**, 235–250 (2007)
15.237 C. Kwok, D. Fox: Reinforcement learning for sensing strategies, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2004)
15.238 T. Yasuda, K. Ohkura: A reinforcement learning technique with an adaptive action generator for a multi-robot system, Int. Conf. Simul. Adapt. Behav. (2008)
15.239 J.H. Piater, S. Jodogne, R. Detry, D. Kraft, N. Krüger, O. Kroemer, J. Peters: Learning visual representations for perception-action systems, Int. J. Robotics Res. **30**(3), 294–307 (2011)
15.240 M. Asada, S. Noda, S. Tawaratsumida, K. Hosoda: Purposive behavior acquisition for a real robot by vision-based reinforcement learning, Mach. Learn. **23**(2/3), 279–303 (1996)

15.241 M. Huber, R.A. Grupen: A feedback control structure for on-line learning tasks, Robotics Auton. Syst. **22**(3/4), 303–315 (1997)

15.242 P. Fidelman, P. Stone: Learning ball acquisition on a physical robot, Int. Symp. Robotics Autom. (2004)

15.243 V. Soni, S.P. Singh: Reinforcement learning of hierarchical skills on the Sony AIBO robot, Int. Conf. Dev. Learn. (2006)

15.244 B. Nemec, M. Tamošiunaitė, F. Wörgötter, A. Ude: Task adaptation through exploration and action sequencing, IEEE-RAS Int. Conf. Humanoid Robots (2009)

15.245 M.J. Matarić: Reinforcement learning in the multi-robot domain, Auton. Robots **4**, 73–83 (1997)

15.246 M.J. Matarić: Reward functions for accelerated learning, Int. Conf. Mach. Learn. (ICML) (1994)

15.247 R. Platt, R.A. Grupen, A.H. Fagg: Improving grasp skills using schema structured learning, Int. Conf. Dev. Learn. (2006)

15.248 M. Dorigo, M. Colombetti: *Robot Shaping: Developing Situated Agents Through Learning*, Technical Report (International Computer Science Institute, Berkeley 1993)

15.249 G.D. Konidaris, S. Kuindersma, R. Grupen, A.G. Barto: Autonomous skill acquisition on a mobile manipulator, AAAI Conf. Artif. Intell. (2011)

15.250 G.D. Konidaris, S. Kuindersma, R. Grupen, A.G. Barto: Robot learning from demonstration by constructing skill trees, Int. J. Robotics Res. **31**(3), 360–375 (2012)

15.251 A. Cocora, K. Kersting, C. Plagemann, W. Burgard, L. de Raedt: Learning relational navigation policies, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2006)

15.252 D. Katz, Y. Pyuro, O. Brock: Learning to manipulate articulated objects in unstructured environments using a grounded relational representation. In: *Robotics: Science and Systems*, Vol. IV, ed. by O. Brock, J. Trinkle, F. Ramos (MIT, Cambridge 2008)

15.253 C.H. An, C.G. Atkeson, J.M. Hollerbach: *Model-Based Control of a Robot Manipulator* (MIT, Press, Cambridge 1988)

15.254 C. Gaskett, L. Fletcher, A. Zelinsky: Reinforcement learning for a vision based mobile robot, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2000)

15.255 Y. Duan, B. Cui, H. Yang: Robot navigation based on fuzzy RL algorithm, Int. Symp. Neural Netw. (2008)

15.256 H. Benbrahim, J.A. Franklin: Biped dynamic walking using reinforcement learning, Robotics Auton. Syst. **22**(3/4), 283–302 (1997)

15.257 W.D. Smart, L. Pack Kaelbling: A framework for reinforcement learning on real robots, Natl. Conf. Artif. Intell./Innov. Appl. Artif. Intell. (1989)

15.258 D.C. Bentivegna: *Learning from Observation Using Primitives* (Georgia Institute of Technology, Atlanta 2004)

15.259 A. Rottmann, C. Plagemann, P. Hilgers, W. Burgard: Autonomous blimp control using model-free reinforcement learning in a continuous state and action space, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2007)

15.260 K. Gräve, J. Stückler, S. Behnke: Learning motion skills from expert demonstrations and own experience using Gaussian process regression, Jt. Int. Symp. Robotics (ISR) Ger. Conf. Robotics (ROBOTIK) (2010)

15.261 O. Kroemer, R. Detry, J. Piater, J. Peters: Active learning using mean shift optimization for robot grasping, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2009)

15.262 O. Kroemer, R. Detry, J. Piater, J. Peters: Combining active learning and reactive control for robot grasping, Robotics Auton. Syst. **58**(9), 1105–1116 (2010)

15.263 T. Tamei, T. Shibata: Policy gradient learning of cooperative interaction with a robot using user's biological signals, Int. Conf. Neural Inf. Process. (2009)

15.264 A.J. Ijspeert, J. Nakanishi, S. Schaal: Learning attractor landscapes for learning motor primitives, Adv. Neural Inform. Process. Syst., Vol. 15 (2003) pp. 1547–1554

15.265 S. Schaal, P. Mohajerian, A.J. Ijspeert: Dynamics systems vs. optimal control – A unifying view, Prog. Brain Res. **165**(1), 425–445 (2007)

15.266 H.-I. Lin, C.-C. Lai: Learning collision-free reaching skill from primitives, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2012)

15.267 J. Kober, B. Mohler, J. Peters: Learning perceptual coupling for motor primitives, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2008)

15.268 S. Bitzer, M. Howard, S. Vijayakumar: Using dimensionality reduction to exploit constraints in reinforcement learning, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2010)

15.269 J. Buchli, F. Stulp, E. Theodorou, S. Schaal: Learning variable impedance control, Int. J. Robotics Res. **30**(7), 820–833 (2011)

15.270 P. Pastor, M. Kalakrishnan, S. Chitta, E. Theodorou, S. Schaal: Skill learning and task outcome prediction for manipulation, IEEE Int. Conf. Robotics Autom. (2011)

15.271 M. Kalakrishnan, L. Righetti, P. Pastor, S. Schaal: Learning force control policies for compliant manipulation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2011)

15.272 D.C. Bentivegna, C.G. Atkeson, G. Cheng: Learning from observation and practice using behavioral primitives: Marble maze, 11th Int. Symp. Robotics Res. (2004)

15.273 F. Kirchner: Q-learning of complex behaviours on a six-legged walking machine, EUROMICRO Workshop Adv. Mobile Robots (1997)

15.274 J. Morimoto, K. Doya: Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning, Robotics Auton. Syst. **36**(1), 37–51 (2001)

15.275 J.-Y. Donnart, J.-A. Meyer: Learning reactive and planning rules in a motivationally autonomous animat, Syst. Man Cybern. B **26**(3), 381–395 (1996)

15.276 C. Daniel, G. Neumann, J. Peters: Learning concurrent motor skills in versatile solution spaces, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2012)
15.277 E.C. Whitman, C.G. Atkeson: Control of instantaneously coupled systems applied to humanoid walking, IEEE-RAS Int. Conf. Humanoid Robots (2010)
15.278 X. Huang, J. Weng: Novelty and reinforcement learning in the value system of developmental robots, 2nd Int. Workshop Epigenetic Robotics Model. Cognit. Dev. Robotic Syst. (2002)
15.279 M. Pendrith: Reinforcement learning in situated agents: Some theoretical problems and practical solutions, Eur. Workshop Learn. Robots (1999)
15.280 B. Wang, J.W. Li, H. Liu: A heuristic reinforcement learning for robot approaching objects, IEEE Conf. Robotics Autom. Mechatron. (2006)
15.281 L.P. Kaelbling: Learning in Embedded Systems (Stanford University, Stanford 1990)
15.282 R.S. Sutton: Integrated architectures for learning, planning, and reacting based on approximating dynamic programming, Int. Conf. Mach. Learn. (1990)
15.283 A.W. Moore, C.G. Atkeson: Prioritized sweeping: Reinforcement learning with less data and less time, Mach. Learn. **13**(1), 103–130 (1993)
15.284 J. Peng, R.J. Williams: Incremental multi-step Q-learning, Mach. Learn. **22**(1), 283–290 (1996)
15.285 N. Jakobi, P. Husbands, I. Harvey: Noise and the reality gap: The use of simulation in evolutionary robotics, 3rd Eur. Conf. Artif. Life (1995)

Multimedia Contents

Part B Design

Part B Design

Ed. by Frank C. Park

16 Design and Performance Evaluation
Jorge Angeles, Montreal, Canada
Frank C. Park, Seoul, Korea

17 Limbed Systems
Shuuji Kajita, Tsukuba, Japan
Christian Ott, Wessling, Germany

18 Parallel Mechanisms
Jean-Pierre Merlet, Sophia-Antipolis, France
Clément Gosselin, Quebec, Canada
Tian Huang, Tianjin, China

19 Robot Hands
Claudio Melchiorri, Bologna, Italy
Makoto Kaneko, Suita, Japan

20 Snake-Like and Continuum Robots
Ian D. Walker, Clemson, USA
Howie Choset, Pittsburgh, USA
Gregory S. Chirikjian, Baltimore, USA

21 Actuators for Soft Robotics
Alin Albu-Schäffer, Wessling, Germany
Antonio Bicchi, Pisa, Italy

22 Modular Robots
I-Ming Chen, Singapore, Singapore
Mark Yim, Philadelphia, USA

23 Biomimetic Robots
Kyu-Jin Cho, Seoul, Korea
Robert Wood, Cambridge, USA

24 Wheeled Robots
Woojin Chung, Seoul, Korea
Karl Iagnemma, Cambridge, USA

25 Underwater Robots
Hyun-Taek Choi, Daejeon, Korea
Junku Yuh, Seoul, Korea

26 Flying Robots
Stefan Leutenegger, London, UK
Christoph Hürzeler, Baden, Switzerland
Amanda K. Stowers, Stanford, USA
Kostas Alexis, Zurich, Switzerland
Markus W. Achtelik, Zurich, Switzerland
David Lentink, Stanford, USA
Paul Y. Oh, Las Vegas, USA
Roland Siegwart, Zurich, Switzerland

27 Micro-/Nanorobots
Bradley J. Nelson, Zurich, Switzerland
Lixin Dong, East Lansing, USA
Fumihito Arai, Nagoya, Japan

The chapters contained in Part B, Design, are concerned with the design and modeling of the actual physical realizations of a robot. Some of the more obvious mechanical structures that come to mind are arms, legs, and hands. To this list we can add wheeled vehicles and platforms; snake-like and continuum robots; robots capable of swimming and flying; and robot structures at the micro- and nanoscales. Even for that most basic robotic device, the arm, an incredibly diverse set of structures is possible, depending on the number and types of joints and actuators, and the presence of closed loops in the kinematic structure, or flexibility in the joints and links. Constructing models, and planning and control algorithms for these diverse structures represents an even greater set of challenges.

The topics addressed in these chapters are essential to creating not only the physical robot itself, but also to creating and controlling movements, and manipulating objects in desired ways. As such the connections with the chapters on Robot Foundations (Part A) – particularly the chapters on Kinematics (Chap. 1), Dynamics (Chap. 2), and Mechanisms and Actuation (Chap. 3) – are self-evident. What ultimately distinguishes robotics from other disciplines that study intelligence is that, by definition, robots require a physical manifestation, and by extension must physically interact with the environment. In this regard the topics addressed in these chapters can be said to constitute the most basic layer of this endeavor.

Just as it is difficult to examine human intelligence from a purely abstract perspective, remotely detached from the physical body, so it is difficult to separate the contents of the remaining parts without including in the discussion the actual medium of interaction with the physical world, the (physical) robots themselves. For example, the question of how to coordinate sensing and perception with action (Part C), how to grasp and manipulate objects (Part D), and how to teach robots to move in the world (Part E), must inevitably consider the physical structure of the robot. Robots specialized to various applications and environments (Part F), particularly those intended for direct interaction with humans (Part G), naturally must also consider the robot's physical structure.

With this overview of Part B, we now provide a brief synopsis of each chapter.

Chapter 16, Performance Evaluation and Design Criteria, provides a concise overview of the robot design process, and surveys some of the criteria and tools used in the mechanical design and performance evaluation of robots. Criteria such as workspace volume, local and global dexterity, and elastostatic and elastodynamic performance are not only applicable to determining the topological structure and physical dimensions of the robot, but can also be useful for, e.g., workpiece placement and kinematic redundancy resolution.

Chapter 17, Limbed Systems, discusses the myriad issues involved in the design, analysis, and control of robots with limbs. Defining a limbed system as a robot consisting of a body and at least one limb such that it is able to support and propel itself, the chapter begins with an overview of the design process for limbed systems,from conceptual to detailed, and the basic dynamics of passive and controlled walking. The chapter also examines numerous case studies illustrating the diversity of limbed robot designs, and schemes for their actuation and locomotion. Multi-legged robots, such as dynamic quadrupeds inspired by mammals and behavior-based multilegged robots, are also discussed, as are hybrid leg-wheel-arm robots, tethered walking robots, and even legged robots capable of climbing walls.

Chapter 18, Parallel Mechanisms and Robots, presents an introduction to the kinematics and dynamics of parallel mechanisms such as the well-known Stewart–Gough platform. Parallel mechanisms contain closed loops in their kinematic structure, and as such methods for their analysis differ considerably from those for their serial counterparts. This chapter discusses topics ranging from type synthesis and forward and inverse kinematic solutions of parallel mechanisms, to an investigation of their singularity behavior, workspace characterization, static and dynamic analysis, and practical issues in their design.

Chapter 19, Robot Hands, investigates the principal issues behind the design, modeling, and control of robot hands. Beginning with a discussion of levels of anthropomorphism, and the characterization of robot hand dexterity, the chapter investigates the relevant design issues for robot hands, actuation and transmission architectures, and available sensing technologies. The dynamic modeling and control of robot hands are made challenging not only by the complex kinematic structure, but also by the flexible transmission elements, and the chapter devotes particular attention to these issues.

Chapter 20, Snake-Like and Continuum Robots, begins with a history of snake robots, starting with the pioneering work of Shigeo Hirosein in the early 1970s. While snake-like and continuum robots have very similar exterior appearances, there is considerable diversity in their mechanical design, and the ways in which they are actuated.This chapter describes the mechanical design, actuation, modeling,motion planning, and control of such robots. The chapter also provides case studies of a wide range of existing snake-like and continuum

robots that illustrate the diversity of designs as well as applications.

Chapter 21, Soft Robots, begins with the premise that robots of the future will not resemble the bulky rigid machines found in today's factory floors, but will be compliant and adaptable, and able to safely interact with humans – in other words, soft. This chapter discusses the design, modeling, and control of actuators for this new generation of soft robots.The chapter surveys the different principles and technologies that can be used to design and implement actuators for soft robotics. Many of the concepts are organized so as to allow a direct analogy with natural muscles. Variable impedance actuators are examined in some detail, from their mathematical modeling to motion and force planning and control.

Chapter 22, Modular Robots, provides an overview of the design of modular robots. The chapter begins with a discussion of the concept of modularity, and a definition and classification of modular robots. The chapter then examines reconfigurable modular manipulators, from the earliest designs that were conceived in an industrial automation setting, to more recent self-reconfigurable modular robots. Issues related to the design of modules and the interfaces between them, and the determination of optimal configurations, are also discussed in this chapter.

Chapter 23, Biomimetic Robots, broadly examines the ways in which biological principles can be applied to the design of robotic mechanisms. The challenges of biomimetic design include developing a deep understanding of the relevant natural system, and translating this understanding into a set of engineering design rules; this often entails the development of novel fabrication and actuation to realize the biomimetic design. This chapter discusses the basic design principles underlying biomimetic robots and their contrast with bio-inspired robots, and the fundamental components for developing a biomimetic robot. The chapter also provides detailed reviews of biomimetic designs that have been developed for flapping-wing flight, jumping, crawling, wall climbing, and swimming, as well as the enabling material and fabrication technologies for these biomimetic designs.

Chapter 24, Wheeled Robots, provides a general and comprehensive description of wheeled mobile robots. The chapter begins with a discussion of robot mobility based on the types of wheels and the nature of the kinematic constraints, followed by a classification of wheeled robot structures according to the number and type of wheels and how they are arranged. Omnimobile robots and articulated robot realizations are described, and wheel–terrain interaction models for computing contact forces are also presented. The chapter concludes with a classification of wheel–terrain interaction cases depending on the relative stiffnesses of the wheel and terrain, and the structure and dyamics of suspension systems that enable movement of wheeled robots over uneven surfaces.

Chapter 25, Underwater Robots, examines the design issues for underwater robots, with a focus on remotely operated vehicles and autonomous underwater vehicles. The major components of an underwater robot – from the mechanical elements and subsystems including any attached manipulators, to power sources, actuators and sensors, and architectures for computing and communications and control, are discussed in this chapter. Aspects of the mathematical modeling and control of underwater robots is covered in a separate chapter later in this handbook.

Chapter 26, Flying Robots, provides an overview of the core elements of flying robots. The reader will be guided through the design process of aerial robots, beginning with a qualitative characterization of the different types of flying robot. Design and modeling are particularly closely intertwined in flying robots, and the chapter provides an overview of the underlying aerodynamics and tools for their analysis. The chapter then shows how these tools can be applied to the design and analysis of various types of flying robots, including fixed-wing, rotary-wing, and flapping wing systems, with case studies illustrating these design principles.

Chapter 27, Micro/Nanorobots, provides an overview of the state of the art in micro- and nanorobotics. The former entails robotic manipulation of objects with dimensions in the millimeter to micrometer range, as well as the design and fabrication of autonomous robotic agents within this size range (nanorobotics is defined in the same way, but for dimensions smaller than a micrometer). The chapter outlines scaling effects, actuation, and sensing and fabrication at these scales, and also applications to microassembly, biotechnology, and the construction and characterization of micro- and nano-electromechanical systems.

Multimedia Contents

16. Design and Performance Evaluation

Jorge Angeles, Frank C. Park

In this chapter we survey some of the tools and criteria used in the mechanical design and performance evaluation of robots. Our focus is on robots that are (a) primarily intended for manipulation tasks and (b) constructed with one or more serial kinematic chains. The kinematics of parallel robots is addressed in detail in Chap. 18; their elastostatics is the subject of Sect. 16.5.1. Wheeled robots, walking robots, multifingered hands, and robots intended for outdoor applications, i. e., those encompassing what is known as *field robotics*, are studied in their own chapters; here we provide an overview of the main classes of these robots as relating to design.

16.1 **The Robot Design Process** 400
16.2 **Workspace Criteria** 401
16.2.1 Reaching a Set of Goal Frames 403
16.2.2 Workspace Volume and Topology 404
16.3 **Dexterity Indices** 405
16.3.1 Local Dexterity for Open Chains 405
16.3.2 Dynamics-Based Local Performance Evaluation 407
16.3.3 Global Dexterity Measures 407
16.3.4 Closed-Chain Dexterity Indices ... 407
16.3.5 Alternative Dexterity-Like Measures 407
16.4 **Other Performance Indices** 408
16.4.1 Acceleration Radius 408
16.4.2 Elastostatic Performance 408
16.4.3 Elastodynamic Performance 410
16.5 **Other Robot Types** 411
16.5.1 Robots Mounted on a Fixed Base: Parallel-Kinematics Machines..... 412
16.5.2 Mobile Robots 414
16.5.3 Aquatic Robots 415
16.5.4 Flying Robots 416
16.6 **Summary** 416
References 416

The most obvious application of the criteria and tools described in this chapter is in the mechanical design of a robot. Robot design differs from the design of 1-degree-of-freedom (DOF) machinery in that the latter is intended for one specific task, e.g., picking up a workpiece from a belt conveyor and placing it on a magazine. Moreover, the conveyor is synchronized with the manipulating machine and the magazine is stationary, with well-defined locations where each workpiece is to be placed. Manipulation robots, in contrast, are not intended for one specific task, but rather for a *family of tasks* falling within one class of workpiece motions, e.g., planar, spherical, translational, or motions produced by systems of the selective compliance assembly robot arm (SCARA) type, also known as *Schönflies displacements* [16.1]. The challenge that robot designers face is therefore one of *uncertainty* in the specific task that the robot will be required to execute. Design criteria have been devised to help the designer cope with uncertainty, as discussed herein.

16.1 The Robot Design Process

Given a family of tasks that constitute the *functional requirements* in the design process, besides more-detailed *design specifications*, the role of the designer consists in producing a robot that will meet all the requirements and specifications. The various stages in the robot design job at hand are intended to:

1. Determine the *topology* of the kinematic chain underlying the mechanical structure. Under this item we consider first the robot type: serial, parallel or hybrid. Then, a decision is to be made on the layout of the various *subchains* in terms of the type of joints, most commonly, revolute and prismatic. Recently, one additional type has been recognized to be equally useful, the Π-joint, coupling two links under relative translation by means of two other links undergoing identical angular displacements, although about different parallel axes. The four links form a parallelogram four-bar linkage [16.2].
2. Determine the *geometric dimensions* of the various links defining the *robotic architecture*, as required to fill a table of *Denavit–Hartenberg (DH) parameters* [16.3] so as to satisfy workspace requirements. Although these parameters are usually understood to include the joint variables, these variables do not affect the *robot architecture*; they determine instead the *robot posture*.
3. Determine the structural dimensioning of the various links and joints, as needed to meet static load requirements, where load includes both forces and moments – wrenches – under either the most demanding or the most likely operation conditions, depending on the design philosophy adopted at the outset.
4. Determine the structural dimensioning of the various links and joints, as needed to meet dynamic load requirements, where loads are inertia effects of links and manipulated object.
5. Determine the elastodynamic dimensioning of the overall mechanical structure, including the actuator dynamics, to avoid a specific spectrum of excitation frequencies under either the most demanding or the most likely operation conditions.
6. Select the actuators and their mechanical transmissions for the operation conditions adopted at the outset to cope with task uncertainty.

The above stages can be performed sequentially, in the order given above: (i) first, the topology is determined based on the family of tasks specified at the outset and the *shape* of the workspace, as discussed in Sect. 16.2.2; (ii) the link geometry is defined based on the workspace requirements, which include the *maximum reach*, and the topology defined in stage 1; (iii) with the link geometry thus defined, the structural dimensioning of links and joints (unless the robot under design is parallel, which does not fall within the scope of this chapter, all joints are actuated) is undertaken, so as to support the static loads assumed at the outset; (iv) with the links and joints dimensioned for static-load conditions, the link centers of mass and link inertia matrices are determined for a preliminary evaluation of the motor torque requirements (this evaluation is preliminary in that it does not consider the dynamic load brought about by the actuators; this load can be significant, even in the case of parallel robots, which can have all their motors fixed to the robot base); (v) with the links assumed rigid, joint stiffness is assumed, based on experience or using data from a similar robot, which then leads to an elastodynamic model whose natural modes and frequencies can be determined at a selected set of robot postures (dynamic behavior of the structure is dependent on robot posture) by means of scientific code such as Matlab or computer-aided engineering (CAE) code such as CATIA, Siemens PLM, Pro/Engineer or ANSYS; and (vi) if the frequency spectrum of the robot structure is acceptable, the designer can continue to motor selection; otherwise, a redimensioning is required, which means returning to stage 3.

Even though a design cycle can be completed as outlined above, the designer must now incorporate into the elastodynamic model the structural and inertial data provided by the motor manufacturer. This requires a return to stage 5 and a new elastodynamic analysis. It is thus apparent that the robot design process has one element in common with engineering design in general: both are iterative and open-ended [16.4]. Remarkably, however, the various items driving each design stage are, to a large extent, independent of each other, e.g., topology and geometry can be determined independently from motor selection. Obviously, all issues interact in the overall design process, but, within certain design specifications, the various items do not contradict each other, as to warrant a multiobjective design approach. That is, the optimum design of serial robots can be accomplished fairly well by means of a sequence of single-objective optimization jobs. Again, the results of the last stage, motor selection, must be integrated into an overall mathematical model to test the overall performance. One reference addressing practical optimization issues in the conceptual design of industrial robots is [16.5].

Only when the physical limits of components have been exhausted may a radical redesign requiring a return to stage 1 be warranted. This is the case with SCARA systems. Current industrial robots of this class bear topologies that are largely of the serial type, but some parallel SCARA systems have now appeared in the market. Indeed, the quest for shorter cycle time, as pertaining to an industry test cycle – described in Sect. 16.2.1 – has prompted the industry to look for alternatives to serial architectures. This is how ABBRobotics developed a hybrid parallel-serial robot, the *FlexPicker*, built upon *Clavel*'s Delta robot [16.6], to which a fourth axis has been added in series with the first three. The latter are laid out in a symmetric, parallel architecture that enables Delta to produce pure translations of its moving platform. Adept Technology's Quattro s650H is reportedly capable of three cycles/s for the same test cycle.

This chapter is organized according to the various stages of the robot design process outlined earlier. Noting that topology selection and geometric dimensioning are tightly coupled in the kinematic design process, we first begin with an examination of workspace criteria: we review methods for determining the topology of the kinematic chain, followed by the geometric dimensions so as to satisfy workspace requirements. We then review in detail the various criteria developed for characterizing a robot's manipulating capability, focusing on quantitative notions of dexterity based on both kinematic and dynamic models. We then examine methods for structural dimensioning of the links and joints so as to meet both static and dynamic load requirements. Finally, we discuss elastodynamic dimensioning, and actuator and gear sizing, taking into account properties such as the natural frequency of the robot, and force and acceleration capability requirements. We end the chapter with an overview of mobile, aquatic, and flying robots.

16.2 Workspace Criteria

The most obvious consideration in designing a robot is that its workspace has a set of *required characteristics*. This is a fundamental problem in classical mechanism design, and raises the obvious question of how a user can specify those characteristics.

Issues to consider here pertain, mostly, to what *Vijaykumar* et al. [16.7] termed the *regional structure* of a manipulator. This applies to manipulators with a *decoupled architecture*, whose last three revolutes have concurrent axes, thereby forming a *spherical wrist*, the point of concurrency being the *wrist center*. The manipulation task of architectures of this kind thus allows for a *decoupling* of the positioning and the orientation subtasks: the regional structure, consisting of the first three joints, is first postured so as to locate the center of its wrist at a specified point $C(x, y, z)$; then, the *local structure*, i. e., the wrist, is postured so as to make the end-effector (EE) attain a specified orientation with respect to a frame fixed to the base, given by a rotation matrix.

Most algorithms reported in the literature to determine the workspace of a given robot refer to the workspace of the regional structure. Here, we should distinguish between the workspace of the kinematic chain, regardless of the physical implementation of the chain, and that of the physical robot. In the former, all revolute joints are capable of unlimited rotations about their axes; in the latter, joint limits are needed, for example, to avoid wire entanglement. In the early stages of robot design, joint limits need not be considered, the workspace thus exhibiting symmetries that are proper of the type of joints of the regional structure. If the first joint is a revolute, the workspace has an axis of symmetry, namely, the axis of this revolute joint; if the first joint is prismatic, the workspace has an extrusion symmetry, with the direction of extrusion given by the direction of motion of this joint. As prismatic joints are infinitely extensive, so is the kinematic workspace of a robot with a prismatic joint. The kinematic workspaces of robots with prismatic joints are usually displayed for a finite portion of this workspace.

In the case of parallel robots, to be studied in full detail in Chap. 14, the regional structure is elusive, in general. The usual practice when displaying the workspace for these robots is to assume a constant orientation of the moving plate, the counterpart of the EE of serial robots [16.8]. A common architecture of parallel robots, which arises quite naturally in the design process, entails identical legs symmetrically placed both on the base platform and on the moving platform. Each leg is, in turn, a serial kinematic chain with one or two active joints, all others being passive. The workspace of this kind of robots also exhibits certain symmetries, but no axial symmetry. The symmetries are dictated by the number of legs and the types of actuated joints.

Coming back to serial robots, the workspace can be defined by an envelope that is essentially of one of two types, either a manifold or a surface that is

smooth *almost everywhere*, i. e., smooth everywhere except for a set of points of *measure zero* in the Lebesgue sense [16.10]. Broadly speaking, a set of measure zero on a surface is a curve, e.g., a meridian on a sphere, or a set of isolated points on a line, e.g., the set of rational numbers on the real line. A paradigm for this second kind of workspace is that of the Puma robot, whose kinematic chain is displayed in Fig. 16.1. In this figure, the regional and the local structures are clearly distinguished, the former being *fully extended*. The workspace of this robot is obtained upon locking all joints but the second, when the robot is in the posture shown in Fig. 16.1. Then, the second joint is fully rotated about its axis, the center C of the wrist then describing a circle of radius R equal to the distance of C from the line L_2, the plane of the circle being normal to this line and lying a distance b_3 from the axis L_1 of the first joint. This distance is known as the *shoulder offset*. Now, with all joints locked again, but this time with the first joint unlocked, the robot is turned as a rigid body about L_1. The result is the toroid of Fig. 16.4. Notice that the solid enclosed by this surface is the result of the Boolean operation $S-C$, where S is the sphere of radius R centered at point O_2 of Fig. 16.1, while C is the infinite cylinder of radius b_3 and axis L_1, which appears as Z_1 in Fig. 16.2. It is noteworthy that, although this workspace can be readily generated by a simple Boolean operation, it cannot possibly be generated by an implicit function of the form $f(x, y, z) = 0$ because the surface is not a manifold.

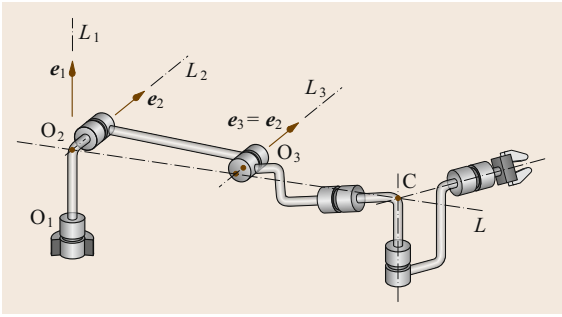


Fig. 16.1 A Puma robot in a fully stretched posture (after [16.9])

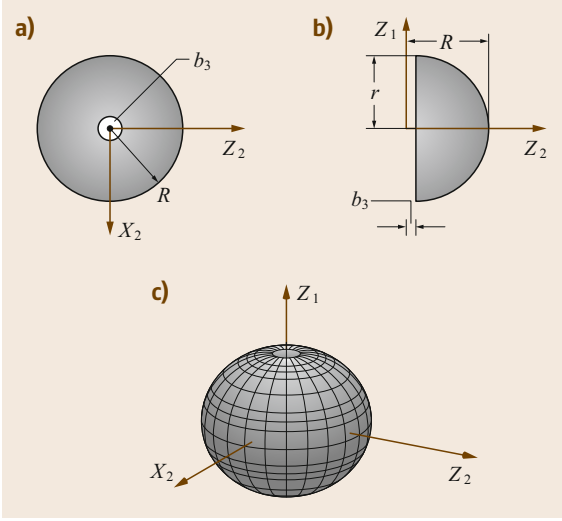


Fig. 16.2 The workspace of a Puma robot (after [16.9])

Robots with manifold workspaces are not common in industry. We display in Fig. 16.3 an architecture for the regional structure of a six-axis decoupled robot, with its neighboring axes mutually orthogonal and at the same distance a from each other. The common normals to the two pairs of axes, X_2 and X_3, also lie at the same distance a, as do X_4 from X_3 and C from Z_3. Point C is the center of the spherical wrist, the latter not being included in the figure. The workspace of this robot admits a representation of the form $f(x, y, z) = 0$ [16.9], which produces the manifold workspace of Fig. 16.4. The shaded internal region of the workspace includes all points admitting four real inverse-kinematics solutions, all other points admitting only two.

Given that any point of the workspace boundary represents a positioning singularity – different from an orientation singularity – manipulators with workspace boundaries that are not manifolds exhibit double singularities at the edges of their workspace boundary, which means that at edge points the rank of the robot Jacobian becomes deficient by two. At any other point of the workspace boundary the rank deficiency is by one.

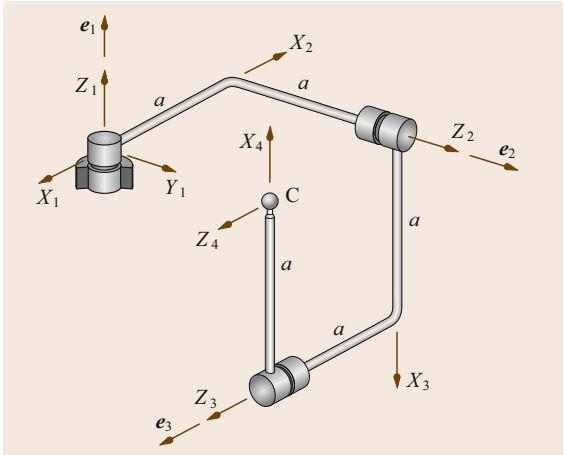


Fig. 16.3 An orthogonal three-revolute robot (after [16.9])

Design rules based on the *shape* of the workspace can now be drawn:

1. If the workspace required is axially symmetric and finite, use a serial robot with a regional structure composed of revolute joints only.
2. If the workspace required has a symmetry of extrusion and is unbounded, use a serial robot with regional structure having one first joint of the prismatic type. Here, *unbounded* is used in a restricted sense, meaning much larger in one direction than the others. Moreover:
   - If one direction is required to be much larger than the others, then practical implementations of prismatic joints are available in the form of rails either overhead, thereby giving rise to *gantry* robots, or on the floor.
   - If two directions are required to be much larger than the other, then use a wheeled mobile robot carrying a manipulator on top. A famous realization of this concept is the National Aeronautical and Space Agency's (NASA) *Sojourner* used in the *Pathfinder* mission to Mars in 1997.
3. If axial symmetry is not required, but rather a workspace with various coplanar axes of symmetry, similar to those of regular polygons, use a parallel robot.

16.2.1 Reaching a Set of Goal Frames

Closely related to the problem of workspace specification is that of task specification. In mechanism design it is customary to specify a set of coordinate frames in space, and to design a mechanism with an a priori specified topology that can visit these frames. An order in which the frames must be reached may be given. In the event that not all of the frames are reachable, then one may seek a mechanism that comes *closest*, in some suitable sense, to the specified frames. The literature on this classical mechanism design problem is vast [16.1, 11, 12] and the references cited therein. Some further remarks in connection with this goal-frame approach to robot dimensioning are noteworthy:

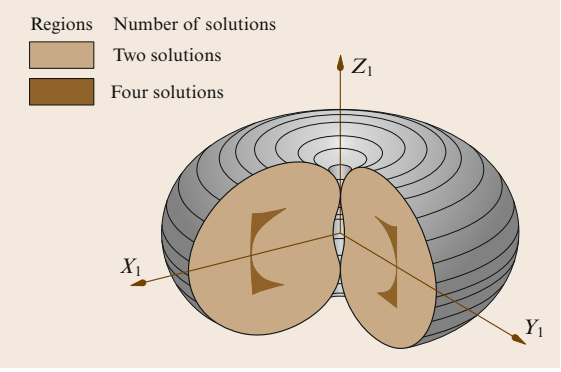


Fig. 16.4 The workspace of the orthogonal robot of Fig. 16.3 (after [16.9])

1. Reaching *exactly* the desired frames may not always be desired or possible: in some cases it is better to use an optimization approach that allows for solutions that will visit the desired poses within a minimum error (provided that an error norm can be suitably engineered, of course, as suggested in [16.13]).
2. It has been claimed [16.8] that interval analysis allows not only a discrete set of desired poses but also a full six-dimensional (6-D) workspace to be met while taking into account manufacturing errors.
3. The branching problem occurring in 1-DOF mechanisms may also occur in robot design: a design solution based on via points may indeed visit the prescribed poses, but not all of these may be reachable within the same assembly mode. This problem is exacerbated in the design of serial robots, as a 6-DOF, revolute-coupled robot may admit up to 16 distinct postures – branches – for one given EE pose [16.14, 15].
4. While a robot designed to visit a set of prescribed poses via its EE will be able to visit that set, we should not forget that the purpose of using robots is first and foremost to be able to perform not one single task, but rather a family of tasks. In this light, the set of poses for which a robot is designed might as well be a task that is *representative* of that family.

In connection with remark 4 above, we can cite the design or evaluation of SCARA systems. A SCARA system is a 4-DOF serial robot capable of tasks that lie within the *Schönflies* subgroup of the group of rigid-body displacements [16.16, 17], namely the set of three-dimensional (3-D) displacements augmented with a rotation about an axis of fixed direction. In these systems, the task at hand is given by two vertical segments joined by one horizontal segment. Moreover, the length of the vertical segments is 25.0 mm, that of the horizontal segment being 300.0 mm. While the EE is traversing the horizontal segment, moreover, it should rotate about a vertical axis through an angle of 180°. This task specification, which has been adopted by SCARA manufacturers, does not indicate how to negotiate the corners, which is left to the imagination of the robotics engineer.

16.2.2 Workspace Volume and Topology

Reachable and Dexterous Workspace

Beginning with the early work of *Roth* [16.18], there have been many studies on the relationship between manipulator kinematic geometry and its workspace. Most studies have focused on a classification of the workspace into two components, the *reachable* and the *dexterous* workspace [16.19]. Given a reference point P attached to a manipulator EE, such as the center of the spherical wrist, or a point on the EE, the reachable workspace is defined to be the set of points in physical space that can be reached by P. The dexterous workspace, on the other hand, is the set of points that can be reached by P with arbitrary EE orientations.

The early literature on workspace focuses on numerical and algebraic methods to characterize these workspaces. Reachable and dexterous workspaces have been analyzed using numerical techniques by *Kumar* and *Waldron* [16.19], *Yang* and *Lee* [16.20], and *Tsai* and *Soni* [16.21], among others. The advantage of these schemes over algebraic approaches is that kinematic constraints can be readily included. More-general design principles or insights, however, are more difficult to come by using these techniques. Among the algebraic approaches to workspace characterization, a topological analysis of robot workspace is given by *Gupta* and *Roth* [16.22] and *Gupta* [16.23]; here the concept of workspace holes and voids is defined, and conditions for their existence are identified. The shape of the reachable and dexterous workspaces is also analyzed as a function of P.

Further studies of workspace analysis were reported by *Freudenstein* and *Primrose* [16.24] and by *Lin* and *Freudenstein* [16.25], where precise relationships between kinematic and workspace parameters are developed, and a class of three-joint manipulators is optimized for workspace volume. A more general analysis of workspace optimization is given in *Vijaykumar* et al. [16.7]. Performance criteria for manipulators are defined here in terms of the dexterous workspace; given that a manipulator satisfies certain constraints on its Denavit–Hartenberg parameters, it is shown that the optimal six-revolute (6R) design is the elbow manipulator.

A typical design of robot regional architecture is of the *orthogonal type*, consisting of one revolute of the vertical axis and two revolutes of the horizontal axes, one of which intersects the vertical axis. Moreover, the most common architecture includes intermediate and distal links of identical lengths. The workspace of this architecture is thus a sphere of radius equal to twice that common link length. The volume of this workspace is thus determined by the length in question. As shown by *Yoshikawa* [16.26], the workspace of the two-link planar manipulator defined by the last two links of the foregoing regional architecture is of maximum area for a prescribed reach and equal link lengths. As a result, the volume of the same regional architecture is similarly of maximum volume for a prescribed reach.

Differential-Geometric Workspace Characterization

Workspace can also be approached from a differential-geometric perspective, by regarding the configuration space of a robotic EE frame as a subset of the special Euclidean group SE(3). An important physical consideration in defining the workspace volume of spatial mechanisms is that it should not depend on the choice of fixed reference frame. Less obvious but just as important is the requirement that the volume should not depend on which point of the last link the EE frame is fixed to. This last condition has the following physical significance: if the EE were enlarged or shrunk, then the robot would have the same workspace volume. The workspace volume of a robot therefore depends only on the joint axes.

The workspace volume of a robot is defined by regarding SE(3) as a Riemannian manifold, so that the workspace volume is simply the volume of the image of the forward kinematic map f with respect to the volume form on SE(3). It is known that SE(3) has a bi-invariant volume form, that is, the notion of volume is invariant with respect to the choice of both the fixed (base) and moving (EE) frames – see, e.g., *Loncaric* [16.27]. *Paden* and *Sastry* [16.28] provide the following visualization for this volume form: Suppose an airplane is restricted to move within a cube of airspace of length 1 km on a side. At each point within this cube the airplane can point itself anywhere in a 4π solid angle and roll 2π about the direction it is pointing. The orientation volume at such an airplane is $4\pi \times 2\pi = 8\pi^2\,\mathrm{rad}^3$. Multiplying by the positional volume one obtains $8\pi^2\,\mathrm{rad}^3\mathrm{km}^3$ for the volume of the free configuration space of the aircraft.

This depiction is the notion of workspace volume used for robots; it has the advantage of being able to trade off orientation freedom for positional freedom smoothly, unlike the popular notion of dexterous workspace. Note that the actual numerical value one obtains will depend on the choice of length scale for physical space; this in itself does not pose a serious problem for workspace volumes, as long as the same length scale is maintained when comparing different workspaces.

In [16.28] *Paden* and *Sastry* show that the optimal 6R manipulator that maximizes the workspace volume subject to a kinematic length constraint is the

elbow manipulator. This result is consistent with the earlier finding of *Vijaykumar* et al. [16.7], but the authors employ the geometric framework outlined above. Moreover, the results are obtained without some of the a priori assumptions on the kinematic structure made by Vijaykumar.

16.3 Dexterity Indices

16.3.1 Local Dexterity for Open Chains

Dexterity can be defined as the ability to move and apply forces and torques in arbitrary directions with equal ease, the concept thus belonging to the realm of *kinetostatics*, which is the study of the interplay between *feasible twists* and *constraint wrenches* in multibody mechanical systems under *static conservative conditions*. Here, twist is the 6-D array of velocity variables of a rigid body, involving three components of a landmark-point velocity and three of angular velocity; wrench, in turn, is the 6-D array of static variables acting on a rigid body, three accounting for the resultant force applied at the same landmark point and three for the concomitant moment acting on the same body.

Salisbury and *Craig* [16.29] introduced the concept of dexterity when working on the design of articulated hands. At issue is the way in which input joint velocity errors propagate to the output velocities of each fingertip. To illustrate this concept, let $\mathbf{J}(\boldsymbol{\theta})$ denote the Jacobian of the forward kinematic map, i. e.,

$$\boldsymbol{t} = \mathbf{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \,, \tag{16.1}$$

in which $\boldsymbol{\theta}$ and $\dot{\boldsymbol{\theta}}$ denote the vectors of joint variables and joint rates, respectively, while $\boldsymbol{t}$ is the EE *twist*, defined, in turn, as,

$$\boldsymbol{t} = \begin{pmatrix} \boldsymbol{\omega} \\ \dot{\boldsymbol{p}} \end{pmatrix} \,, \tag{16.2}$$

with $\boldsymbol{\omega}$ denoting the angular velocity of the EE and $\dot{\boldsymbol{p}}$ the velocity of the *operation point* P of the EE, at which the task is specified.

Assuming the Jacobian $\mathbf{J}$ is $m \times n$, the singular value decomposition of J can be expressed in the form

$$\mathbf{J} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}} \,, \tag{16.3}$$

where $\mathbf{U}$ and $\mathbf{V}$ are respectively $m \times m$ and $n \times n$ unitary matrices, and $\boldsymbol{\Sigma}$ is a $m \times n$ matrix with zeros everywhere, except for its (i, i) entries, for $i = 1, \ldots n$ if $m > n$; otherwise, for $i = 1, \ldots m$. The non-zero elements are the (non-negative) singular values of $\mathbf{J}$. At nonsingular postures, the Jacobian is invertible and we can write

$$\dot{\boldsymbol{\theta}} = \mathbf{V}\boldsymbol{\Sigma}^{-1}\mathbf{U}^{\mathrm{T}}\boldsymbol{t} \,. \tag{16.4}$$

Furthermore, if we assume that all the components of both $\boldsymbol{t}$ and $\dot{\boldsymbol{\theta}}$ have the same physical units, which is the case for purely positioning or purely orienting manipulators with only revolute joints, then we can take the Euclidean norm of both sides of (16.4), thereby obtaining

$$\|\dot{\boldsymbol{\theta}}\|^2 = \boldsymbol{t}^{\mathrm{T}}\mathbf{U}\left(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{\mathrm{T}}\right)^{-1}\mathbf{U}^{\mathrm{T}}\boldsymbol{t} \,. \tag{16.5}$$

If we let $\boldsymbol{v} = \mathbf{U}^{\mathrm{T}}\boldsymbol{t}$ then the above expression for $\|\dot{\boldsymbol{\theta}}\|^2$ becomes

$$\boldsymbol{v}^{\mathrm{T}}\left(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{\mathrm{T}}\right)^{-1}\boldsymbol{v} = \|\dot{\boldsymbol{\theta}}\|^2 \,. \tag{16.6}$$

Now, if the i-th component of $\boldsymbol{v}$ is denoted by v_i, for $i = 1, \ldots, n$, and we look at the mapping of the unit ball in $\mathcal{J}$, $\|\dot{\boldsymbol{\theta}}\|^2 = 1$, (16.6) leads to

$$\frac{v_1}{\sigma_1^2} + \frac{v_2}{\sigma_2^2} + \cdots + \frac{v_n}{\sigma_n^2} = 1 \,, \tag{16.7}$$

which is the canonical equation of an ellipsoid of semi-axes $\{\sigma_i\}_1^n$ in the $\mathcal{G}$-space, i. e., the space of Cartesian velocities, or twists of the EE. Notice that the ellipsoid in question takes its canonical form when represented in a coordinate frame of axes oriented in the directions of the eigenvectors of $\mathbf{U}$.

In summary the unit ball in joint space is mapped by the Jacobian-inverse $\mathbf{J}^{-1}$ into an ellipsoid whose semi-axes are the singular values of $\mathbf{J}$. That is, $\mathbf{J}$ *distorts* the unit ball in the joint-rate space into an ellipsoid in the EE-twist space. Hence, a measure of the quality of motion and force transmission of the robotic architecture from the joints to the EE is given by the above distortion; *the smaller the distortion, the higher the quality of the transmission*.

A measure of the Jacobian-induced distortion can thus be defined as the ratio of the largest σ_{M} to the smallest σ_{m} singular values of $\mathbf{J}$, which is nothing but the *condition number* κ_2 of $\mathbf{J}$ based on the matrix 2-norm [16.30], i. e.,

$$\kappa_2 = \frac{\sigma_{\mathrm{M}}}{\sigma_{\mathrm{m}}} \,. \tag{16.8}$$

Actually, (16.8) is only one possibility of computing the condition number of $\mathbf{J}$, or of any $m \times n$ matrix

for that matter, and certainly not the most economical. Notice that this definition requires knowledge of the singular values of the Jacobian. However, computing the singular values of a matrix is as computationally intensive as computing eigenvalues, with the added cost of a polar decomposition [16.31]; the combined operation is slightly less expensive than the singular-value decomposition [16.32]. The most general definition of condition number, for $n \times n$ matrices, is [16.30]

$$\kappa(\mathbf{A}) = \|\mathbf{A}\|\|\mathbf{A}^{-1}\| \,. \tag{16.9}$$

The expression (16.8) is obtained when the matrix 2-norm is adopted in (16.9). The matrix 2-norm is defined as

$$\|\mathbf{A}\|_2 \equiv \max_i \{\, \sigma_i \,\} \,. \tag{16.10}$$

If, on the other hand, the *weighted matrix Frobenius norm* is adopted, which is defined as

$$\|\mathbf{A}\|_\mathrm{F} \equiv \sqrt{\frac{1}{n}\mathrm{tr}(\mathbf{A}\mathbf{A}^\mathrm{T})} \equiv \sqrt{\frac{1}{n}\mathrm{tr}(\mathbf{A}^\mathrm{T}\mathbf{A})} \,, \tag{16.11}$$

then, apparently, the computation of the singular values can be obviated. When the weight $1/n$ is omitted in the above definition, the standard Frobenius norm is obtained. Notice, however, that the weighted Frobenius norm is more significant in engineering, for it does not depend on the number of rows and columns of the matrix at hand. The weighted Frobenius norm, in fact, yields the root-mean-square (rms) value of the set of singular values.

The *Frobenius condition number* κ_F of the Jacobian $\mathbf{J}$, based on the matrix Frobenius norm, is then

$$\begin{aligned}\kappa_\mathrm{F}(\mathbf{J}) &= \frac{1}{n}\sqrt{\mathrm{tr}(\mathbf{J}\mathbf{J}^\mathrm{T})}\sqrt{\mathrm{tr}[(\mathbf{J}\mathbf{J}^\mathrm{T})^{-1}]} \\ &= \frac{1}{n}\sqrt{\mathrm{tr}(\mathbf{J}^\mathrm{T}\mathbf{J})}\sqrt{\mathrm{tr}[(\mathbf{J}^\mathrm{T}\mathbf{J})^{-1}]} \,. \end{aligned} \tag{16.12}$$

One more important difference between the two forms of computing the matrix condition number is worth pointing out: $\kappa_2(\cdot)$ is not an *analytic function* of its matrix argument, while $\kappa_\mathrm{F}(\cdot)$ is. Hence, the condition number based on the Frobenius norm can be applied to great advantage in robot architecture design, as $\kappa_\mathrm{F}(\cdot)$ is differentiable and lends itself to gradient-dependent optimization methods, which are much faster than direct methods based only on function evaluations. In robot control, which requires real-time computations, $\kappa_\mathrm{F}(\cdot)$ also offers practical advantages, for its computation obviates the knowledge of the singular values, only requiring a matrix inversion, which is a rather simple task.

The significance of the condition number in robot design and control can be best understood if we recall that the concept stems from numerical analysis in connection with the solution of the linear system of equations (16.1) for $\dot{\boldsymbol{\theta}}$. Given that $\mathbf{J}$ is a function of both the architecture parameters and the posture variables $\boldsymbol{\theta}$, $\mathbf{J}$ is known only to within the error with which those quantities are known. Further, let the architecture parameters, namely, the constant values in the Denavit–Hartenberg parameter list, be stored in an array $\boldsymbol{p}$. Both $\boldsymbol{p}$ and $\boldsymbol{\theta}$ are known up to measurement errors $\delta\boldsymbol{p}$ and $\delta\boldsymbol{\theta}$, respectively. Moreover, the twist $\boldsymbol{t}$ is input into the control software of the robot with an unavoidable error $\delta\boldsymbol{t}$.

In solving (16.1) for $\dot{\boldsymbol{\theta}}$ with floating-point arithmetic, the computed value is contaminated with a roundoff error $\delta\dot{\boldsymbol{\theta}}$. The *relative error* in the computed $\dot{\boldsymbol{\theta}}$ is related to the relative errors in the architecture parameters and posture variables by the relation [16.30]

$$\frac{\|\delta\dot{\boldsymbol{\theta}}\|}{\|\dot{\boldsymbol{\theta}}\|} \le \kappa(\mathbf{J})\left(\frac{\|\delta\boldsymbol{p}\|}{\|\boldsymbol{p}\|} + \frac{\|\delta\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|} + \frac{\|\delta\boldsymbol{t}\|}{\|\boldsymbol{t}\|}\right) \,, \tag{16.13}$$

where $\boldsymbol{p}$ and $\boldsymbol{\theta}$ represent the (unknown) actual values of these vectors and $\boldsymbol{t}$ the nominal value of the twist.

Nevertheless, the foregoing discussion applies to tasks involving either positioning or orientation, but not both. Most frequently, robotic tasks involve both positioning and orientation, which leads to Jacobian matrices with entries bearing disparate physical units, the consequence being that the Jacobian singular values have disparate units as well. Indeed, singular values associated with positioning bear units of length, whereas those associated with orientation are dimensionless. As a consequence, it is impossible to either order all singular values from smallest to largest or to add them all.

To cope with the issue of disparate units in the Jacobian entries, and to allow for the computation of the Jacobian condition number, the concept of *characteristic length* has been proposed [16.9]. The characteristic length L has been defined as the length by which the entries of the Jacobian with units of length are to be divided to render the Jacobian condition number a minimum at an optimum posture. This definition, while sound, lacks an immediate geometric interpretation, which has made its adoption in the robotics community difficult. In order to provide for a geometric interpretation, the concept of *homogeneous space* was recently introduced [16.33]. Using this concept, the robot architecture is designed in a *dimensionless space*, with points whose coordinates are dimensionless real numbers. In this way, the six Plücker coordinates [16.34] of lines are all dimensionless, and hence the columns

of the robot Jacobian matrix, comprising the Plücker coordinates of the revolute axes, are dimensionless as well. As a consequence, the Jacobian singular values are all dimensionless, and the Jacobian condition number can be defined. Once the robotic architecture has been designed for minimum condition number, under geometric constraints on link-length ratios and angles between neighboring revolute axes, for example, the maximum reach of the robot can be calculated. The maximum reach r will thus be a dimensionless quantity. When this quantity is compared with the prescribed maximum reach R, with units of length, the characteristic length is computed as the ratio $L = R/r$.

16.3.2 Dynamics-Based Local Performance Evaluation

Since motions are caused by forces and torques acting on rigid bodies, it seems reasonable to formulate performance indices that take into account the inertial properties of the mechanism. *Asada* [16.35] defines the *generalized-inertia ellipsoid* (GIE) as the ellipsoid described by the product $\mathbf{G} = \mathbf{J}^{-\mathrm{T}}\mathbf{M}\mathbf{J}^{-1}$, where $\mathbf{M}$ denotes the inertia matrix of the manipulator. This ellipsoid is that with semiaxes given by the singular values of the foregoing product. *Yoshikawa* [16.36], in turn, defines a corresponding dynamic manipulability measure as $\det[\mathbf{J}\mathbf{M}^{-1}(\mathbf{J}\mathbf{M}^{-1})^{\mathrm{T}}]$. Physically these concepts represent two distinct phenomena. Suppose the robot is viewed as an input–output device, which, given a joint torque, produces an acceleration at the EE. Yoshikawa's index measures the uniformity of this torque–acceleration gain, whereas Asada's GIE characterizes the inverse of this gain. If a human operator were holding the EE and attempting to move it about, the GIE would measure the resistance of the robot to this EE motion.

Other measures that attempt to capture robot performance as a function of the dynamics can be cited: *Voglewede* and *Ebert-Uphoff* [16.37] propose performance indices based on joint stiffness and link inertia, with the aim of establishing a distance to singularity for any robot posture, while *Bowling* and *Khatib* [16.38] propose a general framework for capturing the dynamic capability of a general robot manipulator that includes the velocity and acceleration characteristics of the EE, taking into account factors such as torque and the velocity limits of the actuators.

16.3.3 Global Dexterity Measures

The above measures are local in the sense that they characterize the dexterity of a robot at a given posture. Local measures are useful for applications ranging from redundancy resolution to workpiece positioning, but for design applications a global measure may be more desirable. One straightforward way of extending local measures to global ones is to integrate them over the allowable joint space. In [16.39] *Gosselin* and *Angeles* integrate the Jacobian condition number over the workspace to define a global measure, thereby producing a *global conditioning index*. For the simpler cases of planar positioning and spherical manipulators, the global conditioning index was found to coincide with its local counterpart.

16.3.4 Closed-Chain Dexterity Indices

The formulation of dexterity for closed chains presents a number of subtleties. The first obvious difference is that the joint configuration space of a closed chain will no longer be a flat space; in the general case it will be a curved multidimensional surface embedded in a higher-dimensional (typically flat) space. Also, dual to the open chain case, the forward kinematics for closed chains is generally more difficult to solve than the inverse kinematics, with the possibility of multiple solutions. Another important difference is that only a subset of the joints may be actuated, and that the number of actuated joints may even exceed the mechanism's kinematic degrees of freedom.

Several coordinate-based formulations for closed-chain dexterity have been proposed for specific mechanisms [16.40] and for cooperating robot systems whose joints are all assumed to be actuated [16.41, 42], at least some of which have led to apparently paradoxical results [16.43, 44]. Because of the nonlinear characteristics unique to closed-chain mechanisms discussed above, particular care must be exercised in formulating coordinate-based dexterity measures.

Another recent line of work has attempted a coordinate-invariant differential-geometric formulation of dexterity for closed chains. In this framework the joint and EE configuration spaces are regarded as Riemannian manifolds with an appropriate choice of Riemannian metric, with the choice of joint space metric reflecting the characteristics of the joint actuators. The previous notions of ellipsoid developed for serial chains can be extended to general closed chains, containing both passive and active joints, and possibly redundantly actuated [16.45, 46].

16.3.5 Alternative Dexterity-Like Measures

The various definitions of dexterity discussed above all refer to the same qualitative feature of the ability of a robot to move and apply forces in arbitrary directions. A different viewpoint is taken in the work of

Liégeois [16.47] and *Klein* and *Huang* [16.48], where dexterity is quantified in terms of joint-range availability. The driving motivation here lies in that most robots have joint limits; therefore, one should minimize the possibility of a joint reaching a stop.

Hollerbach [16.49] takes an alternative approach to designing a redundant seven revolute 7R manipulator, by considering: (a) elimination of internal singularities, (b) ability to avoid obstacles in the workspace, (c) the solvability of the kinematic equations, and (d) mechanical constructability. Based on these four criteria, he derived a particular 7R design with the morphology of the human arm, i. e., composed of two spherical joints defining the shoulder and the wrist plus an intermediate revolute playing the role of the elbow. Now, while a redundant architecture should remain fully capable of performing 6-DOF tasks upon locking one joint, the architecture reported in this reference may end up by losing this capability if the elbow joint is locked.

From a control perspective, *Spong* [16.50] shows that, if the inertia matrix of a manipulator has a vanishing Riemannian curvature, there exists a set of coordinates in which the equations of motion assume a particularly simple form. The curvature of the inertia matrix also reflects the sensitivity of the dynamics to certain robot parameters. Minimizing the curvature, therefore, is another possible criterion for robot design.

16.4 Other Performance Indices

16.4.1 Acceleration Radius

Another measure that attempts to capture the dynamic capabilities of a robotic manipulator is the *acceleration radius*. Initially proposed by *Graettinger* and *Krogh* in [16.51], the acceleration radius measures the minimum acceleration capability of the EE, in arbitrary directions, for the given torque bounds on the actuators. Specifically, given the dynamics equations for a serial chain in the form

$$\boldsymbol{\tau} = \mathbf{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} \,, \tag{16.14}$$

in which $\mathbf{M}$ is the robot *mass matrix* – also known as the *inertia matrix* – in *joint space*, and $\mathbf{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ is the matrix mapping the joint-rate vector to the vector of Coriolis and centrifugal forces in the same space. Moreover, the actuators are assumed to have joint torque limits of the form

$$\boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{\max} \,, \tag{16.15}$$

where the lower and upper limits $\boldsymbol{\tau}_{\min}, \boldsymbol{\tau}_{\max} \in \mathbb{R}^n$ are constant or functions of the manipulator posture $\boldsymbol{\theta}$. The EE *twist rate*, denoted by $\dot{\boldsymbol{t}}$, can be expressed as

$$\dot{\boldsymbol{t}} = \mathbf{J}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \dot{\mathbf{J}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} \,, \tag{16.16}$$

where $\dot{\mathbf{J}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ is the Jacobian time-derivative. The Jacobian $\mathbf{J}$ was introduced in (16.1).

Under the assumption that $\mathbf{J}(\boldsymbol{\theta})$ is nonsingular, one can write

$$\ddot{\boldsymbol{\theta}} = \mathbf{J}(\boldsymbol{\theta})^{-1}\dot{\boldsymbol{t}} - \boldsymbol{J}(\boldsymbol{\theta})^{-1}\dot{\mathbf{J}}(\boldsymbol{\theta}, \boldsymbol{t})\dot{\boldsymbol{\theta}} \,. \tag{16.17}$$

Substituting the above expression into the dynamic equations (16.14) leads to

$$\boldsymbol{\tau}(\boldsymbol{\theta}, \boldsymbol{t}, \dot{\boldsymbol{t}}) = \mathbf{M}'(\boldsymbol{\theta})\dot{\boldsymbol{t}} + \mathbf{C}'(\boldsymbol{\theta}, \boldsymbol{t})\boldsymbol{t} \,, \tag{16.18}$$

where

$$\mathbf{M}'(\boldsymbol{\theta}) = \mathbf{M}(\boldsymbol{\theta})\mathbf{J}(\boldsymbol{\theta})^{-1}$$
$$\mathbf{C}'(\boldsymbol{\theta}, \boldsymbol{t}) = [\mathbf{C}(\boldsymbol{\theta}, \boldsymbol{t}) - \mathbf{M}(\boldsymbol{\theta})\mathbf{J}(\boldsymbol{\theta})^{-1}\dot{\mathbf{J}}(\boldsymbol{\theta}, \boldsymbol{t})]\mathbf{J}^{-1}(\boldsymbol{\theta}) \,.$$

For a given state $(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$, the linear torque bounds (16.15) now define a polytope in the twist-rate space. *Graettinger* and *Krogh* [16.51] define the acceleration radius to be the largest sphere centered at the origin that is contained in this polytope; its radius reflects the minimal guaranteed EE acceleration in arbitrary directions. This concept is applied to measure the EE acceleration capability of a manipulator, as well as to determine the actuator sizes for achieving a desired acceleration radius. *Bowling* and *Khatib* [16.38] generalize this concept to capture both force and acceleration capabilities of the EE, with a view to quantifying the worst-case dynamic performance capability of a manipulator.

16.4.2 Elastostatic Performance

Elastostatic performance refers to the robotic system's response to the applied load – force and moment, i. e., wrench – under static equilibrium. This response may be measured in terms of the *stiffness* of the manipulator, which determines the translation and angular deflections when the EE is subjected to an applied wrench.

Robot deflections have two sources: link and joint deflection. In the presence of long links, as in the space

robot Canadarm2, link compliance is a major source of deflection. However, in the majority of today's serial robots, the main source of deflection occurs at the joints.

In this chapter, we assume that the manipulator links are rigid, and model the joints as linearly elastic torsional springs. The more complex problem of link flexibility is studied in detail in Chap. 11. That is, for the elastostatic model we base our analysis on the assumption that, under a positioning task, the joints are locked at a certain posture $\boldsymbol{\theta}_0$, while the EE is subject to a perturbation wrench $\Delta\boldsymbol{w}$ that is balanced by an elastic joint torque $\Delta\boldsymbol{\tau}$. Under these conditions, $\Delta\boldsymbol{\theta}$ and $\Delta\boldsymbol{\tau}$ obey the well-known linear relation

$$\mathbf{K}\Delta\boldsymbol{\theta} = \Delta\boldsymbol{\tau} \ , \tag{16.19}$$

in which $\mathbf{K}$ is the *stiffness matrix* in joint space at the given posture. Moreover, the matrix $\mathbf{K}$ is diagonal, with its entries equal to the torsional stiffnesses of the corresponding joints, $\mathbf{K}$ is therefore *posture independent*, i. e., constant throughout the whole robot workspace. Moreover, since all joints exhibit a finite, nonzero stiffness, $\mathbf{K}$ is invertible, its inverse $\mathbf{C}$ being termed the *compliance matrix*. We can thus express the inverse relation of (16.19) as

$$\Delta\boldsymbol{\theta} = \mathbf{C}\Delta\boldsymbol{\tau} \ . \tag{16.20}$$

It should be apparent that $\Delta\boldsymbol{\theta}$ and $\Delta\boldsymbol{\tau}$ have an *incremental* nature, for both are measured from the equilibrium posture, at which $\Delta\boldsymbol{\tau}$ and $\Delta\boldsymbol{\theta}$ vanish.

On the issue of stiffness matrix, *Griffis* and *Duffy* [16.52] proposed a mapping from an incremental rigid-body displacement $\Delta\boldsymbol{x}$ into an incremental wrench $\Delta\boldsymbol{w}$ that turned out to be nonsymmetric. The concept behind this mapping was formalized by *Howard*, *Zefran* and *Kumar* [16.53] by means of Lie algebra. However, in the foregoing papers, $\Delta\boldsymbol{x}$ and $\Delta\boldsymbol{w}$ turn out to be incompatible in the sense that their reciprocal product does not yield incremental work – the point at which $\Delta\boldsymbol{x}$ is defined is distinct from that at which $\Delta\boldsymbol{w}$ is applied. For this reason, the array representation of the same mapping cannot be, properly speaking, termed a stiffness matrix.

For a constant magnitude of $\Delta\boldsymbol{\tau}$, the deflection attains its maximum value in the direction of the eigenvector associated with the maximum eigenvalue of $\mathbf{C}$ or, equivalently, with the minimum eigenvalue of $\mathbf{K}$, denoted by $\kappa_{\min}$. In terms of elastostatic performance, we aim to (a) make the maximum deflection a minimum, i. e., we want to maximize $\kappa_{\min}$, and (b) make the magnitude of the deflection $||\Delta\boldsymbol{\theta}||$ as insensitive as possible to changes in the direction of the applied load $\Delta\boldsymbol{\tau}$. This can be done by rendering $\kappa_{\min}$ as close as possible to $\kappa_{\max}$. The first aim is associated with the stiffness constants, i. e., the higher the constants the lower the deflections. The later, however, is associated with the concept of isotropy, the ideal case being when all the eigenvalues of $\mathbf{K}$ are identical, which means that $\mathbf{K}$ is isotropic. Due to the *pyramidal effect* of serial robots, in which downstream motors carry their upstream counterparts, the joint stiffness is bound to be largest for the proximal joints. Hence an isotropic stiffness matrix is impossible for serial robots.

Notice that (16.19) and (16.20) may also be formulated in the task space, i. e.,

$$\mathbf{K}_{\mathrm{C}}\Delta\boldsymbol{x} = \Delta\boldsymbol{w} \ , \tag{16.21}$$

where $\Delta\boldsymbol{x} \equiv \boldsymbol{t}\Delta t$, with Δt denoting a *small* time interval producing a correspondingly *small* change $\Delta\boldsymbol{x}$ in the pose of the EE. That is,

$$\Delta\boldsymbol{x} = \mathbf{J}\dot{\boldsymbol{\theta}}\,\Delta t = \mathbf{J}\Delta\boldsymbol{\theta} \ , \tag{16.22}$$

which is a linear transformation of the joint-vector increment into the pose-increment vector. We show next that *the stiffness matrix is not frame invariant*. This means that, under the linear transformation from joint to Cartesian coordinates, *the stiffness matrix does not obey a similarity transformation*. For quick reference, we recall below the definition of similarity transformation: if $\boldsymbol{y} = \mathbf{L}\boldsymbol{x}$ is a linear transformation of $\mathbb{R}^n$ into itself, and we introduce a change of vector basis, $\boldsymbol{x}' = \mathbf{A}\boldsymbol{x}$, $\boldsymbol{y}' = \mathbf{A}\boldsymbol{y}$, then $\mathbf{L}$ changes to $\mathbf{L}'$, which is given by

$$\mathbf{L}' = \mathbf{A}\mathbf{L}\mathbf{A}^{-1} \ . \tag{16.23}$$

The above transformation of any vector of $\mathbb{R}^n$ into another one of the same space, and of a matrix $\mathbf{L}$ into $\mathbf{L}'$, as given by (16.23), is termed a *similarity transformation*. Notice that $\mathbf{A}$ is guaranteed to be invertible, as it represents a change of coordinate frame.

Now, under the change of coordinates given by (16.22), (16.21) leads to

$$\mathbf{K}_{\mathrm{C}}\mathbf{J}\Delta\boldsymbol{\theta} = \mathbf{J}^{-\mathrm{T}}\Delta\boldsymbol{\tau} \ , \tag{16.24}$$

where we have used the kinetostatic relation [16.9]

$$\mathbf{J}^{\mathrm{T}}\Delta\boldsymbol{w} = \Delta\boldsymbol{\tau} \ .$$

The exponent $-\mathrm{T}$ denotes the transpose of the inverse or, equivalently, the inverse of the transpose. Upon multiplication of both sides of (16.24) by $\mathbf{J}^{\mathrm{T}}$ from the left, we end up with

$$\mathbf{J}^{\mathrm{T}}\mathbf{K}_{\mathrm{C}}\mathbf{J}\Delta\boldsymbol{\theta} = \Delta\boldsymbol{\tau} \ . \tag{16.25}$$

If we now compare (16.19) with (16.25), we can readily derive the relations between the stiffness matrix $\mathbf{K}$ in joint space and the stiffness matrix $\mathbf{K}_C$ in Cartesian space

$$\mathbf{K} = \mathbf{J}^T \mathbf{K}_C \mathbf{J} \quad \text{and} \quad \mathbf{K}_C = \mathbf{J}^{-T} \mathbf{K} \mathbf{J}^{-1} , \tag{16.26}$$

which apparently is not a similarity transformation between $\mathbf{K}$ and $\mathbf{K}_C$. What this means is that the two matrices do not share the same set of eigenvalues and their eigenvectors are not related by the linear relation (16.22). As a matter of fact, if the robot is revolute-coupled, the entries of its stiffness matrix $\mathbf{K}$ all have units of Nm, i. e., of *torsional stiffness*, while the entries of $\mathbf{K}_C$ have disparate units. To show this, the Jacobian matrix, the inverse Jacobian and the two stiffness matrices are partitioned correspondingly into four 3×3 blocks, i. e., as

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_{11} & \mathbf{J}_{12} \\ \mathbf{J}_{21} & \mathbf{J}_{22} \end{pmatrix} , \quad \mathbf{J}^{-1} = \begin{pmatrix} \mathbf{J}'_{11} & \mathbf{J}'_{12} \\ \mathbf{J}'_{21} & \mathbf{J}'_{22} \end{pmatrix} ,$$
$$\mathbf{K} = \begin{pmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{12}^T & \mathbf{K}_{22} \end{pmatrix} , \quad \mathbf{K}_C = \begin{pmatrix} \mathbf{K}_{C11} & \mathbf{K}_{C12} \\ \mathbf{K}_{C12}^T & \mathbf{K}_{C22} \end{pmatrix} .$$

Furthermore, in light of the definition of the twist, (16.2), the two upper blocks of $\mathbf{J}$ are dimensionless, while its two lower blocks have units of length [16.9]. As a consequence, the two left blocks of $\mathbf{J}^{-1}$ are dimensionless, while its two right blocks have units of inverse length. Now, the blocks of $\mathbf{K}_C$ are computed from the corresponding relation in (16.26), thereby obtaining

$$\begin{aligned}
\mathbf{K}_{C11} &= \mathbf{J}_{11}'^T (\mathbf{K}_{11} \mathbf{J}'_{11} + \mathbf{K}_{12} \mathbf{J}'_{21}) \\
&\quad + \mathbf{J}_{21}'^T (\mathbf{K}_{12}^T \mathbf{J}'_{11} + \mathbf{K}_{22} \mathbf{J}'_{21}) \\
\mathbf{K}_{C12} &= \mathbf{J}_{11}'^T (\mathbf{K}_{11} \mathbf{J}'_{12} + \mathbf{K}_{12} \mathbf{J}'_{22}) \\
&\quad + \mathbf{J}_{21}'^T (\mathbf{K}_{12}^T \mathbf{J}'_{12} + \mathbf{K}_{22} \mathbf{J}'_{22}) \\
\mathbf{K}_{C21} &= \mathbf{K}_{C12}^T \\
\mathbf{K}_{C22} &= \mathbf{J}_{12}'^T (\mathbf{K}_{11} \mathbf{J}'_{12} + \mathbf{K}_{12} \mathbf{J}'_{22}) \\
&\quad + \mathbf{J}_{22}'^T (\mathbf{K}_{12}^T \mathbf{J}'_{12} + \mathbf{K}_{22} \mathbf{J}'_{22}) .
\end{aligned}$$

It is now apparent that: $\mathbf{K}_{C11}$ has entries with units of Nm, i. e., of torsional stiffness; $\mathbf{K}_{C12}$ and $\mathbf{K}_{C21}$ have entries with units of N; and $\mathbf{K}_{C22}$ has entries with units of N/m, i. e., of translational stiffness.

The outcome of the foregoing discussion is that a norm for $\mathbf{K}$ is possible, but not one for $\mathbf{K}_C$, unless of course a characteristic length is introduced to render all the entries of $\mathbf{K}_C$ dimensionally homogeneous. A norm of a matrix is useful as it indicates *how large* the entries of the matrix are. We would like to characterize how stiff a robot is in both joint and Cartesian spaces. In joint space we could adopt any norm, but notice that the 2-norm, introduced in (16.8), would be misleading, as this norm would assign the value of the strongest joint to the overall robot stiffness. A more suitable norm would be the weighted Frobenius norm, introduced in (16.11), which would assign the rms value of the joint stiffnesses to the overall robot stiffness.

To design a robot optimally, we would therefore aim to maximize the Frobenius norm of its stiffness matrix in joint space, while observing constraints on the robot weight, as stiffer joints lead to heavier joints if the same material is used for all joints.

16.4.3 Elastodynamic Performance

For a general design problem, not only the kinetostatic and elastostatic performances, but also the elastodynamic performance have to be considered. In this regard, we introduce the assumptions of Sect. 16.4.2, with the added condition that inertia forces due to the link masses and moments of inertia are now taken into consideration.

The linearized model of a serial robot at the posture given by $\boldsymbol{\theta}_0$, if we neglect damping, is

$$\mathbf{M} \Delta \ddot{\boldsymbol{\theta}} + \mathbf{K} \Delta \boldsymbol{\theta} = \Delta \boldsymbol{\tau} , \tag{16.27}$$

in which $\mathbf{M}$ is the $n \times n$ positive-definite mass matrix introduced in Sect. 16.4.1, while $\mathbf{K}$ was defined in Sect. 16.4.2 as the $n \times n$ positive-definite stiffness matrix in joint space. Both $\mathbf{K}$ and $\mathbf{M}$ were defined in joint-space coordinates, $\Delta\boldsymbol{\theta}$, representing the vector of joint-variable elastic displacements, as in Sect. 16.4.2. These displacements are produced when, as the joints are all locked at a value $\boldsymbol{\theta}_0$ and thereby become ideal linear springs, the robot is subject to a perturbation $\Delta\boldsymbol{\tau}$, to nonzero initial conditions, or to a combination of both.

Under *free vibration*, i. e., under a motion of the system (16.27) caused by nonzero initial conditions and a zero excitation $\Delta\boldsymbol{\tau}$, the foregoing equation can be solved for $\Delta\ddot{\boldsymbol{\theta}}$

$$\Delta \ddot{\boldsymbol{\theta}} = -\mathbf{D} \Delta \boldsymbol{\theta} , \quad \mathbf{D} \equiv \mathbf{M}^{-1} \mathbf{K} , \tag{16.28}$$

in which the matrix $\mathbf{D}$ is known as the *dynamic matrix*. This matrix determines the behavior of the system under consideration, as its eigenvalues are the *natural frequencies* of the system and its eigenvectors the *modal vectors*. Let $\{\omega_i\}_1^n$ and $\{\boldsymbol{f}_i\}_1^n$ denote the sets of eigenvalues and the corresponding eigenvectors of $\mathbf{D}$, respectively. Under the initial conditions $[\Delta\boldsymbol{\theta}(0), \Delta\dot{\boldsymbol{\theta}}(0)]^T$, in which $\Delta\boldsymbol{\theta}(0)$ is proportional to the i-th

eigenvector of $\mathbf{D}$ and $\Delta\dot{\boldsymbol{\theta}}(0) = \mathbf{0}$, the ensuing motion is of the form $\Delta\boldsymbol{\theta}(t) = \Delta\boldsymbol{\theta}(0)\cos\omega_i t$ [16.54].

Furthermore, under the change of variables given by (16.22), the model (16.27) changes to

$$\mathbf{M}\mathbf{J}^{-1}\Delta\ddot{\boldsymbol{x}} + \mathbf{K}\mathbf{J}^{-1}\Delta\boldsymbol{x} = \mathbf{J}^{\mathrm{T}}\Delta\boldsymbol{w}\,.$$

If we now multiply both sides of the above equation by $\boldsymbol{J}^{-\mathrm{T}}$, we obtain the elastodynamic model (16.27) in Cartesian coordinates, namely

$$\mathbf{J}^{-\mathrm{T}}\mathbf{M}\mathbf{J}^{-1}\Delta\ddot{\boldsymbol{x}} + \mathbf{J}^{-\mathrm{T}}\mathbf{K}\mathbf{J}^{-1}\Delta\boldsymbol{x} = \Delta\boldsymbol{w}\,,$$

in which the first matrix coefficient is the mass matrix $\mathbf{M}_{\mathrm{C}}$ in Cartesian coordinates, and the second is identified as $\mathbf{K}_{\mathrm{C}}$, which was introduced in (16.26), i. e.,

$$\mathbf{M}_{\mathrm{C}} \equiv \mathbf{J}^{-\mathrm{T}}\mathbf{M}\mathbf{J}^{-1}\,. \tag{16.29}$$

The elastodynamic model in Cartesian coordinates thus takes the form

$$\mathbf{M}_{\mathrm{C}}\Delta\ddot{\boldsymbol{x}} + \mathbf{K}_{\mathrm{C}}\Delta\boldsymbol{x} = \Delta\boldsymbol{w}\,. \tag{16.30}$$

Again, by virtue of the transformation (16.29), it is apparent that the mass matrix, like its stiffness counterpart, is not invariant under a change of coordinates. Moreover, in a revolute-coupled robot, all the entries of $\mathbf{M}$ have units of kg m^2; however, the entries of $\mathbf{M}_{\mathrm{C}}$ have disparate units. An analysis similar to that conducted in Sect. 16.4.2 for the stiffness matrix in Cartesian space reveals that, if $\mathbf{M}_{\mathrm{C}}$ is partitioned into four 3×3 blocks, then its upper-left block has units of moment of inertia, its lower-right block has units of mass, while its off-diagonal blocks have units of kg m.

Correspondingly, the dynamic matrix in Cartesian coordinates becomes

$$\mathbf{D}_{\mathrm{C}} = \mathbf{M}_{\mathrm{C}}^{-1}\mathbf{K}_{\mathrm{C}}\,. \tag{16.31}$$

It is now a simple matter to prove that *the dynamic matrix is invariant under a change of coordinates*. Indeed, if we substitute transformations (16.26) and (16.29) into (16.31), we obtain

$$\mathbf{D}_{\mathrm{C}} = \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^{\mathrm{T}}\mathbf{J}^{-\mathrm{T}}\mathbf{K}\mathbf{J}^{-1} = \mathbf{J}\mathbf{M}^{-1}\mathbf{K}\mathbf{J}^{-1}\,,$$

in which the dynamic matrix $\mathbf{D}$ in joint coordinates can be readily identified in the final expression, and hence,

$$\mathbf{D}_{\mathrm{C}} = \mathbf{J}\mathbf{D}\mathbf{J}^{-1}\,, \tag{16.32}$$

which shows that $\mathbf{D}_{\mathrm{C}}$ is indeed a similarity transformation of $\mathbf{D}$. As a consequence, the dynamic matrix is indeed invariant under a change of coordinates, the two matrices thus sharing the same sets of eigenvalues, their eigenvectors being related by the same similarity transformation. That is, if the modal vectors in joint space – the eigenvectors of $\mathbf{D}$ – are denoted by $\{\boldsymbol{f}_i\}_1^n$ and the modal vectors in Cartesian space are denoted by $\{\boldsymbol{g}_i\}_1^n$, then these two sets are related by

$$\boldsymbol{g}_i = \mathbf{J}\boldsymbol{f}_i\,, \quad i = 1,\ldots,n\,. \tag{16.33}$$

Therefore, the natural frequencies of the elastodynamic model are the same, regardless of the space in which they are calculated, while their natural modes of vibration change under a similarity transformation.

Under an excitation of the form $\Delta\boldsymbol{\tau} = \boldsymbol{\theta}_0\cos\omega t$ and zero initial conditions, the system is known to respond with a harmonic motion of frequency ω and magnitude that depends on both ω and the system *frequency spectrum* $\{\omega_i\}_1^n$ [16.54]. When ω equals one of the natural frequencies of the system, the response magnitude grows unbounded, a phenomenon known as *resonance*. For this reason, when designing a robot, it is imperative that its frequency spectrum does not involve any of the expected operation frequencies, which can be achieved by designing the robot with stiffness and mass matrices that yield a frequency spectrum outside of the frequency range of the operation conditions.

This design task is not straightforward. Indeed, while the stiffness matrix in joint space is constant, the mass matrix is posture dependent, i. e., $\mathbf{M} = \mathbf{M}(\boldsymbol{\theta})$. Because of this feature, the elastodynamic design of a robot is bound to be iterative: the design can be conducted for a *straw-man task*, i. e., a typical task in the target applications, thus defining a set of postures that lead in turn to a set of mass-matrix values. Then, the frequency spectrum for all these postures can be designed to lie above the frequency range of the straw-man task. Since the robot will eventually be commanded to execute other tasks than the straw-man task, simulation of alternative tasks is required to ensure that the design is safe from a resonance viewpoint.

16.5 Other Robot Types

Serial robots were fully developed, to a great extent, in the nineties, with SCARA systems capable of executing the standard manipulation task defined in Sect. 16.2.1 at the rate of two cycles per second. Other, more sophisticated robots started appearing in the 1980s. Parallel robots were a natural extension of serial robots. They

presented new challenges to researchers, as their forward kinematics turned out to be quite more complex than the inverse kinematics of serial robots. Nowadays a few parallel and hybrid, i. e., serial-parallel robots, are found in applications in industry and other environments. A remarkable type of robot that emerged in the 1990s is *humanoids*, with their own design and control challenges. Flying and swimming robots are also developing fast. An outline of the design challenges brought about by these new types of robots is included below.

16.5.1 Robots Mounted on a Fixed Base: Parallel-Kinematics Machines

Under the umbrella *parallel-kinematics machines* (PKMs) are included robots comprising one base plate (BP), one moving plate (MP) and a set of limbs that couple the two. The earliest robot of this kind that attracted the attention of researchers is what is known as the *Gough–Stewart platform* (GSP), consisting of six limbs, each of the SPU type, where S, P and U stand for spherical, prismatic and universal joint, respectively. In fact, the first two belong to what is known as the six lower kinematic pairs [16.3] – the other four are R, H, C, and F, which are termed revolute, helical, cylindrical and planar pairs. The U joint is a concatenation of two R pairs with axes intersecting at right angles. Three of these six pairs, R, P and H, allow for 1-DOF motion between the two links that they couple; C allows for 2-DOF, while S and F allow for three. Each limb of the GSP thus allows for 6-DOF of the MP with respect to the BP.

The first and foremost task when designing a PKM is the choice of both the type and the number of limbs. As to the latter, it varies from two to six. Apparently, the DOF of a limb should be at least that of the PKM under design. The early PKMs studied, mainly in the 1980s, were invariably 6-DOF systems. *Clavel* [16.6] patented in 1990 what may be the first PKM with *reduced mobility*, i. e., with a DOF = 3, namely, a robot that allows pure translations of its MP. Spherical robots, like *Gosselin*'s *Agile Eye* [16.55], are also of reduced mobility. Different from Clavel's Delta, all links of spherical linkages undergo spherical motion, rotating about the same point – the center of the sphere. Delta is supplied with links that undergo other motions than those of their MP, purely translational.

Other PKMs with reduced mobility appeared later, the best known being the H4, a 4-DOF system, patented in 2001 [16.56], and designed to implement Schönflies displacements (Sect. 16.2.1) of its MP. The foregoing patent came as a response to the need of ever faster pick-and-place robots for the packaging and the electronics industry sectors, when it was apparent that the SCARA systems of the mid eighties had reached their limits because of their serial architecture. The H4 is manufactured by Adept Technology, Inc. under the name *Adept Quattro s650HS parallel robot*.

Motivated by the growing interest in fast robots with reduced mobility, two approaches to the design of PKMs with reduced mobility have been put forth: *screw theory* [16.57] and *group theory* [16.58]. By far, the most favored approach is the former, as screw theory is well established within the community of robot mechanicists. However, group theory offers features not found in screw theory, which is *local* in that its scope is limited to the *velocity level*. Group theory is applicable at the *displacement level*. What is at stake in this context is the *topology* of the robot, namely, the number of joints, links, limbs and the type of joints, what *Hartenberg* and *Denavit* [16.3] term the *number synthesis* and the *type synthesis*. These two activities correspond to the *qualitative synthesis* underlying the design process, which appears at the *conceptual* stage, as opposed to its *quantitative* counterpart; the latter appears at the *embodiment* and the *detail* stages of the design process [16.4]. Recently, a design paradigm based on the concept of complexity has been proposed [16.59], aimed at assessing the complexity of a linkage design variant based on the types of its joints.

As an extension of the Delta robot, ABB Robotics developed the *Flexpicker IRB360*, a hybrid system that comprises a Delta robot in series with a revolute joint at its MP that adds one fourth DOF to the purely translational motion of the Delta. The result is, thus, a hybrid Schönflies motion generator. ABB's robot is used in the food-processing industry.

A major bottleneck in the development of fast robots with reduced mobility is their structural stiffness. With the packaging and electronics industry requiring ever faster pick-and-place robots, capable of speeds greater than 3 cycles/s, high excitation frequencies are generated, which calls for structures much stiffer than those found in other applications areas. The ideal fast robot should have a parallel structure whose mass is negligible when compared to that of its payload and, at the same time, highly stiff. As the robots of interest are intended for the production of Schönflies displacements, any motion of their MP outside of this subgroup, namely, any tilt of the MP, is considered parasitical and should be of as small an amplitude as possible. This calls for a structure that exhibits a high rotational stiffness about any horizontal axis – under the assumption that the only rotational motion of the MP is about a vertical axis, which is usually the case. The challenge here is how to characterize this stiffness. A simple elastodynamic model of the structure is derived by assuming

that the limbs provide a linearly elastic suspension for the MP, which is assumed to be rigid. The structure can be regarded as a *generalized spring* [16.60], namely, a linearly elastic suspension that is fixed to one stationary base at one end and carries a moving rigid body at another end. Moreover, the suspension allows full 6-DOF *small-amplitude* displacements of the body. What is meant by the latter is rotations that entail a *small* angle – as small as to allow for an approximation of the type $\sin\theta \approx \theta$, with θ denoting the angle – and translations small with respect to the dimensions of the body. The motion of the MP can then be represented using a unit screw [16.57] multiplied by a *small* angle Θ. Under these conditions, the Cartesian stiffness matrix is of 6×6, given that the screw is of dimension 6.

In characterizing the rotational stiffness of PKMs, one may think of resorting to the eigenvalues and eigenvectors of their stiffness matrix. There are some issues with this approach, however, for, in light of the different units of the four blocks of this matrix (Sect. 16.4.2) the eigenvalue problem associated with the matrix is meaningful only in a *generalized sense* [16.61], as explained presently. Let us recall the 6×6 Cartesian stiffness matrix $\mathbf{K}_C$, introduced in Sect. 16.4.2, and let us denote with $\boldsymbol{k}_i$ its i-th eigenvector, a unit screw, and with κ_i its corresponding eigenvalue. The generalized eigenvalue problem associated with $\mathbf{K}_C$ now can be stated as

$$\mathbf{K}_C\boldsymbol{k}_i = \kappa_i\boldsymbol{\Gamma}\boldsymbol{k}_i,\ \boldsymbol{k}_i \equiv \begin{pmatrix}\boldsymbol{e}_i\\ \boldsymbol{\mu}_i\end{pmatrix},\ i=1,\ \ldots\ 6\,, \tag{16.34}$$

in which $\boldsymbol{e}_i$ is the unit vector denoting the direction of the axis of the i-th eigenscrew, $\boldsymbol{\mu}_i$ is the *moment* of the axis with respect to the origin and $\boldsymbol{\Gamma}$ is a permutation matrix that exchanges the top and the bottom blocks of a screw array, these two items being defined as

$$\boldsymbol{\Gamma} = \begin{pmatrix}\mathbf{O} & \mathbf{1}\\ \mathbf{1} & \mathbf{O}\end{pmatrix},\ \boldsymbol{\mu}_i = \boldsymbol{e}_i\times\boldsymbol{p}_i + p_i\boldsymbol{e}_i\,, \tag{16.35}$$

for $i=1,\ \ldots\ 6$, $\mathbf{O}$ and $\mathbf{1}$ denoting the 3×3 zero and identity matrices, respectively, while $\boldsymbol{p}_i$ is the position vector of a point on the axis of the eigenscrew $\boldsymbol{k}_i$, while p_i is the *pitch* of $\boldsymbol{k}_i$. The moment of a line $\mathcal{L}$ can be interpreted as that of a unit force whose line of action coincides with $\mathcal{L}$. The generalized eigenvalue problem of (16.34) now leads to

$$\mathbf{A}_i\boldsymbol{k}_i = \mathbf{0},\ \mathbf{A}_i = \begin{pmatrix}\mathbf{K}_{rr} & \mathbf{K}_{rt}-\kappa_i\mathbf{1}\\ \mathbf{K}_{rt}^{\mathrm{T}}-\kappa_i\mathbf{1} & \mathbf{K}_{tt}\end{pmatrix}, \tag{16.36}$$

for $i=1,\ \ldots\ 6$. In light of its definition, (16.34), $\boldsymbol{k}_i \neq \mathbf{0}$, and hence, (16.36) must admit a non-trivial solution, which means that $\mathbf{A}_i$ must be singular, thereby leading to the characteristic equation of $\mathbf{K}_C$, namely,

$$\det\begin{pmatrix}\mathbf{K}_{rr} & \mathbf{K}_{rt}-\kappa\mathbf{1}\\ \mathbf{K}_{rt}^{\mathrm{T}}-\kappa\mathbf{1} & \mathbf{K}_{tt}\end{pmatrix} = 0\,, \tag{16.37}$$

where subscript i has been deleted from κ, for it is no longer needed. In light of the linear homogeneous form in κ of the foregoing 6×6 matrix, its determinant is a sextic polynomial in κ, the eigenvalue sought.

It is worth mentioning that the set of screws does not form, properly speaking, a vector space, for screws do not have a zero – they include a three-dimensional unit vector, which hence cannot vanish. By the same token, the set of screws does not admit a scalar product. Indeed, if two screw arrays are dot-multiplied, their product is meaningless, because it involves additions of quantities with disparate units. The pertinent concept is *reciprocity*: the *reciprocal product* of two unit screws, $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ is defined via matrix $\boldsymbol{\Gamma}$, introduced in (16.35), namely, $\boldsymbol{s}_1^{\mathrm{T}}\boldsymbol{\Gamma}\boldsymbol{s}_2$, which rightfully has units of length. When the reciprocal product of two unit screws vanishes, the screws are said to be *reciprocal* to each other. The physical significance of the concept lies in that a unit screw multiplied by an amplitude F with units of force produces a *wrench*; one multiplied by an amplitude Ω with units of frequency yields a twist. If the wrench happens to be a pure force with line of action intersecting the axis of the twist – the case of a force applied at a door whose line of action intersects the axis of the hinges – then wrench and twist are reciprocal to each other in that the power developed by the wrench on the door undergoing a *feasible* twist is zero.

Now we have a result that parallels one applicable to symmetric $n\times n$ matrices:

Theorem 16.1

The eigenvalues of $\mathbf{K}$ are real and the product $\kappa_i p_i$ is non-negative, while the eigenvectors $\{\boldsymbol{k}_i\}_1^6$ are mutually reciprocal.

its proof being found in [16.62] and [16.63].

Further, *decoupling* of the stiffness matrix means finding a new coordinate frame in which its two off-diagonal blocks vanish. Such a decoupling is not always possible, as shown below (See the *corrigenda* in [16.63]). It is known that the entries of the Cartesian stiffness matrix change according with a similarity transformation applicable to screws, which involves both a shift of origin and a change of the orientation of the original frame [16.64]. If the latter is labeled $\mathcal{A}$ and the new frame $\mathcal{B}$, then $\mathcal{A}$ is carried into $\mathcal{B}$ by means of a shift of the origin given by vector $\boldsymbol{d}$ and a rotation matrix $\mathbf{Q}$. In the sequel, the cross-product matrix

of a vector $\boldsymbol{v} \in \mathbb{R}^3$ will be needed. This is known to be a skew symmetric matrix $\mathbf{V}$ with the property

$$\mathbf{V}\boldsymbol{x} \equiv \boldsymbol{v} \times \boldsymbol{x}, \quad \forall\, \boldsymbol{x} \in \mathbb{R}^3, \quad \mathbf{V} \equiv \mathrm{CPM}(\boldsymbol{v})\,. \tag{16.38}$$

Now let $\mathbf{D} \equiv \mathrm{CPM}(\boldsymbol{d})$, with $\boldsymbol{d}$ defined above as the displacement that takes the origin of $\mathcal{A}$ into that of $\mathcal{B}$. The components of a unit screw $\hat{\mathbf{s}}$ in $\mathcal{B}$ change to those in $\mathcal{A}$ according to the similarity transformation

$$[\hat{\mathbf{s}}]_{\mathcal{A}} = \mathbf{S}[\hat{\mathbf{s}}]_{\mathcal{B}}, \quad \mathbf{S} = \begin{pmatrix} \mathbf{Q} & \mathbf{O} \\ \mathbf{DQ} & \mathbf{Q} \end{pmatrix}, \tag{16.39}$$

the inverse of the 6×6 matrix $\mathbf{S}$ being

$$\mathbf{S}^{-1} = \begin{pmatrix} \mathbf{Q}^{\mathrm{T}} & \mathbf{O} \\ -\mathbf{Q}^{\mathrm{T}}\mathbf{D} & \mathbf{Q}^{\mathrm{T}} \end{pmatrix}. \tag{16.40}$$

Accordingly, the components of the stiffness matrix change, under the same similarity transformation, as [16.63]

$$[\mathbf{K}]_{\mathcal{A}} = \mathbf{\Gamma}\mathbf{S}\mathbf{\Gamma}[\mathbf{K}]_{\mathcal{B}}\mathbf{S}^{-1}, \tag{16.41}$$

which, were it not for matrix $\mathbf{\Gamma}$, looks like the similarity transformation of a $n \times n$ matrix. The presence of $\mathbf{\Gamma}$ is essential by virtue of the definition of the reciprocal product. Let, now,

$$[\mathbf{K}]_{\mathcal{A}} = \begin{pmatrix} \mathbf{K}'_{rr} & \mathbf{K}'_{rt} \\ (\mathbf{K}'_{rt})^{\mathrm{T}} & \mathbf{K}'_{tt} \end{pmatrix}, \; [\mathbf{K}]_{\mathcal{B}} = \begin{pmatrix} \mathbf{K}_{rr} & \mathbf{K}_{rt} \\ \mathbf{K}_{rt}^{\mathrm{T}} & \mathbf{K}_{tt} \end{pmatrix}. \tag{16.42}$$

The distinct blocks of the two matrices are related by

$$\begin{aligned} \mathbf{K}'_{rr} &= \mathbf{Q}(\mathbf{K}_{rr} - \mathbf{K}_{rt}\mathbf{D})\mathbf{Q}^{\mathrm{T}} + \mathbf{DQ}(\mathbf{K}_{rt}^{\mathrm{T}} - \mathbf{K}_{tt}\mathbf{D})\mathbf{Q}^{\mathrm{T}} \\ \mathbf{K}'_{rt} &= (\mathbf{QK}_{rt} + \mathbf{DQK}_{tt})\mathbf{Q}^{\mathrm{T}} \\ \mathbf{K}'_{tt} &= \mathbf{QK}_{tt}\mathbf{Q}^{\mathrm{T}} \end{aligned},$$

the decoupling condition following upon setting $\mathbf{K}'_{rt} = \mathbf{O}$, which leads to

$$\mathbf{Q}^{\mathrm{T}}\mathbf{DQK}_{tt} = -\mathbf{K}_{rt}\,, \tag{16.43}$$

Now, since $\mathbf{D}$ is a cross-product matrix, it is singular – its null space is spanned by vector $\boldsymbol{d}$. Based on Sylvester's Theorem [16.31], which states that the rank of the product of two matrices is at most the smaller of the matrix ranks, the left-hand side of (16.43) is singular, and hence, the right-hand side must also be singular, which is the decoupling condition sought. If block $\mathbf{K}_{rt}$ is singular, then decoupling can be achieved by means of a shift of the origin, without changing the orientation of the frame, i. e., upon choosing $\mathbf{Q} = \mathbf{1}$ in (16.43). This result is summarized below:

Theorem 16.2

The stiffness matrix can be decoupled if and only if its off-diagonal block is singular. When this is the case, decoupling is achieved by a *similarity transformation* involving only a shift of origin.

The design criterion to apply here is, thus, to aim at a singular coupling block of the Cartesian stiffness matrix. This can be achieved by means of finite elements, but these work for one single robot posture. However, with the help of the concept of generalized spring, *Taghvaiepour* et al. [16.65] showed that it is possible to obtain the stiffness matrix of deformable links using that obtained with finite element analysis (FEA) at one particular posture, then transforming this matrix to obtain its counterpart at an alternate posture. The details on how to compute the displacement vector $\boldsymbol{d}$ that helps decouple the stiffness matrix are given in [16.63].

Once the stiffness matrix is decoupled, the eigenvalues of the rotational and translational blocks can be readily found, and assess the performance of the robot from the stiffness viewpoint. As a matter of fact, in the presence of a decoupled matrix, the characteristic equation (16.37) reduces to

$$\det \begin{pmatrix} \mathbf{K}_{rr} & -\kappa\mathbf{1} \\ -\kappa\mathbf{1} & \mathbf{K}_{tt} \end{pmatrix} = 0\,, \tag{16.44}$$

Upon expansion, using the expression for the determinant of a block-matrix [16.66], the foregoing equation leads to

$$\det(\mathbf{K}_{rr})\det(\mathbf{K}_{tt} - \kappa^2\mathbf{K}_{rr}^{-1}) = 0\,,$$

the characteristic equation in κ readily following

$$P(\kappa) \equiv \det(\mathbf{K}_{tt} - \kappa^2\mathbf{K}_{rr}^{-1}) = 0\,, \tag{16.45}$$

which is apparently a cubic equation in κ^2, the outcome being that the eigenvalues of the *decoupled* 6×6 Cartesian stiffness matrix occur in symmetric pairs, i. e., the six eigenscrews of the matrix occur in pairs of opposite chirality, while the absolute values of each pair are identical [16.62, 63].

16.5.2 Mobile Robots

Mobile robots bear an architecture similar to PKMs. Different from the latter, their limbs are not attached to a base plate, but free to move. In fact, their links proximal to the ground end by either feet or wheels, thereby leading to *weheeled* or, correspondingly, *legged* mobile robots. Mobile robots termed *hybrid* have been designed with a combination of wheeled and legged limbs. These robots take on various morphologies, as outlined below.

Wheeled Robots

The earliest of these robots were termed *autonomous guided vehicles* (AGV). They were provided with a chassis and two motors, to implement the 2-DOF proper of any terrestrial vehicle equipped with conventional – as opposed to omnidirectional – wheels. These systems were capable of following a trajectory printed on the flat ground of a manufacturing facility. They had thus a rather limited mobility, but their load-carrying capacity was virtually unlimited.

Robots with conventional wheels are generally equipped with two pairs of pneumatic tires, that allow them to adapt themselves to irregular surfaces. Moreover, these robots can either bear steerable wheels, like MDA's *Rover Chassis Prototype (RCP)*, or change course by means of maneuvers involving different angular velocities of their tires mounted on axles fixed to the chassis, like the *Khepera* robot.

Extraterrestrial exploration has prompted the development of *rovers*, namely, wheeled robots intended for roaming on unstructured terrain, either soft, like the Moon's or Mars's regolith, or hard, like rocks. Design issues here pertain mainly to the wheels, as these must adapt themselves to a rich variety of conditions and terrain constitution. Rovers are currently designed to travel at very low speeds, sometimes of a few meters per day, the case of NASA's *Curiosity* (This rover landed on Mars on August 6, 2012.). The main issue here is for the wheels to provide enough traction and prevent spinning without advancing. The latter is guaranteed by the use of individual motors on each wheel. Traction is enhanced by means of *grousers* on the wheel periphery. *Curiosity* and *RCP* feature metal wheels with grousers.

Legged Robots

This kind of robots has evolved tremendously since the earlier systems developed in the eighties: *Ohio state university (OSU) hexapod* and the *OSU adaptive suspension vehicle (ASV)*, among others. Current legged robots range from those with six legs, *X-RHEX Lite*, with a planar symmetry and three identical legs on each side of the body, or the *multi-appendage robotic system (MARS)*, with its legs distributed along the vertices of a regular hexagon, to four-legged robots with an architecture similar to that of mammals, like *Big Dog* and *Cheetah*. Special mention deserve bipeds, of which the best known are humanoids, with the architecture of the human body, namely, *ASIMO* (advanced step in innovative mobility) and its competitor, *QRIO* (quest for curiosity). Design issues vary, depending on the number of legs. Hexapods offer a combination of static stability and maneuverability, which their four-legged counterparts do not. By far, humanoids are the highest demanding in terms of static and dynamic stability, but offer the highest mobility.

Hybrid Robots

Hybrid robots stem from the combination of the two foregoing types, wheeled with legged. These are systems that look like wheeled robots, but that feature wheels instead of feet. They offer the advantages – and the disadvantages – of the two foregoing types. A recent example of this kind of robots is NASA's ATHLETE (all-terrain hex-legged extra-terrestrial explorer) robot, that features six articulated legs stemming from the edges of a regular hexagon defining the robot body and ending in wheels. Its design is intended for service to the planned Lunar base. According to NASA, the 15-ton lunar habitat would be [16.67]

> *mounted on top of the six-legged robot. The habitat could walk right off of the lunar lander, and then proceed to any desired location. Wheeled locomotion would be used for level ground; more challenging terrain could be negotiated with the full use of the flexible legs.*

16.5.3 Aquatic Robots

Aquatic robots are currently being marketed for operation on water, whether on the surface or underneath. An example of a surface robot is the *Mantra*, intended to act as a lifeguard and help swimmers in distress. Some aquatic robots have been designed with the morphology of fish [16.68]. One swimming robot designed with a hexapod morphology, featuring six flippers in lieu of legs is *Aqua*, developed at McGill University, and depicted in Fig. 16.5.

By far, the aquatic robot with the highest profile is the robotic submarine *Bluefin*, which became a celebrity

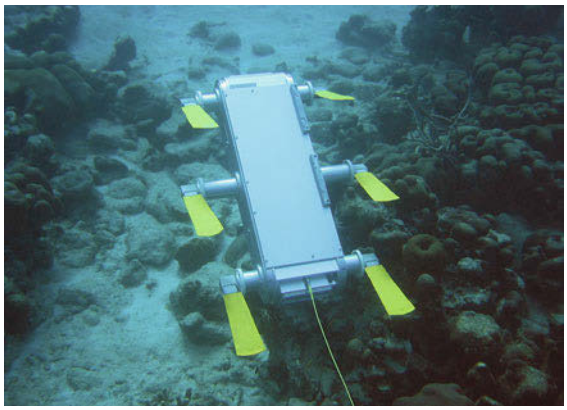


Fig. 16.5 Aqua, an amphibious robot

in the search for the debris of Flight MHS370 (Flight went missing on March 7, 2014 at 16:41 GMT.).

16.5.4 Flying Robots

This field is becoming quite active, with some robots mimicking the morphology of insects, and falling into the category of micro-robots [16.69]; other flying robots are designed as unmanned aerial vehicles (UAV) [16.70]. Intensive research is now being reported in the design and control of two novel types of UAVs, namely, drones and quadrotors. The control of drones undergoing fast maneuvers has been made possible by the inception of gyro-free inertial measurement systems fabricated with microelectromechanical systems (MEMS) technology. Highly maneuverable quadrotors, especially suitable for surveillance and reconnaissance missions, are small-size, light-weight flying machines supplied with *two pairs of counter-rotating rotors and propellers, located at the vertices of a square frame* [16.71]. A recent issue of IEEE's *Robotics & Automation Magazine* [16.72] includes a state-of-the-art account of quadrotor technology.

16.6 Summary

This chapter is devoted to the design of robots, with a focus on serial architectures. In this regard, we started by proposing a stepwise design procedure; then, we recalled the main issues in robot design. These issues pertain to workspace geometry, and to the kinetostatic, the dynamic, the elastostatic and the elastodynamic performances. In doing this, the mathematics behind the concepts addressed was briefly outlined, so as to make the chapter self-contained.

A major design issue in the design of PKM, namely, stiffness, was given due attention. Design issues pertaining to other kinds of robots, like wheeled, legged, flying and aquatic, were outlined.

References

16.1 O. Bottema, B. Roth: *Theoretical Kinematics* (North-Holland, Amsterdam 1979), also available by Dover Publishing, New York 1990

16.2 J. Angeles: The qualitative synthesis of parallel manipulators, ASME J. Mech. Des. **126**(4), 617–624 (2004)

16.3 R. Hartenberg, J. Denavit: *Kinematic Synthesis of Linkages* (McGraw-Hill, New York 1964)

16.4 G. Pahl, W. Beitz: *Engineering Design. A Systematic Approach*, 3rd edn. (Springer, London 2007), translated from the original Sixth Edition in German

16.5 M. Petterson, J. Andersson, P. Krus, X. Feng, D. Wäppling: Industrial robot design optimization in the conceptual design phase, Proc. IEEE Int. Conf. Mechatron. Robotics, Vol. 2, ed. by P. Drews (2004) pp. 125–130

16.6 R. Clavel: Device for the movement and positioning of an element in space, US Patent 497 6582 (1990)

16.7 R. Vijaykumar, K.J. Waldron, M.J. Tsai: Geometric optimization of serial chain manipulator structures for working volume and dexterity, Int. J. Robotics Res. **5**(2), 91–103 (1986)

16.8 J.P. Merlet: *Parallel Robots* (Kluwer, Boston 2006)

16.9 J. Angeles: *Fundamentals of Robotic Mechanical Systems*, 3rd edn. (Springer, New York 2007)

16.10 K. Hoffman: *Analysis in Euclidean Space* (Prentice Hall, Englewood Cliffs 1975)

16.11 L. Burmester: *Lehrbuch der Kinematik* (Arthur Felix, Leipzig 1886)

16.12 J.M. McCarthy: *Geometric Design of Linkages* (Springer, New York 2000)

16.13 J. Angeles: Is there a characteristic length of a rigid-body displacement?, Mech. Mach. Theory **41**, 884–896 (2006)

16.14 H. Li: Ein Verfahren zur vollständigen Lösung der Rückwärtstransformation für Industrieroboter mit allegemeiner Geometrie, Ph.D. Thesis (Universität-Gesamthochschule Duisburg, Duisburg 1990)

16.15 M. Raghavan, B. Roth: Kinematic analysis of the 6R manipulator of general geometry, Proc. 5th Int. Symp. Robotics Res., ed. by H. Miura, S. Arimoto (MIT, Cambridge 1990)

16.16 J. Angeles: The degree of freedom of parallel robots: A group-theoretic approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Barcelona (2005) pp. 1017–1024

16.17 C.C. Lee, J.M. Hervé: Translational parallel manipulators with doubly planar limbs, Mech. Mach. Theory **41**, 433–455 (2006)

16.18 B. Roth: Performance evaluation of manipulators from a kinematic viewpoint, National Bureau of Standards – NBS SP **495**, 39–62 (1976)

16.19 A. Kumar, K.J. Waldron: The workspaces of a mechanical manipulator, ASME J. Mech. Des. **103**, 665–672 (1981)

16.20 D.C.H. Yang, T.W. Lee: On the workspace of mechanical manipulators, ASME J. Mech. Trans. Autom. Des. **105**, 62–69 (1983)

16.21 Y.C. Tsai, A.H. Soni: An algorithm for the workspace of a general n-R robot, ASME J. Mech. Trans. Autom. Des. **105**, 52–57 (1985)

16.22 K.C. Gupta, B. Roth: Design considerations for manipulator workspace, ASME J. Mech. Des. **104**, 704–711 (1982)

16.23 K.C. Gupta: On the nature of robot workspace, Int. J. Robotics Res. **5**(2), 112–121 (1986)

16.24 F. Freudenstein, E. Primrose: On the analysis and synthesis of the workspace of a three-link, turning-pair connected robot arm, ASME J. Mech. Trans. Autom. Des. **106**, 365–370 (1984)

16.25 C.C. Lin, F. Freudenstein: Optimization of the workspace of a three-link turning-pair connected robot arm, Int. J. Robotics Res. **5**(2), 91–103 (1986)

16.26 T. Yoshikawa: Manipulability of robotic mechanisms, Int. J. Robotics Res. **4**(2), 3–9 (1985)

16.27 J. Loncaric: Geometric Analysis of Compliant Mechanisms in Robotics, Ph.D. Thesis (Harvard University, Cambridge 1985)

16.28 B. Paden, S. Sastry: Optimal kinematic design of 6R manipulators, Int. J. Robotics Res. **7**(2), 43–61 (1988)

16.29 J.K. Salisbury, J.J. Craig: Articulated hands: Force control and kinematic issues, Int. J. Robotics Res. **1**(1), 4–17 (1982)

16.30 G.H. Golub, C.F. Van Loan: *Matrix Computations* (Johns Hopkins Univ. Press, Baltimore 1989)

16.31 G. Strang: *Linear Algebra and Its Applications*, 3rd edn. (Harcourt Brace Jovanovich, San Diego 1988)

16.32 A. Dubrulle: An optimum iteration for the matrix polar decomposition, Electron. Trans. Numer. Anal. **8**, 21–25 (1999)

16.33 W.A. Khan, J. Angeles: The kinetostatic optimization of robotic manipulators: The inverse and the direct problems, ASME J. Mech. Des. **128**, 168–178 (2006)

16.34 H. Pottmann, J. Wallner: *Computational Line Geometry* (Springer, Berlin, Heidelberg 2001)

16.35 H. Asada: A geometrical representation of manipulator dynamics and its application to arm design, Trans. ASME J. Dyn. Sys. Meas. Contr. **105**(3), 131–135 (1983)

16.36 T. Yoshikawa: Dynamic manipulability of robot manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1985) pp. 1033–1038

16.37 P.A. Voglewede, I. Ebert-Uphoff: Measuring closeness to singularities for parallel manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4539–4544

16.38 A. Bowling, O. Khatib: The dynamic capability equations: A new tool for analyzing robotic manipulator performance, IEEE Trans. Robotics **21**(1), 115–123 (2005)

16.39 C.M. Gosselin, J. Angeles: A new performance index for the kinematic optimization of robotic manipulators, Proc. 20th ASME Mech. Conf. (1988) pp. 441–447

16.40 C. Gosselin, J. Angeles: The optimum kinematic design of a planar three-degree-of-freedom parallel manipulator, ASME J. Mech. Trans. Autom. Des. **110**, 35–41 (1988)

16.41 A. Bicchi, C. Melchiorri, D. Balluchi: On the mobility and manipulability of general multiple limb robots, IEEE Trans. Robotics Autom. **11**(2), 232–235 (1995)

16.42 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Global task space manipulability ellipsoids for multiple-arm systems, IEEE Trans. Robotics Autom. **7**, 678–685 (1991)

16.43 C. Melchiorri: Comments on Global task space manipulability ellipsoids for multiple-arm systems and further considerations, IEEE Trans. Robotics Autom. **9**, 232–235 (1993)

16.44 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Reply to comments on Global task space manipulability ellipsoids for multiple-arm systems' and further considerations, IEEE Trans. Robotics Autom. **9**, 235–236 (1993)

16.45 F.C. Park: Optimal robot design and differential geometry, ASME J. Mech. Des. **117**, 87–92 (1995)

16.46 F.C. Park, J. Kim: Manipulability of closed kinematic chains, ASME J. Mech. Des. **120**(4), 542–548 (1998)

16.47 A. Liégeois: Automatic supervisory control for the configuration and behavior of multibody mechanisms, IEEE Trans. Sys. Man. Cyber. **7**(12), 842–868 (1977)

16.48 C.A. Klein, C.H. Huang: Review of pseudo-inverse control for use with kinematically redundant manipulators, IEEE Trans. Syst. Man Cybern. **13**(2), 245–250 (1983)

16.49 J.M. Hollerbach: Optimum kinematic design of a seven degree of freedom manipulator. In: *Robotics Research: The Second International Symposium*, ed. by H. Hanafusa, H. Inoue (MIT, Cambridge 1985)

16.50 M.W. Spong: Remarks on robot dynamics: Canonical transformations and riemannian geometry, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1992) pp. 454–472

16.51 T.J. Graettinger, B.H. Krogh: The acceleration radius: A global performance measure for robotic manipulators, IEEE J. Robotics Autom. **4**(11), 60–69 (1988)

16.52 M. Griffis, J. Duffy: Global stiffness modeling of a class of simple compliant couplings, Mech. Mach. Theory **28**, 207–224 (1993)

16.53 S. Howard, M. Zefran, V. Kumar: On the 6×6 cartesian stiffness matrix for three-dimensional motions, Mech. Mach. Theory **33**, 389–408 (1998)

16.54 L. Meirovitch: *Fundamentals of vibrations* (McGraw-Hill, Boston 2001)

16.55 C.M. Gosselin, J.-F. Hamel: The agile eye: A high-performance three-degree-of-freedom camera-orienting device, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 781–786

16.56 O. Company, F. Pierrot, T. Shibukawa, K. Morita: Four-Degree-of-Freedom Parallel Robot, EU Patent No. EP 108 4802 (2001)

16.57 X. Kong, C. Gosselin: *Type synthesis of parallel mechanisms*, Springer Tract. Adv. Robotics, Vol. 33 (Springer, Berlin, Heidelberg 2007)

16.58 J. Hervé: The Lie group of rigid body displacements, a fundamental tool for mechanism design, Mech. Mach. Theory **34**, 719–730 (1999)
16.59 W.A. Khan, J. Angeles: A novel paradigm for the qualitative synthesis of simple kinematic chains based on complexity measures, ASME J. Mech. Robotics **3**(3), 0310161–03101611 (2011)
16.60 J. Lončarivc: Normal forms of stiffness and compliance matrices, IEEE J. Robotics Autom. **3**(6), 567–572 (1987)
16.61 X. Ding, J.M. Selig: On the compliance of coiled springs, Int. J. Mech. Sci. **46**, 703–727 (2004)
16.62 J.M. Selig: The spatial stiffness matrix from simple stretched springs, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 3314–3319
16.63 J. Angeles: On the nature of the Cartesian stiffness matrix, Rev. Soc. Mex. Ing. Mec. **3**(5), 163–170 (2010)
16.64 A.K. Pradeep, P.J. Yoder, R. Mukundan: On the use of dual-matrix exponentials in robotic kinematics, Int. J. Robotics Res. **8**(5), 57–66 (1989)
16.65 A. Taghvaeipour, J. Angeles, L. Lessard: On the elastostatic analysis of mechanical systems, Mech. Mach. Theory **58**, 202–216 (2013)
16.66 D. Zwillinger (Ed.): *CRC Standard Mathematical Tables and Formulae*, 31st edn. (CRC, Boca Raton 2002)
16.67 B. Christensen: Robotic lunar base with legs changes everything, http://www.space.com/5216-robotic-lunar-base-legs.html
16.68 L. Wen, T. Wang, G. Wu, J. Li: A novel method based on a force-feedback technique for the hydrodynamic investigation of kinematic effects on robotic fish, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011), Paper No. 0828
16.69 L.L. Hines, V. Arabagi, M. Sitti: Free-flight simulation and pitch-and-roll control experiments of a sub-gram flapping-flight micro aerial vehicle, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011), Paper No. 0602
16.70 C.E. Thorne, M. Yim: Towards the dvelopment of gyroscopically controled micro air vehicles, Proc. 2011 IEEE Int. Conf. Robotics Autom. (2011), Paper No. 1800
16.71 T. Lee, M. Leok, N.H. McClamroch: Geometric tracking control of a quadrotor UAV on SE(3), Proc. 49th IEEE Conf. Decis. Control (2010)
16.72 Robots take flight: Quadroto unmanned aerial vehicles, IEEE Robotics Autom. Mag. **19**(3) (2012), http://online.qmags.com/RAM0912?pg=3&mode=2#pg1&mode2

Multimedia Contents

17. Limbed Systems

Shuuji Kajita, Christian Ott

A limbed system is a mobile robot with a body, legs and arms. First, its general design process is discussed in Sect. 17.1. Then we consider issues of conceptual design and observe designs of various existing robots in Sect. 17.2. As an example in detail, the design of a humanoid robot HRP-4C is shown in Sect. 17.3. To design a limbed system of good performance, it is important to take into account of actuation and control, like gravity compensation, limit cycle dynamics, template models, and backdrivable actuation. These are discussed in Sect. 17.4.

In Sect. 17.5, we overview divergence of limbed systems. We see odd legged walkers, leg–wheel hybrid robots, leg–arm hybrid robots, tethered walking robots, and wall-climbing robots. To compare limbed systems of different configurations, we can use performance indices such as the gait sensitivity norm, the Froude number, and the specific resistance, etc., which are introduced in Sect. 17.6.

17.1 **Design of Limbed Systems** 420

17.2 **Conceptual Design** 420
17.2.1 Conceptual Design Issues 421
17.2.2 Case Studies 421

17.3 **Whole Design Process Example** 423
17.3.1 Conceptual Design of HRP-4C 423
17.3.2 Mechanical Design 424
17.3.3 Electrical System Design 425

17.4 **Model Induced Design** 427
17.4.1 Dynamic Walking 427
17.4.2 Template Models 430
17.4.3 Robots for Control Based on Backdrivable Actuation 432

17.5 **Various Limbed Systems** 434
17.5.1 Odd Legged Walkers 434
17.5.2 Leg-Wheel Hybrid Robots 435
17.5.3 Leg-Arm Hybrid Robots 436
17.5.4 Tethered Walking Robots 436
17.5.5 Wall-Climbing Robots 436

17.6 **Performance Indices** 437
17.6.1 Stability Margins 437
17.6.2 Gait Sensitivity Norm 437
17.6.3 Duty Factor and Froude Number 438
17.6.4 Specific Resistance 439

Video-References 439

References 440

In this chapter, we refer a limbed system as a mobile robot which consists of a body, legs and arms. Its general concept is illustrated in Fig. 17.1.

Let us define that a limbed system must have a body and at least one leg (lower limb) to support and to propel itself. A leg interacts with the environment by its end effector (foot). It can also have an arbitrary number of arms (upper limb) to manipulate target objects by their end effectors (hands).

From a body-centered point of view, legs can be regarded as special arms designed for manipulating the earth. This justifies us to treat both of the legs and the arms as limbs in a unified manner, thus we chose the word *limbed systems* as the title of this chapter.

17.1 Design of Limbed Systems

General robot design process has already been discussed in the previous 16, therefore, we will concentrate on the specific issues on limbed systems. A design process of actual limbed system can be seen in literatures [17.1–3]. A typical development process may takes the following steps:

- *Step 1*: *Basic requirements.* A designer must specify the class of tasks performed by the limbed system and its requirements, target velocity, payload, expected environment, etc.
- *Step 2*: *Conceptual design.* The number of legs and arms, their topology, and associated gait are decided simultaneously. They must be designed at the same time, because the leg structure and gait are tightly coupled, thus they should not be considered separately. Geometric dimensions of links are also determined based on the basic requirements.
- *Step 3*: *Detailed design.* A designer selects actuators, sensors, reduction gears, and other mechanical/electrical components to realize the conceptual design. Then the structural elements are designed to integrate the devices.
- *Step 4*: *Evaluation.* Evaluate the design by building a prototype model or performing computer simulations. In our experience, the first design can never meet the basic requirement, and we are forced to return to Step 3 to change the motor type or the reduction ratio. Sometimes, the simulation results suggest us reconsidering the conceptual design or even rethinking the basic requirements.

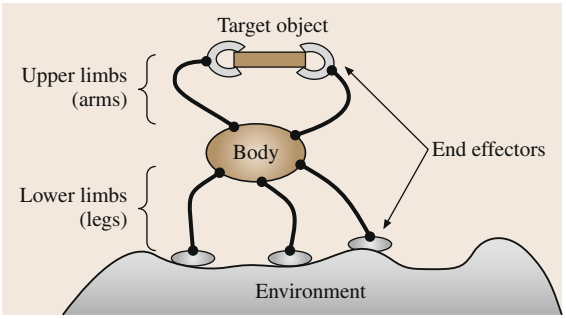


Fig. 17.1 General concept of a limbed system

Figure 17.2 overviews the development process of a limbed system. Note that a development of a limbed system is an extremely dynamic process, rather than a static top-down development. The detail of each step will be discussed in the following subsections.

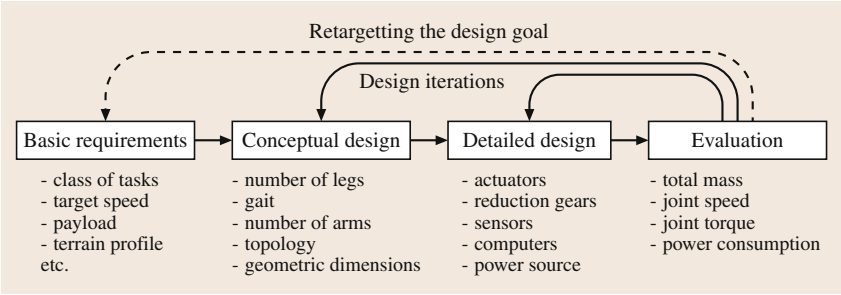


Fig. 17.2 Typical development process of a limbed system

17.2 Conceptual Design

At the beginning, a designer must have *a class of tasks* in his mind, for example:

- Enter a damaged nuclear power plant, climb stairs, and close specified valves
- Explore a planet to seek a new lifeforms
- Carry a person and climb stairs
- Perform dances to entertain its audience, etc..

Note that it must be a class of tasks and should not be a single task, because the latter can be realized by a single degree-of-freedom machinery as expressed in the former chapter.

A class of tasks should be compiled into more specific goals. For example, target speed of traveling, maximum step length, maximum payload, or expected terrain profiles (ex. maximum height of obstacles).

Those figures give a solid guide line for a design process.

17.2.1 Conceptual Design Issues

The goal of conceptual design is to decide the basic kinematic property of the robot, the number of legs and arms, limb topology, and geometric dimensions of the links. A robot designer should take into account the following issues:

- *Gait type*: Leg motion pattern for walking or running is called *gait* and it greatly affects limbed system design. For example, a robot for static gait should be designed to have sprawled posture to maximize its stability. Whereas, a robot for dynamic gait is designed to have erect posture with high center of mass to get a longer time constant of its falling. The gait generation is discussed in 75.
- *Biomimesis*: Some robots are designed to mimic the mechanical structure of a living creature as accurate as possible. This is beneficial for deeper understanding of biological systems. For example, *Quinn* and *Ritzmann* have built a hexapod robot with kinematics that are remarkably similar to the cockroach *Blaberus discoidalis* [17.4] (VIDEO 521). Biomimesis is also meaningful in terms of entertainment applications. *Hirukawa* et al. developed biped walking dinosaur robots of *Tyrannosaurus Rex* and *Parasaurolophus* for an exposition [17.5]. Those robots were built on 30% scale of the real dinosaurs due to safety issues; nevertheless, they could attract many audience by their realistic outlooks at the 2005 World Exposition, Aichi, Japan.
- *Bioinspired dynamics*: To reproduce the robust and versatile locomotion of animals, some designers pay their attention on essential dynamics of the locomotion, rather than the mechanical similarity. *Koditschek* et al. emphasized this standpoint on the development of their hexapod robot RHex, which equipped with only six actuators and yet can run on various irregular terrain [17.6].
- *Mechanical simplicity*: It is preferred to realize the target tasks with a mechanism as simple as possible. Therefore, the design is always biased to use fewer number of actuators. VIDEO 520 shows an example of robot design with minimum numbers of actuators.
- *Limb workspace*: A limb must have at least three degrees of freedom (DOF) to freely locate the tip of a limb in 3-D space. It must have also at least six DOF for the arbitrary orientation of the end effector in 3-D space.
- *Load bearing*: Proper joint assignment can reduce the joint torque to support the body weight. In VIDEO 517, we can see a unique hexapod leg design for efficient load bearing.

17.2.2 Case Studies

Let us observe how abovementioned issues are reflected in the conceptual design of actual limbed systems.

Underactuatedness Versus Redundancy

We compare two monopods which were designed under different policies on actuation. Figure 17.3a is the 3-D one-legged hopping machine developed by *Raibert* [17.1]. This robot is driven by one pneumatic cylinder in its leg and two hydraulic actuators in its hip joints. Therefore, it has only three actuated DOF while 3-D rigid body motion requires 6. Despite its underactuatedness, it can freely travel in 3-D space keeping its body upright thanks to its balance control.

Figure 17.3b shows the one-legged robot developed by *Tajima* and *Suga* [17.7]. It has seven actuated DOFs in total, three at the hip, one at the knee, two at the ankle, and one at the toe. Therefore, this robot is a redundant system having one extra DOF in addition to the six required for 3-D motion. This redundancy can help the robot to reduce the excessive joint speed required at hopping. As a result, successful hopping motion control was realized by using servo motors with reduction gears for all joints.

Serial Versus Parallel

Legs of a limbed system can be configured either as a serial or parallel structures. In most cases, each leg of a biped robot is designed as serial chain of six ac-

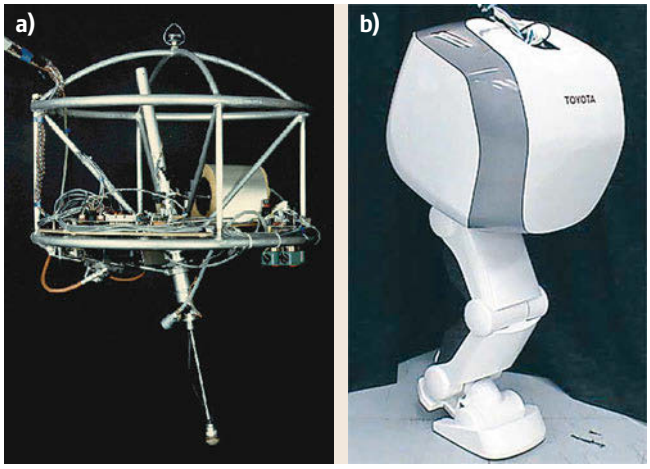


Fig.17.3a,b Monopod robot designs. **(a)** MIT 3-D One-Leg Hopper (1983); **(b)** Toyota monopod (2006)

tuated DOF, where three are used in the hip, one in the knee and two in the ankle. HRP-2 L developed by *Kaneko* et al. [17.8], as shown in Fig. 17.4a, is one of such examples. For such robots, the most difficult part is the hip joint design. A hip joint mechanism must contain three axis of rotation (roll/pitch/yaw) bearing high torque caused by the upper body weight. This is serious especially for a robot to carry heavy loads.

Takanishi et al. solved this problem by introducing legs of the Stewart platform for their WL-16R, a human carrying biped locomotor (Fig. 17.4b). Each leg consists of six linear actuators driven by DC-geared motors and ball screws. In this design, their robot can carry a human of 50 kg weight [17.9].

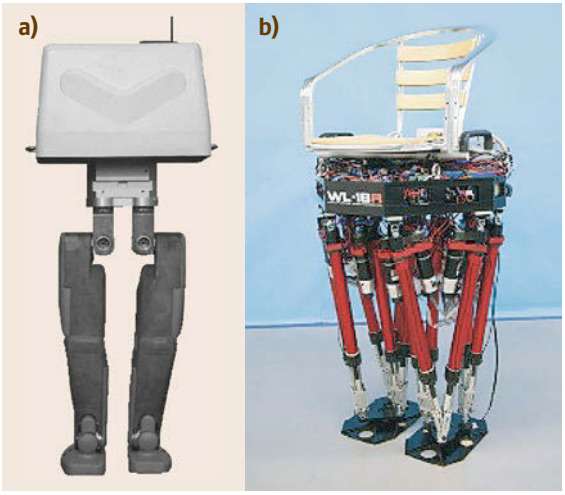


Fig.17.4a,b Biped robot designs. **(a)** HRP-2L (2001); **(b)** WL-16R (2003)

Static Gait Versus Dynamic Gait

Figure 17.5a is TITAN III developed by Hirose and his colleagues [17.10]. This is one of the successful quadruped robots in 1980s. Each leg is a 3-D pantograph mechanism driven (PANTOMEC) by three linear actuators (ball screw with DC motor) equipped on its body, thus it has 12 DOF. To perform statically balanced walk, this robot was designed to have a low center of mass and widely spread legs.

Figure 17.5b is BigDog, an energy autonomous hydraulic quadruped robot of 1 m tall, 1 m long, and 90 kg developed by *Buehler* et al. [17.11]. Each leg has one passive linear pneumatic compliance in the lower leg, three active joints for knee, hip pitch, and roll. There-

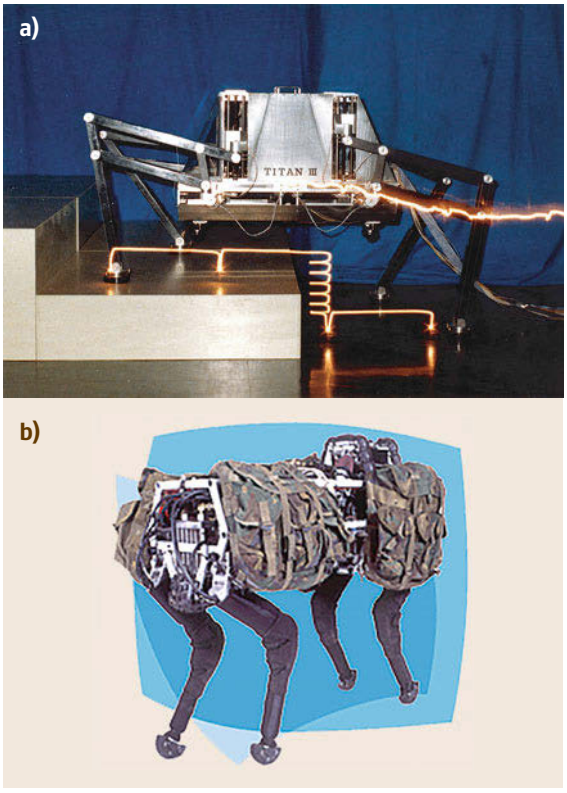


Fig.17.5a,b Quadruped designs. **(a)** TITAN III (1984); **(b)** BigDog (2005)

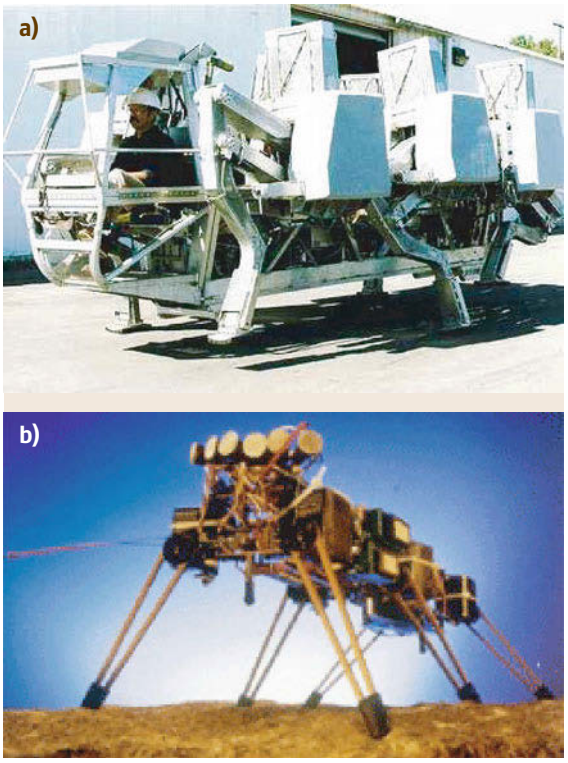


Fig.17.6a,b Hexapod designs. **(a)** ASV (1986); **(b)** Genghis (1989)

fore it has 12 actuated DOF and four passive DOF. This robot can demonstrate remarkable walking performance in an outdoor environment, as well as can maintain its balance even when kicked by a human. This ability of dynamic balancing is helped by its design of high center of mass with the leg assignment with narrow lateral separation.

Big Versus Small

Design concept is deeply affected by the robot size. Figure 17.6a shows one of the most famous hexapods, the adaptive suspension vehicle (ASV) developed by *Waldron* et al. [17.2, 12]. The ASV is a hydraulically driven hexapod robot which can carry a parson over rough terrain. Its length and height are 5.2 m and 3.0 m, respectively, and weighs 2700 kg. To walk around in 3-D space, each leg requires 3 DOFs, thus the robot has 18 DOF in total.

Figure 17.6b shows a small hexapod robot Genghis (35 cm length and 1 kg weight) developed by *Brooks* [17.13]. This robot could also demonstrate robust walking in 3-D space, but its weight is less than a thousandth of the ASV. As a result, Genghis does not need precise 3-D foot positioning for its walking control. Each leg has only two DOF and the robot has 12 DOF in total. Standing with three legs, the body position and orientation can be fully controlled as long as the feet is allowed to slip on the ground.

Degree of Biomimesis

We can increase the degree of biomimesis by adding DOF; however, fewer DOF is preferable in terms of engineering. The first version of ASIMO has 26 DOF in total, 6 for each leg, 5 for each arm, one for each hand, and two for the head (Fig. 17.7a) [17.14]. As a biped walking humanoid robot, this is a reasonable configuration.

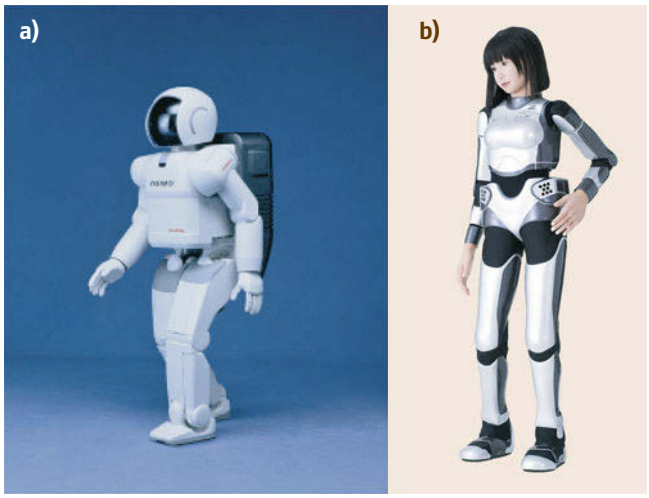


Fig.17.7a,b Humanoid designs. **(a)** ASIMO (2000); **(b)** HRP-4C (2009)

Cybernetic human HRP-4C (Fig. 17.7b) was designed to be as close as a human in consideration of applications in entertainment industry [17.15, 16]. It has 44 DOF in total, 7 for each leg, 6 for each arm, 2 for each hand and, three for its waist, 3 for its neck, and 8 for its face. Its design detail is discussed in the next section.

17.3 Whole Design Process Example

In this section, we explain the design process of a humanoid robot HRP-4C. We do this to give the readers an example of the whole development process of a limbed system. The readers are also recommended to see the comprehensive development report for the humanoid robot LOLA written by *Lohmeier* [17.3]. Also please watch another successful humanoid designs in VIDEO 522 and VIDEO 526. Note that a general mechanical design and construction process have already been discussed in this handbook (Chap. 4).

17.3.1 Conceptual Design of HRP-4C

At the beginning of our project, we defined *Cybernetic human* as a humanoid robot with the following features:

1. Have the appearance and shape of a human being.
2. Can walk and move like a human being.
3. Can interact with humans using speech recognition and so forth.

Such robots can be used in the entertainment industry, for example, exhibitions and fashion shows. It can also be used as a human simulator to evaluate devices for humans.

As the successor of our previous humanoid robots HRP-2 and HRP-3 [17.17, 18], we call our new humanoid robot HRP-4C, C stands for cybernetic human.

To determine the target shape and dimensions of HRP-4C, we used the anthropometric database for Japanese population, which was measured and compiled by *Kouchi* et al. [17.19]. The database provides

the dimensions of different Japanese groups of ages and sex. Considering the entertainment applications like fashion show, we picked the average young female data (age from 19 to 29). Figure 17.8 shows part of the dimensions provided by the database.

To reproduce women's graceful motion, we asked a professional walking model to perform walking, turning, sitting on chair, and other motions. The performance was motion-captured and used to evaluate possible joint configurations proposed for HRP-4C structure. From the captured data, the required joint workspace was estimated. In addition, we estimated the motor power during biped walking to design the appropriate leg joint configuration.

Figure 17.9 shows the joint configuration we finally decided for HRP-4C. To reproduce the human motion, three DOFs (roll, pitch, yaw) were assigned for each of the waist and the neck joints. In addition, we put active toe joints to realize biped walking with wide stride.

17.3.2 Mechanical Design

Figure 17.10 shows the mechanism to drive the knees of HRP-4C. The rotation of the servo motor is transmitted to the input of the harmonic drive gear by way of the pulleys and the timing belt, and the output of the harmonic gear is connected to the lower leg link. By this way, we can obtain high torque output without backlash which is required for dynamic biped walking. Note that

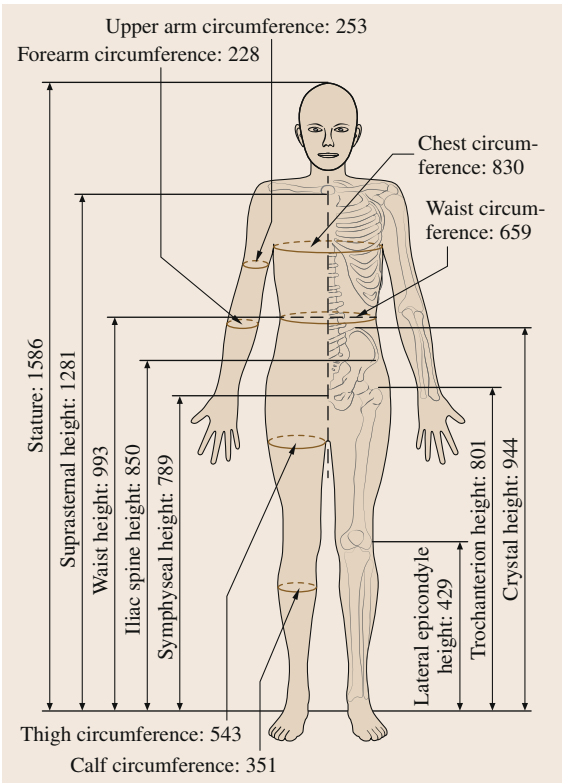


Fig. 17.8 Anthropometric data of average young Japanese female (after [17.19], all measures in mm)

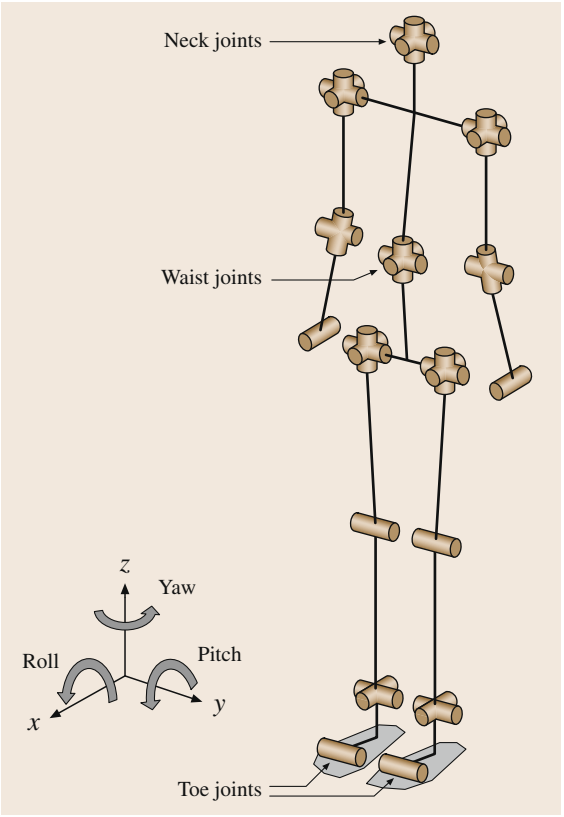


Fig. 17.9 Joint configuration of HRP-4C body (joints for the face and hands are omitted)

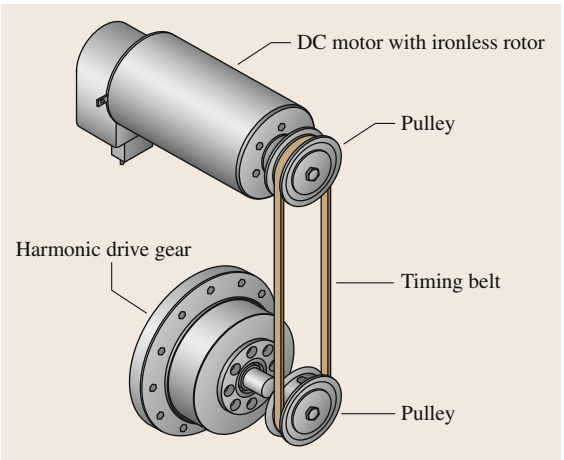


Fig. 17.10 Joint drive system (after [17.8])

the pulleys and the timing belt offer us a flexible choice of the motor placement as well as the total reduction ratio. The belt compliance becomes negligible thanks to the high reduction ratio of the harmonic gear at the final stage.

The same mechanisms of Fig. 17.10 are used for most of the joints of HRP-4C; however, there are some exceptions. The first example is the ankle joint. Figure 17.11 illustrates the mechanism of the left lower leg link and the ankle. The ankle pitch and roll joints are driven by two servomotors (Motor #1 and Motor #2) embedded in the shin link. Motor #1 rotates the ball screw to create the linear motion via timing belt and pulleys. Then the ball screw's nut pushes or pulls the connecting rod that finally drives the ankle pitch joint.

Motor #2's rotation is transmitted to the bevel gear #1, timing belt #2, and the second bevel gear. The bevel gear #2 rotates the harmonic drive gear and it finally rotates the ankle roll joint. This elegant yet complicated mechanism was necessary to realize the slim outer shape of the HRP-4C ankle which must have close dimensions of human ankle.

Figure 17.12 is the toe joint mechanism of HRP-4C. To avoid sole plate separation which may catch obstacles between the plates, the toe joint axis should locate as close as possible to the floor surface. For this purpose, a four bar linkage is used to drive the toe sole plate by the harmonic drive gear mounted on the heel sole plate.

As already mentioned, harmonic drive gears are mainly used because they are backlash free. On the other hand, for the mechanical parts, which do not require backlash-less motion, we can use alternative devices. For this reason, servo motors with planetary gear heads were used as the actuators for facial expressions and hands.

Figure 17.13 is the final mechanical design of HRP-4C (Fig. 17.13a) and its exterior (Fig. 17.13b). The covers for the arms, legs and body are made of fiber reinforced plastics (FRP) and the facial skin and hand skin are made of silicone rubber.

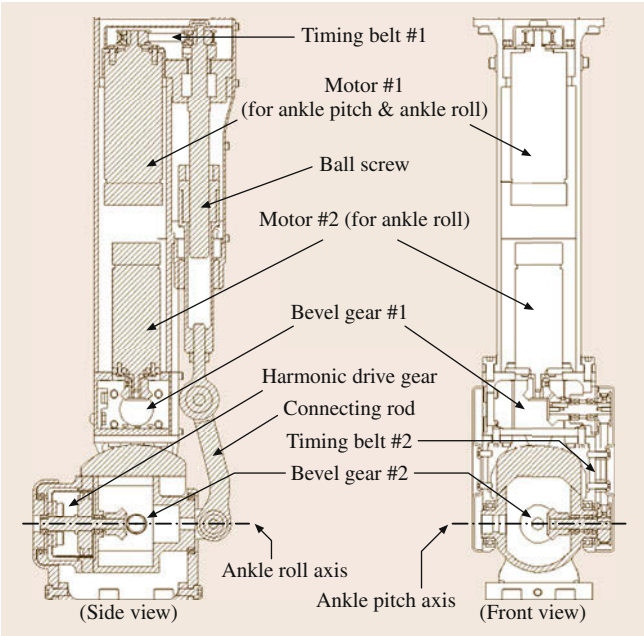


Fig. 17.11 Ankle joint drive mechanism of HRP-4C

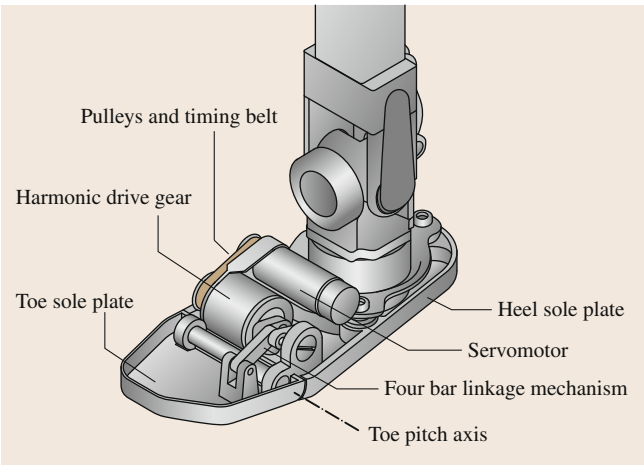


Fig. 17.12 Toe joint mechanism

17.3.3 Electrical System Design

Figure 17.14 shows the electronics architecture of HRP-4C. The entire system is controlled by a single board computer of the PCI-104 form factor with Intel Pentium M 1.6 GHz. Two peripheral boards, the 10 ch controller area network CAN interface board and the force sensor interface board are stacked on the CPU board.

To realize a human-like slender outlook, we used compact motor drivers which were developed in our previous project [17.18]. This device is an intelligent driver for a single servo motor, which takes an incremental encoder signal and provides motor current by proportional–integral–derivative (PID) control. The driver is small enough to be embedded in leg/arm links nearby the target motor. Since it has a CAN interface which allows cascade connection from the CAN interface board, the wiring can be simplified and reduced.

In addition, we used another CAN-interfaced intelligent drivers for small motors for the face and the hands. This driver can drive multiple motors by handling each incremental encoder, PID feedback, and the current control. We developed this type of

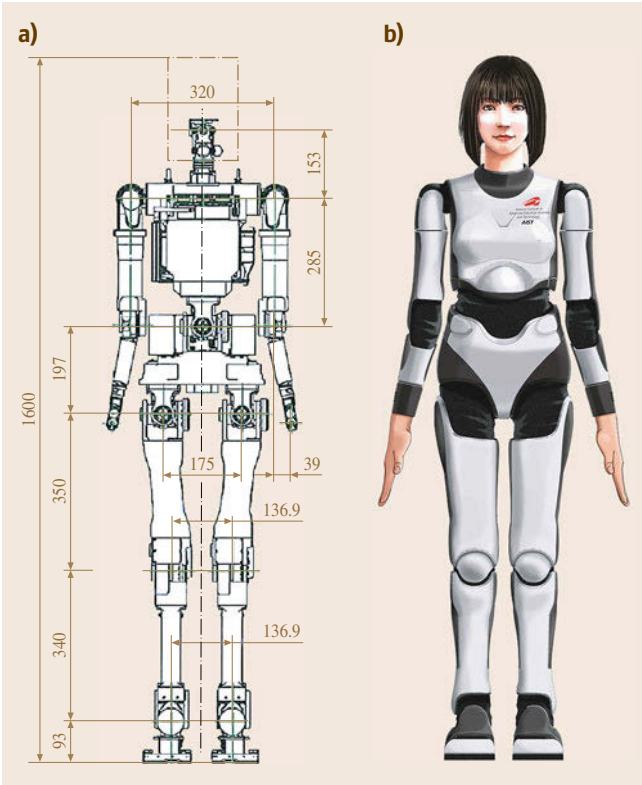


Fig.17.13a,b Mechanism of (a) HRP-4C and its (b) exterior

Table 17.1 Principal specifications of HRP-4C

Height		1,600 mm
Weight		48 kg (with batteries)
Total DOF		44 DOF
	Face	8 DOF
	Neck	3 DOF
	Arm	6 DOF × 2
	Hand	2 DOF × 2
	Waist	3 DOF
	Leg	7 DOF × 2
CPU		Intel Pentium M 1.6 GHz
Sensors	Head	CCD camera
	Body	Inertial measurement unit (IMU)
	Sole	6-axis force sensor × 2
Batteries		NiMH 48 V

driver by modifying the device for our multifingered hand [17.20].

By using these motor drivers and the CAN interface board, we can control 44 servo motors in total. Furthermore, a commercial inertial measurement unit (IMU) is connected via CAN to measure and control of the body posture during biped walk.

The feet of HRP-4C are equipped with six-axis force sensors to measure the zero-moment point (ZMP) for its balance control. These sensor signals are interfaced by the force sensor interface board.

Table 17.1 shows the principal dimensions and specification of the final design of HRP-4C. It becomes

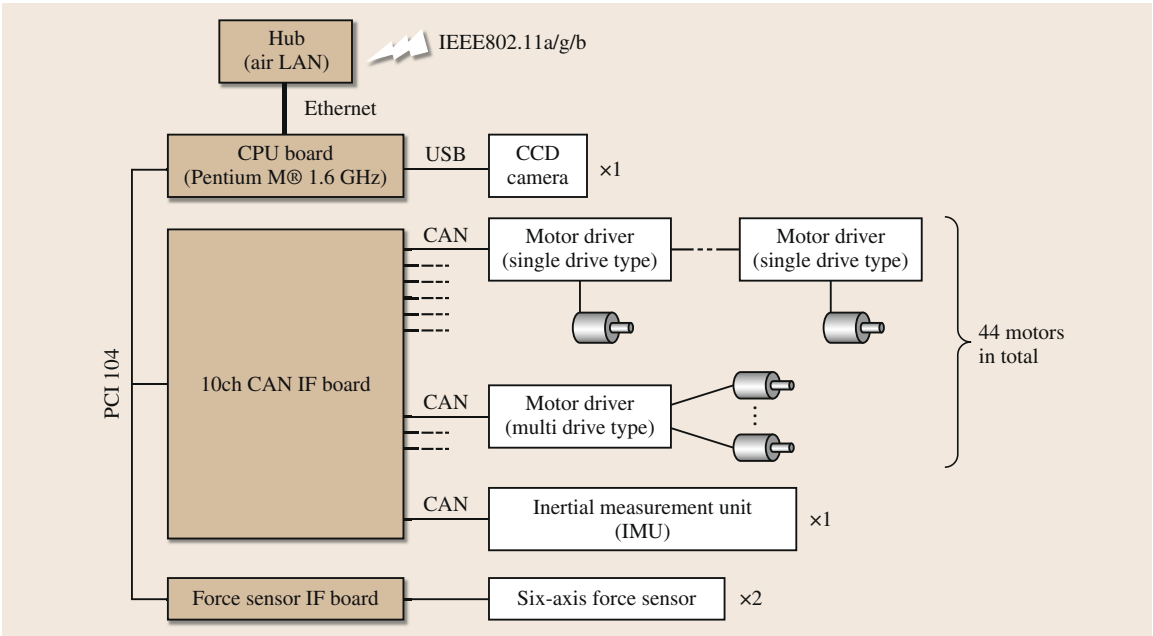


Fig. 17.14 Electronics architecture of HRP-4C

a humanoid robot with 1.6 m height, 46 kg weight, and 44 actuated DOF. The on-board nickel metal hydride battery (NiMH) provides power for 20 min operation time without an external cable. HRP-4C's human-like walking demonstration and a quick slip-turn experiment are shown in VIDEO 522 and VIDEO 525, respectively.

17.4 Model Induced Design

The design of multilimbed robotic systems is a mechatronic challenge involving expertise in diverse fields such as mechanical design, electrical design, communication and control, as well as software engineering. A task-oriented perspective on the overall system allows in some cases to incorporate a desired function directly into the system design rather than enforcing this behavior by active control. This idea can be found in many robotic designs. The most prominent example might be the use of counterweights for passive gravity compensation employed in robot manipulators. A clever mechanical design thus leads to a smaller power consumption of the actuators and consequently leads to a simplified system design. Passive gravity balancing has been realized via several approaches involving counterweights, springs (Fig. 17.15), and parallel mechanisms [17.21]. Such a gravity compensation based on springs is for example employed in the robot PR2 from Willow Garage (Fig. 17.16) [17.22, 23]. In this section, we will highlight several examples of legged robots where a desired function or property of the final system is realized at least partly via clever mechanical design.

17.4.1 Dynamic Walking

Passive Dynamic Walking

The concept of passive dynamic walking was pioneered by McGeer (VIDEO 527). In the seminal paper [17.24], a class of purely passive mechanical systems on an inclined surface were analyzed which exhibit a periodic gait that is intrinsically, i. e., open loop, stable. The resulting periodic motion is characterized by the fact that the increase of energy due to the slope is exactly balanced by the energy loss at the impact. McGeer proposed to analyze these systems in terms of orbital stability using Poincaré maps.

A detailed analysis of simple walking models with point feet, which are based on this principle, can be found for instance in [17.25] or [17.26]. Figure 17.17 shows a conceptual model of a passive walking machine which is able to walk down on an inclined slope. For analyzing the dynamics of this system, the following two components are needed: 1) The open-loop dynamics of the swing phase: Under the assumption that the stance leg is standing firmly on the ground (without lift-off or slipping) until the next touch-down, the open loop dynamics can be derived from standard Lagrangian multibody dynamics. 2) An impact model

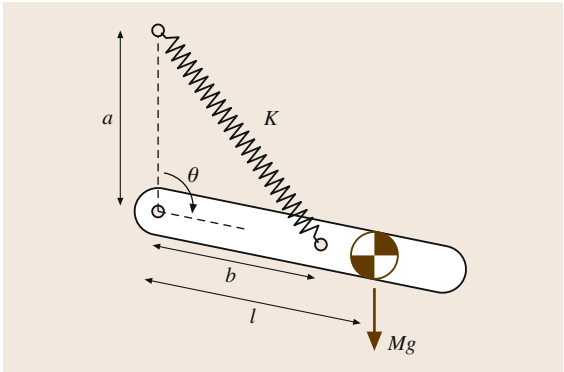


Fig. 17.15 Passive gravity compensation by a spring: For $Kab = Mgl$ the torques around the hinge produced by gravity and by the spring counter-balance each other

Fig. 17.16 PR2 robot from Willow Garage using springs for passive gravity compensation (after [17.22, 23])

at the time of touch-down of the swing leg, which determines the state variables after the impact based on the state variables before the impact. Often, a perfectly inelastic impact is assumed.

In a fully passive mechanism with given dynamics parameters, the motion is completely determined by the initial condition of the system. The mapping from the initial state to the state at the beginning of the next step is called *stride function* or *return map*. A periodic gait can be characterized by a fixed point of the stride function, and the search for a periodic gait can accordingly be formulated as an optimization problem on the stride function. In general, optimization may involve the initial condition as well as the dynamics parameters including the inclination of the slope.

In order to analyze the stability properties of a periodic gait, McGeer proposed to interpret the stride function as a Poincaré map of the system. Local (orbital) stability can then be decided based on the linearization of the Poincaré map. Since computing the Poincaré map (i. e., computing the stride function) involves solving the open loop dynamics of the system until the next touch-down, this often can only be implemented numerically.

Figure 17.18 shows technical realizations of planar passive dynamic walkers. The walker shown in Fig. 17.18a from Technical University of Delft has the simplest possible configuration with straight legs and point-feet. Premature impact of the swing leg with the ground is avoided by a stepping pattern on the inclined slope. The mechanism in Fig. 17.18b was built at Cornell University based on ideas from McGeer's seminal paper. The arc shape of the foot implements a rolling motion during stance. The ground clearance of the swing leg was realized by a passive knee joint.

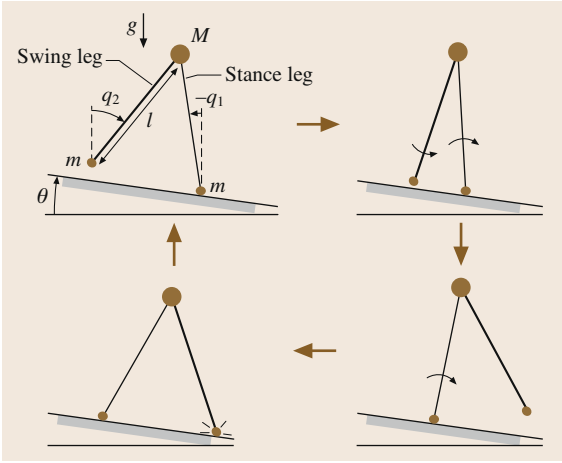


Fig. 17.17 Conceptual model of a simple passive dynamic walker

Figure 17.19 shows a fully passive walking machine in 3-D [17.27] (VIDEO 532). It employs soft heels for improving the contact transition. The swinging motion of the arms allows for reducing the angular momentum around the vertical axis as well as side-to-side rocking. Knee extension is limited to a straight configuration by a locking mechanism.

Semipassive Limit Cycle Walker

Purely passive dynamic walkers can produce a stable limit cycle when walking down a shallow slope. Thereby, the energy loss at each impact is compensated

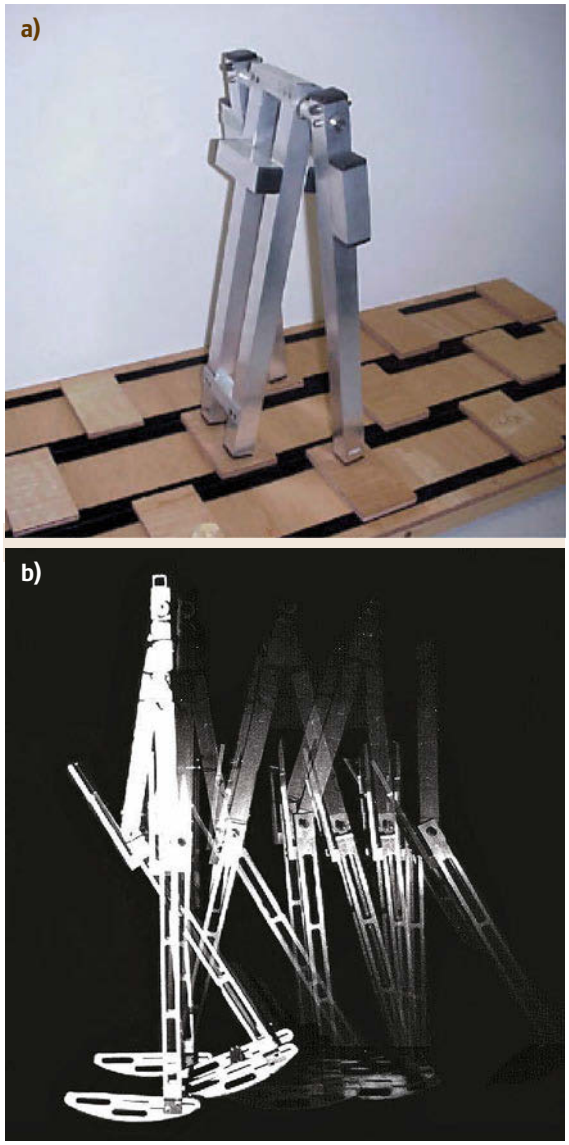


Fig.17.18a,b Examples of planar passive dynamic walkers with **(a)** point feet, **(b)** arc feet, and knees

by the energy gain from the decent along the slope, i. e., these systems are *powered* by gravity.

Several attempts have been made for increasing the versatility and robustness of limit cycle walkers by adding active power sources. Figure 17.20 shows some successful designs reported in [17.28]. In these systems, simple actuation is used for substituting gravitational power in order to allow for walking on level ground. While these systems are not fully passive, their energy consumption is much less than compared to fully actuated walking robots. The Cornell biped, as shown in Fig. 17.20a, utilizes actuation only in the ankle. The actuation is used such that each ankle joint extends when the opposite foot hits the ground. The Delft biped, as shown in Fig. 17.20b, instead is powered at the hip and utilizes passive ankles.

The superior energy efficiency which can be achieved by dynamic limit cycle walkers has been demonstrated in the Cornell Ranger [17.29] that succeeded to walk over a distance of 65 km without recharging or direct human intervention except for the remotely controlled steering.

Fig. 17.19 A 3-D passive walker with upper body (after [17.27])

Actuated Limit Cycle Walkers

Passive and semipassive limit cycle walkers strongly rely on the passive dynamics for realizing a specific gait. In actuated limit cycle walkers the control of some joints is used for imposing a limit cycle on the remaining dynamics. The systems shown in Fig. 17.21 are fully actuated, except for some unactuated degrees of freedom, e.g., at the foot–ground contact, and/or underactuation due to elasticity.

In the framework of the *hybrid zero dynamics* [17.32], a cyclic variable like the virtual leg angle (angle from the hip to the stance foot) is introduced and the active degrees of freedom are controlled via partial feedback linearization to implement a virtual constraint which describes how the active degrees of freedom are coupled to the cyclic variable. The shape of the virtual constraint is then optimized aiming at a stable limit cycle for the remaining two-dimensional (2-D) zero dynamics. A comprehensive treatment of this approach can be found in [17.33]. Such control approaches were implemented on the robots RABBIT [17.34] and MABLE [17.35]. While RABBIT employed position control in the active degrees of freedom, MABLE was implemented with series elastic actuators allowing for torque-based control (VIDEO 533).

The design of the 3-D walking machine Flame in Fig. 17.21c was also motivated by the aim of combining limit cycle walking with active control [17.36]. The actuation of this robot was also based on series elastic

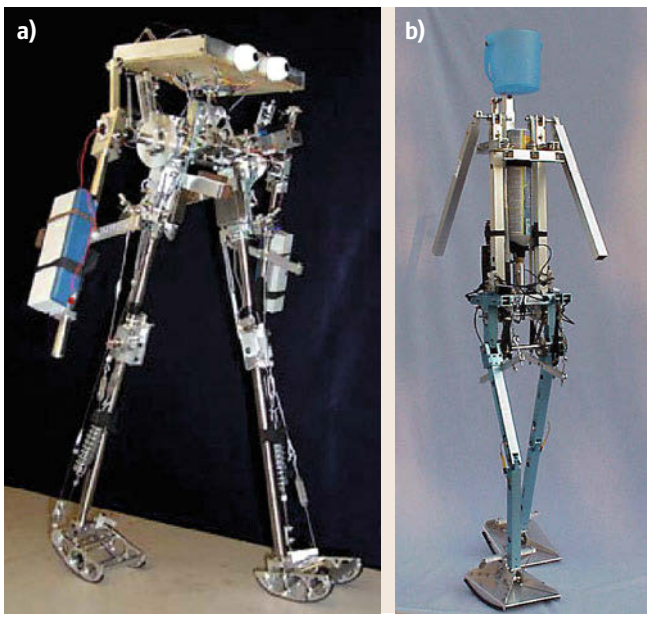


Fig.17.20a,b Underactuated biped robots: **(a)** 3-D passive walker (Cornell Biped, 2003) developed at Cornell Univ. (after [17.30]); **(b)** Denise (2004) (after [17.31])

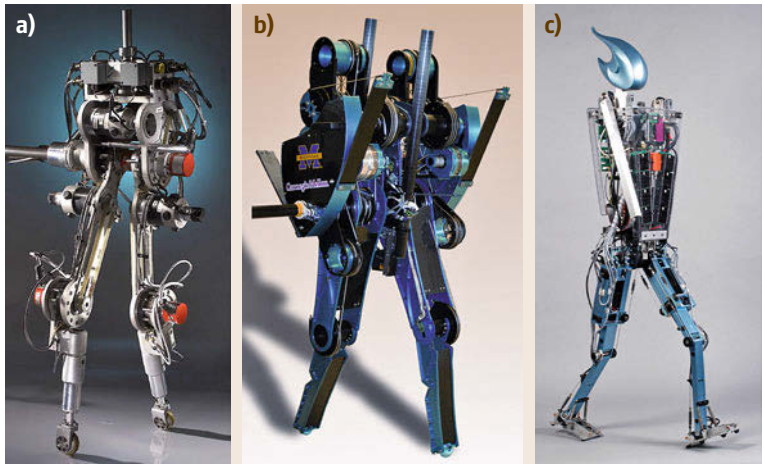


Fig.17.21a–c Actuated limit cycle walkers **(a)** RABBIT (2002); **(b)** MABEL (2008); **(c)** Flame (2007)

actuation and it was controlled by an event-based state machine in combination with active foot placement for gait stabilization.

17.4.2 Template Models

Conceptual models play an important role as templates for the design and control of legged walking machines. Template models can also be used for generating hypotheses about the neuro-mechanical control in biological systems [17.37]. In this context, they provide a possible answer to Bernstein's DOF problem, how the redundancy lying in the human neuro-muscular system is coordinated. We will discuss two conceptual models which have proven useful for legged locomotion.

The linear inverted pendulum (LIP) model has been proposed as a template model for bipedal walking (VIDEO 512). Figure 17.22a shows the main forces acting on the center of mass (COM) of a legged robot. If all the mass is concentrated in one single point, which implies mass-less legs, the acceleration forces of the COM are counter balanced by the ground interaction force F acting at the ZMP p. If the COM is kept at a constant height, i. e., $\ddot{z} = 0$, one immediately obtains the dynamical equations of the linear inverted pendulum model

$$\ddot{x} = \frac{g}{z}(x - p) \,. \tag{17.1}$$

This model has been the starting point for many successful methods aiming at trajectory generation and feedback control of bipedal humanoid robots [17.41–43]. More details on the related control issues can be found in 48. By using this model, the implicit assumption is made that the change of angular momentum around the COM, e.g., due to the swing leg dynamics,

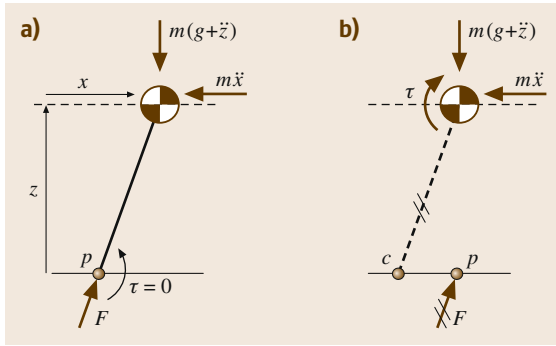


Fig. 17.22 **(a)** The linear inverted pendulum model (after [17.38]) **(b)** Definition of the centroidal moment pivot (after [17.39, 40])

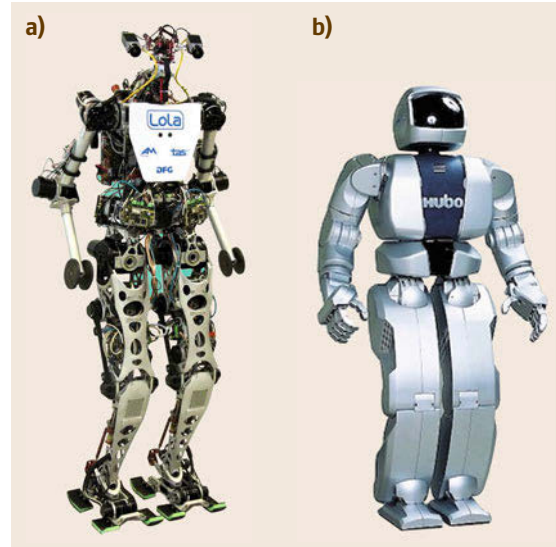


Fig.17.23a,b Bipedal robots **(a)** LOLA (2009); **(b)** HUBO (2005)

can be neglected. This assumption plays an important role also for the mechanical design. The bipedal robot LOLA (Fig. 17.23a) developed at Technical University of Munich (TUM) is optimized for low inertia in the legs by moving the knee and ankle actuators into thigh [17.44]. In the robot HUBO from KAIST (Fig. 17.23b) the design aimed at concentrating the main inertia into the torso while keeping the mass of the moving parts in the arms and legs small [17.45].

Aiming at a better representation of the change of angular momentum the centroidal moment pivot (CMP) (Fig. 17.22b) was proposed as an additional ground reference point [17.39, 40]. For the more general model in Fig. 17.22b the force balance results in

$$\ddot{x} = \frac{g+\ddot{z}}{z}(x-p) + \frac{\tau}{mz} \,. \tag{17.2}$$

The CMP is defined as the hypothetical location of the ZMP for the case $\tau = 0$, such that the distance between the ZMP and the CMP is proportional to the change of angular momentum, i. e.,

$$c = p - \frac{\tau}{F_z} \,, \tag{17.3}$$

where $F_z = m(g+\ddot{z})$. In undisturbed human walking it was observed that the distance between ZMP and CMP is kept small [17.40], while the change of angular momentum plays a stronger role for push recovery [17.46].

An alternative way of modeling the effect of the limb motion (e.g., swing foot dynamics) on the overall dynamics is to consider a three-mass model, which has been utilized in the control of LOLA [17.44] and the Honda humanoid robot ASIMO [17.47].

While the LIP model has been introduced as a conceptual model for walking, it cannot describe running motions in which a flight phase is present. In biomechanics, the spring-loaded inverted pendulum (SLIP) model (Fig. 17.24) was introduced in which the leg is modeled as a linear spring [17.48].

During the stance phase, the dynamics is given by

$$m\ddot{\boldsymbol{r}} = m\boldsymbol{g} + k\left(\frac{l_0}{|\boldsymbol{r}-\boldsymbol{r}_\mathrm{F}|} - 1\right)(\boldsymbol{r}-\boldsymbol{r}_\mathrm{F}) \,, \tag{17.4}$$

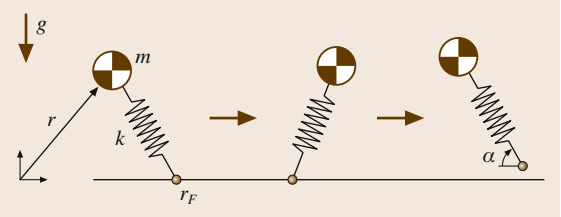


Fig. 17.24 The spring loaded inverted pendulum (SLIP) model

where l_0 and k are the rest length and stiffness of the spring, respectively, and $\boldsymbol{r}_\mathrm{F}$ is the location of the stance foot. During the flight phase, one simply has $\ddot{\boldsymbol{r}} = \boldsymbol{g}$, and it is assumed that the leg angle instantaneously adjusts to an angle α for preparing the next touch-down. The model contains no damping and impact phenomena are eliminated by assuming mass-less legs. Therefore,

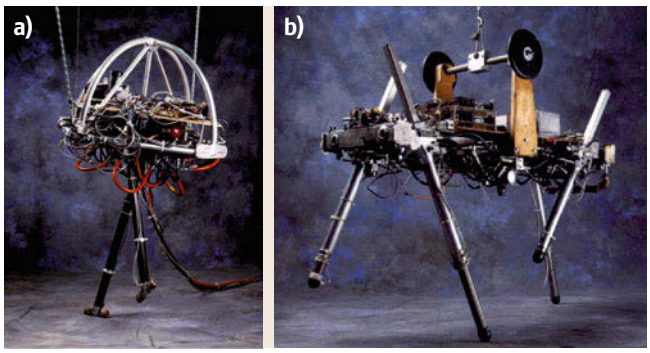


Fig.17.25a,b Hopping robots developed in the MIT LegLab **(a)** 3-D Biped (1989); **(b)** Quadruped (1984) (after [17.1])

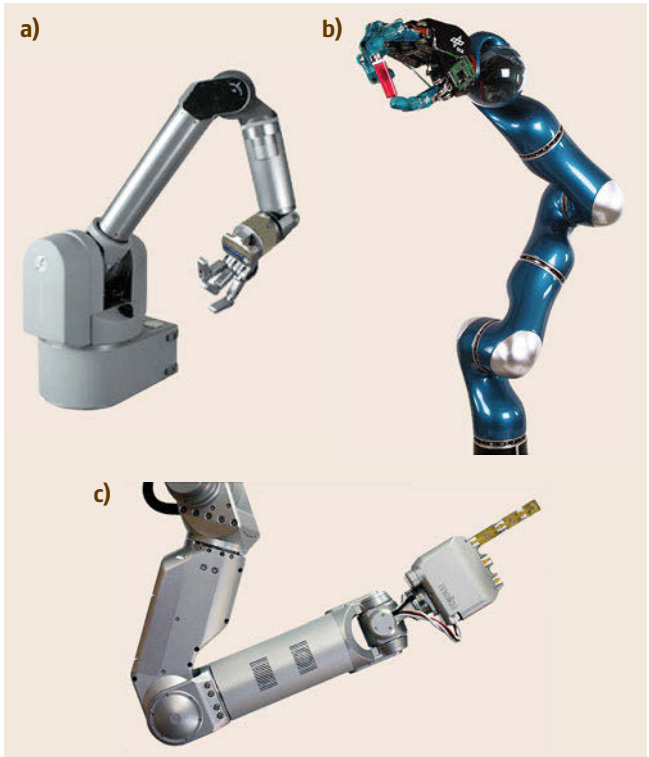


Fig.17.26a–c Robotic arms for implementing torque control: **(a)** WAM arm (Barrett Technologies) utilizing direct drive actuation in combination with cable transmissions, **(b)** the DLR lightweight-robot-III with integrated joint torque sensors, **(c)** the robot arm A2 (Mekka Robotics) based on series elastic actuators

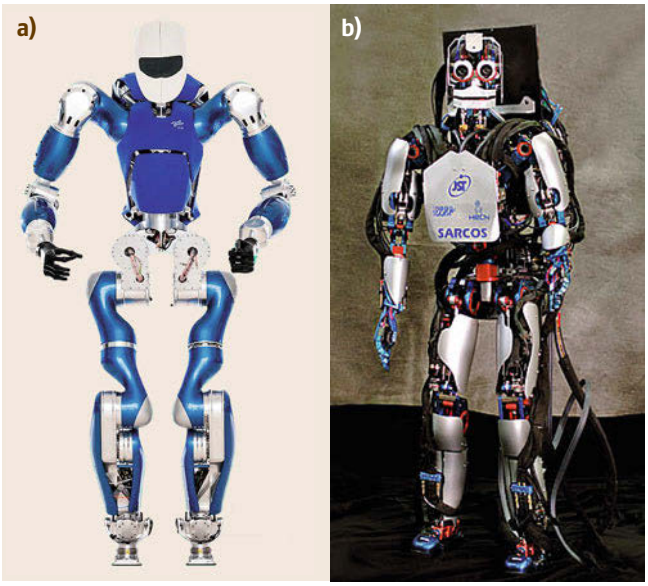


Fig.17.27a,b Fully torque controlled humanoid robots: **(a)** DLRs robot TORO (2013); **(b)** Humanoid robot CB based on hydraulic actuation (CB-i, 2008)

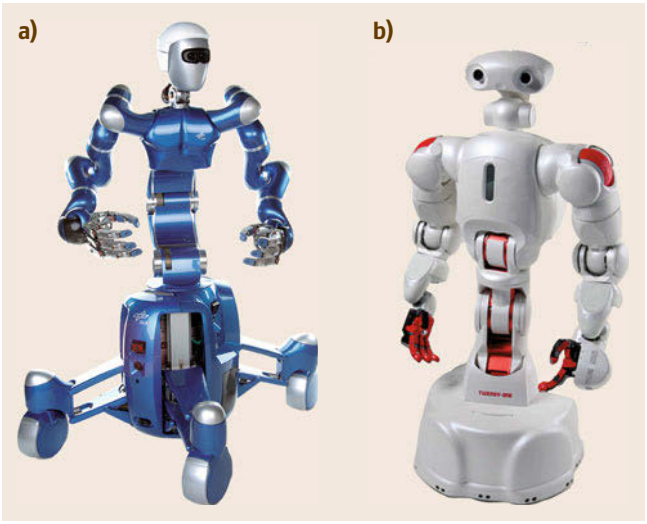


Fig.17.28a,b Humanoid upper body robots which allow whole body control based on joint torque: **(a)** DLRs torque controlled robot Justin and **(b)** Twendy-one from Waseda Univ. using SEA

this model is conservative. It allows to generate periodic hopping trajectories representing running motions. Moreover, the contact force profile resulting from these motions shows similarity to data from human running motions. An extension of the SLIP dynamics to bipedal walking including double support phases was analyzed in [17.49].

The SLIP model can be considered as the simplest model of one-legged hopping and bipedal running. Its compliant leg function can be of interest for the design and control of future legged robots, which aim at highly dynamic motions like fast running, hopping, and jumping.

Gait generation and control of compliant hopping robots have been studied in the 1980s by Raibert in the MIT LegLab. A detailed treatment of these developments is summarized in [17.1]. The basic idea of Raibert's dynamic locomotion approach can be explained by considering a planar one-legged hopping robot. It can be shown that this system can be controlled in a surprisingly simple way by the combination of three separate components:

1. Altitude control which is implemented by a fixed thrust during stance.
2. Control of the forward velocity by placing the foot at a fixed distance from the hip at touch-down.
3. Control of the body attitude during stance using actuation at the hip.

The same principle was applied to a one-legged hopper in 3-D, as well as to bipedal and quadrupedal hopping robots (Fig. 17.25). In the quadrupedal case, the combined effect of alternating pairs of legs was summarized by considering a single virtual leg.

17.4.3 Robots for Control Based on Backdrivable Actuation

A large number of algorithms for model-based control of robotic systems assume a rigid-body model of the robot in which the joint torques act as the control input. The capability of controlling the torque in practice is heavily influenced by the properties of the actuator transmission. While direct drive actuation allows a precise feed-forward control of the actuator torque via the motor current, this leads to rather heavy drive units. The WAM arm from Barrett Technologies (Fig. 17.26a) utilizes direct drive actuation in combination with cable transmissions in order to store the leading four drives into the base of the robot. In this way, a large inertia of the moving elements is avoided. The use of gears with high transmission ratios allows to reduce the overall weight, but leads to slower drive units which are hardly or not at all backdrivable. Joint torque sensing and serial elastic actuation present two hardware-oriented approaches for improving the torque controllability of highly geared actuators.

Joint torque sensing assumes a dedicated sensor for measuring the link side torque τ and was applied in a series of lightweight robot arms de-

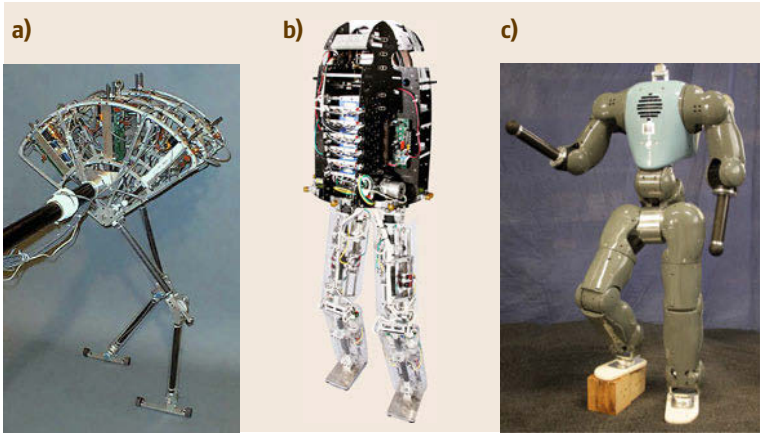


Fig.17.29a–c Bipedal robots using serial elastic actuators: **(a)** Spring Flamingo (1996) (after [17.50]); **(b)** M2V2 (2008) (after [17.51]); **(c)** COMAN (2011) developed at IIT utilizing SEA in selected joints (hip, knee, ankle)

veloped at DLR, including the light-weight robot in Fig. 17.26b [17.52] and the upper body robot Justin as shown in Fig. 17.28a [17.53]. Recently, the same drive technology has been integrated in the design of the torque-controlled humanoid robot TORO (Fig. 17.27a) [17.54] (VIDEO 531). Torque sensing and control have also been employed in the hydraulic humanoid CB developed by Sarcos (Fig. 17.27b).

In series elastic actuators (SEAs), the joint torque sensing is implemented indirectly by measuring the deflection of an elastic element which is introduced into the drive train [17.55]. Early robots with compliant actuators were the two armed systems COG [17.56] from MIT and Wendy [17.57] from Waseda university, which even allowed the adaptation of the compliance. More recently, series elastic actuators are used in the compliant arm A2 from Mekka (Fig. 17.26c) and the humanoid upper body robot Twendy-one from Waseda university (Fig. 17.28). They have been also applied to bipedal systems in the robots as shown in Fig. 17.29 (also watch VIDEO 529 and VIDEO 530).

While the mathematical models of torque controlled robots and robots with SEA have the same structure it should be mentioned that the source of elasticity is different in these approaches. In SEA, the deflective element for the torque sensing represents the main source of elasticity and its role is to decouple the motor and link side dynamics for high frequency disturbances like impacts. A well-designed torque sensor instead should not introduce a big effect on the overall joint stiffness (in this case usually the gear introduces the main source of elasticity). Despite their conceptual similarity, the literature of SEA and elastic joint robots also differs in the way that in the context of SEA usually the linear dynamics of a single compliant actuator is considered for the controller design, while in the context of elastic joint robots the full nonlinear multibody dynamics is considered.

From a modeling and control point of view, the use of torque sensors in the joints leads to a robot model with elastic joints. For actuators with a high transmission ratio, a common modeling assumption is that the kinetic energy of each rotor depends only on its own spinning motion and not on the rigid body motion of the other joints [17.58]. Let the motor and link side joint angles be denoted by $\theta \in \mathbb{R}^n$ and $q \in \mathbb{R}^n$, then the *reduced flexible joint model* (Chap 11 for a detailed derivation) is given by

$$\mathbf{M}(q)\ddot{q} + \mathbf{C}(q,\dot{q})\dot{q} + \boldsymbol{g}(q) = \boldsymbol{\tau} + \tau_{\mathrm{f},q}\,, \tag{17.5}$$

$$\mathbf{B}\ddot{\theta} + \boldsymbol{\tau} = \tau_{\mathrm{m}} + \tau_{\mathrm{f},\theta}\,, \tag{17.6}$$

where $\mathbf{M}(q)$, $\mathbf{C}(q,\dot{q})\dot{q}$, and $\boldsymbol{g}(q)$ represent the link side inertia matrix, the centrifugal and Coriolis forces, and the gravity term of the rigid body dynamics. The motor inertia is given by the diagonal matrix $\mathbf{B}$. The vector $\boldsymbol{\tau}$ represents the joint torques. In the case of an elastic joint with linear joint stiffness k_i the torque is given by $\tau_i = k_i(\theta_i - q_i)$. The terms $\tau_{\mathrm{f},q}$ and $\tau_{\mathrm{f},\theta}$ represent friction terms at the link side and the motor side, respectively.

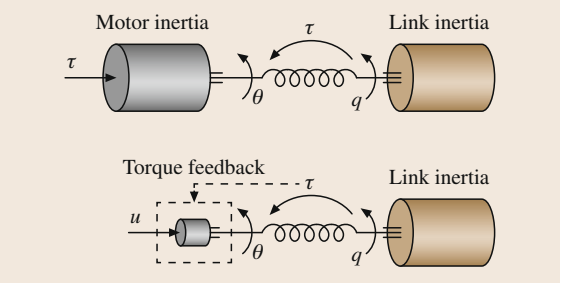


Fig. 17.30 Conceptual model of an elastic joint: A proportional feedback of the joint torque reduces the effective motor inertia and motor friction

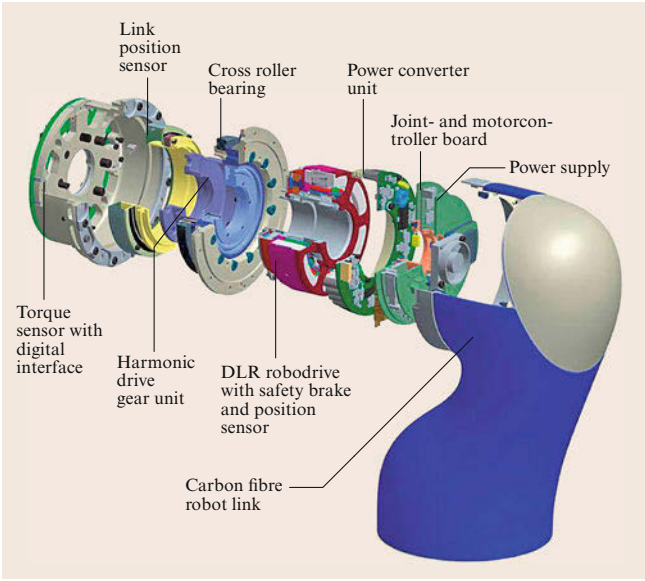


Fig. 17.31 The torque sensing based joint technology of DLR's light weight robot arm

A purely proportional feedback of the joint torques

$$\tau_{\mathrm{m}} = -(\mathbf{K}_\tau - I)\tau + \mathbf{K}_\tau u \tag{17.7}$$

has the effect that the motor dynamics equation (17.6) shows a controlled behavior as

$$\mathbf{K}_\tau^{-1}\mathbf{B}\ddot{\theta} + \tau = u + \mathbf{K}_\tau^{-1}\tau_{\mathrm{f},\theta} \,. \tag{17.8}$$

Let us assume that $\mathbf{K}_\tau$ is chosen as a diagonal matrix. Then we see that the matrix $\mathbf{K}_\tau^{-1}\mathbf{B}$ corresponds to an effective (i. e., virtual) motor inertia matrix. By choosing the gains in $\mathbf{K}_\tau$ larger than unity, we can thus reduce the effective motor inertia (Fig. 17.30). Moreover also the effects of motor friction are reduced such that a torque controlled joint ultimately exhibits a backdrivable behavior even if the underlying gear mechanism is not or only hardly backdrivable.

From this interpretation of torque feedback, we can derive some general guidelines for the mechanical design of torque-controlled robots. Since the effective motor side friction can be reduced by a simple proportional torque feedback but link side friction is not affected, we conclude that the mechanical design should aim at low friction at the link side and the torque sensing should be implemented at the power output side of the gear. Since the assembly of gear and torque sensor is assumed to behave like an ideal spring, any masses of moving parts in the transmission should be kept low in order to minimize unmodeled dynamics which would affect the validity of (17.7). As an example of a successful design, Fig. 17.31 shows a cross-section of a joint of the torque-controlled DLR-light-weight-robot-III [17.52] in which a harmonic drive transmission is combined with a strain gauge-based torque sensor.

Instead of the simple proportional control of (17.7), a complete torque controller can be designed based on the torque dynamics

$$\mathbf{B}\mathbf{K}^{-1}\ddot{\tau} + \tau = \tau_{\mathrm{m}} + \tau_{\mathrm{f},\theta} + \mathbf{B}\ddot{q} \,, \tag{17.9}$$

resulting from (17.6) and $\tau = K(\theta - q)$. Torque tracking is achieved via

$$\tau_{\mathrm{m}} = \tau_{\mathrm{d}} + \mathbf{B}\mathbf{K}^{-1}\ddot{\tau}_{\mathrm{d}} + \tau_{\mathrm{f},\theta} + K_q\mathbf{B}\ddot{q} + u(\tau_{\mathrm{d}} - \tau) \,, \tag{17.10}$$

where $u(\tau_{\mathrm{d}} - \tau)$ represents a proportional–derivative (PD) or PID control law and the nonnegative factor $K_q < 1$ is chosen less than 1 in order to avoid an overcompensation of the link side acceleration. In the context of SEA such a controller was proposed based on a linear system representation of a single joint in [17.55]. Cascaded torque control of a single SEA-based joint using an inner loop velocity controller has been analyzed in [17.59, 60], also using a linear model. In the context of torque-controlled elastic joint robots, in [17.61] a combination with a Cartesian impedance controller was proven to be stable for the full nonlinear dynamics based on the stability theory of cascaded systems (using $K_q = 1$). Passivity-based control has been analyzed in [17.62].

17.5 Various Limbed Systems

In this section, we explore the divergence of the limbed system design.

17.5.1 Odd Legged Walkers

A robot can be designed by pure engineering. For example, the tripodal walking robots STriDER (self-excited tripodal dynamic experimental robot) 1 and 2 (Fig. 17.32) can demonstrate a unique locomotion which have never developed by evolution [17.63] and VIDEO 534. It has three identical legs each of them has four DOF, three at the hip and one at the knee. Supporting the body with two legs, the robot takes a dynamic step by swinging the rest leg. Since the body flips

Fig. 17.32 STriDER 1 and 2

with 180 ∘ about the aligned pelvis joints connecting the support legs, the robot can continue its dynamic walk.

Five legged walkers have been developed by some groups. For example, *Besari* et al. developed a five-legged robot inspired from starfish, and applied reinforcement learning to obtain its optimal gait [17.64].

17.5.2 Leg–Wheel Hybrid Robots

Unlike animals or insects, a robot can be designed to have wheels which can rotate infinitely. By mixing the efficiency of wheels and the flexibility of legs, we can expect a robot of maximum terrain adaptivity with minimum power consumption. Figure 17.33 shows examples of such a design concept.

Figure 17.33a shows a stair climbing of a biped leg-wheeled robot developed by *Matsumoto* et al. [17.65]. The robot is a planer biped with telescopic legs, but the tip of each leg is equipped with a powered wheel. During a single leg support phase, the robot is controlled as a wheeled inverted pendulum. In addition, a controller was developed to realize the smooth transient between single support and statically stable double support phase.

Figure 17.33b is RollerWalker developed by *Hirose* et al. [17.66, 69]. RollerWalker is a 12 DOF quadruped robot equipped with passive wheels on the tip of legs. It uses roller-skating mode on a flat floor, while it can walk on an uneven terrain by retracting the passive wheels (VIDEO 535).

Figure 17.33c is RHex developed by Buehler, *Koditchek* et al. [17.67]. Although it was originally inspired from the locomotion of cockroach, RHex has only six active DOF, that is, one actuator for each hip. Moreover, the legs can rotate full circle around the pitch axis. By this unique design, RHex can walk and run over rugged, broken, and obstacle-ridden ground (VIDEO 536). Recently, it also demonstrated biped running with its rear legs [17.70].

Figure 17.33d is Whegs II, another cockroach-inspired robot developed by *Allen* et al. [17.68]. This robot has only 4 active DOFs, one for propulsion, two for staring and one for body flexion. Each leg is equipped with three spring-loaded spoke and driven by

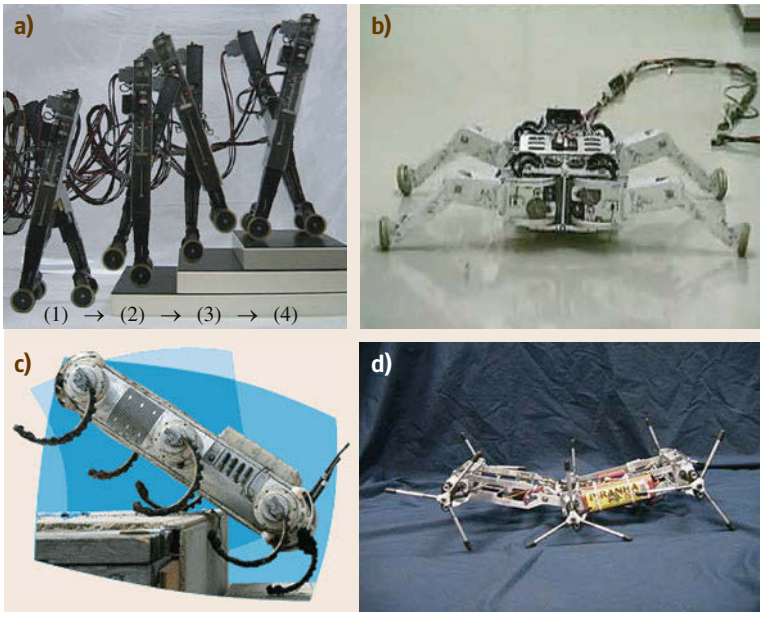
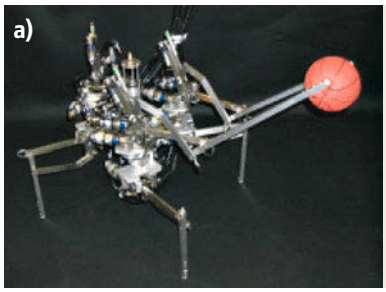


Fig.17.33a–d Leg-wheel hybrid robots. **(a)** Biped leg–wheel robot (1998); **(b)** Roller walker (1996); **(c)** RHex (2001); **(d)** Whegs II (2003) (after [17.65–68])



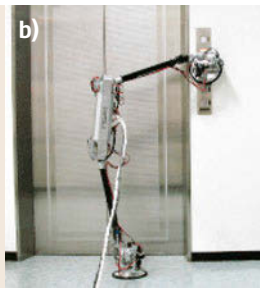


Fig.17.34a,b Leg-arm hybrid robots **(a)** MELMANTIS-1 (1996); **(b)** Yanbo3 (2003) (after [17.72, 73]) ◄

the same actuator (VIDEO 537). Whegs II can realize comparable mobility to RHex, while it uses fewer actuators.

17.5.3 Leg–Arm Hybrid Robots

Another design concept is a leg–arm hybrid robot. Since legs have inherently many degree of freedom, it is possible to use them as manipulators. By this way, we can minimize total DOF, complexity, weight and power consumption of the walking robot. MELMANTIS-1 (Fig. 17.34a) developed by *Koyachi* et al. is a hexapod walker of 22 DOF, which can transform its legs into manipulators [17.72]. The robot can manipulate an object by two legs while standing with four other legs, after traveling by six legs with the maximum stability.

Yanbo3 is a biped walker of 8 DOF developed by the group of *Ota* et al. [17.73]. It is designed to have minimum DOF necessary for a biped robot as well as for a manipulator when it is in the single support. In Fig. 17.34b, the robot is pressing the elevator button by its foot.

17.5.4 Tethered Walking Robots

Figure 17.35 shows Dante II, an eight-legged tethered walking robot developed by CMU Field Robotics Center in 1994. It was used at an Alaskan volcano for scientific exploration. To descend down steep crater walls in a rappelling-like manner, the robot uses tether cable anchored at the crater rim [17.71]. *Hirose* et al. also developed tethered quadruped for construction work [17.74].

Fig. 17.35 Dante II (1994) (after [17.71])

17.5.5 Wall-Climbing Robots

Wall-climbing robots are characterized by their foot mechanisms and leg configurations. The vital part is the foot mechanism to generate pulling force and the use of vacuum suction cups, electromagnets (for steel wall), adhesive materials, or miniature spine array has been proposed.

Figure 17.36a shows a wall-climbing quadruped NINJA-1 developed by *Hirose* et al. [17.75] Each foot of NINJA-1 is equipped with a specially designed suction pad which can minimize its vacuum leakage. Another reliable wall-climbing robot with suction cups was developed by *Yano* et al. [17.76]

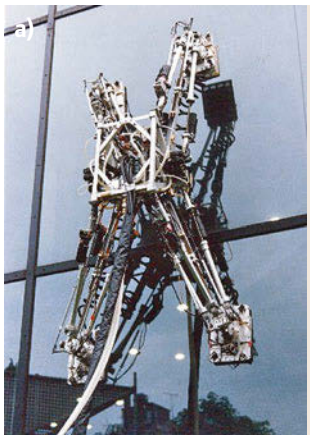


Fig.17.36a,b Wall-climbing robots. **(a)** NINJA-1 (1991); **(b)** RiSE (2005) (after [17.75, 77]) ◄

Figure 17.36b shows a wall-climbing hexapod RiSE developed by *Kim* et al. [17.77, 78]. Each foot of RiSE is equipped with arrays of miniature spines observed in some insects and spiders. The robot can reliably climb on a wide variety of outdoor surfaces including concrete, stucco, brick, and dressed sandstone. As a wall-climbing robot more recently developed, watch Stickybot III in VIDEO 540 and Waalbot in VIDEO 541.

17.6 Performance Indices

In this section, we introduce useful performance indices, that can be used to evaluate legged robots of different configurations.

17.6.1 Stability Margins

The stability margin was originally proposed for a degree of stability of the statically walking multilegged robots by *McGhee* and *Frank* [17.79]. By neglecting the inertial effects caused by body and leg acceleration, we can guarantee the robot keep balance if the projection of center of mass (COM) exists inside the support polygon as shown in Fig. 17.37. (Note: In multilegged robot research, the word *support pattern* is frequently used for support polygon [17.2, 79, 80].)

For a given configuration of a walking robot, the *stability margin* S_m is defined as the minimum distance of the vertical projection of COM to the boundaries of the support pattern in the horizontal plane as illustrated in Fig. 17.38a.

In addition, an alternative index was proposed to obtain the optimal gait analytically. That is the *longitudinal stability margin* S_l which is defined as the minimum distance from the vertical projection of COM to the support pattern boundaries along the line parallel to the body motion.

For a dynamic walking robot, we can define a stability margin as the minimum distance of the ZMP to the boundaries of the support polygon, since ZMP is the natural extension of a projected COM on the ground.

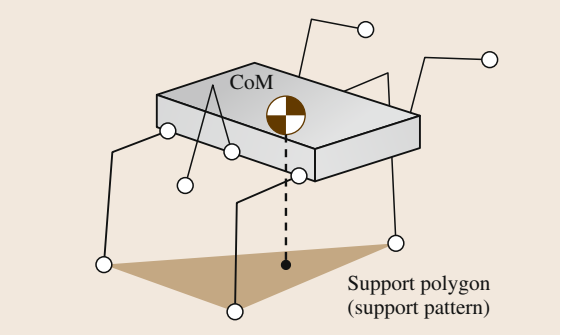


Fig. 17.37 Support polygon (support pattern) of multilegged robot

This fact has already been mentioned in the original work of ZMP [17.81], and has been implicitly used by many researchers. Explicit definition of this *ZMP stability margin* can be seen, for example, in *Huang* et al. [17.82].

For a legged robot on a rough terrain, *Messuri* and *Klein* defined the *energy stability margin* as the minimum potential energy required to tumble the robot as,

$$S_E = \min_i (Mgh_i) \,, \tag{17.11}$$

where h_i is the COM height variation during the tumble around the i-th segment of the support polygon and M is the total mass of the robot [17.83]. This concept is widely accepted, and there are some proposals for improvement [17.14, 84].

17.6.2 Gait Sensitivity Norm

Hobbelen and *Wisse* proposed a gait sensitivity norm (GSN) as a disturbance rejection measure of limit cycle walking machines [17.85]. GSN is defined as

$$\mathrm{GSN} = \left\| \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{e}} \right\|_2 \,, \tag{17.12}$$

where $\boldsymbol{e}$ is the set of disturbances and $\boldsymbol{g}$ is the gait indicator which characterizes the *failure mode*, motion finally end up a tumble. For a 2-D limit cycle walker, they took floor irregularities (step height to walk over) and step time as $\boldsymbol{e}$ and $\boldsymbol{g}$, respectively.

For real robots, GSN can be experimentally obtained from the response of gait indicators $\boldsymbol{g}$ to a single

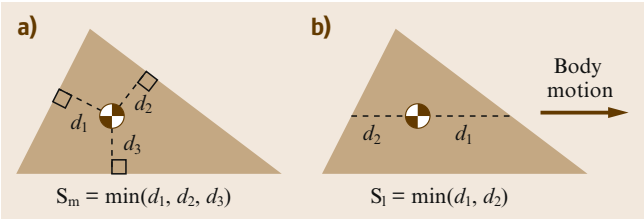


Fig.17.38a,b Definition of stability margins. **(a)** Stability margin; **(b)** Longitudinal stability margin

disturbance e_0 as

$$\left\| \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{e}} \right\|_2 = \frac{1}{|e_0|} \sqrt{\sum_{i=1}^{q} \sum_{k=0}^{\infty} [\boldsymbol{g}_k(i) - \boldsymbol{g}^*(i)]^2} \,, \qquad (17.13)$$

where $\boldsymbol{g}_k(i)$ is the i-th gait indicator value at k steps after the disturbance e_0 has applied, q is the number of gait indicators. GSN has a good correlation with the basin of attraction, hence it can be used for a design of a robust walking robot and its controller.

17.6.3 Duty Factor and Froude Number

Throughout this chapter, we have observed various walking robots which might fit best for certain environment and purpose. In some cases, however, we need to compare walking robots which have different masses, sizes, and leg numbers by using a certain performance index. Such an index should be dimensionless like a Mach number or a Raynords number in fluid mechanics.

One of the useful indices for walking machines is a duty factor β defined as

$$\beta = \frac{\text{(support period)}}{\text{(cycle time)}} \,.$$

Duty factors can be used to make the distinction between walks and runs, since we have $\beta \geq 0.5$ for walking and $\beta < 0.5$ for running [17.87].

Froude number is used in fluid mechanics to explain the behavior of surface waves. Since both of surface waves and legged locomotion are dynamic motion in gravity, *Alexander* used it to characterize animal locomotion [17.87, 88]. He calculated a Froude number by

$$Fr_2 = \frac{V^2}{gh} \,, \qquad (17.14)$$

where V is the walking or running speed, g is the gravity acceleration, and h is the height of hip joint from the ground. He showed animals of different sizes uses similar gaits when they travel with equal Froude numbers. Especially, most animals changes

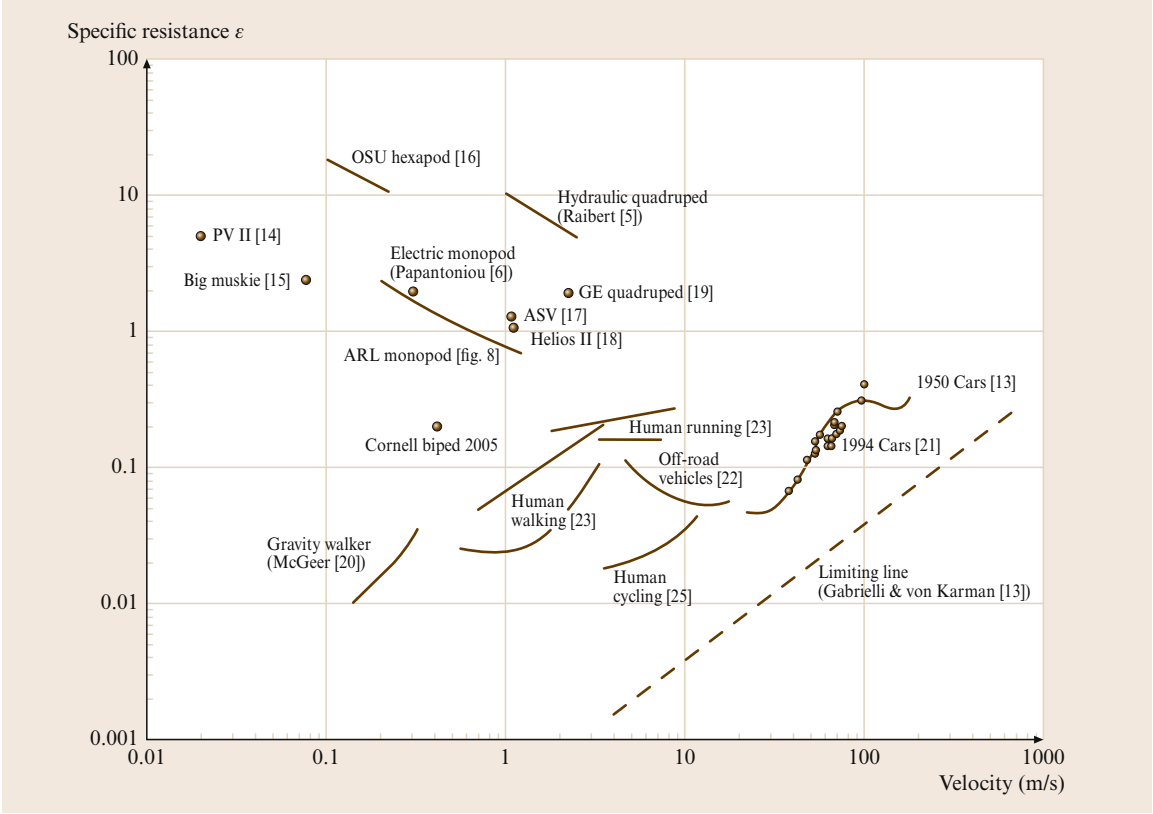


Fig. 17.39 Gabrielli–von Karman diagram (after [17.86])

their gait from walking to running at the speed of $Fr_2 = 1$.

Froude number is also defined as

$$Fr_1 = \frac{V}{\sqrt{gh}}\,, \tag{17.15}$$

which is the square root of Fr_2 and can be used non dimensional speed of animals or legged robots.

17.6.4 Specific Resistance

The *specific resistance* is another important dimensionless number, which is used to evaluate the energy efficiency of a mobile robot.

Gabrielli and von Karman discussed the performance of various vehicles using the power consumption per unit distance. That is

$$\epsilon = \frac{E}{Mgd}\,, \tag{17.16}$$

where E is the total energy consumption for a travel of distance d, M is the total mass of the vehicle and g is gravity acceleration [17.89]. Note that when we push a box of mass M for a distance d on a floor with a friction coefficient μ, we consume the power $Mg\mu d$ and the specific resistance becomes $\epsilon = \mu$. Therefore, we can say that the specific resistance indicates how smooth the locomotion is.

In the original work, Gabrielli and von Karman plotted the specific resistance as the function of speed for various vehicles (Fig. 17.39). This is called the *Gabrielli–von Karman diagram* and it was used to compare various styles of locomotion by *Umetani* and *Hirose* [17.90]. *Gregorio* et al. also showed the specific resistance of recent walking robots including their efficient hopping robot, ARL monopod [17.86]. *Collins* et al. developed robots based on passive-dynamics with small actuators and achieved energy efficiency close to human walking [17.91]. In the paper, the specific resistance is referred to as *specific energetic cost of transport*.

Video-References

VIDEO 512 Linear inverted pendulum mode
available from http://handbookofrobotics.org/view-chapter/17/videodetails/512

VIDEO 517 Hexapod robot *Ambler*
available from http://handbookofrobotics.org/view-chapter/17/videodetails/517

VIDEO 520 Hexapod ParaWalker-II
available from http://handbookofrobotics.org/view-chapter/17/videodetails/520

VIDEO 521 Cockroach-like hexapod
available from http://handbookofrobotics.org/view-chapter/17/videodetails/521

VIDEO 522 Bipedal humanoid robot: WABIAN
available from http://handbookofrobotics.org/view-chapter/17/videodetails/522

VIDEO 522 Cybernetic human HRP-4C walking
available from http://handbookofrobotics.org/view-chapter/17/videodetails/524

VIDEO 525 Cybernetic human HRP-4C quick turn
available from http://handbookofrobotics.org/view-chapter/17/videodetails/525

VIDEO 526 Development of a humanoid robot DARwIn
available from http://handbookofrobotics.org/view-chapter/17/videodetails/526

VIDEO 527 Passive dynamic walking with knees
available from http://handbookofrobotics.org/view-chapter/17/videodetails/527

VIDEO 529 Intuitive control of a planar bipedal walking robot
available from http://handbookofrobotics.org/view-chapter/17/videodetails/529

VIDEO 530 IHMC/Yobotics biped
available from http://handbookofrobotics.org/view-chapter/17/videodetails/530

VIDEO 531 Torque controlled humanoid robot TORO
available from http://handbookofrobotics.org/view-chapter/17/videodetails/531

VIDEO 532 3-D passive dynamic walking robot
available from http://handbookofrobotics.org/view-chapter/17/videodetails/532

VIDEO 533 Biped running robot MABEL
available from http://handbookofrobotics.org/view-chapter/17/videodetails/533

VIDEO 534 STriDER: Self-excited tripedal dynamic experimental robot
available from http://handbookofrobotics.org/view-chapter/17/videodetails/534

VIDEO 535 Roller-Walker: Leg-wheel hybrid vehicle
available from http://handbookofrobotics.org/view-chapter/17/videodetails/535

VIDEO 536 RHex rough-terrain robot
available from http://handbookofrobotics.org/view-chapter/17/videodetails/536
VIDEO 537 Whegs II: A mobile robot using abstracted biological principles
available from http://handbookofrobotics.org/view-chapter/17/videodetails/537
VIDEO 540 StickybotIII climbing robot
available from http://handbookofrobotics.org/view-chapter/17/videodetails/540
VIDEO 541 Waalbot: agile climbing with synthetic fibrillar dry adhesives
available from http://handbookofrobotics.org/view-chapter/17/videodetails/541

References

17.1 M.H. Raibert: *Legged Robots That Balance* (MIT Press, Cambridge 1986)
17.2 S.-M. Song, K.J. Waldron: *Machines That Walk: The Adaptive Suspension Vehicle* (MIT Press, Cambridge 1989)
17.3 S. Lohmeier: Design and Realization of a Humanoid Robot for Fast and Autonomous Bipedal Locomotion (Technische Universität München, München 2010)
17.4 R.D. Quinn, R.E. Ritzmann: Construction of a hexapod robot with cockroach kinematics benefits both robotics and biology, Connect. Sci. **10**(3), 239–254 (1998)
17.5 H. Hirukawa, F. Kanehiro, K. Kaneko, S. Kajita, M. Morisawa: Dinosaur robotics for entertainment applications, IEEE Robotics Autom. Mag. **14**(3), 43–51 (2007)
17.6 D.E. Koditschek, R.J. Full, M. Buehler: Mechanical aspects of legged locomotion control, Arthropod Struct. Dev. **33**, 251–272 (2004)
17.7 R. Tajima, K. Suga: Motion having a flight phase: Experiments involving a one-legged robot, Proc. Int. Conf. Intell. Robots Syst. (IROS), Beijing (2006) pp. 1727–1731
17.8 K. Kaneko, S. Kajita, F. Kanehiro, K. Yokoi, K. Fujiwara, H. Hirukawa, T. Kawasaki, M. Hirata, T. Isozumi: Design of advanced leg module for humanoid robotics project of METI, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 38–45
17.9 Y. Suhagara, H. Lim, T. Hosobata, Y. Mikuriya, H. Sunazuka, A. Takanishi: Realization of dynamic human-carrying walking by a biped locomotor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 3055–3060
17.10 S. Hirose, T. Masui, H. Kikuchi, Y. Fukuda, Y. Umetani: TITAN III: A quadruped walking vehicle – Its structure and basic characteristics, Proc. Int. Symp. Robotics Res., Kyoto (1984) pp. 325–331
17.11 M. Buehler, R. Playter, M. Raibert: Robots step outside, Proc. Int. Symp. Adapt. Motion Anim. Mach. (AMAM), Ilmenau (2005)
17.12 K.J. Waldron, R.B. McGhee: The adaptive suspension vehicle, IEEE Control Syst. Mag. **6**, 7–12 (1986)
17.13 R.A. Brooks: A robot that walks; Emergent behavior from a carefully evolved network, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Scottsdale (1989) pp. 292–296
17.14 M. Hirose, Y. Haikawa, T. Takenaka, K. Hirai: Development of humanoid robot ASIMO, Proc. Int. Conf. Intell. Robots Syst. (IROS) – Workshop 2 (2001)
17.15 K. Kaneko, F. Kanehiro, M. Morisawa, K. Miura, S. Nakaoka, S. Kajita: Cybernetic Human HRP-4C, IEEE-RAS Int. Conf. Humanoid Robots, Paris (2009) pp. 7–14
17.16 K. Kaneko, F. Kanehiro, M. Morisawa, T. Tsuji, K. Mira, S. Nakaoka, S. Kajita, K. Yokoi: Hardware improvement of cybernetic human HRP-4C towards entertainent use, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), San Fransisco (2011) pp. 4392–4399
17.17 K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, T. Isozumi: Humanoid Robot HRP-2, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 1083–1090
17.18 K. Kaneko, K. Harada, F. Kanehiro, G. Miyamori, K. Akachi: Humanoid Robot HRP-3, Proc. Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 2471–2478
17.19 M. Kouchi, M. Mochimaru, H. Iwasawa, S. Mitani: Anthropometric database for Japanese population 1997-98, Japanese Industrial Standards Center, AIST, MITI http://riodb.ibase.aist.go.jp/dhbodydb/ (Tokyo 2000)
17.20 K. Kaneko, K. Harada, F. Kanehiro: Development of multi-fingered hand for life-size humanoid robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 913–920
17.21 Q. Lu, C. Ortega, O. Ma: Passive gravity compensation mechanisms: Technologies and applications, Recent Pat. Eng. **5**(1), 32–44 (2011)
17.22 K.A. Wyrobek, E.H. Berger, H.F.M. Van der Loos, J.K. Salisbury: Towards a personal robotics development platform: Rationale and design of an intrinsically safe personal robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008)
17.23 Willow Garage Inc., 68 Willow Road, Menlo Park, CA 94025, USA: http://www.willowgarage.com/pages/pr2/
17.24 T. McGeer: Passive dynamic walking, Int. J. Robotics Res. **9**(2), 62–82 (1990)
17.25 M. Garcia, A. Chatterjee, A. Ruina, M. Coleman: The simplest walking model: Stability, complexity, and scaling, ASME J. Biomech. Eng. **120**, 281–288 (1998)
17.26 A. Goswami, B. Thuilot, B. Espiau: A study of the passive gait of a compass-like biped robot: Symmetry and chaos, Int. J. Robotics Res. **17**, 1282–1301 (1998)

17.27 S.H. Collins, M. Wisse, A. Ruina: A three-dimensional passive-dynamic walking robot with two legs and knees, Int. J. Robotics Res. **20**(2), 607–615 (2001)
17.28 S.H. Collins, A. Ruina, R. Tedrake, M. Wisse: Efficient bipedal robots based on passive dynamic walkers, Sci. Mag. **307**, 1082–1085 (2005)
17.29 P.A. Bhounsule, J. Cortell, A. Ruina: Design and control of Ranger: an energy-efficient, dynamic walking robot, 15th Int. Conf. Climb. Walk. Robots (CLAWAR), Baltimore (2012) pp. 441–448
17.30 S.H. Collins, M. Wisse, A. Ruina: Three-dimensional passive-dynamic walking robot with two legs and knees, Int. J. Robotics Res. **20**(2), 607–615 (2001)
17.31 M. Wisse, L. Schwab, F.L.T. Van der Helm: Passive walking dynamic model with upper body, Robotica **22**(6), 681–688 (2004)
17.32 E.R. Westervelt, J.W. Grizzle, D.E. Koditschek: Hybrid zero dynamics of planar biped walkers, IEEE Trans. Autom. Control **48**(1), 42–56 (2003)
17.33 E.R. Westervelt, J.W. Grizzle, C. Chevallereau, J.H. Choi, B. Morris: *Feedback Control of Dynamic Bipedal Robot Locomotion* (CRC, Boca Raton 2007)
17.34 C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E.R. Westervelt, C. Canudas-de-Wit, J.W. Grizzle: RABBIT: A testbed for advanced control theory, IEEE Control Syst. Mag. **23**(5), 57–79 (2003)
17.35 J.W. Grizzle, J. Hurst, B. Morris, H.W. Park, K. Sreenath: MABEL, A new robotic bipedal walker and runner, Proc. IEEE Am. Control Conf. (2009)
17.36 D. Hobbelen, T. de Boer, M. Wisse: System overview of bipedal robots Flame and TUlip: Tailor-made for Limit Cycle Walking, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Nice (2008) pp. 2486–2491
17.37 R.J. Full, D.E. Koditschek: Templates and anchors: neuromechanical hypotheses of legged locomotion on land, J. Exp. Biol. **202**, 3325–3332 (1999)
17.38 S. Kajita, K. Tani: Study of dynamic biped locomotion on rugged terrain – Derivation and application of the linear inverted pendulum mode, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1991) pp. 1405–1411
17.39 M.B. Popovic, A. Goswami, H. Herr: Angular momentum regulation during human walking: Biomechanics and control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 2405–2411
17.40 M.B. Popovic, A. Goswami, H. Herr: Ground reference points in legged locomotion: Definitions, biological trajectories and control implications, Int. J. Robotics Res. **24**(12), 1013–1032 (2005)
17.41 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Biped walking pattern generation by using preview control of zero-moment point, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 1620–1626
17.42 Y. Choi, D. Kim, Y. Oh, B.J. You: Posture/walking control for humanoid robot based on kinematic resolution of com Jacobian with embedded motion, IEEE Trans. Robotics **23**(6), 1285–1293 (2007)
17.43 J. Englsberger, C. Ott, M. Roa, A. Albu-Schaeffer, G. Hirzinger: Bipedal walking control based on capture point dynamics, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 4420–4427
17.44 S. Lohmeier, T. Bushmann, H. Ulbrich: Humanoid Robot LOLA, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Kobe (2009) pp. 775–780
17.45 I.W. Park, J.-Y. Kim, J. Lee, J.H. Oh: Mechanical design of humanoid robot platform KHR-3 (KAIST Humanoid Robot 3: HUBO), Proc. IEEE-RAS Int. Conf. Humanoid Robots (2005) pp. 321–326
17.46 B. Stephens: Humanoid push recovery, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2007)
17.47 T. Takenaka, T. Matsumoto, T. Yoshiike: Real time motion generation and control for biped robot – 1st Report: Walking gait pattern generation, Proc. IEEE /RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 1084–1091
17.48 R. Blickhan: The spring mass model for running and hopping, J. Biomech. **22**(11–12), 1217–1227 (1989)
17.49 H. Geyer, A. Seyfarth, R. Blickhan: Compliant leg behaviour explains basic dynamics of walking and running, Proc. Biol. Sci. **273**(1603), 2861–2867 (2006)
17.50 J. Pratt, G. Pratt: Exploiting natural dynamics in the control of a planar bipedal walking robot, Proc. 36th Ann. Allerton Conf. Commun. (1998)
17.51 J. Pratt, B. Krupp: Design of a bipedal walking robot, SPIE Def. Sec. Symp., Bellingham (2008)
17.52 G. Hirzinger, N. Sporer, A. Albu-Schaeffer, M. Haehnle, R. Krenn, A. Pascucci, M. Schedl: DLR's torque-controlled light weight robot III – are we reaching the technological limits now?, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002) pp. 1710–1716
17.53 C. Ott, O. Eiberger, W. Friedl, B. Baeuml, U. Hillenbrand, C. Borst, A. Albu-Schaeffer, B. Brunner, H. Hirschmueller, S. Kielhoefer, R. Konietschke, M. Suppa, T. Wimboeck, F. Zacharias, G. Hirzinger: A humanoid two-arm system for dexterous manipulation, Proc. IEEE-RAS Int. Conf. Humanoid Robots, Genova (2006) pp. 276–283
17.54 C. Ott, C. Baumgaertner, J. Mayr, M. Fuchs, R. Burger, D. Lee, O. Eiberger, A. Albu-Schaeffer, M. Grebenstein, G. Hirzinger: Development of a biped robot with torque controlled joints, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2010) pp. 167–173
17.55 G.A. Pratt, M.M. Williamson: Series elastic actuators, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1995) pp. 399–406
17.56 R. Brooks, C. Breazeal, M. Marjanovic, B. Scassellati, M. Williamson: The Cog project: Building a humanoid robot, Lect. Notes Comput. Sci. **1562**, 52–87 (1999)
17.57 H. Iwata, S. Sugano: Development of human symbiotic robot: WENDY, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999)
17.58 M.W. Spong: Modeling and control of elastic joint robots, Trans. ASME: J. Dyn. Syst. Meas. Control **109**, 310–318 (1987)
17.59 G. Wyeth: Control issues for velocity sourced series elastic actuators, Proc. Australasian Conf. Robotics Autom. (2006)
17.60 H. Vallery, R. Ekkelenkamp, H. van der Kooij, M. Buss: Passive and accurate torque control of se-

ries elastic actuators, IEEE/RSJ Proc. Int. Conf. Intell. Robots Syst. (IROS) (2007)
17.61 C. Ott, A. Albu-Schaeffer, G. Hirzinger: Decoupling based cartesian impedance control of flexible joint robots, IEEE Int. Conf. Robotics Autom. (ICRA) (2003)
17.62 C. Ott, A. Albu-Schaeffer, A. Kugi, G. Hirzinger: On the passivity based impedance control of flexible joint robots, IEEE Trans. Robotics **24**(2), 416–429 (2008)
17.63 J. Heaston, D. Hong, I. Morazzani, P. Ren, G. Goldman: STriDER: Self-excited tripedal dynamic experimental robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Roma (2007) pp. 2776–2777
17.64 A. Rachmat, A. Besari, R. Zamri, A. Satria Prabuwono, S. Kuswadi: The study on optimal gait for five-legged robot with reinforcement learning, Int. Conf. Intell. Robots Appl. (2009) pp. 1170–1175
17.65 O. Matsumoto, S. Kajita, M. Saigo, K. Tani: Dynamic trajectory control of passing over stairs by a biped type leg-wheeled robot with nominal reference of static gait, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998) pp. 406–412
17.66 S. Hirose, H. Takeuchi: Study on roller-walk (basic characteristics and its control), Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 3265–3270
17.67 U. Saranli, M. Buehler, D.E. Koditschek: RHex: A Simple and Highly Mobile Hexapod Robot, Int. J. Robotics Res. **20**(7), 616–631 (2001)
17.68 T.J. Allen, R.D. Quinn, R.J. Bachmann, R.E. Ritzmann: Abstracted biological principles applied with reduced actuation improve mobility of legged vehicles, Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS), Las Vegas (2003) pp. 1370–1375
17.69 G. Endo, S. Hirose: Study on roller-walker: System integration and basic experiments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Detroit (1999) pp. 2032–2037
17.70 N. Neville, M. Buehler, I. Sharf: A bipedal running robot with one actuator per leg, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Orlando (2006) pp. 848–853
17.71 J. Bares, D. Wettergreen: Dante II: Technical description, results and lessons learned, Int. J. Robotics Res. **18**(7), 621–649 (1999)
17.72 N. Koyachi, H. Adachi, M. Izumi, T. Hirose, N. Senjo, R. Murata, T. Arai: Multimodal control of hexapod mobile manipulator MELMANTIS-1, Proc. 5th Int. Conf. Climb. Walk. Robots (2002) pp. 471–478
17.73 Y. Ota, T. Tamaki, K. Yoneda, S. Hirose: Development of walking manipulator with versatile locomotion, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 477–483
17.74 S. Hirose, K. Yoneda, H. Tsukagoshi: TITAN VII: Quadruped walking and manipulating robot on a steep slope, IEEE Int. Conf. Robotics Autom. (ICRA), Albuquerque (1997) pp. 494–500
17.75 S. Hirose, A. Nagakubo, R. Toyama: Machine that can walk and climb on floors, walls and ceilings, Proc. 5th Int. Conf. Adv. Robotics (ICAR), Pisa (1991) pp. 753–758
17.76 T. Yano, S. Numao, Y. Kitamura: Development of a self-contained wall climbing robot with scanning type suction cups, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 1 (1998) pp. 249–254
17.77 S. Kim, A. Asbeck, W. Provancher, M.R. Cutkosky: SpinybotII: Climbing hard walls with compliant microspines, Proc. Int. Conf. Adv. Robotics (ICAR), Seattle (2005) pp. 18–20
17.78 A.T. Asbeck, S. Kim, A. McClung, A. Parness, M.R. Cutkosky: Climbing walls with microspines (Video), Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Orlando (2006)
17.79 R.B. McGhee, A.A. Frank: On the stability properties of quadruped creeping gaits, Math. Biosci. **3**, 331–351 (1968)
17.80 R.B. McGhee: Vehicular legged locomotion. In: *Advances in Automation and Robotics*, ed. by G.N. Saridis (JAI, Greenwich 1985) pp. 259–284
17.81 M. Vukobratović, J. Stepanenko: On the stability of anthropomorphic systems, Math. Biosci. **15**, 1–37 (1972)
17.82 Q. Huang, K. Yokoi, S. Kajita, K. Kaneko, H. Arai, N. Koyachi, K. Tanie: Planning walking patterns for a biped robot, IEEE Trans. Robotics Autom. **17**(3), 280–289 (2001)
17.83 D.A. Messuri, C.A. Klein: Automatic body regulation for maintaining stability of a legged vehicle during rough-terrain locomotion, IEEE J. Robotics Autom. **RA-1**(3), 132–141 (1985)
17.84 E. Garcia, P. de Gonzalez Santos: An improved energy stability margin for walking machines subject to dynamic effects, Robotica **23**(1), 13–20 (2005)
17.85 D.G.E. Hobbelen, M. Wisse: A disturbance rejection measure for limit cycle walkers: The gait sensitivity norm, IEEE Trans. Robotics **23**(6), 1213–1224 (2007)
17.86 P. Gregorio, M. Ahmadi, M. Buehler: Design, control, and energetics of an electrically actuated legged robot, IEEE Trans. Syst. Man Cybern. **B27**(4), 626–634 (1997)
17.87 R. McNeill Alexander: The gait of bipedal and quadrupedal animals, Int. J. Robotics Res. **3**(2), 49–59 (1984)
17.88 R. McNeill Alexander: *Exploring Biomechanics – Animals in Motion* (Freeman, Boston 1992)
17.89 G. Gabrielli, T. von Karman: What price speed – Specific power required for propulsion of vehicles, Mechan. Eng. **72**(10), 775–781 (1950)
17.90 Y. Umetani, S. Hirose: Biomechanical study on serpentine locomotion – Mechanical analysis and zoological experiment for the stationary straightforward movement, Trans. Soc. Instrum. Control Eng. **6**, 724–731 (1973), in Japanese
17.91 S. Collins, A. Ruina, R. Tedrake, M. Wisse: Efficient Bipedal Robots Based on Passive-Dynamic Walkers, Science **307**, 1082–1085 (2005)

Multimedia Contents

18. Parallel Mechanisms

Jean-Pierre Merlet, Clément Gosselin, Tian Huang

This chapter presents an introduction to the kinematics and dynamics of parallel mechanisms, also referred to as parallel robots. As opposed to classical serial manipulators, the kinematic architecture of parallel robots includes *closed-loop kinematic chains*. As a consequence, their analysis differs considerably from that of their serial counterparts. This chapter aims at presenting the fundamental formulations and techniques used in their analysis.

18.1 **Definitions** 443
18.2 **Type Synthesis of Parallel Mechanisms** 445
18.3 **Kinematics** 446
18.3.1 Inverse Kinematics 446
18.3.2 Forward Kinematics 446
18.4 **Velocity and Accuracy Analysis** 447
18.5 **Singularity Analysis** 448
18.5.1 Parallel Singularity Analysis 448
18.6 **Workspace Analysis** 450
18.7 **Static Analysis** 451
18.8 **Dynamic Analysis** 452
18.9 **Design** 452
18.10 **Wire-Driven Parallel Robots** 453
18.10.1 Cable-Suspended Parallel Mechanisms 454
18.10.2 Fully Constrained Wire-Driven Parallel Mechanisms 454
18.10.3 Design and Prototyping 455
18.11 **Application Examples** 455
18.12 **Conclusion and Further Reading** 455
Video-References 456
References 456

18.1 Definitions

A *closed-loop kinematic chain* is one in which the links and joints are arranged such that at least one closed loop exists. Furthermore, a *complex closed-loop kinematic chain* is obtained when one of the links – other than the base – has a degree of connectivity greater than or equal to 3, i. e., one link other than the base is connected through joints to at least three other links. A *parallel manipulator* can be defined as a closed-loop mechanism composed of an end effector having n degrees of freedom (DOF) and a fixed base, linked together by at least two independent kinematic chains.

An example of such a structure was patented in 1928 by *Gwinnett* [18.1] to be used as a platform for a movie theatre. In 1947, *Gough* [18.2] established the basic principles of a mechanism with a closed-loop kinematic structure (Fig. 18.1) that allows the positioning and the orientation of a moving platform so as to test tire wear and tear. He built a prototype of this machine in 1955.

For this mechanism, the moving effector is a hexagonal platform whose vertices are all connected to a link by a ball-and-socket (spherical) joint. The other end of the link is attached to the base by a universal joint. A linear actuator allows the modification of the total length of the link; this mechanism is, therefore, a closed-loop kinematic structure, actuated by six linear actuators.

In 1965, *Stewart* [18.3] suggested the use of such a structure for flight simulators and the Gough mechanism is sometimes referred to as the *Stewart platform*

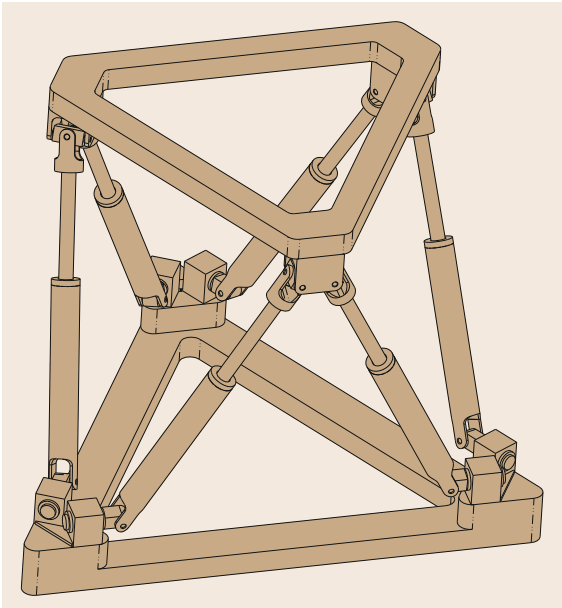


Fig. 18.1 Gough platform

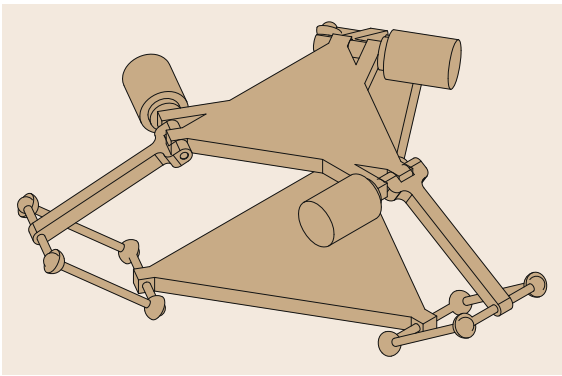


Fig. 18.2 The Delta robot

or the *Gough–Stewart platform*. The same architecture was also concurrently proposed by *Kappel* as a motion simulation mechanism [18.4]. Nowadays, the Gough platform and its variants are the platform of choice for many applications, for example, flight simulators. This application provides a convincing illustration of the main advantage of parallel robots namely, their load-carrying capacity. Indeed, while the ratio of the mass of the payload over the mass of the robot is typically smaller than 0.15 for serial 6R industrial robots, this ratio can be larger than 10 for parallel structures. Another advantage of the Gough platform is its very good positioning accuracy (although it has been questioned [18.5]). This high accuracy arises from the fact that the legs work essentially in tension/compression and are subjected to virtually no bending – thereby

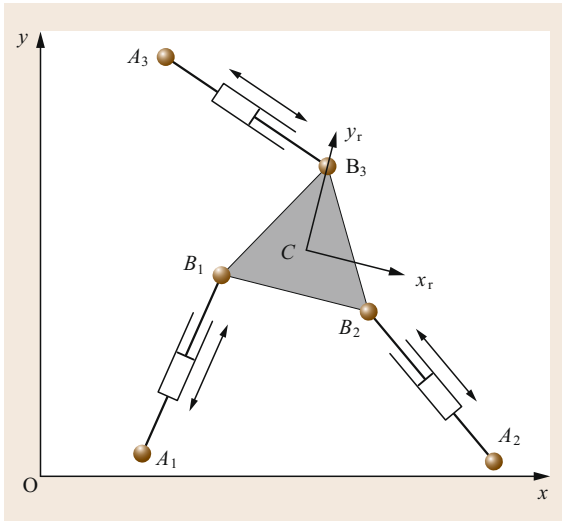


Fig. 18.3 The $3-\underline{RPR}$ planar parallel robot

leading to very small deformations – and from the fact that the errors in the internal sensors of the robot (measurement of the lengths of the legs for the Gough platform) are mapped into very small errors of the platform position. Parallel robots are also almost insensitive to scaling (the same structure can be used for large or micro robots) and they can be built using almost any type of joint, actuator, or transmission, including exotic ones such as tape spring and flexure joints [18.6] or binary actuators [18.7]. The main drawbacks of parallel robots are their small workspace and the singularities that can appear within this workspace. However, large workspaces may be obtained using wire transmissions (see, for example, the Robocrane [18.8]): such robots will be specifically addressed in Sect. 18.10 as their analysis is somewhat different. Apart from the Gough platform, the most successful designs of parallel robots are the Delta robot proposed by *Clavel* [18.9] (Fig. 18.2) and some planar parallel robots. The most common planar parallel robots have three identical legs each having an $\underline{RPR}$ or $\underline{RRR}$ architecture, where the underlined joint is the actuated one (such robots are often denoted as $3-\underline{RPR}$ ($3-\underline{RRR}$; Fig. 18.3).

The geometric arrangement of the joints and links of the Delta structure provides three translational degrees of freedom at the platform. Numerous other types of parallel robots have been proposed in the recent years. Although most existing architectures are based on the intuition of their designer, the synthesis of parallel mechanisms can be dealt with systematically. An overview of the main approaches to the type synthesis of parallel mechanisms is given in the next section.

18.2 Type Synthesis of Parallel Mechanisms

Determining all potential mechanical architectures of parallel robots that can produce a given motion pattern at the moving platform is a challenging problem. Several authors have addressed this problem, referred to as type synthesis. The approaches proposed in the literature can be divided into four groups:

- *Approaches based on graph theory*: The enumeration of all possible structures having a given number of DOFs can be performed by considering that there is only a finite set of possible kinematic pairs, and hence a very large, but finite, set of possible structure combinations (e.g., [18.10]). Classical mobility formulae – for example, the Chebychev–Grübler–Kutzbach formula – are then used to determine the number of DOFs in the structure. Unfortunately, these formulas do not take into account geometric properties of the mechanism that can change the number of DOFs of the platform. Although many parallel robots have been successfully invented and developed for industrial uses, based on graph theory – the Neos Tricept [18.11] and the Sprint Z3 head of DS Technologie Gmbh, for example – the synthesis approaches based strictly on graph theory can produce only limited nonoverconstrained parallel robots and have been largely superseded by the other methods described below.
- *Approaches based on group theory*: The motion of a rigid body has the special structure of a group, the *displacement group*. Subgroups of the group of displacements, such as the spatial translations or all the translational and rotational motions about all axes that are parallel to the axis defined by a given vector (Schönflies motion [18.12]), play an important role as they can be combined through the *intersection* operation [18.13] when elements of subgroups act on the same rigid body. Type synthesis consists in determining all the possible subgroups to which the different kinematic chains that will constitute the legs of the robot may belong to so that their intersection leads to the desired motion pattern for the platform. The synthesis approaches based on group theory led to the discovery of numerous possible architectures [18.14]. Nevertheless, the displacement group has special properties that are not reflected by its group structure alone. Also, the approach is limited to motion patterns that can be described by a subgroup of the displacement group.
- *Approach based on differential geometry*: As a full extension and completion of the Lie-group-algebraic method, the differential geometry method can be used to cover the motion patterns that cannot be modeled by subgroups but regular submanifolds, a plane-hinged parallelogram, for example [18.15]. The additional merit of the geometric method lies in that it enables to link the finite motion with the instantaneous motion of the platform thanks to the integral/differential relationship between the Lie group and Lie algebra.
- *Approaches based on screw theory*: In this approach, the first step is to determine the wrench system S that is reciprocal to the theoretically accessible instantaneous twist (or velocity twist) of the moving platform. Then, the wrenches of each limb whose union spans the system S (and that determine all the possible structures of the kinematic chains that will generate the corresponding wrenches) are enumerated. Although this method is very effective at providing simple and intuitive descriptions of the instantaneous motions of the platform, it is necessary to verify that the mobility of the platform is full cycle and not only instantaneous as all considered twists and wrenches are instantaneous. Thus, techniques to ensure the consistency between the finite and instantaneous motions have also been developed. A systematic implementation of this approach is provided in [18.16].

The above synthesis methods have been used to generate a large number of architectures that cannot be presented in this book, but the website [18.17] presents a comprehensive description of a large number of mechanical architectures. Among others, several remarkable architectures were proposed in the recent years to generate Schönflies motions using the innovative concept of articulated plates, leading to a brand new pick-and-place parallel robot known as Adept Quattro [18.18]. Also, other similar architectures allowing unlimited rotations of the end effector while avoiding all singularities have recently been proposed (see for instance [18.19]).

It should also be pointed out that although these synthesis methods can be used to generate a large number of architectures [18.20], there is a lack of a metric to evaluate which one is better than the other even if they have completely identical motion patterns. Furthermore, it is hard to link the models for type synthesis with those for kinematic, static, and dynamic analyses and design. Open issues, therefore, include:

- To develop suitable performance metrics for each motion type to identify suitable topological structures for specific applications.

- To develop a complete theoretical package that enables type synthesis and parametric design to be integrated into a unified framework that takes into account the unavoidable uncertainties in the manufacturing process, an ultimate objective of design.

18.3 Kinematics

18.3.1 Inverse Kinematics

The solution of the inverse kinematic problem is usually straightforward for parallel robots. This can be illustrated with the Gough platform.

The solution of the inverse kinematic problem consists in determining the vector of leg lengths $\boldsymbol{q}$ for a given pose of the platform defined by the position vector $\boldsymbol{p}$ of a given point on the platform in a fixed reference frame and a rotation matrix $\mathbf{R}$ representing the orientation of the platform with respect to the fixed frame. Let $\boldsymbol{a}_i$ denote the position vector of the anchor point of the i-th leg on the base given in the fixed reference frame and $\boldsymbol{b}_i$ denote the position vector of the anchor point of the i-th leg on the platform, given in a moving frame attached to the platform. The length of the i-th leg is the norm of the vector connecting the two anchor points. This vector, noted $\boldsymbol{s}_i$, can be written as

$$\boldsymbol{s}_i = \boldsymbol{p} + \mathbf{R}\boldsymbol{b}_i - \boldsymbol{a}_i\,, \quad i = 1, \ldots, 6\,. \tag{18.1}$$

Given the pose of the platform (vector $\boldsymbol{p}$ and matrix $\mathbf{R}$), vector $\boldsymbol{s}_i$ is readily calculated using (18.1) and, hence, the leg lengths can be obtained.

18.3.2 Forward Kinematics

The solution of the forward kinematic problem consists in determining the pose of the platform for a given set of actuated joint coordinates (a given vector $\boldsymbol{q}$). This solution is needed for control purposes, calibration, and motion planning.

The forward kinematic problem of a parallel robot is usually much more complex than the inverse kinematic problem. Indeed, the loop closure equations (18.1) are typically highly nonlinear expressions of the pose variables. They form a nonlinear set of equations that generally admits multiple solutions (e.g., the Gough platform can have up to 40 solutions [18.21–23], while a table with the number of solutions for Gough platforms with special geometries is provided in [18.24]). The forward kinematic problem arises in two different contexts. In the first case, no estimate of the current pose of the platform is available (e.g., when starting the robot), while in the second case, a relatively precise estimate of the pose is known (e.g., in real-time control when the forward kinematics has been solved at the previous sampling time of the controller). In the first case, the only known approach is to determine all the solutions of the inverse kinematic equations, although there is no known algorithm to sort the set of solutions. It is often possible to determine an upper bound on the number of real solutions. Consider, for example, the case of the planar $3-\underline{R}PR$ robot (Fig. 18.3). If the joint at point B_3 is disassembled, two separate mechanisms are obtained, namely, a 4-bar linkage and a rotary lever. From the kinematics of 4-bar linkages, it is known that point B_3 moves on a sixth-order algebraic curve. Moreover, it is also known that point B_3 of the lever moves on a circle, that is, a second-order algebraic curve. For a given set of actuated joint coordinates, solutions to the forward kinematic problem arise when the two curves intersect, that is, when the mechanism can be assembled. Bezout's theorem states that algebraic curves of order m, n intersect in nm points, counting their order of multiplicity. In the above case, this means that the curves intersect in 12 points. However, these 12 points include the two circular imaginary points that belong to the coupler curve of the 4-bar linkage and to any circle, thereby to their intersection. These points are counted three times in Bezout's theorem and, hence, the forward kinematic problem will have at most six real solutions, corresponding to the intersection points.

Various methods have been proposed for the solution of the forward kinematic problem: elimination [18.25], continuation [18.23], Gröbner bases [18.26], and interval analysis [18.27]. Elimination is usually not very stable numerically (i. e., it can produce spurious roots and miss solutions) unless special care is taken in the implementation of the resulting univariate equation and the elimination steps. For instance, the transformation of polynomial solutions into eigenvalue problems can be used [18.28]. Polynomial continuation, on the other hand, is much more stable numerically since mature algorithms can be found in the literature [18.29]. The fastest methods – although not appropriate for a real-time use – are Gröbner bases and interval analysis. They also have the advantage of being numerically certified (no roots can be missed and the solution can be computed with an arbitrarily prescribed accuracy).

However, in the simplest cases, elimination can usually be used to produce stable implementations as, for example, for the $3-\underline{RPR}$ robot [18.30, 31]. Finally, it is pointed out that the forward kinematic problem solved for one type of robot may often be used for another architecture: for example, the one developed above for the $3-\underline{RPR}$ robot may be used for the $3-\underline{RRR}$. This is typical for parallel robots.

When a priori information (an initial guess of the solution) is available, the forward kinematics is usually solved using the Newton–Raphson or the Newton–Gauss iterative scheme. Consider the solution to the inverse kinematic problem, written as

$$\boldsymbol{q} = \boldsymbol{f}(\boldsymbol{x}) \,. \tag{18.2}$$

Iteration k of the Newton–Raphson procedure is written as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \mathbf{A}(\boldsymbol{q} - \boldsymbol{f}(\boldsymbol{x}_k)) \,, \tag{18.3}$$

where $\boldsymbol{q}$ is the vector of prescribed joint variables. Matrix $\mathbf{A}$ is usually chosen as $(\partial \boldsymbol{f}/\partial q)^{-1}(\boldsymbol{x}_k)$ (the inverse matrix may not be computed at each iteration or even be chosen as constant). The iterative scheme stops when the magnitude of the difference vector $(\boldsymbol{q} - \boldsymbol{f}(\boldsymbol{x}_k))$ is smaller than a chosen threshold.

Provided that a good initial estimate of the solution is available, the Newton–Raphson algorithm is usually very fast. However, the procedure may not converge or, even worse, it may converge to a solution that is not the correct pose of the platform, that is, it may converge to another assembly mode. Such a situation can occur even if the initial guess is arbitrarily close to the correct pose. If the result is used in a control loop, the consequences can be catastrophic. Fortunately, mathematical tools, such as the Kantorovitch theorem combined with interval analysis can be used to determine if the solution found by the Newton–Raphson scheme is the correct pose of the robot, that is, to certify the result at the cost of a longer computation time, yet still compatible with real time [18.27].

Furthermore, the choice of the inverse kinematics equations plays an important role in the convergence of this method [18.27]. For example, for a Gough platform, a minimal set of equations (six equations involving six unknowns: three for the translation and three orientation angles) can be used but other representations are possible. For instance, the position coordinates of three of the platform anchor points – in the fixed reference frame – can be used as unknowns (the coordinates of the remaining three points are then readily computed upon convergence). Using this representation, nine equations are needed. Six equations are obtained based on the known distances between the anchor points on the base and platform and three additional equations are obtained based on the known distances between all combinations of the three selected platform anchor points.

Another possible approach for the solution of the forward kinematic problem is either to add sensors on the passive joints (e.g., on the U joints of a Gough platform) or to add passive legs with sensed joints. The main issue is then to determine the number and location of the additional sensors that lead to a unique solution [18.32–34] and to determine the effect of the sensor errors on the positioning error of the platform. For example, Stoughton mentions that for a Gough platform with sensors in the U joints, it is still necessary to use the Newton–Raphson scheme to improve the accuracy of the solution because the solutions obtained with the additional sensors are very sensitive to measurement noise [18.35].

18.4 Velocity and Accuracy Analysis

Similarly to serial robots, the actuated joint velocity vector of parallel robots $\dot{\boldsymbol{q}}$ is related linearly to the vector of translational and angular velocities of the platform (for simplicity, the latter vector is denoted here as $\dot{\boldsymbol{p}}$ although the angular velocity is not the time derivative of any set of angular parameters). The linear mapping between the two vectors is provided by the Jacobian matrix $\mathbf{J}$

$$\dot{\boldsymbol{p}} = \mathbf{J}(p)\dot{\boldsymbol{q}} \,. \tag{18.4}$$

However, for parallel robots, a closed-form expression is usually available for the inverse Jacobian $\mathbf{J}^{-1}$ but this is much more difficult for $\mathbf{J}$ (more precisely, the closed-form for most 6-DOF robots will be so complex that it cannot be used in practice). For example, a simple static analysis can be used to show that the i-th row $\mathbf{J}_i^{-1}$ of matrix $\mathbf{J}^{-1}$ for a Gough platform can be written as

$$\mathbf{J}_i^{-1} = (\boldsymbol{n}_i \quad \boldsymbol{c}_i \times \boldsymbol{n}_i) \,, \tag{18.5}$$

where $\boldsymbol{n}_i$ represents the unit vector defined along leg i and $\boldsymbol{c}_i$ is the vector connecting the origin of the mobile frame attached to the platform to the i-th anchor point on the platform.

The effect of the joint sensor errors $\Delta \boldsymbol{q}$ on the positioning error $\Delta \boldsymbol{p}$ follows the same relationship, namely

$$\Delta \boldsymbol{p} = \mathbf{J}(\boldsymbol{p}) \Delta \boldsymbol{q} \,. \tag{18.6}$$

Since matrix $\mathbf{J}$ is difficult to be obtained in a closed form, accuracy analysis (i. e., finding the maximal positioning errors over a given workspace for bounded joint sensor errors) is much more difficult than for serial robots [18.36, 37]. Apart from the measurement errors, there are other sources of inaccuracy in parallel robots, namely, clearance in the passive joints, manufacturing tolerances, thermal errors, gravity induced and dynamic errors [18.38, 39]. The effect of joint clearances on trajectories followed by serial and parallel robots was studied in [18.40–42]. According to these works, it is impossible to determine trends for the effect of the geometric errors: a case-by-case study must be performed since the effect is highly dependent on the mechanical architecture, dimensioning, and workspace of the robot. Thermal effects are sometimes mentioned as possible sources of inaccuracy, although few works substantiate this claim [18.43] and cooling may slightly reduce their effects [18.44, 45].

Calibration is another means of improving the accuracy of parallel robots. This issue was addressed in Chap. 3. The methods and procedures used for parallel robots differ slightly from the ones used for serial robots since only the inverse kinematic equations are available and the positioning of the platform is much less sensitive to the geometric errors than in serial robots [18.46, 47]. Hence, the measurement noise occurring during calibration has a significant impact and may lead to surprising results. For example, the classical least-squares method may lead to parameters that are such that some constraint equations are not satisfied even when the measurement noise is taken into account [18.48] or even to parameter values that are unrealistic. It has also been shown with experimental data that classical parallel robot modeling leads to constraint equations that do not have any solution irrespective of the measurement noise [18.49]. Moreover, calibration is very sensitive to the choice of the calibration poses [18.50]: it seems that the optimal choice is to select poses on the workspace boundary [18.51, 52]. We may also mention an original calibration approach based not on the proprioceptive sensors of the robot but on leg observation with a vision system [18.53].

18.5 Singularity Analysis

The analysis of singularities of parallel mechanisms was first addressed in [18.54]. In this formulation, the kinematic equations were reduced to the input–output relationship between the actuated joint coordinate vector $\boldsymbol{q}$ and the platform Cartesian coordinate vector $\boldsymbol{p}$, namely,

$$\boldsymbol{f}(\boldsymbol{q}, \boldsymbol{p}) = 0 \,. \tag{18.7}$$

Differentiating (18.7) with respect to time leads to

$$\mathbf{B}\dot{\boldsymbol{q}} + \mathbf{A}\dot{\boldsymbol{p}} = \mathbf{0} \,. \tag{18.8}$$

Three types of singularities can then be defined: *i*) when matrix $\mathbf{B}$ is singular (termed serial singularity), *ii*) when matrix $\mathbf{A}$ is singular (termed parallel singularity) and *iii*) when the input–output equations degenerate (termed architecture singularity) in which case both matrices $\mathbf{A}$ and $\mathbf{B}$ can be singular. In a serial singular configuration, the joints can have a nonzero velocity, while the platform is at rest. In a parallel singularity, there exist nonzero platform velocities for which the joint velocities are zero. In the neighborhood of such a configuration, the robot can undergo an infinitesimal motion, while the actuators are locked. Since the mobility of the end effector should be zero when the actuators are locked, it is also said that in such a configuration, the robot *gains* some DOFs. As a consequence, certain degrees of freedom of the platform cannot be controlled, and this is a major problem.

A more general study of singularities was proposed by *Zlatanov et al.* [18.55]. The latter analysis uses the velocity equations involving the full twist of the end effector and all joint velocities (passive or actuated). This approach led to a more detailed classification of singularities and was also used later to reveal special singularities (referred to as *constraint singularities*) [18.56] that could not be found with the analysis presented in [18.54]. Also, the singularity analysis of parallel mechanisms was revisited more recently in [18.57], where hierarchical levels in which different critical phenomena originate are recognized.

The singularity analyses discussed above are of the first order. Second-(and higher-)order singularity analyses can also be performed, although these analyses are much more complex [18.42, 58].

18.5.1 Parallel Singularity Analysis

This type of singularity is especially important for parallel robots because it corresponds to configurations in

which the control of the robot is lost. Furthermore, very large joint forces can occur in the vicinity of singular poses that may lead to a breakdown of the robot. The main issues to be addressed in this context are:

1. The characterization of the singularities.
2. The definition of performance indices representing the closeness to a singularity.
3. The development of algorithms that are capable of detecting the presence of singularities over a given workspace or trajectory.

Parallel singularities arise when the 6×6 *full inverse Jacobian* $\mathbf{J}_f$ (i. e., the matrix that maps the full twist of the platform into active – and eventually passive – joint velocities) is singular, that is, when its determinant $\det(\mathbf{J}_f)$ is equal to 0. It is pointed out that passive joint velocities sometimes have to be included because restricting the analysis to the active joint velocities may not allow the determination of all singular configurations of the mechanism (see the example of the $3-U\underline{P}U$ robot [18.59]). Usually, a proper velocity analysis allows one to establish this matrix in a closed form. But computing the determinant of this matrix may be difficult even with symbolic computation tools (see the example of the Gough platform [18.60, 61]) and it does not provide an insight into the geometric conditions associated with the singularities.

An alternative approach is to use line geometry: indeed, for several parallel robots (although not all), a row of $\mathbf{J}_f$ corresponds to the Plücker vector of some line defined on the links of the robot. For example, for a Gough platform, the rows of $\mathbf{J}_f$ are the normalized Plücker vectors of the lines associated with the legs of the robot. A singularity of matrix $\mathbf{J}_f$, therefore, implies a linear dependency between these vectors (they then constitute a *linear complex*), a situation that may occur only if the lines associated with the vectors verify particular geometric constraints [18.62]. These geometric constraints have been identified by Grassmann for every set of 3, 4, 5, or 6 vectors. Singularity analysis is, thus, reduced to determining conditions on the pose parameters for which these constraints are satisfied, giving geometric information on the singularity variety. A variant of this approach is to use the Grassmann Cayley algebra that in certain cases may lead directly to the geometric singularity conditions [18.63].

Measuring closeness between a pose and a singular configuration is a difficult problem: there exists no mathematical metric defining the distance between a prescribed pose and a given singular pose. Hence, a certain level of arbitrariness must be accepted in the definition of the *distance* to a singularity and none of the proposed indices is perfect. For example, a possible index is the determinant of $\mathbf{J}_f$: unfortunately, when the platform is subjected to both translational and rotational motion, the Jacobian matrix is not dimensionally homogeneous and, hence, the value of the determinant changes according to the physical units used to describe the geometry of the robot. Dexterity indices, as defined in Chap. 10 (although they are much less relevant for parallel robots than for serial robots [18.37]), can also be used, and special indices for parallel robots have also been proposed [18.64–66]. For example, it may be interesting for a Gough platform to define a workspace as singularity-free if for any pose the absolute value of the joint forces is lower than a given threshold [18.67] or to use a physically meaningful index like the kinematic sensitivity [18.68].

Most of the analyses based on the above indices are local, that is, valid only for a given pose, while in practice the problem is to determine if a singular configuration can occur over a given workspace or trajectory. Fortunately, an algorithm exists that allows this verification, even if the geometric model of the robot is uncertain [18.69]. However, it should be pointed out that a singularity-free workspace may not always be optimal for parallel robots. Indeed, other performance requirements may impose the presence of singularities in the workspace or the robot may have part of its workspace singularity-free (for example its working region) while exhibiting singularities outside this region. Therefore, a motion planner producing a trajectory that avoids singularities while remaining close to a desired path is advisable and various approaches have been proposed to address this problem [18.70, 71]. A related problem is to determine if two solutions of the forward kinematics may be joined without crossing a singularity: this has been proved for planar robots [18.72, 73] but also for 6-DOF robots [18.74].

Finally, it is noted that being close to a singular configuration may be useful in some cases. For example, large amplification factors between the end-effector motion and the actuated joint motion may be interesting for fine positioning devices with a very small workspace or for improving the sensitivity along some measurement directions for a parallel robot used as a force sensor [18.75]. It should also be mentioned that parallel robots that remain *permanently* in a singular configuration may be interesting since they are capable of producing complex motions with only one actuator [18.76–78].

18.6 Workspace Analysis

As mentioned earlier, one of the main drawbacks of parallel robots is their small workspace. Moreover, their workspace analysis is usually more complex than for serial robots, especially if the platform has more than 3-DOFs. Indeed, the platform motion capabilities are usually coupled, thereby prohibiting simple graphical representations of the workspace. *Decoupled robots* have also been proposed [18.79–81] (VIDEO 54 and VIDEO 52), but they do not provide the load-carrying capacity of conventional parallel robots. In general, the workspace of parallel robots is restricted by the following:

- The limitations on the actuated joint variables: for example, the lengths of the legs of a Gough platform have a minimum and maximum value.
- The limitations on the range of motion of the passive joints: for example, the U and S joints of a Gough platform have restricted ranges of angular motion.
- The mechanical interferences between the bodies of the mechanism (legs, base, and platform).

Different types of workspaces can be defined such as, for example, the *dextrous* workspace (the set of platform locations in which all orientations are possible), the *maximal* workspace (the set of platform locations that may be reached with at least one orientation), or the *orientation* workspace (the set of orientations that can be reached for a given location of the platform).

A possible approach for representing the workspace of parallel robots with $n > 3$ DOFs is to fix the value of $n-3$ pose parameters in order to be able to plot the workspace over the remaining 3-DOFs. Such plots can be obtained very efficiently using geometric methods if the plotted variables involve only translational motions because a geometric approach usually allows one to establish directly the nature of the boundary of the workspace (e.g., [18.82, 83] for the calculation of the workspace of a Gough platform when its orientation is fixed). Another advantage of this approach is that it allows the computation of the surface and volume of the workspace. Possible alternatives include:

- Discretization methods in which all poses of an n-dimensional grid are checked with respect to all types of kinematic constraints. Such methods are usually simple to implement but are computationally intensive because the computation time increases exponentially with the resolution of the mesh. Also, they require a large storage space.
- Numerical methods that allow the determination of the workspace boundary [18.84, 85].
- Numerical methods based on interval analysis that allow the determination of an approximation of the workspace volume up to an arbitrary accuracy [18.86, 87]. This representation is also appropriate for motion planning problems.

Singularities can also split the workspace calculated from the above kinematic constraints into elementary components, called the *aspects* by *Wenger* and *Chablat* [18.88] and which are separated by a singularity variety. Some very good results have been obtained for planar robots [18.89] but determining the aspects for spatial robots is still an open problem. A parallel robot may not always be able to move from one aspect to another (at least without considering the dynamics of the robot [18.90, 91]) and hence the useful workspace can be reduced.

A problem related to workspace analysis is the *motion planning problem*, which is slightly different from that encountered with serial robots. For parallel robots, the problem is not to avoid obstacles within the workspace but either to determine if a given trajectory lies entirely inside the workspace of the robot and is singularity-free (for which an algorithm is available [18.92]) or to determine such a trajectory between two poses which is much more difficult [18.93]. Indeed, classical serial robot motion planners work in the joint space and assume that there is a one-to-one relationship between the joint space and the operational space. With this assumption, it is possible to determine a set of points in the joint space that is collision-free and to use this knowledge to build a collision-free path between two poses. However, this assumption is not valid for parallel robots because the mapping between the joint space and the operational space is not one to one: a point in the joint space may either correspond to multiple points in the operational space or have no correspondence because the closure equations of the mechanism are not satisfied. For parallel robots, the most efficient motion planner seems to be an adaptation of a probabilistic motion planner that takes into account – to a certain extent – the closure equations [18.94, 95].

Another motion planning problem is related to tasks where a limited number of DOFs of the robot are used, while the others may be used to increase the machine's workspace, to avoid singularities or to optimize some performance index of the robot [18.96, 97]. Along the same line, it is possible to define the *part positioning* problem [18.98] as determining the pose of a rigid body with respect to the workspace of the robot so that this pose satisfies some constraints.

18.7 Static Analysis

Similarly to serial robots, the static analysis of parallel robots can be readily performed using the Jacobian matrix. Indeed, the mapping between the vector of actuated joint forces/torques $\boldsymbol{\tau}$ and the external wrench $\boldsymbol{f}$ exerted by the platform can be written as

$$\boldsymbol{\tau} = \mathbf{J}^{\mathbf{T}} \boldsymbol{f} \,, \tag{18.9}$$

where $\mathbf{J}^{\mathbf{T}}$ is the transpose of the Jacobian matrix of the robot. Equation (18.9) can be used for various purposes, namely:

- During the *design* process in order to determine the maximal actuator forces/torques over the robot workspace (for actuator selection). This is a complex issue since $\mathbf{J}^{\mathbf{T}}$ is not known in a closed form.
- In applications where the robot is used as a *force sensor*: If $\boldsymbol{\tau}$ is measured and the pose of the platform is known, then (18.9) can be used to calculate $\boldsymbol{f}$ so that the robot can be used both as a motion and sensing platform [18.99–101].

The stiffness matrix $\mathbf{K}$ of a parallel robot is defined similarly to that of serial robots as

$$\mathbf{K} = \mathbf{J}^{-\mathbf{T}} \mathbf{K}_{\boldsymbol{j}} \mathbf{J}^{-\mathbf{1}} \,, \tag{18.10}$$

where $\mathbf{K}_{\boldsymbol{j}}$ is the diagonal matrix of actuated joint stiffness. *Duffy* [18.102] notes that this derivation is, in general, incomplete. For example, for a Gough platform this derivation assumes that there is no initial load on the elastic element of the link. Assuming that the length of the unloaded link is q_i^0 leads to

$$\Delta \boldsymbol{f} = \sum_{i=1}^{i=6} k \Delta q_i \boldsymbol{n_i} + k_i (q_i - q_i^0) \Delta \boldsymbol{n_i} \,,$$

$$\Delta \boldsymbol{m} = \sum_{i=1}^{i=6} k \Delta q_i \boldsymbol{c_i} \times \boldsymbol{n_i} + k_i (q_i - q_i^0) \Delta (\boldsymbol{c_i} \times \boldsymbol{n_i}) \,,$$

where k_i denotes the axial stiffness of the leg, $\boldsymbol{n_i}$ represents the unit vector defined along the i-th leg, $\boldsymbol{c_i}$ is the vector connecting the origin of the moving reference frame to the i-th anchor point of the platform, and $\boldsymbol{f}, \boldsymbol{m}$ are the external force and moment vectors that are applied by the platform. Consequently, the stiffness matrix as defined in (18.10) is valid only if $q_i = q_i^0$, and is coined the *passive stiffness*.

Moreover, it was pointed out in the literature [18.103] that the formulation of (18.10) is valid only when the external wrench is zero. Indeed, the above formulation does not account for the fact that the Jacobian matrix is configuration dependent and therefore changes with the applied loads. The formulation proposed in [18.103] and termed *conservative congruence transformation* (CCT) takes these changes into account and leads to a stiffness matrix defined as

$$\mathbf{K}_c = \mathbf{J}^{-\mathbf{T}} \mathbf{K}_{\boldsymbol{j}} \mathbf{J}^{-\mathbf{1}} + \left(\frac{\partial \mathbf{J}^{-\mathrm{T}} \boldsymbol{\tau}}{\partial \boldsymbol{p}} \right) \,, \tag{18.11}$$

where $\boldsymbol{p}$ is the vector of Cartesian coordinates and $\boldsymbol{\tau}$ is the vector of actuated joint forces/torques. Whenever possible, the formulation of (18.11) should be used instead of that of (18.10) because it is mechanically consistent.

Another interesting static problem is the static balancing of parallel robots. The static balancing of mechanisms, in general, has been an important research topic for several decades (e.g., [18.104] for an account of the state of the art and many recent new results). Parallel mechanisms are said to be *statically balanced* when the weight of the links does not produce any torque (or force) at the actuators under static conditions, for any configuration of the manipulator or mechanism. This condition is also referred to as *gravity compensation*. The gravity compensation of parallel mechanisms was addressed in [18.105] where the use of counterweights was suggested to balance a 2-DOF parallel robot used for antenna aiming, and in [18.106] for planar robots. This analysis leads to simple and elegant balancing conditions.

In general, static balancing can be achieved using counterweights and/or springs VIDEO 48. When springs are used, static balancing can be defined as the set of conditions for which the total potential energy in the mechanism – including gravitational energy and the elastic energy stored in the springs – is constant for any configuration of the mechanism. When no springs – or other means of storing elastic energy – are used, then static balancing conditions imply that the center of mass of the mechanism does not move in the direction of the gravity vector, for any motion of the mechanism.

Sufficient conditions under which a given mechanism is statically balanced may be obtained. The problem of statically balancing spatial parallel robots is usually complex [18.107, 108]). Among other results, it was pointed out that it is not possible to balance a 6-*UPS* robot in all poses by using only counterweights [18.109] (*U* stands for a Hooke joint, *P* stands for a prismatic joint and *S* stands for a spherical joint). Alternative mechanisms (using parallelograms) that can

be balanced using only springs were suggested [18.107, 110, 111].

A natural extension of static balancing is dynamic balancing. The dynamic balancing problem consists in finding parallel mechanisms that do not produce any reaction force or torque on their base for arbitrary trajectories. This problem was addressed for planar and spatial parallel mechanisms. In [18.112], it was shown that dynamically balanced parallel mechanisms with up to 6-DOFs can be synthesized, although the resulting mechanical architectures may be complex (VIDEO 49).

18.8 Dynamic Analysis

The dynamic model of parallel robots has a form similar to that of serial robots (Chap. 3), namely,

$$\mathbf{M}(\boldsymbol{x})\ddot{\boldsymbol{p}} + \mathbf{C}(\dot{\boldsymbol{p}}, \dot{\boldsymbol{q}}, \boldsymbol{p}, \boldsymbol{q}) + \mathbf{G}(\boldsymbol{p}, \boldsymbol{q}) = \boldsymbol{\tau} \ , \qquad (18.12)$$

where $\mathbf{M}$ is the positive-definite generalized inertia matrix, $\mathbf{G}$ is the gravitational term, and $\mathbf{C}$ is the centrifugal and Coriolis term. However, this equation is more difficult to obtain for parallel robots because the closure equations have to be satisfied. A classical method for calculating the dynamic models of closed-chain mechanisms is to consider first an equivalent tree structure, and then to enforce the kinematic constraints using Lagrange multipliers or d'Alembert's principle [18.113, 114]. Other approaches include the use of the principle of virtual work [18.115–118], the Lagrange formalism [18.119–122], Hamilton's principle [18.123], and Newton–Euler equations [18.124–129].

The main drawback of dynamic models obtained for parallel robots is that they are usually rather complex, they require the determination of dynamic parameters that are often not well known (dynamic identification of closed-loop mechanisms being not trivial [18.130]), and they usually involve solving the forward kinematics. Therefore, their calculation is computer intensive while they must be used in real time.

Using dynamic models for control purposes was proposed [18.131, 132], usually in the context of an adaptive control scheme, in which the tracking errors are used on-line to correct the parameters used in the dynamic equations [18.133, 134]. Control laws were proposed mainly for planar robots and for the Delta robot [18.135, 136], although some implementations have been proposed for general 6-DOF robots [18.137, 138] or vibration platforms [18.139]. However for 6-DOF robots, the benefits of using dynamic models for high-speed motions are difficult to establish because the computational burden of the dynamic model somewhat reduces the potential gains. An original approach is to work directly in the joint space, using information obtained on the legs' orientation through a vision system, allowing to get rid of the forward kinematics [18.140].

Dynamic control is an important and open issue since parallel robots can operate at velocities and accelerations much larger than those of serial robots. For example, some of the Delta robots have reached accelerations of the order of $500\,\mathrm{m/s^2}$ (with an objective of reaching $100\,\mathrm{G}$ [18.141]) while wire-driven robots can probably exceed this value.

Finally, the optimization of the dynamic properties of parallel mechanisms can be addressed by minimizing the variation of the inertia over the workspace [18.142].

18.9 Design

The design of a robot can be decomposed into two main phases:

- *Structural synthesis*: Finding the general mechanical structure of the robot.
- *Geometric synthesis*: Determining the value of the geometric parameters involved in a given structure (here geometric parameters has a loose sense: for example, mass, inertia may also be involved).

The problem of structural synthesis (type synthesis) was addressed in Sect. 18.2. However, performance requirements other than the motion pattern of the robot have to be taken into account in the design of a robot. Serial robots have the advantage that the number of possible mechanical architectures is relatively small and that some of these architectures have clear advantages in terms of performance compared to others (for example the workspace of a $3R$ structure is much larger than the workspace of a Cartesian robot of similar size).

Unfortunately, no such rules exist for parallel robots, for which there are furthermore a large number of possible mechanical designs. Additionally, the performances of parallel robots are very sensitive to their geometric parameters. For example, the extremal stiffness of a Gough platform over a given workspace can change by 700% for a change of only 10% of the platform radius. Consequently, structural synthesis cannot

be dissociated from the geometric synthesis. In fact, it is conjectured that a well-dimensioned robot of any structural type will, in general, perform better than a poorly designed robot with a structure that may seem more appropriate for the task at hand.

Usually, the design process is treated as an optimization problem. To each specified performance requirement is associated a performance index whose value increases with the level of violation of the performance requirement. These performance indices are summed in a weighted real-value function, called the *cost function*, which is a function of the geometric design parameters, and then a numerical optimization procedure is used to find the parameters that minimize the cost function (hence, this approach leads to what is called an *optimal design*) [18.143–145]. There are, however, numerous drawbacks to this approach: the determination and effect of the weights in the cost function are difficult to ascertain, imperative requirements are difficult to incorporate and complexify the optimization process, and the definition of the performance indices is not trivial, only to name a few. The main issues can be stated as follows:

- It is quite difficult to ensure that the global minima is reached in spite of using relatively new optimization methods such as neural network or genetic algorithm.
- The robustness of the design solution obtained with the cost function approach with respect to the uncertainties in the final design. Indeed, the real instantiation of a theoretical solution always differs from the theoretical solution itself because of the manufacturing tolerances and other uncertainties that are inherent to a mechanical system.
- Performance requirements may be antagonistic (e.g., workspace and accuracy), and the approach only provides a compromise between these requirements that is difficult to master through the weights.

An alternative to optimal design is referred to as *appropriate design*, in which no optimization is considered, but the objective is to ensure that desired requirements are satisfied. This approach is based on the definition of the *parameter space* in which each dimension is associated with a design parameter. Performance requirements are considered in turn and the regions of the parameter space which correspond to robots satisfying the requirements are calculated. The design solution is then obtained as the intersection of the regions obtained for each individual requirement.

In practice, only an approximation of the regions is necessary since values close to the boundary cannot be physically realized due to the manufacturing tolerances. For that calculation, interval analysis was successfully used in various applications [18.36, 146].

The appropriate design approach is clearly more complex to implement than the cost function approach but has the advantage of providing all design solutions, with the consequence that manufacturing tolerances may be taken into account to ensure that the real robot also satisfies the desired requirements.

18.10 Wire-Driven Parallel Robots

Parallel mechanisms naturally lead themselves to another paradigm that exploits the structural properties of wires (cables). Cables are mechanical components that can withstand large tensile loads relative to their weight. When applied to parallel mechanisms, a plurality of cables can be attached to a moving platform and driven by actuated reels mounted on the base frame. Because of the parallel architecture, it is possible to ensure that cables can be maintained under tension within a given workspace. Wire-driven parallel mechanisms have been studied by several researchers over the last 20 years (e.g., [18.8, 147] for early work). The advantages of these mechanisms are numerous. When wound on a spool, cables allow motion ranges that are much larger than that produced by conventional articulated systems. Also, because they can only resist to tensile forces, cables are, as mentioned above, much thinner and lighter than most conventional mechanical components, for similar payloads. Thus, they have very low inertia and are particularly suitable for systems in which large ranges of motion are required. Indeed, long-reach articulated robots are heavy and involve the motion of significant mass, whereas cable-driven robots remain light even when the range of motion is very large. This property is clearly exemplified by aerial video camera systems such as the SkyCam [18.148] or in VIDEO 44, which shows an application in the scanning of artifacts for the generation of digital three-dimensional (3-D) models. VIDEO 45 and VIDEO 50 show another application in material handling.

Because wires can only resist tensile loads, wire-driven parallel mechanisms are generally designed based on either one of the two following approaches, namely: (1) the weight (gravity) of the moving platform is used to ensure that all cables are under tension or (2) the number of cables is larger than the number of degrees of freedom of the platform, thereby allowing

a pre-stressing of the cables that can be used to ensure that all cables remain under tension. The wire-driven parallel mechanisms that are designed based on the first approach are referred to as *cable-suspended parallel mechanisms*, while the mechanisms that are based on the second approach are referred to as *fully constrained wire-driven parallel mechanisms*.

18.10.1 Cable-Suspended Parallel Mechanisms

Most cable-suspended parallel mechanisms are not redundantly actuated: they typically include as many actuators as DOFs. These mechanisms have been proposed in the literature as potential candidates for applications that require very large workspaces or mechanisms that can provide effective payload to mass ratios. One of the first cable-suspended mechanisms that was built is the Robocrane [18.8], developed by NIST almost two decades ago. This 6-DOF robot was intended for crane-type operations in which the pose of the payload can be fully controlled. Other cable-suspended mechanisms have also been studied and prototypes were built to validate their performance (e.g., [18.149] and [18.150]). More recently, cable-suspended parallel mechanisms including more actuators than DOFs were proposed, and it was shown that their workspace-to-footprint ratio can be much larger than that of nonredundant cable-suspended parallel mechanisms [18.151]. This is convincingly illustrated in VIDEO 45 and VIDEO 50, where a cable-suspended parallel mechanism with eight cables is demonstrated.

Most of the cable-suspended parallel mechanisms proposed or built in the past were assumed to work under static or quasi-static conditions. Under this assumption, the workspace can be determined based on the static equilibrium of the moving platform. The workspace is defined as the set of platform poses for which static equilibrium can be obtained while maintaining tension in all cables. Techniques to determine the static workspace of cable-suspended robots were proposed in the literature, for instance, in [18.152]. The dynamic trajectory planning and control of cable-suspended parallel mechanisms were also investigated in the recent literature (e.g., [18.153–156]), leading to cable-suspended parallel mechanisms that can be operated beyond their static workspace. The concept of dynamic workspace then arises [18.157], which can be defined as the set of poses that the platform can reach with at least one kinematic state (position, velocity, and acceleration).

Finally, it should be mentioned that the determination of the static pose of an underactuated cable-suspended parallel mechanism – that is, one with fewer cables than DOFs – is a very complex problem that has not yet been completely solved [18.158, 159]. In fact, for a given platform and given cable lengths, it is possible that an equilibrium pose can be reached, although some of the cables are not under tension. Therefore, the determination of the static pose involves not only solving for the actual number of cables, n, but also for all subrobots with $n-1$ cables.

18.10.2 Fully Constrained Wire-Driven Parallel Mechanisms

Fully constrained wire-driven parallel mechanisms have also attracted a great deal of attention. Fully constrained mechanisms include a number of cables larger than the number of degrees of freedom of the mechanism. If a proper geometry is used, they generally exhibit a wrench-closed workspace within which the platform can resist arbitrary external wrenches, while maintaining the cables in tension.

The statics of wire-driven robots can be described by

$$\mathbf{W}\boldsymbol{\tau} = \boldsymbol{f}\,, \tag{18.13}$$

where $\boldsymbol{\tau}$ is the vector of tensions in the wires, $\boldsymbol{f}$ is the wrench at the platform, and $\mathbf{W}$ is the so-called wrench matrix, which is configuration dependent. Since fully constrained wire-driven parallel mechanisms include more cables than DOFs, their wrench matrix $\mathbf{W}$ contains more columns than rows. Based on the above equation, the wrench-closed workspace can then be defined as the set of poses for which there exists at least one vector in the nullspace of $\mathbf{W}$ that has all its components of the same sign. When this condition is satisfied, the mechanism is said to be tensionable [18.160] and can theoretically resist arbitrary external wrenches simply by increasing the pre-stress in the cables.

The determination and the optimization of the wrench-closure workspace is a challenging problem that has attracted significant attention. Although the early work presented in [18.149, 161, 162] was mainly applicable to planar wire-driven robots, it paved the way to more general analyses by providing a mathematical modelling of the problem. Determining whether a given pose of the platform is located within the wrench-closure workspace is a rather straightforward problem. However, discretizing the workspace and applying such a criterion to a mesh of points does not solve the problem satisfactorily because the wrench-closed workspace is generally not convex and may even be composed of disconnected components. Therefore, more powerful techniques are needed in order to obtain reliable and useful results. In [18.163], in-

terval analysis is used to determine the wrench-closed workspace of fully constrained wire-driven parallel robots while providing definite answers. VIDEO 45 and VIDEO 50 show an example of fully constrained wire-driven parallel mechanisms.

18.10.3 Design and Prototyping

Synthesizing a wire-driven parallel mechanism for a prescribed wrench-closed workspace is a very challenging issue [18.164]. Also, in addition to the workspace, several other issues must be considered in the design of wire-driven parallel robots, for example, the determination of the potential interferences between the wires. An elegant geometric solution to this problem is proposed in [18.165] but handling translations and rotations simultaneously remains a challenge. Other design problems include the handling of special configurations and assembly mode changes as well as the accuracy and the practical implementation of the spools [18.166].

Several prototypes of wire-driven parallel robots were built and reported in the literature. Early prototypes include the NIST Robocrane [18.8], the Falcon robot [18.147], and the skycam system [18.148]. Many others were built, including those shown in the videos accompanying this chapter and the potential applications are numerous.

18.11 Application Examples

Parallel robots have been successfully used in many applications, a few of which are now briefly mentioned. Almost all recent land-based telescopes use parallel robots, either as a secondary mirror alignment system (e.g., the University of Arizona MMT or the ESO VISTA) or as a primary mirror pointing device [18.6]. A wire-driven parallel robot flew in the space shuttle mission STS-63 in February 1999 while an octopod (a parallel robot with eight legs) was used to isolate the space shuttle payload from vibration. All flight simulators use a parallel structure as a motion platform. They are nowadays used for driving simulators as well (e.g., the NADS-driving simulator [18.167]). In industry, numerous machine tools based on parallel structures have been designed. Some of them have found a niche market (e.g., the Tricept) and it can be expected that more will be used in the future. Ultra accurate positioning devices based on parallel robots are proposed by companies, such as Physik Instrumente, Alio, Micos, and Symétrie. In the food industry, the Delta robot proposed by ABB and Demaurex is widely used for fast packaging. Other companies such as Adept are also developing fast parallel robots with 3- and 4-DOFs. Medical applications are also growing, for example, in rehabilitation (after the pioneering work of the Rutgers ankle [18.168] and the platform Caren of Motek), in surgery (for example for spine surgery with the Renaissance robot of Mazor or for vitreoretinal surgery [18.169]) or for training and monitoring the motion of human joints [18.170], a topic in which cable-driven robots play an important role, as they do for aerial robots [18.171] or rescue operation (a version of the SkyCam robot has been used to fight oil fires during the Iraqi War).

18.12 Conclusion and Further Reading

The analysis of a parallel mechanism may be partly based on methods that are presented in other chapters of this handbook:

- *Kinematics*: Kinematics background is covered in Chap. 2.
- *Dynamics*: General approaches are presented in Chap. 3 while identification of the dynamic parameters is addressed in Chap. 6.
- *Design*: Design methodologies are covered in Chap. 16.
- *Control*: Control issues are presented in Chaps. 7, 8, and 9 although the closed-loop structure of parallel robots may require some adaptation of the control schemes.

It must also be emphasized that efficient numerical analysis is a key point for many algorithms related to parallel robots. A system solving with Gröbner basis, continuation method, and interval analysis is essential for kinematics, workspace, and singularity analysis. Further information and up-to-date extensive references and papers related to parallel robots can be found on the following two websites [18.38, 100]. Useful complementary readings on parallel robots are [18.140, 147, 164]. Parallel robots are slowly finding their way into various applications, not only in industry but also in field and service robotics, as discussed in Part F of this handbook. Still, compared to their serial counterpart, their analysis is far from being complete.

Video-References

VIDEO 43 3-DOF high-speed 3-RPS parallel robot
available from http://handbookofrobotics.org/view-chapter/18/videodetails/43
VIDEO 44 6-DOF cable-suspended robot
available from http://handbookofrobotics.org/view-chapter/18/videodetails/44
VIDEO 45 CoGiRo
available from http://handbookofrobotics.org/view-chapter/18/videodetails/45
VIDEO 46 Parallel 5R robot
available from http://handbookofrobotics.org/view-chapter/18/videodetails/46
VIDEO 47 Diamond
available from http://handbookofrobotics.org/view-chapter/18/videodetails/47
VIDEO 48 6-DOF statically balanced parallel robot
available from http://handbookofrobotics.org/view-chapter/18/videodetails/48
VIDEO 49 3-DOF dynamically balanced parallel robot
available from http://handbookofrobotics.org/view-chapter/18/videodetails/49
VIDEO 50 IPAnema
available from http://handbookofrobotics.org/view-chapter/18/videodetails/50
VIDEO 51 Par2 robot
available from http://handbookofrobotics.org/view-chapter/18/videodetails/51
VIDEO 52 Quadrupteron robot
available from http://handbookofrobotics.org/view-chapter/18/videodetails/52
VIDEO 53 R4 robot
available from http://handbookofrobotics.org/view-chapter/18/videodetails/53
VIDEO 54 Tripteron robot
available from http://handbookofrobotics.org/view-chapter/18/videodetails/54

References

18.1 J.E. Gwinnett: Amusement device, US Patent 178 9680 (1931)
18.2 V.E. Gough: Contribution to discussion of papers on research in automobile stability, control and tyre performance, 1956–1957. Proc. Auto Div. Inst. Mech. Eng. (1956)
18.3 D. Stewart: A platform with 6 degrees of freedom, Proc. Inst. Mech. Eng. Part 1 15 (1965) pp. 371–386
18.4 I.A. Bonev: The true origins of parallel robots, http://www.parallemic.org/Reviews/Review007.html (2003)
18.5 S. Briot, I.A. Bonev: Are parallel robots more accurate than serial robots?, CSME Transactions **31**(4), 445–456 (2007)
18.6 G. Aridon, D. Rémond, F. Morestin, L. Blanchard, R. Dufour: Self-deployment of a tape-spring hexapod: experimental and numerical investigation, ASME J. Mech. Des. **131**(2), 0210031–0210037 (2009)
18.7 M. Hafez, M.D. Lichter, S. Dubowsky: Optimized binary modular reconfigurable robotic devices, IEEE Trans. Mechatron. **8**(1), 152–162 (2003)
18.8 J. Albus, R. Bostelman, N. Dagalakis: The NIST robocrane, J. Robotic Syst. **10**(5), 709–724 (1993)
18.9 R. Clavel: DELTA, a fast robot with parallel geometry, 18th Int. Symp. Ind. Robots (ISIR), Lausanne (1988) pp. 91–100
18.10 K.H. Hunt: Structural kinematics of in parallel actuated robot arms, J. Mech. Transm. Autom. Des. **105**(4), 705–712 (1983)
18.11 K.E. Neumann: Robot: Neos Product HB Norrtalje Suède. US Patent 4732525, (1988)
18.12 Q. Li, J.M. Hervè: Parallel mechanisms with bifurcation of Schoenflies motion, IEEE Trans. Robotics **25**(1), 158–164 (2009)
18.13 J.M. Hervé: Group mathematics and parallel link mechanisms, 9th IFToMM World Congr. Theory Mach. Mech., Milan (1995) pp. 2079–2082
18.14 C.-C. Lee, J.M. Hervè: Type synthesis of primitive Schönflies-motion generators, Mech. Mach. Theory **44**(10), 1980–1997 (2009)
18.15 J. Meng, G.F. Liu, Z. Li: A geometric theory for analysis and synthesis of sub-6 DOF parallel manipulators, IEEE Trans. Robotics **23**(4), 625–649 (2007)
18.16 X. Kong, C.M. Gosselin: *Type Synthesis of Parallel Mechanisms*, Springer Tracts in Advanced Robotics, Vol. 33 (Springer, Berlin, Heidelberg 2007)
18.17 J.P. Merlet: http://www.sop.inria.fr/members/Jean-Pierre.Merlet//Archi/archi_robot.html (INRIA, France)
18.18 F. Pierrot, V. Nabat, O. Company, S. Krut, P. Poignet: Optimal design of a 4-DOF parallel manipulator: From academia to industry, IEEE Trans. Robotics **25**(2), 213–224 (2009)
18.19 C. Gosselin, T. Lalibertè, A. Veillette: Singularity-free kinematically redundant planar parallel mechanisms with unlimited rotational capability, IEEE Trans. Robotics **31**(2), 457–467 (2015)

18.20 G. Gogu: *Structural Synthesis of Parallel Robots* (Kluwer, Dordrecht 2007)

18.21 P. Dietmaier: The Stewart Gough platform of general geometry can have 40 real postures. In: *Advances in Robot Kinematics: Analysis Control*, ed. by J. Lenarčič, M.L. Husty (Kluwer, Dordrecht 1998) pp. 7–16

18.22 M.L. Husty: An algorithm for solving the direct kinematic of Stewart Gough-type platforms, Mech. Mach. Theory **31**(4), 365–380 (1996)

18.23 M. Raghavan: The Stewart platform of general geometry has 40 configurations, ASME J. Mech. Des. **115**(2), 277–282 (1993)

18.24 J.C. Faugère, D. Lazard: The combinatorial classes of parallel manipulators, Mech. Mach. Theory **30**(6), 765–776 (1995)

18.25 T.-Y. Lee, J.-K. Shim: Improved dyalitic elimination algorithm for the forward kinematics of the general Stewart Gough platform, Mech. Mach. Theory **38**(6), 563–577 (2003)

18.26 F. Rouillier: Real roots counting for some robotics problems. In: *Computational Kinematics*, ed. by J.-P. Merlet, B. Ravani (Kluwer, Dordrecht 1995) pp. 73–82

18.27 J.-P. Merlet: Solving the forward kinematics of a Gough-type parallel manipulator with interval analysis, Int. J. Robotics Res. **23**(3), 221–236 (2004)

18.28 D. Manocha: Algebraic and Numeric Techniques for Modeling and Robotics, Ph.D. Thesis (University of California, Berkeley 1992)

18.29 A.J. Sommese, C.W. Wampler: *The Numerical Solution of Systems of Polynomials: Arising in Engineering And Science* (World Scientific, Singapore 2005)

18.30 C. Gosselin, J. Sefrioui, M.J. Richard: Solution polynomiale au problème de la cinématique directe des manipulateurs parallèles plans à 3 degrés de liberté, Mech. Mach. Theory **27**(2), 107–119 (1992)

18.31 P. Wenger, D. Chablat, M. Zein: Degeneracy study of the forward kinematics of planar 3 — *RPR* parallel manipulators, ASME J. Mech. Des. **129**(12), 1265–1268 (2007)

18.32 L. Baron, J. Angeles: The direct kinematics of parallel manipulators under joint-sensor redundancy, IEEE Trans. Robotics Autom. **16**(1), 12–19 (2000)

18.33 J.-P. Merlet: Closed-form resolution of the direct kinematics of parallel manipulators using extra sensors data, IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 200–204

18.34 R. Vertechy, V. Parenti-Castelli: Robust, fast and accurate solution of the direct position analysis of parallel manipulators by using extra-sensors. In: *Parallel Manipulators: Towards New Applications*, ed. by H. Wu (InTech, Rijeka 2008) pp. 133–154

18.35 R. Stoughton, T. Arai: Kinematic optimization of a chopsticks-type micro-manipulator, Jpn. Symp. Flex. Autom., San Fransisco (1993) pp. 151–157

18.36 J.-P. Merlet, D. Daney: Dimensional synthesis of parallel robots with a guaranteed given accuracy over a specific workspace, IEEE Int. Conf. Robotics Autom. (ICRA), Barcelona (2005)

18.37 J.-P. Merlet: Jacobian, manipulability, condition number, and accuracy of parallel robots, ASME J. Mech. Des. **128**(1), 199–206 (2006)

18.38 F.-T. Niaritsiry, N. Fazenda, R. Clavel: Study of the source of inaccuracy of a 3 DOF flexure hinge-based parallel manipulator, IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 4091–4096

18.39 G. Pritschow, C. Eppler, T. Garber: Influence of the dynamic stiffness on the accuracy of PKM, 3rd Chemnitzer Parallelkinematik Semin. Chemnitz (2002) pp. 313–333

18.40 V. Parenti-Castelli, S. Venanzi: On the joint clearance effects in serial and parallel manipulators, Workshop Fundam. Issues Future Res. Dir. Parallel Mech. Manip. Québec (2002) pp. 215–223

18.41 A. Pott, M. Hiller: A new approach to error analysis in parallel kinematic structures, Adv. Robot Kinemat., Sestri-Levante (2004)

18.42 K. Wohlhart: Degrees of shakiness, Mech. Mach. Theory **34**(7), 1103–1126 (1999)

18.43 K. Tönshoff, B. Denkena, G. Günther, H.C. Möhring: Modelling of error effects on the new hybrid kinematic DUMBO structure, 3rd Chemnitzer Parallelkinematik Semin. Chemnitz (2002) pp. 639–653

18.44 U. Sellgren: Modeling of mechanical interfaces in a systems context, Int. ANSYS Conf., Pittsburgh (2002)

18.45 S. Eastwood, P. Webbb: Compensation of thermal deformation of a hybrid parallel kinematic machine, Robotics Comput. Manuf. **25**(1), 81–90 (2009)

18.46 W. Khalil, S. Besnard: Identificable parameters for the geometric calibration of parallel robots, Arch. Control Sci. **11**(3/4), 263–277 (2001)

18.47 C.W. Wampler, J.M. Hollerbach, T. Arai: An implicit loop method for kinematic calibration and its application to closed-chain mechanisms, IEEE Trans. Robotics Autom. **11**(5), 710–724 (1995)

18.48 D. Daney, Y. Papegay, A. Neumaier: Interval methods for certification of the kinematic calibration of parallel robots, IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 1913–1918

18.49 D. Daney, N. Andreff, G. Chabert, Y. Papegay: Interval method for calibration of parallel robots: a vision-based experimentation, Mech. Mach. Theory **41**(8), 929–944 (2006)

18.50 A. Nahvi, J.M. Hollerbach: The noise amplification index for optimal pose selection in robot calibration, IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (1996) pp. 647–654

18.51 G. Meng, L. Tiemin, Y. Wensheng: Calibration method and experiment of Stewart platform using a laser tracker, Int. Conf Syst. Man Cybern., Hague (2003) pp. 2797–2802

18.52 D. Daney: Optimal measurement configurations for Gough platform calibration, IEEE Int. Conf. Robotics Autom. (ICRA), Washington (2002) pp. 147–152

18.53 P. Renaud, N. Andreff, P. Martinet, G. Gogu: Kinematic calibration of parallel mechanisms: A novel approach using legs observation, IEEE Trans. Robotics **21**(4), 529–538 (2005)

18.54 C. Gosselin, J. Angeles: Singularity analysis of closed-loop kinematic chains, IEEE Trans. Robotics Autom. **6**(3), 281–290 (1990)

18.55 D. Zlatanov, R.G. Fenton, B. Benhabib: A unifying framework for classification and interpretation of mechanism singularities, ASME J. Mech. Des. **117**(4), 566–572 (1995)

18.56 D. Zlatanov, I.A. Bonev, C.M. Gosselin: Constraint singularities of parallel mechanisms, IEEE Int. Conf. Robotics Autom. (ICRA), Washington (2002) pp. 496–502

18.57 M. Conconi, M. Carricato: A new assessment of singularities of parallel kinematic chains, IEEE Trans. Robotics **25**(4), 757–770 (2009)

18.58 G. Liu, Y. Lou, Z. Li: Singularities of parallel manipulators: A geometric treatment, IEEE Trans. Robotics Autom. **19**(4), 579–594 (2003)

18.59 I.A. Bonev, D. Zlatanov: The mystery of the singular SNUtranslational parallel robot, http://www.parallemic.org/Reviews/Review004.html (2001)

18.60 H. Li, C.M. Gosselin, M.J. Richard, B. Mayer St-Onge: Analytic form of the six-dimensional singularity locus of the general Gough-Stewart platform, ASME J. Mech. Des. **128**(1), 279–287 (2006)

18.61 B. Mayer St-Onge, C.M. Gosselin: Singularity analysis and representation of the general Gough-Stewart platform, Int. J. Robotics Res. **19**(3), 271–288 (2000)

18.62 J.-P. Merlet: Singular configurations of parallel manipulators and Grassmann geometry, Int. J. Robotics Res. **8**(5), 45–56 (1989)

18.63 R. Ben-Horin, M. Shoham: Application of Grassmann Cayley algebra to geometrical interpretation of parallel robot singularities, Int. J. Robotics Res. **28**(1), 127–141 (2009)

18.64 H. Pottmann, M. Peternell, B. Ravani: Approximation in line space. Applications in robot kinematics, Adv. Robot Kinemat., Strobl (1998) pp. 403–412

18.65 P.A. Voglewede, I. Ebert-Uphoff: Measuring *closeness* to singularities for parallel manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 4539–4544

18.66 G. Nawratil: New performance indices for 6-DOF UPS and 3-DOF RPR parallel manipulators, Mech. Mach. Theory **44**(1), 208–221 (2009)

18.67 J. Hubert, J.-P. Merlet: Static of parallel manipulators and closeness to singularity, J. Mech. Robotics **1**(1), 1–6 (2009)

18.68 P. Cardou, S. Bouchard, C. Gosselin: Kinematic-sensitivity indices for dimensionally nonhomogeneous jacobian matrices, IEEE Trans. Robotics **26**(1), 166–173 (2010)

18.69 J.-P. Merlet, D. Daney: A formal-numerical approach to determine the presence of singularity within the workspace of a parallel robot. In: *Computational Kinematics*, ed. by F.C. Park, C.C. Iurascu (EJCK, Seoul 2001) pp. 167–176

18.70 S. Bhattacharya, H. Hatwal, A. Ghosh: Comparison of an exact and an approximate method of singularity avoidance in platform type parallel manipulators, Mech. Mach. Theory **33**(7), 965–974 (1998)

18.71 D.N. Nenchev, M. Uchiyama: Singularity-consistent path planning and control of parallel robot motion through instantaneous-self-motion type, IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (1996) pp. 1864–1870

18.72 C. Innocenti, V. Parenti-Castelli: Singularity-free evolution from one configuration to another in serial and fully-parallel manipulators, 22nd Bienn. Mech. Conf., Scottsdale (1992) pp. 553–560

18.73 M. Husty: Non-singular assembly mode change in 3-RPR parallel manipulators. In: *Computational Kinematics*, ed. by A. Kecskeméthy, A. Müller (Springer, Berlin, Heidelberg 2009) pp. 51–60

18.74 S. Caro, P. Wenger, D. Chablat: Non-singular assembly mode changing trajectories of a 6-DOF parallel robot, ASME Int. Des. Eng. Techn. Conf. Comput. Inform. Eng. Conf., Chicago (2012)

18.75 R. Ranganath, P.S. Nair, T.S. Mruthyunjaya, A. Ghosal: A force-torque sensor based on a Stewart platform in a near-singular configuration, Mech. Mach. Theory **39**(9), 971–998 (2004)

18.76 M.L. Husty, A. Karger: Architecture singular parallel manipulators and their self-motions. In: *Adv. Robot Kinematics: Analysis Control*, ed. by J. Lenarčič, M.L. Husty (Springer, Dordrecht 2000) pp. 355–364

18.77 A. Karger: Architecture singular planar parallel manipulators, Mech. Mach. Theory **38**(11), 1149–1164 (2003)

18.78 K. Wohlhart: Mobile 6-SPS parallel manipulators, J. Robotic Syst. **20**(8), 509–516 (2003)

18.79 C. Innocenti, V. Parenti-Castelli: Direct kinematics of the 6-4 fully parallel manipulator with position and orientation uncoupled, Eur. Robotics Intell. Syst. Conf., Corfou (1991)

18.80 G. Gogu: Mobility of mechanisms: A critical review, Mech. Mach. Theory **40**(10), 1068–1097 (2005)

18.81 I. Zabalza, J. Ros, J.J. Gil, J.M. Pintor, J.M. Jimenez: Tri-Scott a new kinematic structure for a 6-DOF decoupled parallel manipulator, Workshop Fundam. Issues Future Res. Dir. Parallel Mech. Manip., Québec (2002) pp. 12–15

18.82 C. Gosselin: Determination of the workspace of 6-DOF parallel manipulators, ASME J. Mech. Des. **11**(3), 331–336 (1990)

18.83 J.-P. Merlet: Geometrical determination of the workspace of a constrained parallel manipulator, Adv. Robot Kinemat., Ferrare (1992) pp. 326–329

18.84 F.A. Adkins, E.J. Haug: Operational envelope of a spatial Stewart platform, ASME J. Mech. Des. **119**(2), 330–332 (1997)

18.85 E.J. Haugh, F.A. Adkins, C.M. Luh: Operational envelopes for working bodies of mechanisms and manipulators, ASME J. Mech. Des. **120**(1), 84–91 (1998)

18.86 O. Bohigas, L. Ros, M. Manubens: A complete method for workspace boundary determination, Adv. Robot Kinemat., Piran (2010) pp. 329–338

18.87 J.-P. Merlet: Determination of 6D workspaces of Gough-type parallel manipulator and comparison between different geometries, Int. J. Robotics Res. **18**(9), 902–916 (1999)

18.88 P. Wenger, D. Chablat: Workspace and assembly modes in fully parallel manipulators: A descriptive study. In: *Advances in Robot Kinematics: Analysis Control*, ed. by J. Lenarčič, M.L. Husty (Springer, Dordrecht 1998) pp. 117–126

18.89 G. Moroz, F. Rouiller, D. Chablat, P. Wenger: On the determination of cusp points of 3-*RPR* parallel manipulators, Mech. Mach. Theory **45**(11), 1555–1567 (2010)

18.90 S. Briot, V. Arakelian: Optimal force generation in parallel manipulators for passing through the singular positions, Int. J. Robotics Res. **27**(2), 967–983 (2008)

18.91 J. Hesselbach, C. Bier, C. Budde, P. Last, J. Maaß, M. Bruhn: Parallel robot specific control fonctionalities, 2nd Int. Colloquium, Collab. Res. Centre, Braunschweig (2005) pp. 93–108

18.92 J.-P. Merlet: An efficient trajectory verifier for motion planning of parallel machine, Parallel Kinemat. Mach. Int. Conf., Ann Arbor (2000)

18.93 R. Ur-Rehman, S. Caro, D. Chablat, P. Wenger: Multi-objective path placement of parallel kinematics machines based on energy consumption, shaking forces and maximum actuator torques: Application to the Orthoglide, Mech. Mach. Theory **45**(8), 1125–1141 (2010)

18.94 J. Cortés, T. Siméon: Probabilistic motion planning for parallel mechanisms, IEEE Int. Conf. Robotics Autom. (ICRA), Taipei (2003) pp. 4354–4359

18.95 J.H. Yakey, S.M. LaValle, L.E. Kavraki: Randomized path planning for linkages with closed kinematic chains, IEEE Trans. Robotics Autom. **17**(6), 951–958 (2001)

18.96 J.-P. Merlet, M.-W. Perng, D. Daney: Optimal trajectory planning of a 5-axis machine tool based on a 6-axis parallel manipulator. In: *Advances in Robot Kinematics*, ed. by J. Lenarčič, M.L. Husty (Kluwer, Dordrecht 2000) pp. 315–322

18.97 D. Shaw, Y.-S. Chen: Cutting path generation of the Stewart platform-based milling machine using an end-mill, Int. J. Prod. Res. **39**(7), 1367–1383 (2001)

18.98 Z. Wang, Z. Wang, W. Liu, Y. Lei: A study on workspace, boundary workspace analysis and workpiece positioning for parallel machine tools, Mech. Mach. Theory **36**(6), 605–622 (2001)

18.99 D.R. Kerr: Analysis, properties, and design of a Stewart-platform transducer, J. Mech. Transm. Autom. Des. **111**(1), 25–28 (1989)

18.100 C.C. Nguyen, S.S. Antrazi, Z.L. Zhou, C.E. Campbell: Analysis and experimentation of a Stewart platform-based force/torque sensor, Int. J. Robotics Autom. **7**(3), 133–141 (1992)

18.101 C. Reboulet, A. Robert: Hybrid control of a manipulator with an active compliant wrist, 3rd Int. Symp. Robotics Res., Gouvieux (1985) pp. 76–80

18.102 J. Duffy: *Statics and Kinematics with Applications to Robotics* (Cambridge Univ. Press, New York 1996)

18.103 C. Huang, W.-H. Hung, I. Kao: New conservative stiffness mapping for the Stewart Gough platform, IEEE Int. Conf. Robotics Autom. (ICRA), Washington (2002) pp. 823–828

18.104 J.L. Herder: Energy-free systems: Theory, Conception and Design of Statically Balanced Spring Mechanisms, Ph.D. Thesis (Delft University of Technology, Delft 2001)

18.105 G.R. Dunlop, T.P. Jones: Gravity counter balancing of a parallel robot for antenna aiming, 6th Int. Symp. Robotics Manuf., Montpellier (1996) pp. 153–158

18.106 M. Jean, C. Gosselin: Static balancing of planar parallel manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (1996) pp. 3732–3737

18.107 I. Ebert-Uphoff, C.M. Gosselin, T. Laliberté: Static balancing of spatial parallel platform-revisited, ASME J. Mech. Des. **122**(1), 43–51 (2000)

18.108 C.M. Gosselin, J. Wang: Static balancing of spatial six-degree-of-freedom parallel mechanisms with revolute actuators, J. Robotic Syst. **17**(3), 159–170 (2000)

18.109 M. Leblond, C.M. Gosselin: Static balancing of spatial and planar parallel manipulators with prismatic actuators, ASME Des. Eng. Tech. Conf., Atlanta (1998) pp. 5187–5193

18.110 B. Monsarrat, C.M. Gosselin: Workspace analysis and optimal design of a 3-leg 6-DOF parallel platform mechanism, IEEE Trans. Robotics Autom. **19**(6), 954–966 (2003)

18.111 J. Wang, C.M. Gosselin: Static balancing of spatial three-degree-of-freedom parallel mechanisms, Mech. Mach. Theory **34**(3), 437–452 (1999)

18.112 Y. Wu, C.M. Gosselin: Synthesis of reactionless spatial 3-DOF and 6-DOF mechanisms without separate counter-rotations, Int. J. Robotics Res. **23**(6), 625–642 (2004)

18.113 M. Ait-Ahmed: Contribution à la Modélisation Géométrique et Dynamique des Robots Parallèles, Ph.D. Thesis (Univ. Paul Sabatier, Toulouse 1993)

18.114 G.F. Liu, X.Z. Wu, Z.X. Li: Inertial equivalence principle and adaptive control of redundant parallel manipulators, IEEE Int. Conf. Robotics Autom. (ICRA), Washington (2002) pp. 835–840

18.115 R. Clavel: Conception d'un robot parallèle rapide à 4 degrés de liberté, Ph.D. Thesis (EPFL, Lausanne, 1991), No. 925.

18.116 J. Gallardo, J.M. Rico, A. Frisoli, D. Checcacci, M. Bergamasco: Dynamics of parallel manipulators by means of screw theory, Mech. Mach. Theory **38**(11), 1113–1131 (2003)

18.117 L.-W. Tsai: Solving the inverse dynamics of a Stewart Gough manipulator by the principle

of virtual work, ASME J. Mech. Des. **122**(1), 3–9 (2000)
18.118 J. Wang, C.M. Gosselin: A new approach for the dynamic analysis of parallel manipulators, Multibody Syst. Dyn. **2**(3), 317–334 (1998)
18.119 Z. Geng, L.S. Haynes: On the dynamic model and kinematic analysis of a class of Stewart platforms, Robotics Auton. Syst. **9**(4), 237–254 (1992)
18.120 K. Liu, F. Lewis, G. Lebret, D. Taylor: The singularities and dynamics of a Stewart platform manipulator, J. Intell. Robotic Syst. **8**(3), 287–308 (1993)
18.121 K. Miller, R. Clavel: The Lagrange-based model of Delta-4 robot dynamics, Robotersysteme **8**(1), 49–54 (1992)
18.122 H. Abdellatif, B. Heimann: Computational efficient inverse dynamics of 6-DOF fully parallel manipulators by using the Lagrangian formalism, Mech. Mach. Theory **44**(1), 192–207 (2009)
18.123 K. Miller: Optimal design and modeling of spatial parallel manipulators, Int. J. Robotics Res. **23**(2), 127–140 (2004)
18.124 A. Codourey, E. Burdet: A body oriented method for finding a linear form of the dynamic equations of fully parallel robot, IEEE Int. Conf. Robotics Autom. (ICRA), Albuquerque (1997) pp. 1612–1618
18.125 B. Dasgupta, P. Choudhury: A general strategy based on the Newton Euler approach for the dynamic formulation of parallel manipulators, Mech. Mach. Theory **34**(6), 801–824 (1999)
18.126 P. Guglielmetti: Model-Based Control of Fast Parallel Robots: a Global Approach in Operational Space, Ph.D. Thesis (EPFL, Lausanne, 1994)
18.127 K. Harib, K. Srinivasan: Kinematic and dynamic analysis of Stewart platform-based machine tool structures, Robotica **21**(5), 541–554 (2003)
18.128 W. Khalil, O. Ibrahim: General solution for the dynamic modeling of parallel robots, IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 3665–3670
18.129 C. Reboulet, T. Berthomieu: Dynamic model of a six degree of freedom parallel manipulator, Int. Conf. Adv. Robotics, Pise (1991) pp. 1153–1157
18.130 H. Abdellatif, B. Heimann: Experimental identification of the dynamics model for 6-DOF parallel manipulators, Robotica **28**(3), 359–368 (2010)
18.131 H. Abdellatif, B. Heimann: Model based control for industrial robots: Uniform approaches for serial and parallel structures. In: *Industrial Robotics: Theory, Modelling and Control*, ed. by S. Cubero (pro literatur Verlag, Augsburg 2007) pp. 523–556
18.132 S. Tadokoro: Control of parallel mechanisms, Adv. Robotics **8**(6), 559–571 (1994)
18.133 M. Honegger, A. Codourey, E. Burdet: Adaptive control of the Hexaglide, a 6 DOF parallel manipulator, IEEE Int. Conf. Robotics Autom. (ICRA), Albuquerque (1997) pp. 543–548
18.134 S. Bhattacharya, H. Hatwal, A. Ghosh: An online estimation scheme for generalized Stewart platform type parallel manipulators, Mech. Mach. Theory **32**(1), 79–89 (1997)
18.135 P. Guglielmetti, R. Longchamp: A closed-form inverse dynamics model of the Delta parallel robot, 4th IFAC Symp. Robot Control, Syroco, Capri (1994) pp. 51–56
18.136 K. Miller: Modeling of dynamics and model-based control of DELTA direct-drive parallel robot, J. Robotics Mechatron. **17**(4), 344–352 (1995)
18.137 E. Burdet, M. Honegger, A. Codourey: Controllers with desired dynamic compensation and their implementation on a 6 DOF parallel manipulator, IEEE Int. Conf. Intell. Robots Syst. (IROS), Takamatsu (2000)
18.138 K. Yamane, Y. Nakamura, M. Okada, N. Komine, K.I. Yoshimoto: Parallel dynamics computation and h_∞ acceleration control of parallel manipulators for acceleration display, ASME J. Dyn. Syst. Meas. Control **127**(2), 185–191 (2005)
18.139 J.E. McInroy: Modeling and design of flexure jointed Stewart platforms for control purposes, IEEE/ASME Trans. Mechatron. **7**(1), 95–99 (2002)
18.140 F. Paccot, N. Andreff, P. Martinet: A review on the dynamic control of parallel kinematic machines: theory and experiments, Int. J. Robotics Res. **28**(3), 395–416 (2009)
18.141 D. Corbel, M. Gouttefarde, O. Company, F. Pierrot: Towards 100G with PKM is actuation redundancy a good solution for pick-and-place?, IEEE Int. Conf. Robotics Autom. (ICRA), Anchorage (2010) pp. 4675–4682
18.142 F. Xi: Dynamic balancing of hexapods for high-speed applications, Robotica **17**(3), 335–342 (1999)
18.143 J. Angeles: The robust design of parallel manipulators, 1st Int. Colloquium, Collab. Res. Centre, Braunschweig (2002) pp. 9–30
18.144 S. Bhattacharya, H. Hatwal, A. Ghosh: On the optimum design of a Stewart platform type parallel manipulators, Robotica **13**(2), 133–140 (1995)
18.145 K.E. Zanganeh, J. Angeles: Kinematic isotropy and the optimum design of parallel manipulators, Int. J. Robotics Res. **16**(2), 185–197 (1997)
18.146 H. Fang, J.-P. Merlet: Multi-criteria optimal design of parallel manipulators based on interval analysis, Mech. Mach. Theory **40**(2), 151–171 (2005)
18.147 S. Kamamura, W. Choe, S. Tanaka, S.R. Pandian: Development of an Ultrahigh Speed Robot FALCON using Wire Drive System, IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 215–220
18.148 L.L. Cone: Skycam: An Aerial Robotic Camera System, Byte **10**(10), 122–132 (1985)
18.149 J. Pusey, A. Farrah, S.K. Agrawal, E. Messina: Design and workspace analysis of a 6-6 cable-suspended parallel robot, Mech. Mach. Theory **39**, 761–778 (2004)
18.150 C. Gosselin, S. Bouchard: A gravity-powered mechanism for extending the workspace of a cable-driven parallel mechanism: Application to the appearance modelling of objects, Int. J. Autom. Technol. **4**(4), 372–379 (2010)

18.151 J. Lamaury, M. Gouttefarde: A tension distribution method with improved computational efficiency. In: *Cable-Driven Parallel Robots*, ed. by T. Bruckmann, A. Pott (Springer, Berlin, Heidelberg 2012) pp. 71–85

18.152 A.T. Riechel, I. Ebert-Uphoff: Force-feasible workspace analysis for underconstrained,point-mass cable robots, IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4956–4962

18.153 D. Cunningham, H. Asada: The Winch-Bot: A cable-suspended, under-actuated robot utilizing parametric self-excitation, IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 1844–1850

18.154 S. Lefrançois, C. Gosselin: Point-to-point motion control of a pendulum-like 3-DOF underactuated cable-driven robot, IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 5187–5193

18.155 D. Zanotto, G. Rosati, S.K. Agrawal: Modeling and control of a 3-DOF pendulum-like manipulator, IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 3964–3969

18.156 C. Gosselin, P. Ren, S. Foucault: Dynamic trajectory planning of a two-DOF cable-suspended parallel robot, IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1476–1481

18.157 G. Barrette, C. Gosselin: Determination of the dynamic workspace of cable-driven planar parallel mechanisms, ASME J. Mech. Des. **127**(2), 242–248 (2005)

18.158 M. Carricato, J.-P. Merlet: Direct geometrico-static problem of under-constrained cable-driven parallel robots with three cables, IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 3011–3017

18.159 J.-F. Collard, P. Cardou: Computing the lowest equilibrium pose of a cable-suspended rigid body, Optim. Eng. **14**, 457–476 (2013)

18.160 S. Behzadipour, A. Khajepour: Trajectory planning in cable-based high-speed parallel robots, IEEE Trans. Robotics **22**(3), 559–563 (2006)

18.161 M. Gouttefarde, C. Gosselin: Analysis of the wrench-closure workspace of planar parallel cable-driven mechanisms, IEEE Trans. Robotics **22**(3), 434–445 (2006)

18.162 E. Stump, V. Kumar: Workspace delienation of cable-actuated parallel manipulators, ASME Int. Des. Eng. Tech. Conf. (2004)

18.163 M. Gouttefarde, D. Daney, J.-P. Merlet: Interval-analysis-based determination of the wrench-feasible workspace of parallel cable-driven robots, IEEE Trans. Robotics **27**(1), 1–13 (2011)

18.164 K. Azizian, P. Cardou: The dimensional synthesis of planar parallel cable-driven mechanisms through convex relaxations, ASME J. Mech. Des. **4**(3), 0310111–03101113 (2012)

18.165 S. Perreault, P. Cardou, C. Gosselin, M.J.D. Otis: Geometric determination of the interference-free constant-orientation workspace of parallel cable-driven mechanisms, ASME J. Mech. Robotics **2**(3), 031016 (2010)

18.166 J.-P. Merlet: MARIONET, a family of modular wire-driven parallel robots, Adv. Robot Kinemat.: Motion Man Mach. (2010) pp. 53–61

18.167 The National Advanced Driving Simulator, The University of Iowa: http://www.nads-sc.uiowa.edu

18.168 M. Girone, G. Burdea, M. Bouzit, V. Popescu, J.E. Deutsch: A Stewart platform-based system for ankle telerehabilitation, Auton. Robots **10**(2), 203–212 (2001)

18.169 T. Nakano, M. Sugita, T. Ueta, Y. Tamaki, M. Mitsuishi: A parallel robot to assist vitreoretinal surgery, Int. J. Comput. Assist. Radiol. Surg. **4**(6), 517–526 (2009)

18.170 M. Wu, T.G. Hornby, J.M. Landry, H. Roth, B.D. Schmit: A cable-driven locomotor training system for restoration of gait in human SCI, Gait Posture **33**(2), 256–260 (2011)

18.171 J. Fink, N. Michael, S. Kim, V. Kumar: Planning and control for cooperative manipulation and transportation with aerial robots, Proc. Intl. Sym. Robot. Res., Luzern (2009)

Multimedia Contents

19. Robot Hands

Claudio Melchiorri, Makoto Kaneko

Multifingered robot hands have a potential capability for achieving dexterous manipulation of objects by using rolling and sliding motions. This chapter addresses design, actuation, sensing and control of multifingered robot hands. From the design viewpoint, they have a strong constraint in actuator implementation due to the space limitation in each joint. After briefly introducing the overview of anthropomorphic end-effector and its dexterity in Sect. 19.1, various approaches for actuation are provided with their advantages and disadvantages in Sect. 19.2. The key classification is (1) remote actuation or build-in actuation and (2) the relationship between the number of joints and the number of actuator. In Sect. 19.3, actuators and sensors used for multifingered hands are described. In Sect. 19.4, modeling and control are introduced by considering both dynamic effects and friction. Applications and trends are given in Sect. 19.5. Finally, this chapter is closed with conclusions and further reading.

19.1 **Basic Concepts** 464
19.1.1 Anthropomorphic End-Effectors . 464
19.1.2 Dexterity of a Robotic Hand 465
19.2 **Design of Robot Hands** 465
19.2.1 Actuators Placement and Motion Transmission 465
19.2.2 Actuation Architectures 467
19.3 **Technologies for Actuation and Sensing** 470
19.3.1 Actuation 470
19.3.2 Sensors 470
19.4 **Modeling and Control of a Robot Hand** . 473
19.4.1 Dynamic Effects of Flexible Transmission Systems 474
19.4.2 Transmission Model of Tendon-Outer-Tube System 476
19.4.3 The Control Through Single-Acting Actuators 476
19.4.4 Control of a Robot Hand 477
19.5 **Applications and Trends** 477
19.6 **Conclusions and Further Reading** 478
Video-References 478
References 479

Human hands have great potentialities not only for grasping objects of various shapes and dimensions, but also for manipulating them in a dexterous manner. It is common experience that, by training, one can perform acrobatic manipulation of stick-shaped objects, manipulate a pencil by using rolling or sliding motions, perform precise operations requiring fine control of small tools or objects. It is obvious that this kind of dexterity cannot be achieved by a simple gripper capable of open/close motion only. A multifingered robot hand can therefore provide a great opportunity for achieving such a dexterous manipulation in a robotic system. Moreover, we have also to consider that human beings do not use hands only for grasping or manipulating objects. Exploration, touch, perception of physical properties (roughness, temperature, weight, just to mention a few) are other fundamental tasks that we usually are able to perform thanks to our hands. We expect this type of capabilities also from robotic end-effectors and therefore, by adding quite advanced sensing equipments and proper control strategies, we may improve the interaction capabilities with the environment, achieving for example active exploration, detection of sensing surface properties (local friction, impedance, and so on), tasks that are usually very hard or impossible for simple grippers. For these and other reasons the study of multifingered robot hands has strongly interested the research community since the early days of Robotics.

It was in late 1970s that Okada developed a multifingered robot hand with a tendon driving system

and demonstrated a nut opening motion [19.1]. In early 1980s, two major projects on multifingered robot hands have been launched: the Stanford/Jet Propulsion Laboratory (JPL; VIDEO 751) hand and the Utah/Massachusetts Institute of Technology (MIT) hand [19.2, 3]. These two robot hands still represent a milestone and a term of comparison for the design of new devices. Since then, several multifingered hands have been designed and developed in a number research institutes all over the world. Among the most known, one can mention the Deutsches Zentrum für Luft- und Raumfahrt (DLR) hand(s) (VIDEO 754, VIDEO 768, and VIDEO 769), Mechanical Engineering Laboratory (MEL) hand, Electro-Technical Laboratory (ETL) hand, Darmstadt hand, Karlsruhe hand, University of Bologna (UB) hand (VIDEO 756, VIDEO 767), Barrett hand (VIDEO 752), Yasukawa hand, Gifu hand, U-Tokyo hand, Hiroshima hand, Soft Pisa/IIT hand (VIDEO 749, VIDEO 750), and many others [19.4–10].

When designing a multifingered hand, on the basis of its utilization, one should first define the following key issues: number and kinematic configuration of the fingers, anthropomorphic or nonanthropomorphic aspect, built-in or remote actuation, transmission system (in case of remote actuation), sensor assignment, integration with a carrying device (robot arm), control. All these aspects are considered in this chapter.

19.1 Basic Concepts

Before illustrating the main issues involved in the design and use of a robotic hand, it is necessary to discuss some basic concepts and definitions often encountered when dealing with these devices. In particular, terms like *dexterity* and *anthropomorphism* must be defined, and their implications on robotic hand design specified.

19.1.1 Anthropomorphic End-Effectors

The term *anthropomorphism* denotes the capability of a robotic end-effector to mimic the human hand, partly or totally, as far as shape, size, consistency, and general aspect (including color, temperature, and so on) are considered. As the word itself suggests, anthropomorphism is related to the external perceivable properties, and is not, itself, a measure of what the hand can do. On the contrary, *dexterity* is related to actual functionality and not to shape or aesthetic factors. In this sense anthropomorphism and dexterity are *orthogonal* concepts, whose reciprocal dependance (at least in the robotic field) has been not proved yet.

As a matter of fact, we can find in the literature anthropomorphic end-effectors with very poor dexterity level, even though they are called *hands*, as the tasks they can perform are limited to very rough grasping procedures [19.11]. Similarly, we can find smart end-effectors, capable of sophisticated manipulation procedures, without any level of anthropomorphism, e.g., the DxGrip-II [19.12]. Anthropomorphism itself is neither necessary nor sufficient to achieve dexterity, although it is quite evident that the human hand achieves a very high level of dexterity and represents a preferential paradigm for dexterous robotic manipulation.

Anthropomorphism is a desirable goal in the design of robotic end-effectors mainly for the following reasons:

- The end-effector can operate in a human-oriented environment (e.g., servicing robots), where tasks may be executed by robots or men as well.
- The end-effector can be tele-operated by a human operator, by means of special-purpose interfaces (e.g., a data-glove), directly reproducing the operator's hand behavior.
- For purposes of entertainment, assistance, and so on, a human-like aspect and behavior may be specifically required, like for humanoid robots.
- For prosthetic devices anthropomorphism is a quite evident design goal. The development of end-effectors for prosthetic purposes [19.13–15] has recently produced so advanced devices that they can be fully considered robotic systems.

While it is difficult to quantify the effective degree of dexterity of a robotic system, its anthropomorphism can be defined in a precise and objective way. In particular, the aspects that mainly contribute to determine the anthropomorphism level of a robotic hand are:

- *Kinematics*: concerning the presence of the main morphological elements (principal upper fingers, secondary upper fingers, opposable thumb, palm).
- *Contact surfaces*: extension and smoothness of the contact surfaces, aspect that reflects on the capability to locate contacts with objects all over the surface of the available links and on the presence of external compliant pads [19.16].

- *Size*: i. e., the size of the robotic hand both referring to the average size of a human hand and the *correct* size ratio between the links.

19.1.2 Dexterity of a Robotic Hand

Besides the *geometrical* reproduction of the human hand, the main research target remains the emulation of those functionalities which make it such a versatile end-effector.

Two are the main capabilities of a human hand:

- *Prehension*, i. e., the hand's ability to grasp and hold objects of different size and shape
- *Apprehension*, or the hand's ability to understand through active touch.

In this sense, the human hand is both an *output* and *input* device [19.17]. As output device, it can apply forces in order to obtain stable grasps or perform manipulation procedures. As input device, it is capable to explore an unknown environment providing information about the state of the interaction with it. The same features are desirable in robot hands. As a matter of fact, the application of robotic systems in unknown environments requires dexterous manipulation abilities to execute complex operations in a flexible way.

A widely accepted definition states that the dexterity of a robotic end-effector is a measure of its capability of changing the configuration of the manipulated object from an initial configuration to a final one, arbitrarily chosen within the device workspace. Generally speaking, with the term *dexterity* we intend the capability of the end-effector, operated by a suitable robotic system, to autonomously perform tasks with a certain level of complexity. An exhaustive review of scientific work developed so far about dexterity of robotic hands, with a quite complete and updated list of references, can be found in [19.18].

Even though the word dexterity itself has a very positive meaning, it may be useful to consider different levels of dexterity, associated with growing complexity and criticality of performable tasks. The dexterity domain for robotic hands can be roughly divided in two main areas, i. e., *grasping* and *internal manipulation*.

Grasping is intended as the capability of constraining objects with a constraint configuration that is substantially invariant with time (the object is fixed with respect to the hand).

Internal manipulation is a controlled motion of the grasped object in the hand workspace, with the constraint configuration changing with time.

Further subdivisions of these two domains have been widely discussed in the literature (different grasp topologies [19.19], different internal manipulation modes based on internal mobility and/or contact sliding or rolling [19.18]).

Although the notion of dexterity is well settled, the way to achieve it remains debated. Factors affecting the actual capabilities of a robotic end-effector are so many that often the analysis and above all the synthesis of dexterous hands do not take into proper consideration some of these elements, namely: morphological features; sensory equipment; control algorithms; task planning strategies; and so on.

19.2 Design of Robot Hands

The mechanical design of an articulated robotic hand can be performed according to many possible design concepts and options, even if a kinematical architecture has already been defined and size and shape specifications imposed. One of the main issues is the design of a proper actuation and transmission system. This aspect is crucial because space and dimensions are usually limited, being in general an anthropomorphic aspect and dimension a design goal to be pursued. Another aspect that is relevant for the design is the adoption of compliant structures (Fig. 19.1), in place of conventional mechanical joints, e.g., rolling pairs [19.20, 21].

Note that, since many solutions and operating concepts can be adopted, what is presented here aims only at illustrating the most significant solutions, and does not pretend to be a complete discussion of all the possible choices.

19.2.1 Actuators Placement and Motion Transmission

In order to actuate the joints of a robot hand, two basic approaches for the placement of the actuators are possible, i. e.:

- Placing the motors as close as possible to each joint, directly in the fingers and sometimes integrating them within the joint itself.
- Placing the motors into the palm or in the forearm; in this case motion is transmitted to each joint by means of (complex) kinematic chains.

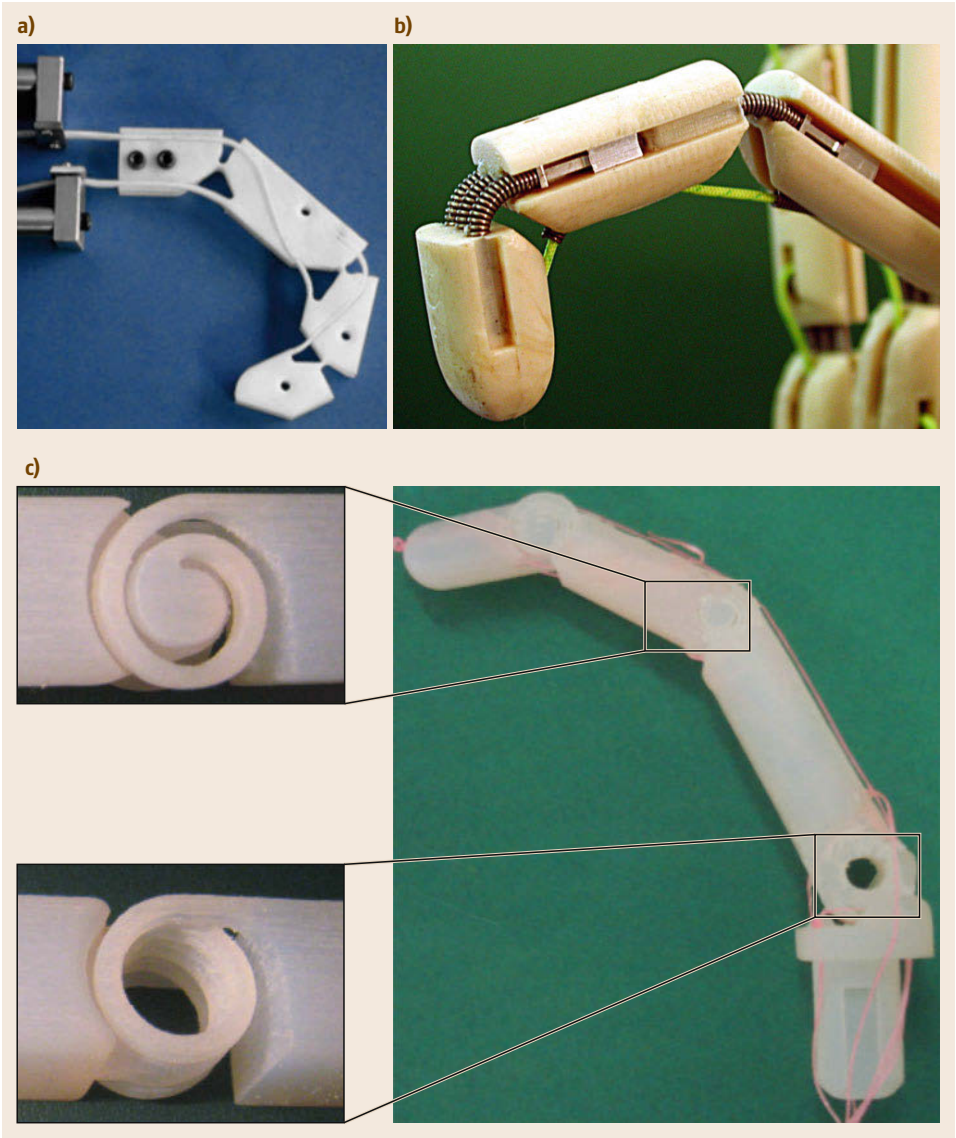


Fig.19.1a–c Three robotic fingers based on compliant joints. **(a)** The finger is obtained in a single teflon piece; **(b)** joint compliance is achieved with metallic springs; **(c)** fast prototyping allows for different compliant mechanisms as joints

In-site actuation can be defined as the case in which the actuator is hosted inside one of the two links connected by the actuated joint or is placed directly inside the joint:

- *Direct-drive actuation*: the actuator is placed directly on the joint, without transmission elements.
- *Link-hosted actuation*: the actuator is placed inside one of the two links constituting the actuated kinematic chain.

In-site actuation simplifies the mechanical configuration of the joint, reducing the transmission chain complexity. In particular, it has the great advantage that the motion of the joint is kinematically independent with respect to other joints. Usually, the size of the finger is imposed by the dimension of the actuators, and for technological reasons it is quite difficult to obtain both an anthropomorphic size and the same grasp strength of the human hand. Furthermore, the motors occupy a large room inside the finger structure, and it is a serious problem to host other elements, like sensors or compliant skin layers. A further negative aspect is that, since the mass of the actuators is concentrated inside the finger, the dynamic behavior of the system and its response bandwidth are reduced.

Nevertheless, the recent advancement of actuator technology enables us to directly implement a quite powerful actuator with reasonable size in each joint.

This built-in actuation has been adopted, e.g., for DLR hand [19.4, 22], ETL hand, Karlsruhe hand, Yasukawa hand, Barrett hand, Gifu hand, U-Tokyo hand, and Hiroshima hand. Since this actuation does not include compliant element like tendons, we can keep a stiff transmission system, which leads to a stable control system even under a high gain (Sect. 19.4). An issue is the routing of wires for both power and signal cables. This issue is more serious in distal joints than for the base joint, since the cables in distal joints produce a relatively large torque disturbance on the first joint, and therefore it is difficult to achieve a precise torque control for this joint.

Remote Actuation

Remote actuation is an alternative solution to in-site actuation. In remote actuation, the joint is driven by actuators placed outside the links connected by the joint itself. Remote actuation requires a motion transmission system, that must pass through the joints between the motor and the actuated joint. In some way, remote actuation must consider the problem of kinematic coupling between the actuated joint and the previous ones. Remote actuation is prevalent in biological structures (e.g., in human hand), where the finger joints are moved by muscles placed on the palm or in the forearm. This human-like approach has been adopted in projects of robotic hands like the UB hand or the National Aeronautics and Space Agency (NASA) Robonaut hand [19.23, 24].

Remote actuation systems can be classified according to the type of adopted transmission elements, i. e., flexible- or rigid-link transmission.

Flexible Link Transmission. Flexible link transmission is based on deformable connections, either flexible or rotational, that can adapt to variations of configuration by changing the transmission path. Linear flexible transmissions are based on flexible elements with translating motion, subject to tension (more frequently) or tension and compression. Two further subcategories can be identified: pulley-routed flexible elements (tendons, chains, belts) or sheath-routed flexible elements (mainly tendon-like elements). Rotational flexible transmissions are based on flexible rotary shafts, that can transmit rotational motion inside the finger structure to the joint, where a final transforming mechanism (a bevel gear or a worm gear) can be used to actuate the joint.

Rigid Link Transmission. Rigid link transmission is mainly based on articulated linkages or on rolling conjugated profiles (mainly gear trains). A further subdivision can be made between parallel and nonparallel axes gear trains, like bevel gears, worm gears, and so on.

19.2.2 Actuation Architectures

Both in-site and remote actuation can be applied according to different types of organization, i. e., by using one ore more actuators for each joint and by making these actuators work in different ways.

In general, we can consider an overall number N of joints for the robotic hand (the wrist joints are not considered) and a number M of actuators that are used to drive, directly or indirectly, the joints. According to different concepts of actuation and transmission, three main categories of actuation schemes can be identified:

- $M < N$: some joints are passive, coupled, or underactuated.
- $M = N$: each joint has its own actuator and there are no passive, coupled or underactuated joints.
- $M > N$: more than one actuator is operating on a single joint.

These architectures strongly depends on the type of motors. In particular, it is possible to recognize two main actuation modalities:

- *Single-acting actuators* – each motor can generate a controlled motion in one direction only: return motion in opposite direction must be obtained by an external action, that can be a passive (e.g., a spring) or an active system (e.g., an antagonistic actuator); this is the case of tendon-based transmission systems.
- *Double-acting actuators* – each motor can generate a controlled motion in both directions and can be used alone to drive the joint or to cooperate with other actuators; in this case the functional redundancy can allow sophisticated drive techniques, like push-pull cooperation.

Each category can be further subdivided. In the following, a brief description of the most frequently adopted schemes is presented.

Single-Acting Actuators with Passive Return Elements

Passive elements, like springs, can store energy during the actuation phase, restituting it during the return stroke (Fig. 19.2a). This mechanism leads to a simplification of actuation scheme, but requires mechanically backdrivable actuators. Other possible drawbacks are related to the loss of available power for the grasp and

the limited response bandwidth in case of low spring stiffness.

Agonistic–Antagonistic Single-Acting Actuators

Two actuators drives the same joint, acting in opposition in different directions (agonistic-antagonistic couple) (Fig. 19.2b). This solution leads to an *N joints-2N actuators* scheme and is quite complex since a large number of actuators must be placed in the hand. On the other hand, it may allow sophisticated control procedures, as both actuators can pull at the same time, with different intensity, generating a driving torque on the joint and a preloading of the joint itself (cocontraction, typical of tendon-driven joints):

- *Pros*: cocontraction strategies, possibility to change the joint stiffness according to the grasping phase and therefore to limit the influence of friction during fast approaching motions; independent position/tension control on each actuator can allow compensation of different path length in case of remote transmission; it is the most flexible solution for driving a joint.
- *Cons*: back-drivability of actuators is required; high difficulty in hosting two actuators for each joint, both in in-site and in remote location; higher control complexity; higher cost.

Single-Acting Actuators Organized According to the Concept of Actuation Net

This is a very interesting case, mimicking biological systems, but has not been implemented yet in robotic hands, except for some preliminary studies. N joints are driven by M actuators, being $N < M < 2N$. Each actuator cooperates in moving more than one joint, thanks to proper net-shaped transmissions:

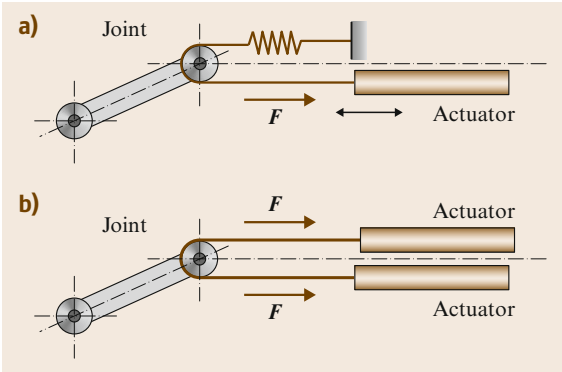


Fig.19.2a,b Single-acting actuator with an antagonist passive element **(a)** and in an agonist-antagonist configuration **(b)**

- *Pros*: cocontraction strategies, possibility to change the joint stiffness according to the grasping phase and therefore to limit the influence of friction during fast approaching motions; reduced number of actuators with respect to the $2N$ actuators scheme.
- *Cons*: back-drivability of actuators is required; high complexity of the kinematic scheme and therefore high complexity in control.

The simplest case of actuation net is represented by the so called $N + 1$ *actuation* (being N in this case the number of joints of a finger), frequently adopted in practice (Fig. 19.3). In this case, all actuators are coupled, and therefore a damage of any of them will result in a general failure.

Double Acting Actuators with $M < N$

In this case, the number of actuators is less than the number of joints. With reference to a single motor and several joints, two main subcases can be defined:

1. The joints are kinematically coupled, in a fixed or variable way, so that the number of degrees of freedom of this subsystems is reduced to one.
2. The joints are selectively actuated by the motor, according to an active or passive selection subsystem.

The former case can be further subdivided:

- *Joints kinematically coupled in a fixed way*: In this kind of kinematical configuration, each motor can move more joints connected by rigid mechanisms with fixed transmission ratios. A typical application is obtained with the use of a gear train: the first link is directly actuated by a motor, while a gear transmission between a wheel fixed to the frame and a final wheel connected to the joint generates the relative motion of the second link (Fig. 19.4a). Should the motion of two parallel fingers be required, their connection could be easily obtained mounting two gear wheels on the same shaft. Another very common way to obtain this kind of kinematical linkage is to use tendon driven devices

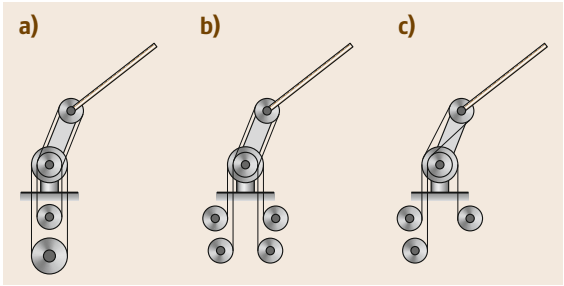


Fig.19.3a–c Remote actuation. **(a)** N-type, **(b)** $2N$-type, **(c)** $N + 1$-type

as shown in Fig. 19.4b. In artificial hand design, the main advantage when using joints driven by fixed-ratio mechanisms is the possibility to know and control the position of the second link. A disadvantage is that this kind of mechanisms does not adapt to the shapes of the grasped objects, and this may cause grasp instability.

- *Joints coupled in a non-fixed way*: This is the case of underactuated mechanisms and deformable passive-driven joints. A mechanism is said to be *underactuated* when the number of actuators is smaller than the number of degrees of freedom. When applied to mechanical fingers, this concept may lead to shape adaptation, i. e., underactuated fingers can envelope the objects to be grasped and adapt to their shape even with a reduced number of actuators. In order to obtain a statically determined system, elastic elements and mechanical limits must be introduced in underactuated systems (simple linear spring are often used). In the case of a finger closing on an object, for instance, the configuration of the finger is determined by the external constraints associated with the object. An example of an underactuated two-degrees of freedom finger is shown in Fig. 19.5 [19.25]. The finger is actuated through the lower link, and a spring is used to maintain the finger fully extended. A mechanical limit is used to keep the phalanges aligned under the action of this spring when no external forces are applied on the phalanges. Since the joints cannot be controlled independently, the behavior of the finger is determined by the design parameters (i. e., the geometric and the stiffness properties). Hence, the choice of these design parameters is a crucial issue.
  Another approach consists in coupling the motion of two adjacent joints by means of deformable linkages. This feature introduces in the kinematical chain the needed compliance to fit to the shapes of the grasped objects. A very simple mechanism of this category is reported in Fig. 19.6. Structurally it is similar to the mechanisms based on a fixed coupling, the only important difference is the addition of a spring to give extensibility to the tendon. This spring allows to decouple the motion between the first and second link when an external force is applied to the distal one. This solution is widely used: a well known example is the DLR hand. The benefits of this solution are mainly due to the possibility to fit to the shapes of objects. A design problem is the choice of the stiffness of the deformable element in order to achieve at the same time a strong grasp and a good shape adaptability.
- *Joints selectively driven by only one motor*: With this solution, the motion generated by only one (large) motor is transmitted and distributed to several joints. Actuation and control of each joint is obtained by means of insertion-disinsertion devices like self-acting or commanded clutches.

Double-Acting Actuation, with $M = N$

This is a very common case: each joint is driven in both directions by the same actuator. The achievable performances are therefore similar (equal) in both directions, but particular attention must be paid to backlash, and it is usually necessary to preload the transmission system. In particular, preload is mandatory in case of transmission by means of flexible elements like tendons (Fig. 19.3a). Furthermore, the adoption of a closed-

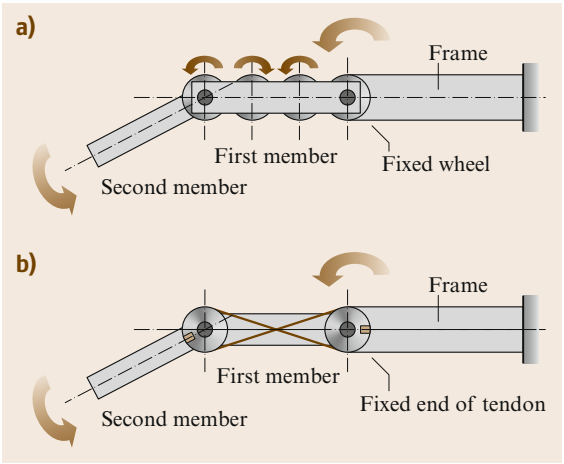


Fig. 19.4 Double-acting actuator with $N = M$ based on gears **(a)** and tendons **(b)**

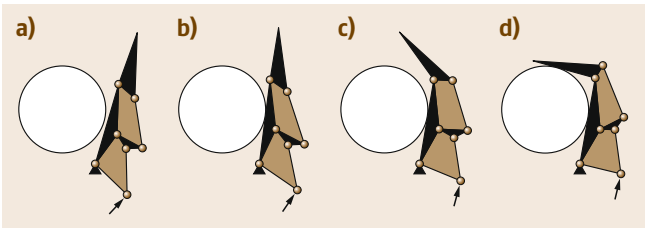


Fig.19.5a–d Grasping sequence performed by a finger based on underactuated mechanism

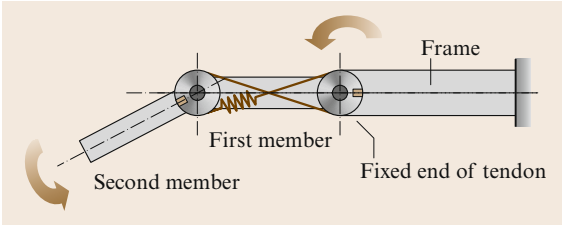


Fig. 19.6 Joints coupled in non fixed way

loop tendon transmission requires that the overall length of the tendon route must be kept constant, according to the concept that winded and unwinded parts of the tendon on the motor pulley have the same length; this involves the need of length-compensating mechanisms (e.g., pulley-trains, cams) every time that changes in the geometry of the finger cause a differential displacement of the tendons. In spite of this required complexity, this actuation scheme has been widely used, with simple pulley routing (UB hand, Okada hand, ...), or sheath-routing (Salisbury hand, Dipartmento di Informatica Sistemica e Telematica (DIST) hand), that has a simpler mechanical structure but must face the problem of sheath-tendon friction (application of high preload is not convenient in this case).

19.3 Technologies for Actuation and Sensing

In this Section, a brief description of the main issues related to technological aspects of actuation and sensing for robot hands is reported. A more general presentation and detailed description of these aspects is given in Part A (Chaps. 4 and 5) and Part C (Chap. 28).

19.3.1 Actuation

Electrical actuators are without doubt the most common choice for actuating robot hands. As a matter of fact, electric motors have very good performance in terms of position/velocity control, have a reasonable mass/power ratio, and are a very common technology, that does not require external devices (as for hydraulic or pneumatic actuators). However, there are several other possibilities. For example: ultrasonic motors (Keio hand [19.26]), chemical actuators, pneumatic actuators (McKibben in the Shadow hand [19.27]; VIDEO 753), spring based actuators (as for the 100G Capturing Robot [19.28] VIDEO 755), twisted string actuators (Fig. 19.7) [19.29] (Dexmart hand [19.30]; VIDEO 767), and others.

In particular, for pursuing quick responses, either pneumatic or spring-based actuators may be good solutions, although it should be noted that a braking system with quick response is essential for achieving good position controllability for this type of actuators.

19.3.2 Sensors

In robot hands, as in other robotic devices, sensors can be classified in two main categories: *proprioceptive* and *exteroceptive* sensors. The first type of sensors measures physical information related to the state of the device itself (e.g., position, velocity, and so on), while the second one is devoted to the measurement of data related to the interaction with objects/environment (e.g., applied forces/torques, friction, shape, and so on).

Joint Position/Velocity Sensors

For control purposes, there is the obvious necessity of measuring position/velocity of the actuated joints. A major problem consists in the limitation of the available space, both for the sensors and for the wires. Different technological solutions can be adopted, but a rather common choice is based on Hall-effect sensors, that are sufficiently small, precise and reliable for this type of application. In case of remote actuation, there is the possibility of having two position/velocity sensors for each joint: one located in the actuator (e.g.,

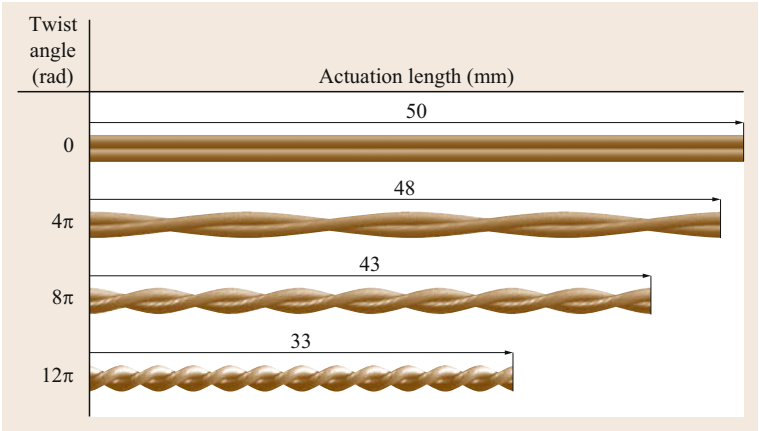


Fig. 19.7 The twisted string concept: by twisting the string, its twisted string concept length is reduced transforming a rotational motion into a linear one

an encoder) and one placed in the joint itself, often necessary because of the non linearities introduced by the transmission system (elasticity, friction, and so on). Quite often, this latter sensor is specifically designed and implemented for the given hand, being commercially available sensors too large and not suitable for installation in the joints.

Tendon Tension Sensor and Joint Torque Sensor

It is well known that humans can control finger tip compliance as well as finger tip force by controlling voluntary muscles. In remote actuation, it is essential to measure the tendon tension for two main reasons: for compensating the friction existing in the transmission system, and for measuring the external contact force. Figure 19.8 shows a way for measuring the tendon tension where the tendon is pressed by an elastic plate with a strain gauge. When a tension is applied to the tendon, the sensor measures a force composed of axial and bending force components. The displacement of the elastic plate due to the axial force component is negligibly small compared with that due to the bending force component. As a result, the bending force component generates a bending deformation for the elastic plate. This deformation is transformed in an electric signal by means of proper transducers, such as strain gauges attached on the surface of the plate or optoelectronic components [19.31, 32]. Now, suppose N-type actuation with two tension sensors, as shown in Fig. 19.9, where joint torque τ is given. Note that $\tau = r(T_1 - T_2)$ where r, T_1, and T_2 are the pulley radius and tendon tensions, respectively. Since we can measure e_1 and e_2 corresponding to T_1 and T_2, τ can be obtained by feeding both e_1 and e_2 into the differential circuit. This approach, however, includes a couple of issues. The main problem is the plastic deformation of the sensor plate under an extreme large pretension. Once such a plastic deformation has happened, the sensor will never work appropriately anymore. Another minor issue is are that two sensors are always necessary for measuring a joint torque. To cope with these issues, the tension-differential-type torque sensor [19.28] can be used as shown in Fig. 19.10. The sensor is designed with just a single body and it partially includes an elastic part where at least one strain gauge is attached. The working principle of the sensor, shown in Fig. 19.10a, supposes that a torque is applied to the joint. This means that T_1 and T_2 have different values. This difference causes a bending force around the strain gauge. The key is that the bending force is kept to zero even under an extremely large tension as far as no joint torque is given. Therefore, we are completely released from the plastic deformation of the elastic plate due to pretension. Furthermore, the sensor is constructed with just a single body. There are couple of variations in this type of torque sensor. As decreasing the pulley distance in Fig. 19.10a, the sensor eventually results in the single-pulley-version with zero distance, as shown in Fig. 19.10b. The single-pulley-version has been implemented into Darmstadt

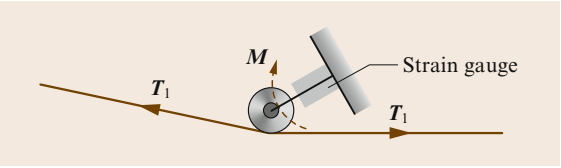


Fig. 19.8 Tendon tension sensing

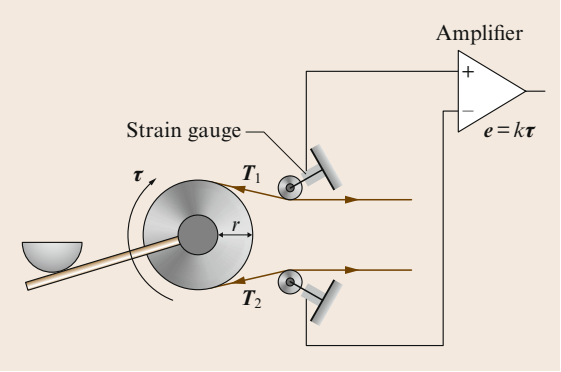


Fig. 19.9 Tension sensor based torque sensing

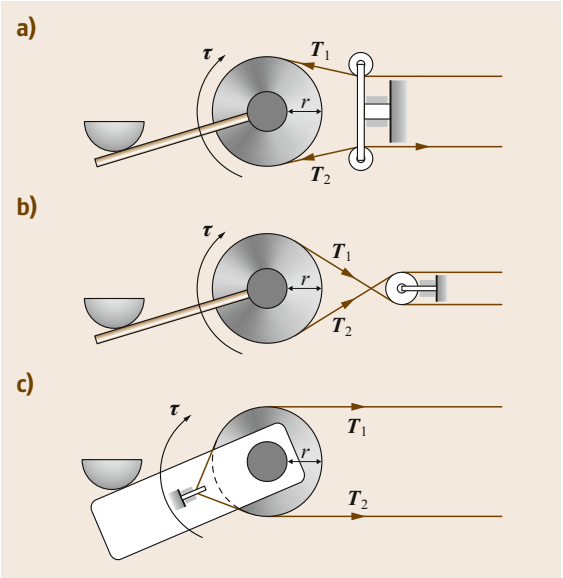


Fig.19.10a–c Tension-differential type (TDT) sensor. **(a)** Double pulley version, **(b)** single pulley version, **(c)** pulley-less version (after [19.28])

hand [19.33] and MEL hand [19.34]. Furthermore, if the sensor is built in the finger link connected by the concerned tendon, there is no relative motion between the sensor and the tendon. As a result, we can remove the pulley, as shown in Fig. 19.10c. This is called as the pulley-less version and has been implemented into Hiroshima hand. The tension-differential-type torque sensor will be a powerful tool for measuring a tendon drive joint.

Finger Tip Tactile (or Force) Sensors

Most robot manipulation and assembly tasks would benefits of the utilization of tactile sensory information. When lifting an object, tactile sensing could detect the onset of slip in time for corrective action to be taken. In addition to the contact point between the finger tip and the object, several objects properties, such as friction coefficient of the object surfaces, surface texture, and weight can be determined by utilizing a finger tip tactile (or force) sensor. A six-axis force sensor allows us to detect contact point as well as contact force between finger and environment, if a single contact is assumed. For the finger model as shown in Fig. 19.11, the following relationship between the sensor output and contact force may be defined

$$\boldsymbol{F}_\mathrm{s} = \boldsymbol{f}\,, \tag{19.1}$$

$$\boldsymbol{M}_\mathrm{s} = \boldsymbol{x}_\mathrm{c} \times \boldsymbol{f}\,, \tag{19.2}$$

where $\boldsymbol{f} \in \mathbb{R}^3$, $\boldsymbol{F}_\mathrm{s} \in \mathbb{R}^3$, $\boldsymbol{M}_\mathrm{s} \in \mathbb{R}^3$, and $\boldsymbol{x}_\mathrm{c} \in \mathbb{R}^3$ are the external force vector, the force vector measured by the six-axis force sensor, the moment vector measured by the six-axis force sensor, and the position vector indicating the contact position, respectively. From the first equation, we can directly obtain the contact force. Putting $\boldsymbol{F}_\mathrm{s}$ into the second equation leads to $\boldsymbol{M}_\mathrm{s} = \boldsymbol{x}_\mathrm{c} \times \boldsymbol{F}_\mathrm{s}$. $\boldsymbol{x}_\mathrm{c}$ is determined in such a way that $\boldsymbol{M}_\mathrm{s} = \boldsymbol{x}_\mathrm{c} \times \boldsymbol{F}_\mathrm{s}$ may be satisfied. For a finger with convex object, we have always two mathematical solutions as shown in Fig. 19.12a where the meaningful solution is the one satisfying $\boldsymbol{f}^\mathrm{t}\boldsymbol{n} < 0$, n being the outward normal direction to the finger's surface (a finger can only push the object). However, for a finger with concave shape, we have at least four mathematical solutions, as shown in Fig. 19.12b where two of those are physically possible. A finger with the six-axis force sensor located in the fingertip, Fig.19.12.c, can avoid multiple solutions. On the other hand, only forces applied to the fingertip can be detected, and if more links are in contact with the object it would be necessary to have a force/torque sensor placed in each of them.

This type of solution, i. e., a multiaxis sensor for measuring not only forces and torques but also the position of the contact point, is known in the literature as the *intrinsic tactile* (IT) principle [19.35]. In general, with respect to the use of traditional tactile sensors, see later, it leads to a simplification in the design since it requires less wires and connections for the sensor.

Tactile Sensors

Another important class of sensing devices consists of tactile sensors, which are used for several purposes, such as shape recognition, contact point determination, pressure/force measurement. A number of tactile sensors have been proposed in the literature, with several different solutions concerning the implementation features: optical, piezoresistive, piezoelectric, and so on. References [19.36, 37] give an overview on technologies and applications.

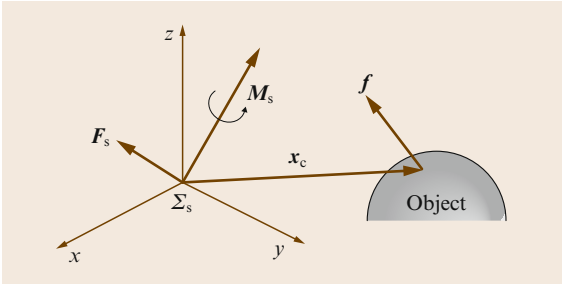


Fig. 19.11 Sensor coordinate system $\sum_\mathrm{S}$

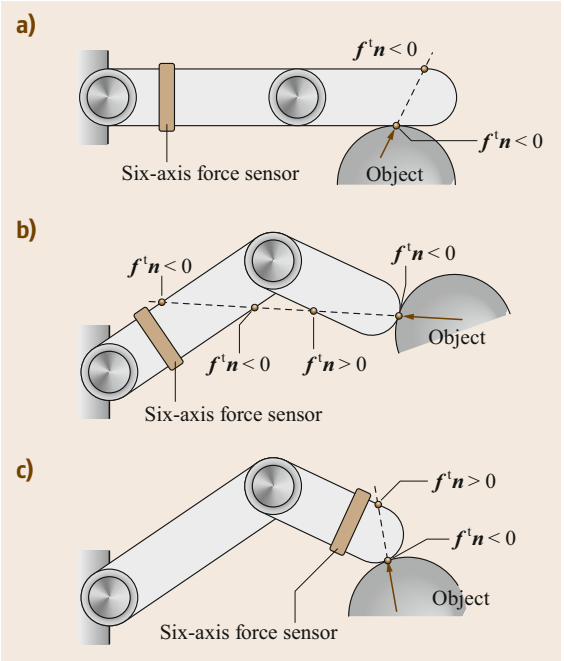


Fig. 19.12 Interpretation of solutions. **(a)** Convex shape, **(b)** finger with a concave shape, **(c)** sensor located in the fingertip

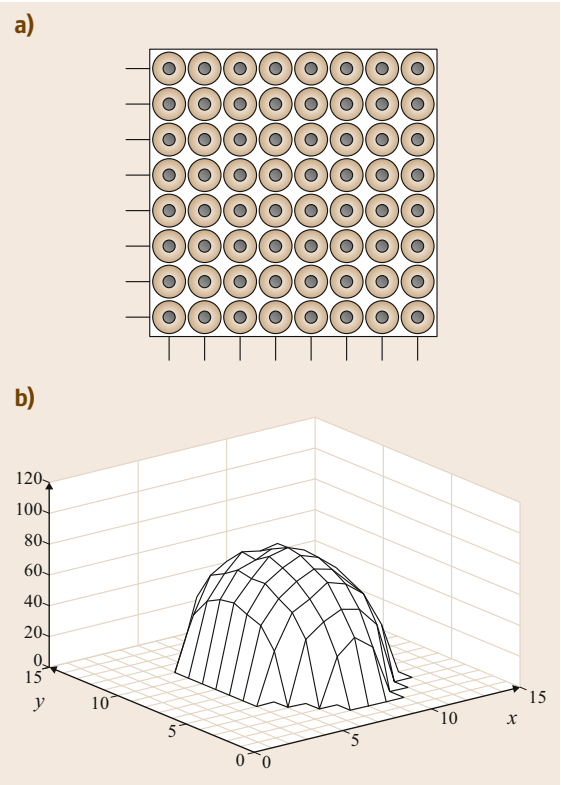


Fig.19.13a,b A Tactile sensor. **(a)** Scheme of a tactile sensor, **(b)** example of data from a tactile sensor

Tactile sensors have been introduced in robotics since the late 1970s. Nowadays, as the force sensors, also tactile sensors are commercially available devices. Probably, they represent the most commonly adopted sensorial class for industrial grippers, even though they are often used as advanced *on/off* devices to check whether a grasp or contact condition occurs.

Usually, they consist in a matrix (array) of sensing elements. Each sensing element is usually referred to as a *taxel* (from *tactile element*), and the whole set of information is called a *tactile image*, Fig. 19.13. Main goal of this class of sensors is to measure the map of pressures over the sensing area.

In general, the types of information that may be obtained from a tactile sensor are:

- *Contact*: This is the most simple information given by the sensor, concerning the presence or absence of a contact.
- *Force*: Each sensing element provides an information related to the amount of locally applied force, which can be used in several manners for successive elaborations.
- *Simple geometrical information*, i. e., position of the contact area, geometrical shape of the contact itself (planar, circular, and so on).
- *Main geometrical features of the object*: By proper elaborations of the data of the taxels, it is possible to deduce the type of object in contact with the sensor, for example a sphere, a cylinder and so on (data relative to the 3-D (three-dimensional) shape).
- *Mechanical properties*, such as friction coefficient, roughness, and so on. Also thermal properties of the object may be measured by a tactile sensor.
- *Slip condition*, i. e., the relative movement between the object and the sensor.

Several technologies have been adopted for the design of tactile sensors, ranging from piezoresistive to magnetic, to optical effects, and so on. Among the most common, one can mention:

- Resistive and conductive effect
- Electromagnetic effect
- Capacitive effect
- Piezoelectric effect
- Optical effect
- Mechanical methods.

Each of these technologies has positive and negative aspects. Common drawbacks, however, are the size of these sensors, usually quite large in comparison with the available space, and the necessity of a high number of electrical connections.

19.4 Modeling and Control of a Robot Hand

The dynamic model of a robot hand with in-site actuation is very similar to the model of a traditional (industrial) robot, and the hand can be considered as a collection of robot manipulators. On the other hand, remote actuation introduces some peculiar features that have to be carefully considered. In particular, the problems tied to nonlinear phenomena (e.g., friction and backlash), compliance of the transmission system, and noncolocation of sensors and actuators are very critical for the design of the control. Moreover, the use of single-acting actuators, such as tendon based actuation systems, requires the adoption of proper control techniques, which allow the imposition of the desired torque at each joint of the hand, despite the coupling among them.

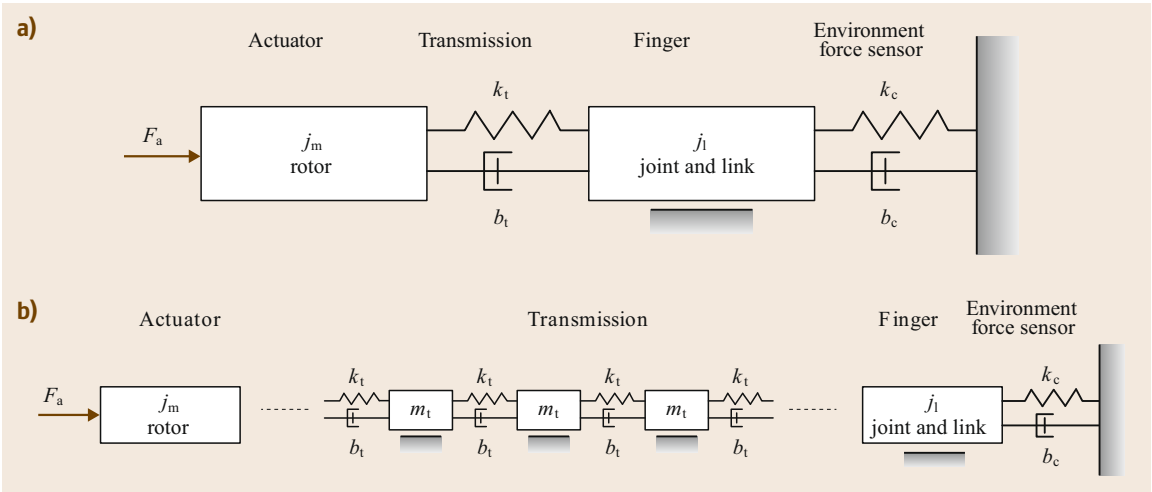


Fig.19.14a,b Model of a robot joint (a) with transmission flexibility, and (b) with tendon based transmission

19.4.1 Dynamic Effects of Flexible Transmission Systems

The transmission system of robot hands with remote actuation is usually characterized by an high level of friction and non negligible dynamic effects which complicate the control problem. A simple representation considers a single axis motion with two inertial elements linked by an elastic transmission. This is the typical representation of elastic joints in which the former element represents the motor inertia, while the latter is related to the inertial properties of the actuated joint/link (Fig. 19.14a). More complex models assume a dynamic model for the transmission system, i. e., the classical representation of tendons based on the serial repetition of masses linked by springs/dampers, reported in Fig. 19.14b. These simple models are particularly useful to understand some drawbacks and limitations due to the fact that actuation system and actuated element are located in two different places and the motion is transmitted by a nonideal (that is not purely static) element. If we consider the capability of the fingers' joint of applying a force on the environment, the effect of the transmission system on the open loop response of the system modeled as in Fig. 19.14a, are a noticeable reduction of the bandwidth, and an important phase delay between the input F_a (the force applied by the motor) and the output F_c (the force exchanged at the contact). As shown in Fig. 19.15, the open-loop transfer function

$$\frac{F_a}{F_c} = \frac{(b_c s + k_c)(b_t s + k_t)}{\left[j_l s^2 + (b_t + b_c)s + k_t + k_c\right](j_m s^2 + b_t s + k_t) - (b_t s + k_t)^2} \tag{19.3}$$

is characterized by four poles that, for growing values of the transmission stiffness k_s, move from their initial locations (that depend on the values of physical parameters j_l, j_m, etc., although for $k_s = 0$ at least one pole is in the origin of the Gauss plane) towards the poles of a system with an infinitely rigid transmission (for k_c tending to ∞, two poles go to infinity) (and a total inertia given by the contributions of both the motor and the link) whose transfer function is

$$\frac{F_a}{F_c} = \frac{(b_c s + k_c)}{(j_l + j_m)s^2 + b_c s + k_c} \tag{19.4}$$

As a consequence, the bandwidth of the system with flexible transmission, that for high values of k_s approx-

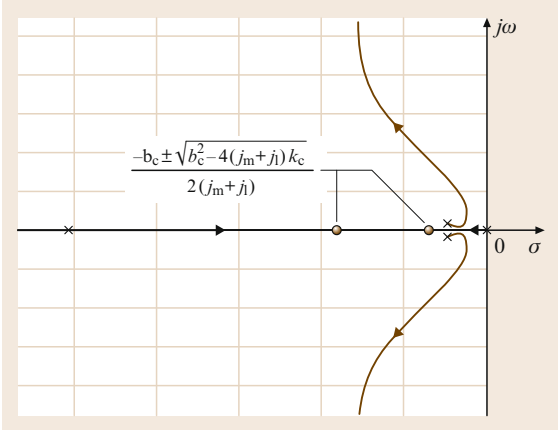


Fig. 19.15 Root contour of the transfer function (19.3) with variable k_t

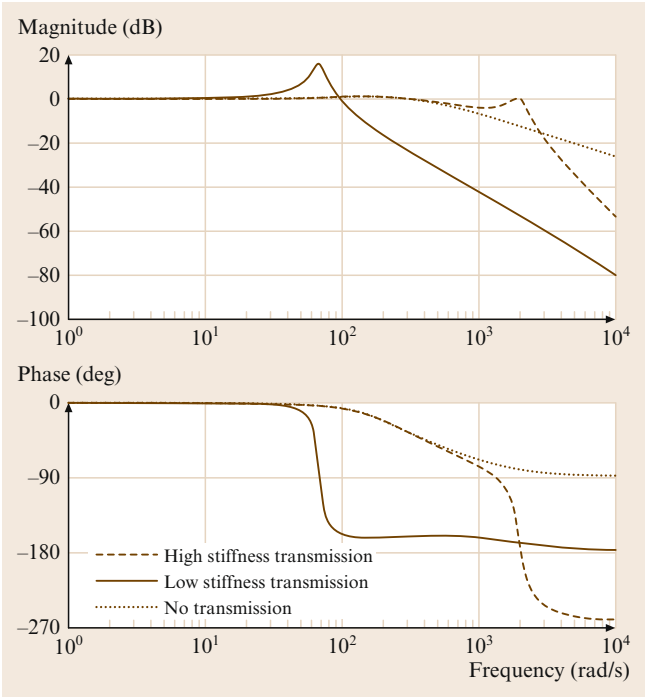


Fig. 19.16 Bode plots of the open loop transfer function (19.3) for low stiffness values (*continuous line*) high stiffness value (*dashed line*) and with no transmission (*dotted line*) ►

imates those of (19.4), decreases when the compliance of the transmission is not negligible, see the Bode plots reported in Fig. 19.16. The bandwidth of the system with flexible transmission is strongly affected by the location of the speed reducer: when the reducer is placed at the joint the bandwidth is K_r times (K_r is the reduction ratio) higher than the one achievable with the reduction applied directly on the motor [19.38] (Fig. 19.17). Moreover, it is worth to notice that for low level of the stiffness k_t a sharp phase drop occurs at the frequency of the flexible mode. Therefore, there are some frequencies (relatively low) at which the force applied by the motor and the one measured with a sensor in the finger's joint are completely out of phase. These effects, which may cause the instability of the overall system under force control (or impedance control) are referred to as *noncolocation*. In general, when actuators and sensors are physically located at different points of a flexible structure (or a structure with flexible transmission), there will be unstable modes in the closed-loop system [19.39].

From the control viewpoint, the problem of mechanical transmission flexibility is further exasperated by the non linear frictional phenomena that inevitably affect remote actuation and motion transmission. As a matter of fact, the linear viscous friction, represented in Fig. 19.14 by the damping coefficient b_t, is accompanied by stiction and Coulomb friction, both of which are discontinuous at zero velocity (Fig. 19.18). These nonlinearities may cause limit cycles and input-dependent stability, and must be accurately taken into account in the design of the robot hand structure as well as of its control architecture [19.40]. For instance, in the design of the Utah/MIT dexterous hand, depicted in Fig. 19.19, in order to reduce static friction, the idea of using tendon sheaths was abandoned in favor of pulleys [19.3]. In order to find an optimal trade-off between complexity and reliability of the mechanical arrangement and achieved friction level, a number of solutions, which combine sheaths and pulleys for routing the tendon from actuators to fingers' joints has been adopted in the design of robot hands, e.g., the Stanford/JPL hand (VIDEO 751) and the UB hand 3 reported in Fig. 19.20. This device is characterized by an extremely simple structure, with the tendons completely routed within sheaths, but on the other hand the friction cannot be absolutely neglected and a precise modeling of the interaction between the tendons and the tube is necessary for control purposes [19.41].

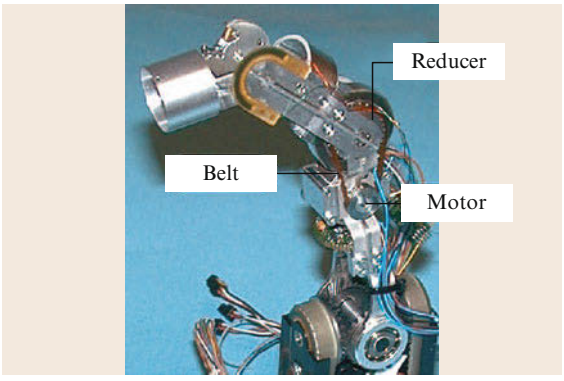


Fig. 19.17 Reducer location on the medial joint of the DLR hand II

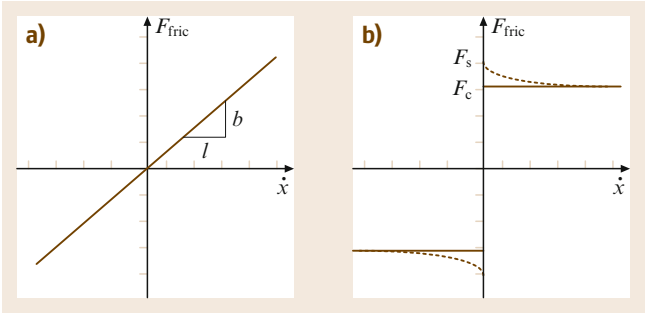


Fig.19.18a,b Frictional phenomena: viscous friction **(a)** and stiction and Coulomb friction **(b)**

19.4.2 Transmission Model of Tendon-Outer-Tube System

This system can be modeled as shown in Fig. 19.21 where T_{in}, T_{out}, T_0, ξ_{in}, ξ_{out}, R_i, x, and L are the tension at the input side, the tension at the output side, the initial pretension, the displacement at the input side, the displacement at the output side, the local radius of routing, the coordinate system along the wire, and the length of the tendon, respectively. The relationship between the tension at output and the input displacement is given by [19.42],

$$T_{out} - T_0 = K_t(\xi_{in}\phi_B)\,, \tag{19.5}$$

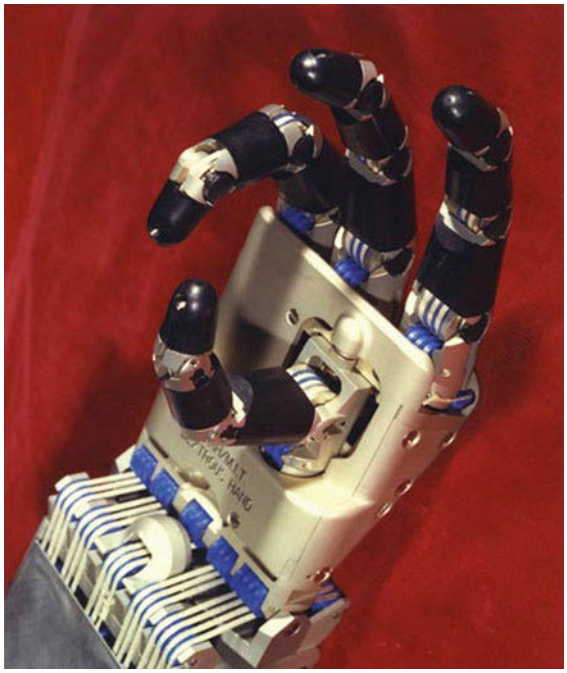


Fig. 19.19 The Utah/MIT robotic hand

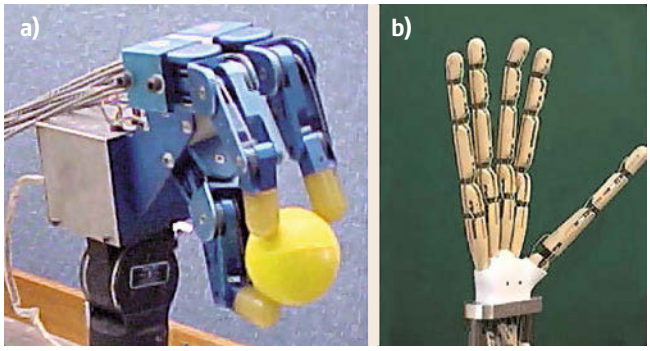


Fig.19.20a,b Tendon based robot hands: Stanford/JPL hand (**a**) and UB hand 3 (**b**)

where K_t and ϕ_B are the total stiffness and the equivalent backlash, respectively, and those are given by,

$$\frac{1}{K_t} = \frac{1}{K_e} + \frac{1}{K_s} + \frac{1}{K_{ap}} \tag{19.6}$$

$$K_{ap} = K_w \frac{\lambda}{\exp(\lambda) - 1} \tag{19.7}$$

$$\phi_B = \frac{T_0 L}{EA} \cdot \frac{\exp(\lambda) - \lambda - 1}{\lambda} \tag{19.8}$$

$$\lambda = \sum |\beta_i| \mu \operatorname{sgn} \xi_{in} \tag{19.9}$$

where K_e, K_s, K_w, K_{ap}, μ, E, A and β_i are the stiffness of environment, the stiffness of force sensor, the apparent stiffness of the tendon, the friction coefficient, Young's modulus, the cross sectional area, and the bending angle of each segment of tendon, respectively. For example,

$$\sum |\beta_i| = 2\pi$$

for the case given in Fig. 19.21. As can be seen from this example, the friction related parameter λ increases dramatically when the tube is heavily bent. While we have a big advantage of choosing a free route for the power transmission, it brings a large nonlinearity for the transmission system. We would note that while both the apparent stiffness of the tendon and the equivalent backlash vary depending upon λ which is the function of the curvature of the route as well as the friction coefficient, ϕ_B and K_{ap} result in $\phi_B = 0$ and $K_{ap} = K_w$ under $\mu = 0$. From the view point of control, such hysteresis is, of course, not desirable. To cope with these issues, each tendon should be designed as short as possible, so that we may keep high stiffness and small backlash in the transmission system.

19.4.3 The Control Through Single-Acting Actuators

The use of single-acting actuators (i. e., standard motors with tendinous transmission), which are commonly

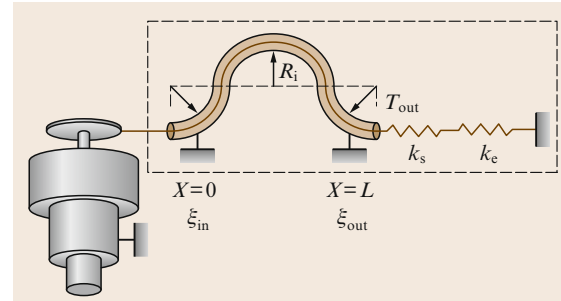


Fig. 19.21 Model of tendon-outer-tube transmission

assumed in the design of robot hands with remote actuation, requires the adoption of specific control techniques in order to guarantee the desired torques at the joints and to maintain at each time a positive tension on the tendons. To this purpose, the tendons are treated like inelastic frictionless elements and the problem is coped in a way completely decoupled from the issue of the device stability, discussed in previous sections.

A tendon, routed in the finger structure through sheaths or/and pulleys, can be modeled by means an *extension function* $l_{\mathrm{i}}(\theta)$ [19.43] relating the joints' configuration with the tendon elongation. In the case of the tendon network represented in Fig. 19.3, the extension functions of the three tendons have the form

$$l_{\mathrm{i}}(\boldsymbol{\theta}) = l_{0\mathrm{i}} \pm R\boldsymbol{\theta}_1 \pm R\boldsymbol{\theta}_2 \,,$$

where R is the radius of the pulleys and

$$\boldsymbol{\theta} = [\boldsymbol{\theta}_1 \boldsymbol{\theta}_2]^{\mathrm{T}}$$

is the vector of the joint variables. Once the extension function has been determined, it is straightforward to derive the relationship between tendons forces and resulting joints torques. As a matter of fact, the relation between the joint speeds $\dot{\boldsymbol{\theta}}$ and tendon speeds $\dot{\boldsymbol{l}}$ can be deduced by simply differentiating the expression of extension functions

$$\dot{\boldsymbol{l}} = \frac{\partial \boldsymbol{l}}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} = \mathbf{P}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \,. \tag{19.10}$$

Because of the conservation of the power, from (19.10), one can achieve

$$\boldsymbol{\tau} = \mathbf{P}^{\mathrm{T}}(\boldsymbol{\theta})\boldsymbol{f} \,, \tag{19.11}$$

where $\boldsymbol{\tau}$ are the torques exerted on the joints, and $\boldsymbol{f}$ are the force applied by tendons. From (19.11) it results that the force transmitted by a tendon may affect (and, in general, will affect) more than one joint.

In order to guarantee the possibility of exerting joint torques in every direction under the constraint of pure tensile forces, for any $\boldsymbol{\tau} \in \mathbb{R}^n$ it must exist a set of forces $\boldsymbol{f}_{\mathrm{i}} \in \mathbb{R}^m$ (n and m are respectively the number of joints and the number of tendons) such that

$$\boldsymbol{\tau} = \mathbf{P}^{\mathrm{T}}(\boldsymbol{\theta})\boldsymbol{f} \quad \text{and} \quad \boldsymbol{f}_i > 0, i = 1, \ldots, m \,. \tag{19.12}$$

In this case the tendon network is said *force closure*. If the condition expressed by (19.12) is verified, given a desired torque vector $\boldsymbol{\tau}$ it is possible to compute the force that the actuators must provide to the tendons according to

$$\boldsymbol{f} = \mathbf{P}^{\dagger}(\boldsymbol{\theta})\boldsymbol{\tau} + \boldsymbol{f}_{\mathrm{N}} \,, \tag{19.13}$$

where

$$\mathbf{P}^{\dagger} = \mathbf{P}(\mathbf{P}^{\mathrm{T}}\mathbf{P})^{-1}$$

is the pseudo-inverse of the coupling matrix $\mathbf{P}^{\mathrm{T}}$ and

$$\boldsymbol{f}_{\mathrm{N}} \in \mathcal{N}(\mathbf{P}^{\mathrm{T}})$$

is a vector of *internal forces* that insures that all tendon tensions are positive. In general, internal forces will be chosen as small as possible, so that the tendons are always taut but are not subject to excessive strains.

19.4.4 Control of a Robot Hand

The modeling and control aspects described in the previous Sections, although very important and fundamental, can be considered as a sort of *low level* problems in the control of a robot hand, in the sense that they are related to the specific physical properties of the device.

There are also other problems that must be faced and solved in order to operate in a profitable manner with a multifingered hand. These problems are solved by a proper design of a *high level* control for the hand, that must take into account the interaction of the hand with the objects and more in general with the environment. In this context, general aspects that must be considered are: the control of forces/torques applied at the contact points, the necessity to model contact compliance/friction effects, the type of mobility both for the fingers and at the contact (rolling, sliding, ...), a suitable planning algorithm for grasping and/or manipulating the objects, and so on.

These problems are illustrated in detail in Part C, Chaps. 37–39.

19.5 Applications and Trends

In the industrial environment, simplicity and cost are the main guidelines for the design of end-effectors, and therefore simple devices, as open-close grippers, are very common and widely used. This situation has led during the years to the development of a number of special-purpose devices, optimized for single

specialized operations but not suitable for other tasks. At the moment, dexterous multifingered hands have not really been applied to any major application, mainly because of problems of reliability, complexity, cost.

On the other hand, more and more operations are currently envisaged for robots working in environments designed for, and utilized by, human operators. Entertainment, maintenance, space/underwater applications, help to disable persons are just a few examples of use of robotic systems in which interaction with tools and objects designed for human beings (or directly with them) is implied. In all these circumstances, the robot must be able to grasp and manipulate objects (different in dimension, shape, weight, ...) similarly to humans, and therefore a robot hand, with a proper number of fingers and joints and also with an anthropomorphic appearance, seems to be the most adequate solution.

There are several projects aiming at developing anthropomorphic robots. Among others, one can mention the NASA/JPL Robonaut [19.24], Fig. 19.22, the devices developed at the DLR, the several projects on humanoid robots currently under development.

Fig. 19.22 The NASA/JPL Robonaut

19.6 Conclusions and Further Reading

The design of multifingered robot hands has attracted the interest of the research community since the early days of robotics, not only as a challenging technical problem itself but, probably, also because of anthropomorphic motivations and the intrinsic interest for a better knowledge of the human beings. In the last decades, has previously discussed, several important projects have been launched, and important examples of robot hands developed. Nevertheless, the current situation is that reliable, flexible, dexterous hands are still not available for real applications. For these motivations, it is easy to foresee also for the future a consistent research activity in this fascinating field, with developments at the technological (sensor, actuator, material, ...) and methodological (control, planning, ...) level. Important connections with other scientific fields are also expected, as for example with cognitive science.

Being this research area so wide, it is not simple to suggest to interest readers further readings, except for quite classical books such as [19.43–45]. As a matter of fact, depending on the specific research area, many publications are available, although often not organized as reference books, but mainly as technical papers published in journals or presented at international conferences. Moreover, since hundreds of new papers are published every year covering the different aspects of this robotic field, it is really quite difficult, and also not fair, to give at the moment specific suggestions for further readings. We can only refer to the citations already provided in the references.

Video-References

VIDEO 749 The PISA-IIT SoftHand
available from http://handbookofrobotics.org/view-chapter/19/videodetails/749

VIDEO 750 The PISA-IIT SoftHand
available from http://handbookofrobotics.org/view-chapter/19/videodetails/750

VIDEO 751 The Salisbury Hand
available from http://handbookofrobotics.org/view-chapter/19/videodetails/751

VIDEO 752 The Barrett Hand
available from http://handbookofrobotics.org/view-chapter/19/videodetails/752

VIDEO 753 The Shadow Hand
available from http://handbookofrobotics.org/view-chapter/19/videodetails/753

VIDEO 754 The DLR Hand
available from http://handbookofrobotics.org/view-chapter/19/videodetails/754
VIDEO 755 A high-speed Hand
available from http://handbookofrobotics.org/view-chapter/19/videodetails/755
VIDEO 756 The UBH2, University of Bologna Hand, ver. 2 (1992)
available from http://handbookofrobotics.org/view-chapter/19/videodetails/756
VIDEO 767 The Dexmart Hand
available from http://handbookofrobotics.org/view-chapter/19/videodetails/767
VIDEO 768 DLR Hand
available from http://handbookofrobotics.org/view-chapter/19/videodetails/768
VIDEO 769 The DLR Hand performing several task
available from http://handbookofrobotics.org/view-chapter/19/videodetails/769

References

19.1 T. Okada: Object-handling system for manual industry, IEEE Trans. Syst. Man Cybern. **2**, 79–86 (1979)
19.2 K.S. Salisbury, B. Roth: Kinematics and force analysis of articulated mechanical hands, J. Mech. Transm. Actuation Des. **105**, 35–41 (1983)
19.3 S.C. Jacobsen, E.K. Iversen, D.F. Knutti, R.T. Johnsan, K.B. Biggers: Design of the Utah/MIT dexterous hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986)
19.4 J. Butterfass, G. Hirzinger, S. Knoch, H. Liu: DLR's Multisensory articulated Hand Part I: Hard- and software architecture, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999)
19.5 A. Albu-Schäffer, T. Bahls, M. Chalon, O. Eiberger, W. Friedl, R. Gruber, S. Haddadin, U. Hagn, R. Haslinger, H. Hoppner, S. Jorg, M. Nickl, A. Nothhelfer, F. Petit, J. Reill, N. Seitz, T. Wimbock, S. Wolf, T. Wusthoff, G. Hirzinger: The DLR hand arm system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)
19.6 C. Melchiorri, G. Vassura: Mechanical and control features of the University of Bologna hand version 2, Proc. IEEE/RSJ Int. Conf. Int. Robots Syst. (IROS), Rayleigh (1992) pp. 187–193
19.7 W.T. Townsend: MCB – Industrial robot feature article- Barrett Hand grasper, Ind. Robot **27**(3), 181–188 (2000)
19.8 H. Kawasaki, T. Komatsu, K. Uchiyama: Dexterous anthropomophic robot hand with distributed tactile sensor: Gifu hand II, IEEE/ASME Trans. Mechatronics **7**(3), 296–303 (2002)
19.9 T.J. Doll, H.J. Sneebeli: The Karlsruhe Hand, Prepr. IFAC Symp. Robot Control (SYROCO) (1988), pp. 37.1–37.6
19.10 M.G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, A. Bicchi: Adaptive synergies for the design and control of the Pisa/IIT SoftHand, Int. J. Robotics Res. **33**, 768–782 (2014)
19.11 N. Fukaya, S. Toyama, T. Asfour, R. Dillmann: Design of the TUAT/Karlsruhe humanoid hand, Robot. Syst. **3**, 1754–1759 (2000)
19.12 A. Bicchi, A. Marigo: Dexterous grippers: Putting nonholonomy to work for fine manipulation, Int. J. Robotics Res. **21**(5/6), 427–442 (2002)
19.13 M.C. Carrozza, C. Suppo, F. Sebastiani, B. Massa, F. Vecchi, R. Lazzarini, M.R. Cutkosky, P. Dario: The SPRING hand: Development of a self-adaptive prosthesis for restoring natural grasping, Auton. Robots **16**(2), 125–141 (2004)
19.14 J.L. Pons, E. Rocon, R. Ceres, D. Reynaerts, B. Saro, S. Levin, W. Van Moorleghem: The MANUS-HAND dextrous robotics upper limb prosthesis: Mechanical and manipulation aspects, Auton. Robots **16**(2), 143–163 (2004)
19.15 Bebionic Prosthetic Hand: RSLSteeper, Leeds, UK (2015) http://www.bebionic.com/
19.16 G. Berselli, M. Piccinini, G. Palli, G. Vassura: Engineering design of fluid-filled soft covers for robotic contact interfaces: Guidelines, nonlinear modeling, and experimental validation, IEEE Trans. Robotics **27**(3), 436–449 (2011)
19.17 T. Iberall, C.L. MacKenzie: Opposition space and human prehension. In: *Dexterous Robot Hands*, (Springer, New York 1990)
19.18 A. Bicchi: Hands for dexterous manipulation and robust grasping: A difficult road toward simplicity, IEEE Trans. Robotics Autom. **16**(6), 652–662 (2000)
19.19 M.R. Cutkosky: On grasp choice, grasp models, and the design of hands for manufacturing tasks, IEEE Trans. Robotics Autom. **5**(3), 269–279 (1989)
19.20 G. Berselli, M. Piccinini, G. Vassura: Comparative evaluation of the selective compliance in elastic joints for robotic structures, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 4626–4631
19.21 L.U. Odhner, A.M. Dollar: Dexterous manipulation with underactuated elastic hands, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)
19.22 J. Butterfass, M. Grebenstein, H. Liu, G. Hirzinger: DLR-Hand II: Next generation of a dextrous robot hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Seoul (2001)
19.23 C. Melchiorri, G. Vassura: Mechanical and control features of the UB hand version II, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1992)
19.24 R.O. Ambrose, H. Aldridge, R.S. Askew, R.R. Burridge, W. Bluethmann, M. Diftler, C. Lovchik, D. Magruder, F. Rehnmark: Robonaut: NASA's space humanoid, IEEE Intell. Syst. (2000)
19.25 L. Birglen, C.M. Gosselin: Kinetostatic analysis of underactuated fingers, IEEE Trans. Robotics Autom. **20**(2), 211–221 (2004)

19.26 I. Yamano, T. Maeno: Five-fingered robot hand using ultrasonic motors and elastic elements, Proc. IEEE Int. Conf. Robotics Autom. (2005) pp. 2684–2689
19.27 Shadow Dexterous Hand: Shadow Robot Co. LTD., London (2015), http://www.shadowrobot.com/
19.28 M. Kaneko, M. Higashimori, R. Takenaka, A. Namiki, M. Ishikawa: The 100G capturing robot – too fast to see, Proc. 8th Int. Symp. Artif. Life Robotics (2003) pp. 291–296
19.29 G. Palli, C. Natale, C. May, C. Melchiorri, T. Würtz: Modeling and control of the twisted string actuation system, IEEE/ASME Trans. Mechatronics **18**(2), 664–673 (2013)
19.30 G. Palli, C. Melchiorri, G. Vassura, G. Berselli, S. Pirozzi, C. Natale, G. De Maria, C. May: Innovative technologies for the next generation of robotic hands, Springer Tracts Adv. Robotics **80**, 173–218 (2012)
19.31 G. Palli, S. Pirozzi: Force sensor based on discrete optoelectronic components and compliant frames, Sensors Actuators A **165**, 239–249 (2011)
19.32 G. Palli, S. Pirozzi: A miniaturized optical force sensor for tendon-driven mechatronic systems: Design and experimental evaluation, Mechatronics **22**(8), 1097–1111 (2012)
19.33 W. Paetsch, M. Kaneko: A three fingered multi-jointed gripper for experimental use, Proc. IEEE Int. Workshop Intell. Robots Syst. (IROS) (1990) pp. 853–858
19.34 H. Maekawa, K. Yokoi, K. Tanie, M. Kaneko, N. Kimura, N. Imamura: Development of a three-fingerd robot hand with stiffness control capability, Mechatronics **2**(5), 483–494 (1992)
19.35 A. Bicchi: A criterion for optimal design of multiaxis force sensors, J. Robotics Auton. Syst. **10**(4), 269–286 (1992)
19.36 A. Pugh: *Robot Sensors: Tactile and Non-Vision*, Vol. 2 (Springer, Berlin, Heidelberg 1986)
19.37 H.R. Nicholls, M.H. Lee: A survey of robot tactile sensing technology, Int. J. Robotics Res. **3**(3), 3–30 (1989)
19.38 W.T. Townsend, J.K. Salisbury: Mechanical bandwidth as a guideline to high-performance manipulator design, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989)
19.39 S.D. Eppinger, W.P. Seering: Three dynamic problems in robot force control, IEEE Trans. Robotics Autom. **8**(6), 751–758 (1992)
19.40 W.T. Townsend, J.K. Salisbury: The effect of Coulomb friction and stiction on force control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987)
19.41 G. Palli, G. Borghesan, C. Melchiorri: Modeling, identification and control of tendon-based actuation systems, IEEE Trans. Robotics **28**(2), 277–290 (2012)
19.42 M. Kaneko, T. Yamashita, K. Tanie: Basic considerations on transmission characteristics for tendon driven robots, Proc. 5th Int. Conf. Adv. Robotics (1991) pp. 827–883
19.43 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)
19.44 J. Mason, J.K. Salisbury: *Robot Hands and the Mechanics of Manipulation* (MIT Press, Cambridge 1985)
19.45 R.M. Cutkosky: *Robotic Grasping and Fine Manipulation* (Springer, New York 1985)

Multimedia Contents

20. Snake-Like and Continuum Robots

Ian D. Walker, Howie Choset, Gregory S. Chirikjian

This chapter provides an overview of the state of the art of snake-like (backbones comprised of many small links) and continuum (continuous backbone) robots. The history of each of these classes of robot is reviewed, focusing on key hardware developments. A review of the existing theory and algorithms for kinematics for both types of robot is presented, followed by a summary of modeling of locomotion for snake-like and continuum mechanisms.

20.1 Snake Robots – Short History 481
20.2 Continuum Robots – Short History 485
20.3 Snake-Like and Continuum Robot Modeling 487
20.4 Modeling of Locomotion for Snake-Like and Continuum Mechanisms 491
20.5 Conclusion and Extensions to Related Areas 492
Video-References 492
References 493

20.1 Snake Robots – Short History

The study of snake robots began in earnest with the pioneering work of Shiego Hirose, a professor at the Tokyo Institute of Technology. He called his snake robots these active chord mechanisms, or ACMs for short. The first successful locomotive snake was ACM III (Fig. 20.1a), which he developed between 1972 and 1975; the ACM III robot had 20 segments and used passive wheels to form a constraint between the mechanism and the environment. Each segment was able to yaw with respect to each other and the combined motion of these degrees of freedom propelled the entire robot forward [20.1]. This type of coordinated control resulting in locomotion is called a *gait*; more formally, a gait is a cyclic motion in the mechanisms internal degrees of freedom that result in net motion for the snake robot. The next step in the development of the Hirose family of snake robots were the OBLIX (Fig. 20.1b) (1978–1979) and MOGURA (1982–1984), two robots that could achieve three-dimensional movements and lift their heads by rotating their oblique joints. The benefit of a single actuator operating an oblique joint over a true three-dimensional joint is simplicity, light weight, and ease to control because each segment only needed to be actuated by one motor [20.2].

Hirose's lab then began exploring possible applications of snake robots with Koryu I and II (1985–present). Each one of Koryu's load-bearing modules uses actuated joints to lift itself, lower itself, and in more recent incarnations, angle its wheel base to fit the ground. With such forms of actuations, these large snake robots were able to traverse uneven terrain and even climb stairs, all while carrying a load, distributed along the entire robot [20.2]. Inspired by the ability of real snakes to traverse uneven terrains, Hirose's lab developed a smaller mechanism, called Souryu I and II (1975–present) for the purposes of search and rescue (Fig. 20.2). The Souryu robots were able to climb over rubble because of their low center of mass and treads all along their bodies [20.3, 4].

In 1999, *Hirose* returned to the ACM with Slim Slime, a robot that was pneumatically actuated and formed a smooth curve instead of the usual sectioned, angular curve of other snake robots [20.5]. After Slim Slime, the lab began working on ACM-RI, R2, R3, and R4 (2001–present.) These robots are much closer to the design of the first ACM, with passive wheels and many-sectioned bodies. These robots, unlike the first ACM, can lift their bodies and move in three dimensions. Each

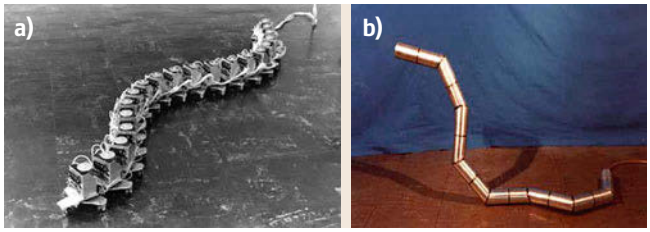


Fig. 20.1 Hirose's snake robots starting **(a)** with the active chord (ACM III) and **(b)** Oblix/Mogura mechanisms (after [20.2])

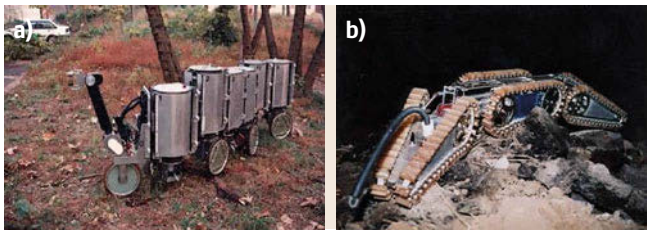


Fig.20.2a,b The Koryu I, II robot drving outside **(a)** (after [20.2]) and **(b)** the Souryu I, II robot maneuvering over rubble (after [20.4])

section has wheels that are set 90° from each other in an alternating fashion. They were created for use in search and rescue [20.1]. The most recent ACM is R5, a fully amphibious robot that uses passive wheels to roll along the ground and paddles to move through the water [20.6] (Fig. 20.3).

Snake robot research began in the United States in the late 1980s. Joel Burdick and his then-student Greg Chirikjian at Caltech developed a continuum theory to model the shape of snake robots in general. Burdick and Chirikjian coined the term *hyper-redundant manipulator*, which means a mechanism that has many more degrees of freedom that are necessary to position and orient its end-effector. They built Snakey in 1992; Snakey consisted of three degree of freedom bays or actuated trusses that could both manipulate and locomote. Their focus was to create a strong, precise mechanism that focused less on snake-like movement and more on the difficulties of programming a mechanism with many degrees of freedom. The Caltech team considered applications such as retrieval of satellites and search and rescue [20.7, 8]. Burdick and his group also designed and built snake robots with the Jet Propulsion Laboratory and the NEC Corporation with Nobuaki Takanashi. These were based on a three-dimensional joint prototype built by Takanashi. This smaller snake was able to execute several movement gaits, and was intended to be used in search and rescue.

Chirikjian moved on to Johns Hopkins University where he created a comprehensive research program on a broad class of hyper-redundant mechanisms, not limited to snake robots. One such nonsnake robot was the metamorphic shape-changing modular robot system comprising hexagonal elements. His group also developed snake robots based on binary actuators, which have two states, say fully elongated and contracted [20.9].

Snake and reconfigurable modular robot research also co-developed at Xerox Parc, and then at the University of Pennsylvania under the direction of Mark Yim. Beginning with PolyPod [20.10] in 1994 as part of Mark Yim's dissertation while he was at Stanford, snake configurations of modular robots were performing many of the conventional snake gaits as well as some innovative slinky-like and rolling gaits [20.11]. This research continued after Yim's move to Xerox's Palo Alto Research with PolyBot in 2000, a more robust version of PolyPod with more abilities for sensing [20.12]. When *Yim* began Modlab at the University of Pennsylvania, PolyBot evolved into CKbot ([20.13], 2009). CKbot can now reconfigure from an exploded position, run on four legs, walk on two, and perform most basic snake gaits [20.14, 15]. They also developed foamBot (2011), a robot that can create other robots, including snake robots, out of CKbot modules and hardening spray foam [20.16].

Hirose, Chirikjian, Burdick, and Yim had a strong influence on Howie Choset's work at Carnegie Mellon University (CMU). Choset's snake robot research began in the CMU Biorobotics Lab as an undergraduate research project which adapted Yim's modular design specifically for snake robot use (Fig. 20.7). Iterating on this modular design, this lab has created a series of snake robots over 15 years [20.17, 18]. In

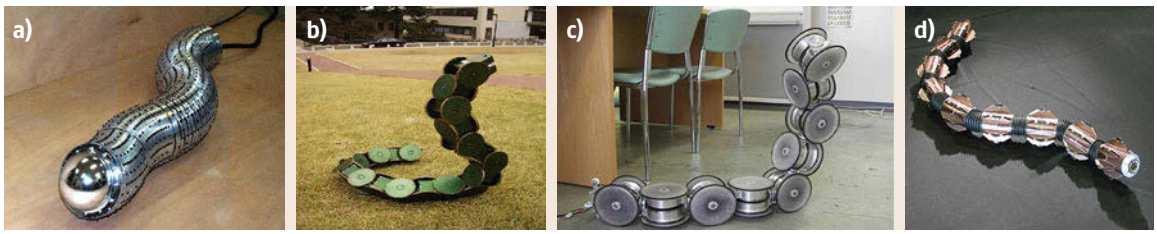


Fig.20.3a–d Slim Slime **(a)** and the new generation of ACM-R snake robots: **(b)** ACM-R3; **(c)** ACM-R4; **(d)** ACM-R5 (after [20.1])

Fig. 20.4 Snakey, the snake robot showing also Joel Burdick, Greg Chirikjian, Howie Choset, Jimbo Ostrowski (after [20.8])

addition to the mechanism, one of the major innovations of this work is gaits, the controllers that prescribe motion for the snake robots. These gaits generalized the gait motion, based on Hirose's serpenoid curve and Chirikjian's modal functions, to perform a large variety of behaviors including climbing up poles, swimming in a pond, maneuvering through pipes, and breaching through a fence [20.19]. Working with Robin Murphy's group at Texas A&M, the snake robots have been fine tuned for urban search and rescue use. This created an ideal platform for infrastructure inspection such as networks of pipes in power plants. In 2012, Choset sent his robots to Egypt for archaeology in confined spaces [20.20].

In Norway, Øyvind Stavdahl in the Gemini Centre for Advanced Robotics created several snake robots, many of which were in collaboration with the Norwegian University of Science and Technology. The first of these major research projects in snake robotics was the Anna Konda (2005-present), a firefighting snake concept. Based on hydraulic actuation, this robot is powered entirely by the pressure of a fire hose, and is designed to reach flames in areas that are either too dangerous or too small for firefighters to reach [20.21].

This team also created AIKO (2006–2010), a snake robot created to experiment with snake locomotion that could later be applied to Anna Konda [20.22]. Mostly focused on object-aided locomotion, AIKO evolved into KULKO (2010–present), a robot with a force-sensing body that uses its simple understanding of the obstacles it is in contact with in order to locomote more effectively. These two robots have produced several innovative movement algorithms for object aided locomotion that could be applied to Anna Konda as well as other nonwheeled snake robots [20.23].

The snake robots described above rely on internal motions to propel themselves forward. *Johann Borenstein* at the University of Michigan created the Omni-Tread (2004) robot [20.24] that propels itself with moving threads situated along side of the robot. Its strong joints are actuated by pneumatic bellows. The bellows can be proportionally controlled, allowing its body to be fully compliant while crossing rubble to have more of its body touching the ground, or strong enough to lift half of its body weight, which is necessary for ledges and gaps. The Omni-Tread excelled at moving over rubble [20.24].

David Anhalt and James McKenna at SAIC took the tread concept to the limit; they designed and built a toroidal skin drive, a drive mechanism that has reciprocating skin that consists of an elongated toroidal skin that covers the entire length of the robot, and a drive unit which propels it. The outer (tubular) layer of the skin slides axially from the head to the tail of the snake robot, and wraps inside itself over a captured ring at the tail. The skin then recirculates from the tail to the head (through the center of the outer tubular layer), and changes direction again (over a second captured ring at the head) to again become the outside layer. In this configuration, the skin forms a continuous toroidal loop. The SAIC team collaborated with Choset's group to build a snake robot using this drive mechanism [20.25].

Finally, we describe some snake robots that use wheels to locomote. Hisashi Date, who heads a research team at the Robotics Lab for National Defense Academy of Japan, built a snake robot in 2007 to trace lines using lateral undulations and a line-tracing sensor arm. Date later developed in 2009 a mechanism that has passive wheels; it is very similar to Shigeo Hirose's first ACM robots, except it is powered by hydraulics. Date's robot also implements touch sensing much like being planned for Anna Konda [20.26].

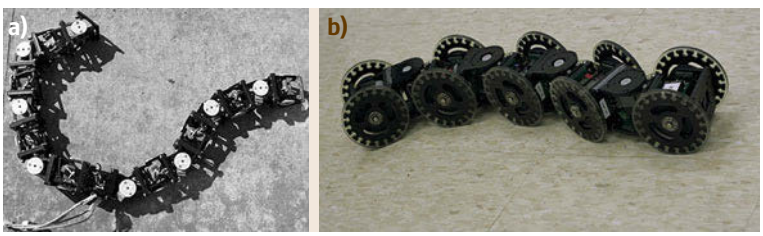


Fig. 20.5 **(a)** Yim's original Polypod robot (after [20.10]) and **(b)** current CK Robot in a wheeled snake configuration (after [20.13])

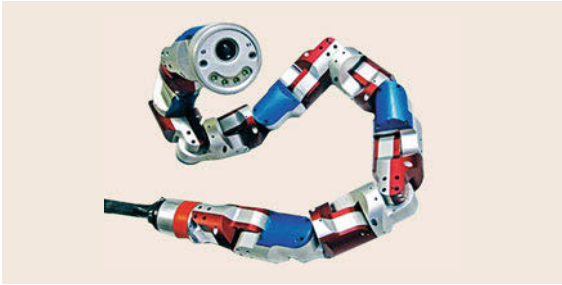


Fig. 20.6 Choset's Modsnake consisting of 16 DOFs (degree of freedom) each orthogonally placed with respect to each other (after [20.17, 18])

Fig. 20.7 Anna Konda snake robot (after [20.21])

Another wheeled snake robot is HAUBOT (2009-present), developed by Akio Ishiguro at Tohoku University. It is being used to study the autonomous decentralized control mechanisms underlying the adaptive locomotion of animals. It uses real-time tunable rubber springs to actuate each joint. Each joint generates torque proportional to the difference between the joint angle and the target angle which is given by the anterior joint angle. By using just this method of simple sensing, the robot can locomote in two dimensions, without requiring a centralized control hub. More recently, HAUBOT has been successfully paired with head tracking software so that it can be driven intuitively [20.27].

At Kyoto University, Fumitoshi Matsuno has also developed several snake robots. Matsuno has made many innovations in the control of snakes with different modes of propulsion. In 2002, Matsuno and his lab built KOHGA, a treaded snake robot made for search and rescue. Its joints can be passive or active, allowing it to move through rough terrain and also lift parts of its body [20.28]. Another robot they have developed is a snake robot that propels itself using screw-inspired modules. Built in 2010, it uses many of the same control algorithms as KOHGA except besides moving forward, backward, and turning, it can also roll sideways perpendicular to its head direction [20.29].

These labs together have created an incredible variety of snake-like robot locomotion. Snake robots can now traverse nonflat fields, climb poles, swim underwater, and even rebuild themselves. They can access incredibly small spaces and transition between swimming to slithering to climbing without having to be reconfigured. Advancing far beyond slithering across flat ground, these robots are nearly ready to begin tackling spaces in the field that no other robot can reach.

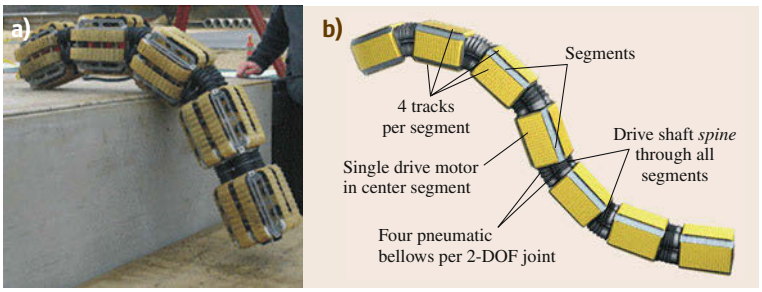


Fig.20.8a,b The Omni-Tread treaded snake robot (a) and an (b) illustration of its design (after [20.24])

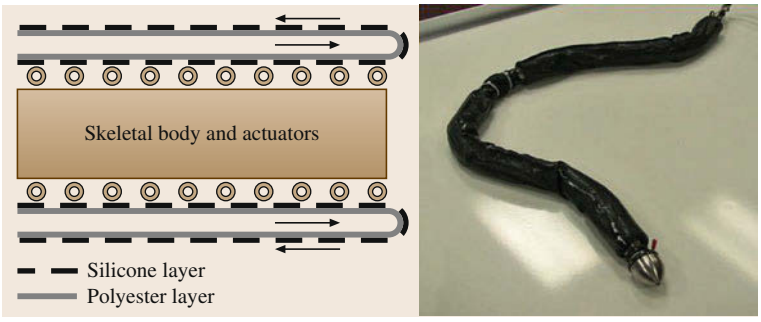


Fig. 20.9 The SAIC skin drive mechanism followed by the SAIC/CMU snake robot powered by the skin drive mechanism (after [20.25])

Soon these robots may be found in the pipes of oil refineries, crawling through nuclear power plants, at the front line of search and rescue operations, fighting fires or exploring underwater environments.

20.2 Continuum Robots – Short History

The term *continuum manipulator*, first coined by *Robinson* and *Davies* in [20.30], which is equivalent to the *continuous morphology manipulator* defined in [20.31], indicates the snake/hyper-redundant robot concept taken to the extreme. In other words, conceptually this category of robots have backbone structures *taken to the limit*, with their number of *joints* tending to infinity, but with their *link lengths* tending to zero, which is also the approach taken in Chirikjian and Burdick's work on backbone curves. While this concept initially appears idealistic, complicated and impossible to implement, the limiting case (a smooth *continuum* curve, with the ability to bend at any point along the backbone) is quite easy to realize in hardware.

Numerous continuum designs have been proposed since the early 1960s beginning with the *Tensor Arm* [20.32] developed based on the original *Orm* concept of Leifer and Scheinman [20.33]. With a flexible backbone bent by remotely driven tendons, the Tensor Arm anticipated numerous later designs. However – a key issue with later continuum robot designs until the 1990s – coordination of the inputs to take advantage of the shapes achievable by the robot proved difficult. This lack of enabling underlying models for motion planning (now much less of an issue, see the following section) hindered the development of the field for many years.

Nevertheless, some researchers, beginning with *Shigeo Hirose* [20.2], continued to explore the problem of constructing continuum robots. Many of these efforts were inspired by an evolving knowledge of the morphology and functionality of biological continuum structures [20.34–36] particularly the trunks of elephants [20.37–39] and the arms of octopuses [20.40–42]. The key robot design issues revolve around how to actuate (bend and possibly extend/contract) the backbone. Two basic design strategies have emerged [20.43, 44]: (1) extrinsic actuation, where the actuators are separate from the backbone structure, e.g., [20.45–49]; and (2) intrinsic actuation, where the actuators represent a fundamental part of the backbone, e.g., [20.50–53].

Extrinsically actuated designs have the advantage that by locating the actuators outside the robot workspace, the backbone itself can be simplified and streamlined. Tendons have proved a popular and generally successful choice for extrinsically actuated continuum designs [20.38, 54, 55]. Groups of tendons are terminated at specific points down the backbone, and combine to bend the backbone to produce a series of (usually constant curvature) *sections*. The tendons bend a backbone that is nominally straight (realized either via an elastic rod [20.56], the incorporation of springs [20.38], or pneumatic chambers [20.57, 58]). A good example of the ability of this design strategy to produce a long, thin backbone with extrinsic actuation is given by NASA Johnson Space Center's Tendril ([20.49], Fig. 20.10), intended to penetrate through small orifices and explore in tight spaces for Space applications.

In recent years, a completely different form of extrinsically actuated continuum robot, based on concentric tubes, has appeared. These concentric tube [20.59, 60] (sometimes termed as active cannula [20.61]) robots have been primarily motivated by medical applications [20.62]. Research has focused on application (e.g., as active endoscopes [20.63–66]) within the human body [20.67, 68], on various types of minimally invasive surgical procedures [20.69–71], and the associated development of *steerable needle* technologies [20.72–74].

Concentric tube robots, as the name suggests, form the robot backbone from several (typically two or three) hollow tubes, with tubes of smaller radius inserted through those of larger radius. Each smaller tube can be pushed through the tube immediately outside it to extend the backbone length. The tubes can also be rotated, to give $2n$ degrees of freedom for an n-tube backbone. The tubes are preformed to have a constant curvature, so can be extended and bent (albeit with amount

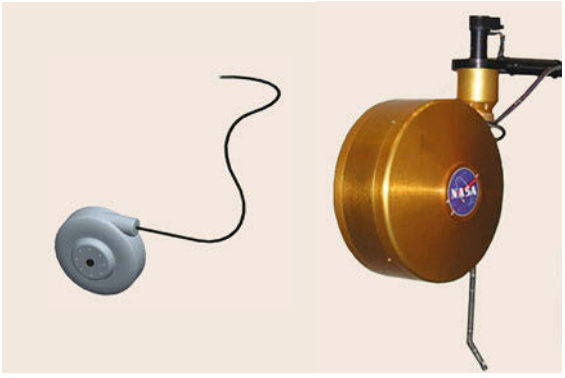


Fig. 20.10 NASA's Tendril, an extrinsically actuated continuum manipulator (after [20.49])

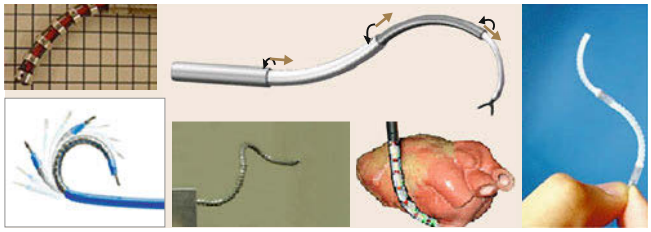


Fig. 20.11 Extrinsically actuated concentric tube continuum manipulators

Fig. 20.12 The OC Robotics *Snake-Arm* robot – an extrinsically actuated *continuum-like* robot with segmented backbone (after [20.75])

Fig. 20.13 The intrinsically actuated *Octarm* pneumatic continuum manipulator and its development team (after [20.42])

of bending preset) in arbitrary directions (Fig. 20.11). This design arrangement is well-suited to the medical applications which have motivated its development. Numerous prototypes have been developed in the last several years, and the results of initial trials are highly promising. This work is being pioneered by the groups of *Pierre Dupont* at Boston University [20.59, 73, 74, 76], *Michael Zinn* et al. at the University of Wisconsin [20.62, 77], and *Robert Webster*, III et al., at Vanderbilt University [20.44, 61, 67, 78].

Note that the backbone for extrinsically actuated robots does not have to be continuous to produce the continuum effect [20.47, 79, 80]. A commercially successful group of extrinsically actuated continuum-like *snake-arm* manipulators are currently manufactured and retailed by OC Robotics (OCR) in the United Kingdom [20.75]. Remotely tendon-driven (designed by Rob Buckingham and Andy Graham of OCR), but with a snake-like segmented backbone (the number and form of segments can be changed) these snake-arms give a continuous backbone effect. OCR's manipulators (Fig. 20.12) are currently the only general purpose continuum-like robot commercially available, and have been deployed in nuclear reactors and inside airframes, among other applications for largely noncontact operations in tight, cluttered spaces. This ability to penetrate and *slither through* complex environments (mirroring NASA's goals for Tendril) is one of the main motivations and anticipated practical advantages of continuum robots.

Intrinsically actuated continuum robots have their actuators within the continuum backbone. Indeed, the backbones of most intrinsically actuated continuum robots are largely formed by their actuators. Artificial muscle technologies, notably pneumatic McKibben muscles, have proved most effective in intrinsically actuated hardware realizations [20.50, 81]. A particular advantage of intrinsic actuation is the ability to produce extensible backbones [20.43, 44].

For example, Ian Walker's group at Clemson led the development of the octopus-arm inspired *Octarm* instrinsically actuated extensible continuum manipulators ([20.42], Fig. 20.13, VIDEO 158). The backbone of the Octarm is formed from its actuators: sets of (3) McKibben muscles arranged in series to give three or four sections. Several generations of Octarm were tested in field trials, mounted on a Foster-Miller Talon mobile base [20.81] (VIDEO 157). The robots were shown to be able to grasp and manipulate objects over a wide range (order of magnitude) of sizes and weights, and of widely varying shapes and textures [20.82]. This

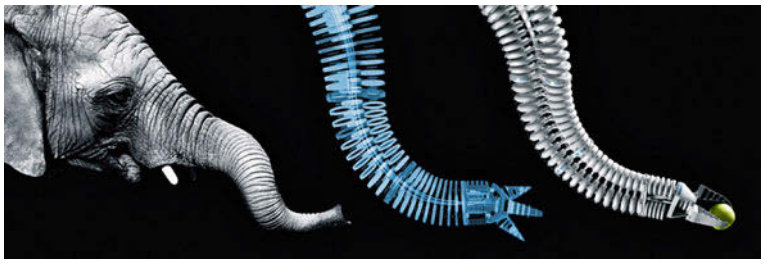


Fig. 20.14 The intrinsically actuated bionic handling assistant by *Festo* (after [20.50])

ability to adapt shape to perform *whole-arm manipulation* [20.83] with the backbone is another prime motivation for continuum robots [20.84].

Another good example of an intrinsically actuated continuum robot is the Bionic Handling Assistant recently developed by *Festo* [20.50] (Fig. 20.14). This continuum robot is unusual in that it is preshaped for its sections to be nonconstant curvature. The curvature is biologically inspired.

Currently, another biologically inspired (once again by the octopus) large project, under EU funding, led by Darwin Caldwell, and featuring researchers in Greece, Israel, Italy, Switzerland, and the UK, is working toward creation of a *European Octopus* robot [20.40, 85] (Fig. 20.15). This robot features multiple continuum limbs [20.85]. The European group has considered both intrinsic [20.86] and extrinsic actuation [20.40] for their limbs.

Many snakelike/continuum robotic devices have found applications in the medical area. This includes very flexible steerable needles [20.87–91], and the concentric-tube robots mentioned earlier [20.73, 74, 92–97]. Other recent medical snake robots include novel articulated designs [20.98, 99], and stiff-continuum-filament cable-driven designs [20.100–103].

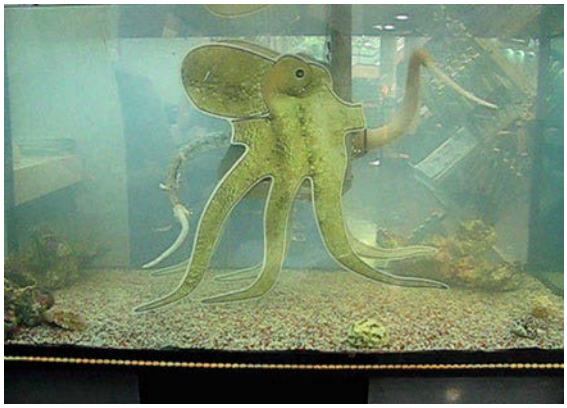


Fig. 20.15 The *European Octopus* continuum-limbed robot (after [20.40]) (currently under development)

20.3 Snake-Like and Continuum Robot Modeling

While continuum manipulators are seen to be the extreme case of snake-like or hyper-redundant structures, it was quickly realized that they represent a fundamentally new class of manipulators [20.43, 44, 104]. A continuous flexible backbone features, at least in theory, an infinite number of degrees of freedom. As such, the traditional tools in robotics for modeling (a finite number of) serial rigid links no longer apply. Additionally, it is obviously not possible to actuate an infinite number of degrees of freedom in practice. Continuum manipulator hardware universally features a finite number of actuators, applying forces/torques to the backbone at a fixed and preselected set of locations. This causes significant complexity in their analysis. However, significant progress has been made, and understanding resulting from analysis of continuum kinematics has been applied, as discussed in the following section.

Clearly, to correctly model continuum manipulators, models based on continuous backbones are required. Interestingly, such models have also proved to be key theoretical resources in the motion planning of rigid-link hyper-redundant systems. While in theory discrete-link hyper-redundant manipulators can be modeled (and their planning based on) traditional manipulator modeling techniques, it was quickly found that the computational complexity of such approaches made them difficult to implement, and the corresponding models hard to visualize. An alternative strategy (and a more successful one in practice) has been to adopt an approach based on continuum manipulator kinematics. In the following, we overview the recent development of continuum manipulator kinematics. We then discuss how the existence of both discrete-link and continuum models have enhanced motion planning for each category of hyper-redundant system.

The lack of distinct links in continuum systems makes the standard robot manipulator modeling strategy of a finite number of coordinate frames (each fixed in one link) inappropriate for their modeling. Instead, the natural approach is to model continuum kinematics via a frame which evolves along a continuous backbone, parameterized by arc length s. Local motion of the backbone at point s is modeled in terms of the local frame at s. This strategy allows computation of *forward kinematics*, and construction of *continuum Jacobians*, analogous to those for rigid-link systems [20.31, 105].

Numerous alternatives for selection of the backbone frames have been proposed [20.31, 106–109]. In all such approaches, positions along the backbone curve can be expressed relative to the unit tangent, $u(s)$, as

$$x(t) = \int_0^t [1 + \varepsilon(s)]u(s)\,\mathrm{d}s\,.$$

In this formulation, $\varepsilon(s)$ is the local extensibility, or stretch, of the curve. If this quantity is equal to zero, then t becomes arc length. Otherwise, equal increments of t can correspond to different lengths along the curve, with a positive value of $\varepsilon(s)$ corresponding to a stretch, and negative values corresponding to a compression. While most continuum robots have a backbone that is essentially inextensible, the reason for introducing this variability is that allowing certain sections to stretch or compress affords greater freedom to hyper-redundant manipulators with discrete stages, making it easier to *fit* the physical robot to the curve. This fitting process takes as an input not only the relative positions of the distal and proximal ends of each stage, but also orientations. If the tangent is taken as one of the axes of a reference frame, a choice for the other two axes remains and can be made in a number of different ways. Once this choice is made, a reference frame

$$g(t) = (R(t), x(t))$$

is established at each value of t and relative rigid-body displacements are computed for stages as

$$[g(t_i)]^{-1} \circ g(t_{i+1}) = \left([R(t_i)]^{\mathrm{T}} R(t_{i+1}), [R(t_i)]^{\mathrm{T}} (x(t_{i+1}) - x(t_i))\right) .$$

In the inextensible case where the curve parameter becomes arclength s one natural choice for defining the orientation $R(t)$ is the well-known Serret–Frenet frame [20.31, 108, 109] which evolves along the backbone according to the following

$$\begin{aligned} \mathrm{d}t/\mathrm{d}s &= \kappa n \\ \mathrm{d}n/\mathrm{d}s &= -\kappa t + \tau b \\ \mathrm{d}b/\mathrm{d}s &= -\tau n \, . \end{aligned}$$

In the above, the frame origin is given by x, where the unit tangent to the curve $t = \mathrm{d}x/\mathrm{d}s$ (denoted as t rather than u), and forms one of the frame axes. The other axes are defined as the normal ($t \cdot n = 0$) and the binormal $b = t \times n$. The curvature κ and torsion τ dictate the shape of the curve. The axes of the Frenet–Serret frame provide an intuitive visualization of local movement: two possible dimensions of bending, corresponding to rotations about the normal and binormal axes, and one of extension/contraction (present and controllable in several continuum robots), corresponding to translation along the tangent axis.

The Frenet–Serret model has the advantage of being a well-established means for modeling continuous spatial curves. The drawback of this formulation is that it is not *optimal* in the sense that for a given curve shape, the total amount of motion of the evolving reference frame required to traverse the curve is not minimal subject to end constraints and the requirement that one axis of the reference frame is the tangent. Variational calculus can be used to obtain the optimal framing of curves, as reviewed in [20.31] in the context of coordinate-dependent optimization, and in [20.110] using a coordinate-free approach.

Moreover, there are multiple approaches to defining the evolution of the tangent, and hence the shape of the curve. Consider an inextensible curve whose shape has not yet been specified, and let the unit tangent to the curve be defined relative to an a priori-unknown arclength-dependent orientation as

$$u(s) = R(s)e_3 \, .$$

The goal is to find $R(s)$ such that the positional constraint

$$x(L) = \int_0^L u(s)\,\mathrm{d}s \, ,$$

and the orientational constraints on $R(L)$ are satisfied when $R(0) = I$. Here, I is the identity matrix and the manipulator has length L.

Treating s as time, then corresponding to $R(s)$ is an angular velocity $\omega = (R^{\mathrm{T}}\,\mathrm{d}R/\mathrm{d}s)^V$, where the V operation extracts the dual vector of a skew-symmetric matrix. We can then ask what $R(s)$ is optimal in the sense of minimizing a cost of the form

$$I = \frac{1}{2}\int_0^L [\omega(s) - b]^{\mathrm{T}} B[\omega(s) - b]\,\mathrm{d}s \, .$$

When $b = 0$ and B is the moment of inertia of a cross section of the manipulator, this is akin to kinetic energy of rotation, and minimizing it provides an evolution of reference frames that minimally rotate. Variational calculus provides the conditions

$$B\dot{\omega} + \omega \times (B\omega - b) = \begin{pmatrix} -\lambda^{\mathrm{T}} R e_2 \\ \lambda^{\mathrm{T}} R e_1 \\ 0 \end{pmatrix} ,$$

that can be integrated numerically and the rotation matrix $R(s)$ is obtained from the angular velocities by integrating

$$\dot{R} = R\left[\sum_{i=1}^{3} \omega_i(s) E_i\right] ,$$

where

$$E_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} ,$$

$$E_2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} ,$$

$$E_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} .$$

The three-dimensional vector of Lagrange multipliers λ and the a priori unspecified initial value of the three-dimensional vector ω at $s = 0$ together provide six degrees of freedom with which to match the end constraints on position and orientation. In this way, curves of fixed length that vary in shape as little as possible while meeting end constraints can be generated.

Alternative frame representations in the continuum robot literature also have selected the axes of the frame to align with the controlled motion axes of particular hardware realizations [20.47, 107]. In either case, given a frame, the key problem is how to use the resulting model to plan motions. The basic issue is that the continuum kinematics models in their basic form feature infinite degrees of freedom (as necessary to model arbitrary spatial curves). However, hyper-redundant robots (discrete-joint or continuum) can be controlled in only a finite number of ways, thus admitting a reduced set of physical solutions. Therefore, research in continuum manipulators has concentrated on how to constrain continuum kinematics models to best represent robot hardware.

The initial major breakthrough in motion planning for hyper-redundant arms was made in a landmark series of publications by *Chirikjian* and *Burdick* [20.31, 105, 111] introducing continuum kinematics and using them to approximate rigid-link hyper-redundant systems [20.105]. Basically, the philosophy is to use a (theoretical) curve to model the physical backbone of a hyper-redundant robot. Motion planning is performed for the curve, and the (discrete) robot backbone then *fit* to the resulting (continuous) solution curve. This approach has proved quite effective, and the associated research introduced several key theoretical concepts to the area. In particular, the use of a modal approach [20.105, 112] (restricting the classes of allowable solutions to shapes generated via simple linear combinations of *modes*), arose from this approach to redundancy resolution for hyper-redundant manipulators. This concept can be viewed as a *top-down* approach of building a general model, and adapting it (via mode selection and curve-fitting) to hardware.

In the planar case, the angle of the rotation matrix $R(s)$ is simply the integral of the curvature of the curve. Expressing this curvature as a weighted sum of two functions, or modes, then restricts the manipulator to act as if it only has two degrees of freedom. As an example of this modal approach, a closed form solution can be obtained for inextensible manipulators of unit length by restricting the curvature to be a linear combination of $\cos 2\pi s$ and $\sin 2\pi s$. If the coefficients in front of these functions are, respectively, $2\pi a_1$ and $2\pi a_2$, then

$$x_{ee} = \sin(a_2) J_0 \left[(a_1^2 + a_2^2)^{\frac{1}{2}} \right] ,$$

$$y_{ee} = \cos(a_2) J_0 \left[(a_1^2 + a_2^2)^{\frac{1}{2}} \right] ,$$

where J_0 is the zeroth-order Bessel function.

And inverse kinematics can be achieved in closed form by computing

$$\begin{aligned} a_1 &= \hat{a}_1^{\pm}(\bar{x}_{ee}) \\ &= \pm \left(\left\{ J_0^{-1} \left[(x_{ee}^2 + y_{ee}^2)^{\frac{1}{2}} \right] \right\}^2 \right. \\ &\quad \left. - [\mathrm{Atan2}(x_{ee}, y_{ee})]^2 \right)^{\frac{1}{2}} \\ a_2 &= \hat{a}_2(\bar{x}_{ee}) = \mathrm{Atan2}(x_{ee}, y_{ee}) . \end{aligned}$$

In this planar example the convention that the tangent to the curve at the base of the manipulator points along the y axis has been used.

This closed-form solution has also been used to generate closed-form solutions for spatial manipulators, as well as to define waves that travel along the manipulator for the purpose of obstacle avoidance, locomotion and manipulation. This is illustrated in Figs. 20.16–20.18 [20.31] with a variable-geometry truss superimposed on the backbone curve. In the context of obstacle avoidance, the modal solution given above is scaled in order that the part of the manipulator outside of the obstacle field terminates at o position to allow entry, and the section inside of the obstacle field has curvature that is piecewise constant to maneuver around the obstacles.

In the context of locomotion, the modal approach is used to ensure that the end conditions on a traveling wave match those of terrain. In the context of grasping

and manipulation, the modal approach is used in two ways. First, it defines the shape of a locomotion-like traveling wave that traverses the grasped object. This extends the reach of the manipulator around the object. Then, the bottom part of the manipulator contracts to form a shape defined by the modes while the part of the manipulator that is closest to the base yet in contact with

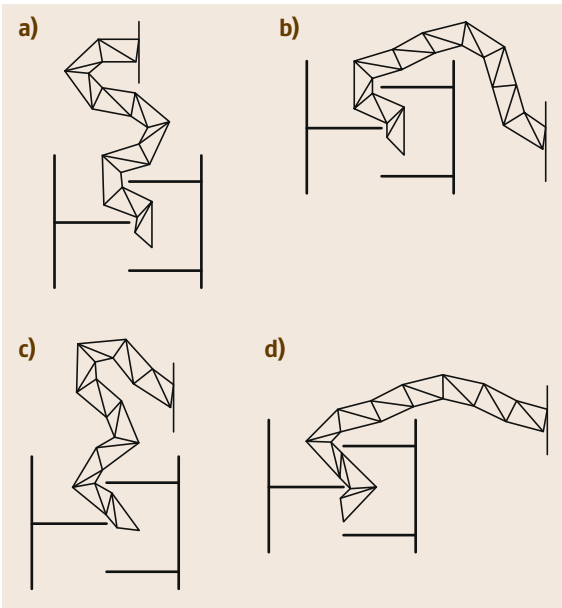


Fig.20.16a–d Using the modal approach to position part of a manipulator at the entry of an obstacle field (after [20.31])

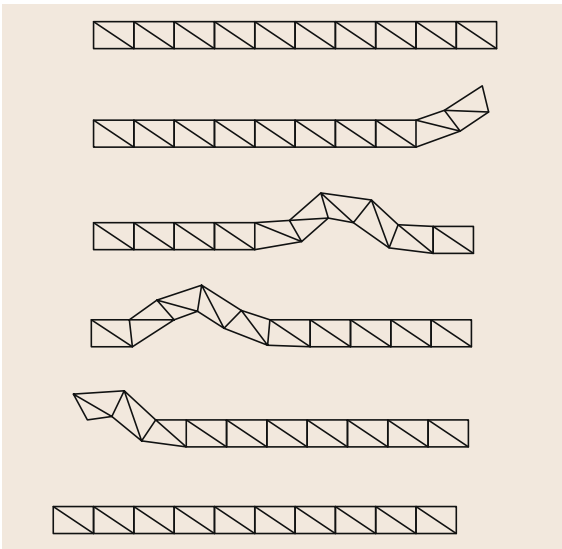


Fig. 20.17 Net motion to the left by wave propagation (after [20.31])

the grasped object unwinds. The result is a combination of waves and unwinding that produce net rotation of grasped objects without ever letting go, as described in [20.31, 113].

More recent research has concentrated on the notion *dual* to that of the previous section, using the physical constraints imposed on the backbones of specific continuum robot hardware types to construct continuum kinematics. This can be viewed as a *bottom-up* approach, focused on particular hardware classes, with the main aim of sufficiently modeling the hardware to avoid the use of approximations in the motion planning. Examples of this approach are given in [20.44, 107] where the key constraint imposed on a general continuum kinematics model arises from the observation that many continuum hardware implementations resolve into a finite number of constant curvature sections. (This arises naturally from the imposition of a finite number of input force/torques on a stiff continuum backbone). In [20.106], it is noted that the resulting constant curvature models can be viewed as a particular case of the general modal approach of [20.105].

The approach shown [20.44, 107] involves several transformations [20.114], to transition the map between task and actuator space via a *section* space. As key part of this transformation, a conventional (theoretical) rigid-link model can be used to model the kinematics of each section. Thus, rigid-link kinematics have proven key to resolving redundancy for continuum manipulators, as continuum kinematics play a key part in redundancy resolution for rigid-link hyperredundant arms.

Finally, a series of Jacobians are formed. The difference here is that the continuum Jacobian is a function of locally meaningful variables (bending angle, curvature, and extension) defining section shape or the direct values of actuators which determine these variables. Given the existence of continuum Jacobians, the redundancy is usually resolved as

$$\dot{\mathbf{l}} = J_E^{\dagger} \dot{l}_E + \left(I - J_E^{\dagger} J_E\right) \dot{\mathbf{l}}_0$$

in the same way as for conventional redundant manipulators (and with the same general corresponding advantages and issues), as discussed in Chap. 10.

Recent efforts have extended the progress in continuum robot kinematics to develop dynamic models for continuum robots [20.84, 104]. New models based on the Lagrangian [20.108, 115] and Newton–Euler [20.116] formulations have been introduced. Progress is impeded by the complexity of the dynamics of continuous backbones with large deformations, and many models remain too complex for practical

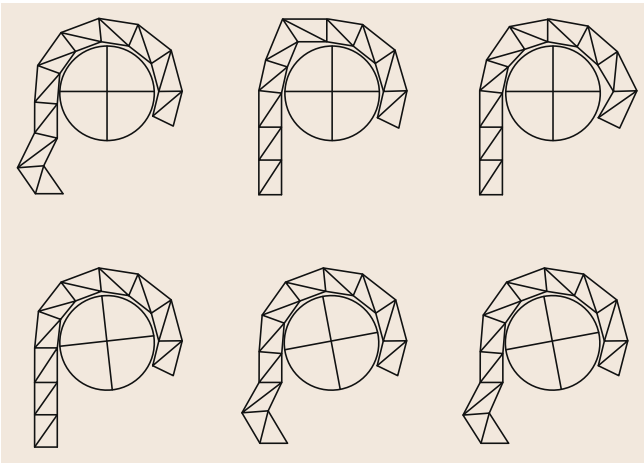


Fig. 20.18 Net rotation of an object by wave initiation at base and wave propagation (after [20.31]) ►

implementation. Hence, recent work has focused on implementable dynamic models based on simple, lumped-parameter models [20.117].

The new kinematic and dynamic models are in turn now enabling research in continuum robot motion planning [20.118, 119] and control [20.120, 121]. New models for force estimation [20.122], handling [20.76, 123], and sensing [20.124] using continuum robots have been introduced. Both model-based [20.54] and non-model based [20.125] controllers have been proposed. Controllers have been proposed in both configuration and task [20.54, 126] space. However, at this time, implementation of model-based controllers is rare.

20.4 Modeling of Locomotion for Snake-Like and Continuum Mechanisms

Snake and continuum robots have the ability to use their entire bodies to interact with the environment in order to produce purposeful motion [20.127–136]. Locomotion is a challenging motion planning and control problem because a planner/controller must contend with many degrees of freedom at once, even if they were somehow reduced with approaches like the backbone method, discussed above. Many researchers achieve this by modeling the snake robot's motion with a *gait*, a cyclic motion in the internal degrees of freedom of a mechanism that produces net motion. It makes sense to implement gaits because all locomoting animals exhibit gaits: horses gallop, trot, and walk; snakes sidewind, laterally undulate, and linearly progress.

The benefit of the gait is that one can reduce a complex system down to a few set of parameters that describe the gait and then by cycling through the phase of the gait, a planner can achieve locomotion. So, if the gait were just a sinusoid, the user specifies amplitude, frequency, and phase and the mechanism is then *fit* to the sinusoid, which requires very little computation as was seen with the backbone method. To achieve motion in three dimensions, we describe gaits for the snake robot with two sinusoidal waves lying in two mutually perpendicular planes, a horizontal plane parallel to the ground and a vertical plane perpendicular to the ground. Recall that some of the mechanisms, described above, consists of single degree of freedom joints, each placed orthogonally to its neighboring joints. For the sake of notation, we ascribe the even joints to modules that bend in the vertical plane and odd joints to modules that bend in the horizontal plane. So, if a vertical wave perpendicular to the ground is sent through the mechanism, only the even modules participate, while the odd modules remain stationary. Likewise, only the odd modules participate in sending lateral waves parallel to the ground through the robot. Should the snake robot fall onto its side, then even modules simply become the lateral modules and the odd become the vertical modules.

Given that the even modules are responsible for propagating vertical waves and the odd for propagating lateral waves, we use the terms $\text{offset}_{\text{vertical}}$, $\text{offset}_{\text{lateral}}$, $\text{amplitude}_{\text{vertical}}$, and $\text{amplitude}_{\text{ground}}$ to denote the parameters affecting the two orthogonal waves. Finally, we assume that both offset parameters are set to zero, unless otherwise noted. All of our differentiable gaits can be described by an angle of the n-th module at time t as

$$\begin{aligned}
&\text{angle}(n,t) \\
&\quad = \begin{cases} \text{offset}_{\text{even}} + \text{amplitude}_{\text{even}} * \sin(\theta)\,, \\ \quad n = \text{even}\,, \\ \text{offset}_{\text{odd}} + \text{amplitude}_{\text{odd}} * \sin(\theta + \delta)\,, \\ \quad n = \text{odd}\,, \end{cases}\,, \\
&\theta = \left(\frac{\mathrm{d}\theta}{\mathrm{d}n} * n + \frac{\mathrm{d}\theta}{\mathrm{d}t} * t \right)\,.
\end{aligned} \tag{20.1}$$

The offset is generally used to aim the robot when locomoting on the ground. For instance, when the snake is straight or climbing a pipe, the offset is zero. The am-

plitude terms describe the amplitudes of the mutually perpendicular waves. In most cases, the amplitude and speed of the robot are directly correlated and as both increase, the robot can better reach over large obstacles. However, larger amplitude comes at the cost of reduced stability. Intuitively, given a fixed number of modules, the fewer waves in the robot, the larger each wave can be. Thus, smaller values of $d\theta/dn$ allow more modules to participate in any one wave, which in turn allows for larger waves.

The frequency of the sine wave is determined by θ, which increases with time and along the length of the snake robot. The rate at which θ changes is controlled by the $d\theta/dn$ and $d\theta/dt$ terms. Note that the sine wave moving through the robot has a spatial and temporal component: 1) $d\theta/dn$ determines the spatial component, meaning if we freeze time (fix t), it tells us what the robot as a whole looks like, and 2) $d\theta/dt$ determines the frequency of the temporal component, meaning that if we examine the sinusoidal trajectory of one individual actuator (fix n), we determine how it moves. A zero value for $d\theta/dn$ causes identical angles on all of the modules on one axis, generating an arc along that axis, and a zero value for $d\theta/dt$ has the effect of freezing time. Finally, δ is simply an offset to control the timing between the motions of the two orthogonal waves. These parameters can be modified to change the type of and nature of the gait performed by the robot.

Current work includes combining the gait-equation-based approach with backbone curves. We are developing techniques that take an animated sequence of backbone curves and infer the gait parameters from that sequence. The backbone curve approach is easy to visualize and therefore geometrically manipulate. The benefit of the gait approach is that it contains an analytic expression which permits evaluation, and hence optimization. With this new current approach, we can obtain the best of both worlds.

20.5 Conclusion and Extensions to Related Areas

The field of snake-like and continuum robots has blossomed over the past 40 years. The work has taken many different directions including novel designs for locomotion, manipulation, and medical applications, as well as different avenues for applying the resulting modeling techniques including nonholonomic planning and the conformational flexibility and statistical mechanics of biological macromolecules.

It is worth noting that other related works in nonrobotic fields and application areas have developed either in parallel or as a consequence of the efforts mentioned before. For example, continuum devices were developed essentially outside the field of robotics in [20.137, 138] from the perspective of mechanics. Continuum curve approaches applied to variable-geometry trusses were developed in [20.139–141], essentially in parallel with [20.9]. In recent work, wormlike robots [20.142] have been designed that execute peristaltic waves, motivated by biology akin to those described in [20.113]. Finally, analysis for understanding curves and workspace formulations for snake and continuum robots have been used to study deoxyribonucleic acid (DNA) [20.110, 143–148].

Video-References

VIDEO 157 Field experiments with the OctArm continuum manipulator
available from http://handbookofrobotics.org/view-chapter/20/videodetails/157

VIDEO 158 OctArms I-V
available from http://handbookofrobotics.org/view-chapter/20/videodetails/158

VIDEO 159 One dimensional binary manipulator
available from http://handbookofrobotics.org/view-chapter/20/videodetails/159

VIDEO 160 Two dimensional binary manipulator
available from http://handbookofrobotics.org/view-chapter/20/videodetails/160

VIDEO 161 Three dimensional binary manipulator
available from http://handbookofrobotics.org/view-chapter/20/videodetails/161

VIDEO 162 Binary manipulator grasping
available from http://handbookofrobotics.org/view-chapter/20/videodetails/162

VIDEO 163 Binary manipulator obstacle navigation
available from http://handbookofrobotics.org/view-chapter/20/videodetails/163

VIDEO 164 Binary manipulator object recovery
available from http://handbookofrobotics.org/view-chapter/20/videodetails/164
VIDEO 165 Modsnake fence navigation
available from http://handbookofrobotics.org/view-chapter/20/videodetails/165
VIDEO 166 Modsnake autonomous pole-climbing
available from http://handbookofrobotics.org/view-chapter/20/videodetails/166
VIDEO 167 Modsnake pipe inspection
available from http://handbookofrobotics.org/view-chapter/20/videodetails/167
VIDEO 168 Modsnake climbing a tree
available from http://handbookofrobotics.org/view-chapter/20/videodetails/168
VIDEO 169 Modsnake swimming
available from http://handbookofrobotics.org/view-chapter/20/videodetails/169
VIDEO 171 Modsnake pole climb
available from http://handbookofrobotics.org/view-chapter/20/videodetails/171
VIDEO 174 Modsnake sidewinding
available from http://handbookofrobotics.org/view-chapter/20/videodetails/174
VIDEO 175 CMU medical snake robot
available from http://handbookofrobotics.org/view-chapter/20/videodetails/175
VIDEO 244 Active compliant insertion
available from http://handbookofrobotics.org/view-chapter/20/videodetails/244
VIDEO 245 Automatic insertion implant calibration
available from http://handbookofrobotics.org/view-chapter/20/videodetails/245
VIDEO 246 IREP tagging spikes
available from http://handbookofrobotics.org/view-chapter/20/videodetails/246
VIDEO 247 RDP experimental results
available from http://handbookofrobotics.org/view-chapter/20/videodetails/247
VIDEO 248 Stenting deployment system
available from http://handbookofrobotics.org/view-chapter/20/videodetails/248
VIDEO 249 Bimanual dissection
available from http://handbookofrobotics.org/view-chapter/20/videodetails/249
VIDEO 250 First concentric tube robot teleoperation
available from http://handbookofrobotics.org/view-chapter/20/videodetails/250
VIDEO 251 Shoe decoration using concentric tube robot
available from http://handbookofrobotics.org/view-chapter/20/videodetails/251
VIDEO 252 Concentric tube robot at TEDMED 2010
available from http://handbookofrobotics.org/view-chapter/20/videodetails/252
VIDEO 253 Aiko obstacle-aided locomotion
available from http://handbookofrobotics.org/view-chapter/20/videodetails/253
VIDEO 254 Aiko sidewinding
available from http://handbookofrobotics.org/view-chapter/20/videodetails/254
VIDEO 255 Anna Konda – Motion
available from http://handbookofrobotics.org/view-chapter/20/videodetails/255

References

20.1 B.Y.S. Hirose, H. Yamada: Snake-like robots, IEEE Robot. Autom. Mag. **16**(1), 88–98 (2009)
20.2 S. Hirose: *Biologically Inspired Robots: Snake-Like Locomotors and Manipulators* (Oxford Univ. Press, New York 1993)
20.3 Y. Tanaka, M. Arai, S. Hirose, T. Shingo: Development of *Souryu-V* with mono-tread-crawlers and elastic-rods joint, IEEE Int. Workshop Saf. Secur. Rescue Robot. (2006)
20.4 Hirose Fukushima Lab: *Souryu Robot* http://www-robot.mes.titech.ac.jp/robot/snake/soryu/soryu.html
20.5 H. Ohno, S. Hirose: Design of slim slime robot and its gait of locomotion, Proc. IEEE/RSJ Int. Conf. Intel Robot. Syst. (2001) pp. 707–715
20.6 H. Yamada, M. Mori, K. Takita, S. Ogami, S. Hirose: Development of amphibious snake-like robot ACM-R5, 36th Int. Symp. Robot. (2005)
20.7 G.S. Chirikjian, J.W. Burdick: Design and experiments with a 30 DOF robot, Proc. IEEE ICRA (1992) pp. 113–119
20.8 J.W. Burdick: Robots that crawl, walk, and slither, Eng. Sci. **55**(4), 2–13 (1992)
20.9 G.S. Chirikjian: Inverse kinematics of binary manipulators using a continuum model, J. Intel. Robot. Syst. **19**, 5–22 (1997)
20.10 J. Bluck: Squaring off with a robotic serpent. http://astrobiology.nasa.gov/articles/2001/2/21/squaring-off-with-a-robotic-serpent/

20.11 University of Pennsylvania: Modlab, http://modlabupenn.org/multimedia/
20.12 M. Yim: New locomotion gaits, Proc. IEEE ICRA (1994) pp. 2508–2514
20.13 M. Yim, D. Duff: Modular robots, IEEE Spectr. **39**(2), 30–34 (2002)
20.14 M. Park, M. Yim: Distributed control and communication fault tolerance for the CKBot, IEEE Int. Conf. Reconfig. Mech. Robot. (2009) pp. 682–688
20.15 M. Yim, B. Shirmohammadi, J. Sastra, M. Park, M. Dugan, C.J. Taylor: Towards robotic self-reassembly after explosion, Proc. IEEE/RSJ Int. Conf. Intel. Robot. Syst. (2007) pp. 2767–2772
20.16 S. Revzen, M. Bhoite, A. Macasieb, M. Yim: Structure synthesis on-the-fly in a modular robot, Int. Conf. Intel. Robot. Syst. (2011) pp. 4797–4802
20.17 C. Wright, A. Johnson, A. Peck, Z. McCord, A. Naaktgeboren, P. Gianfortoni, M. Gonzalez-Rivero, R.L. Hatton, H. Choset: Design of a modular snake robot, Int. Conf. Intel. Robot. Syst. (2007) pp. 2609–2614
20.18 A. Johnson, C. Wright, M. Tesch, K. Lipkin, H. Choset: *A Novel Architecture for Modular Snake Robots, Tech. Report CMU-RI-TR-11-29* (Carnegie Mellon Univ., Pittsburgh 2011)
20.19 K. Lipkin, I. Brown, A. Peck, H. Choset, J. Rembisz, P. Gianfortoni, A. Naaktgeboren: Differentiable and piecewise differentiable gaits for snake robots, Int. Conf. Intel. Robot. Syst. (2007) pp. 1864–1869
20.20 Biorobotics Laboratory, Carnegie Mellon University http://biorobotics.org (for videos of Choset's robots)
20.21 P. Liljeback, O. Stavdahl, A. Beitnes: SnakeFighter – Development of a water hydraulic fire fighting snake robot, 9th Int. Conf. Control Autom. Robot. Vision (2006) pp. 1–6
20.22 P. Liljeback, K.Y. Pettersen, O. Stavdahl, J.T. Gravdahl: Experimental investigation of obstacle-aided locomotion with a snake robot, IEEE Trans. Robot. **99**, 1–8 (2011)
20.23 P. Liljeback, S. Fjerdingen, K.Y. Pettersen, Ø. Stavdahl: A snake robot joint mechanism with a contact force measurement system, Proc. IEEE ICRA (2009) pp. 3815–3820
20.24 J. Borenstein, M. Hansen, A. Borrell: The OmniTread OT-4 serpentine robot–design and performance, J. Field Robot. **24**(7), 601–621 (2007)
20.25 J.C. McKenna, D.J. Anhalt, F.M. Bronson, H.B. Brown, M. Schwerin, E. Shammas, H. Choset: Toroidal skin drive for snake robot locomotion, Proc. IEEE ICRA (2008) pp. 1150–1155
20.26 H. Date, Y. Takita: An electricity-free snake-like propulsion mechanism driven and controlled by fluids, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2009) pp. 3637–3642
20.27 T. Sato, T. Kano, A. Ishiguro: A decentralized control scheme for an effective coordination of phasic and tonic control in a snake-like robot, Bioinspir. Biomim. **7**(1), 016005 (2012)
20.28 T. Kamegawa, T. Yarnasaki, H. Igarashi, F. Matsuno: Development of the snake-like rescue robot *kohga*, Proc. IEEE Int. Conf. Robot. Autom. (2004) pp. 5081–5086
20.29 M. Hara, S. Satomura, H. Fukushima, T. Kamegawa, H. Igarashi, F. Matsuno: Control of a snake-like robot using the screw drive mechanism, Proc. IEEE Int. Conf. Robot. Autom. (2007) pp. 3883–3888
20.30 G. Robinson, J.B.C. Davies: Continuum robots – A state of the art, Proc. IEEE Int. Conf. Robot. Autom., Detroit (1999) pp. 2849–2854
20.31 G.S. Chirikjian: Theory and applications of hyper-redundant robotic mechanisms (Department of Applied Mechanics, California Institute of Technology, Pasadena 1992)
20.32 V.C. Anderson, R.C. Horn: Tensor arm manipulator design, Mech. Eng. **89**(8), 54–65 (1967)
20.33 B. Roth, J. Rastegar, V. Scheinman: On the design of computer controlled manipulators, 1st CISM-IFTMM Symp. Theory Pract. Robot. Manip. (1973) pp. 93–113
20.34 W.M. Kier, K.K. Smith: Tongues, tentacles and trunks: The biomechanics of movement in muscular-hydrostats, Zool. J. Linneaan Soc. **83**, 307–324 (1985)
20.35 F. Martin, C. Niemitz: How do African elephants (Loxodonta Africana) optimize goal-directed trunk movements?, Jahresvers. Dt. Zool. Ges. Dt. Ges. Parasitol. **96**, 159 (2003)
20.36 Y. Yekutieli, R. Sagiv-Zohar, B. Hochner, T. Flash: Dynamics model of the octopus arm. II. Control of reaching movements, J. Neurophysiol. **94**, 1459–1468 (2005)
20.37 R. Cieslak, A. Morecki: Elephant trunk type elastic manipulator – A tool for bulk and liquid type materials transportation, Robotica **17**, 11–16 (1999)
20.38 M.W. Hannan, I.D. Walker: Analysis and experiments with an elephant's trunk robot, Adv. Robot. **15**(8), 847–858 (2001)
20.39 H. Tsukagoshi, A. Kitagawa, M. Segawa: Active hose: An artificial elephant's nose with maneuverability for rescue operation, Proc. IEEE Int. Conf. Robot. Autom., Seoul (2001) pp. 2454–2459
20.40 E. Guglielmino, N. Tsagarakis, D.G. Caldwell: An octopus-anatomy inspired robotics arm, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., Taipei (2010) pp. 3091–3096
20.41 W. McMahan, B.A. Jones, I.D. Walker, V. Chitrakaran, A. Seshadri, D. Dawson: Robotic manipulators inspired by cephalopod limbs, Proc. CDEN Des. Conf., Montreal (2004) pp. 1–10
20.42 I.D. Walker, D. Dawson, T. Flash, F. Grasso, R. Hanlon, B. Hochner, W.M. Kier, C. Pagano, C.D. Rahn, Q. Zhang: Continuum robot arms inspired by cephalopods, Proc. 7th SPIE Conf. Unmanned Ground Veh. Technol., Orlando (2005) pp. 303–314
20.43 D. Trivedi, C.D. Rahn, W.M. Kier, I.D. Walker: Soft robotics: Biological inspiration, state of the art, and future research, Appl. Bionics Biomech. **5**(2), 99–117 (2008)
20.44 R.J. Webster III, B.A. Jones: Design and kinematic modeling of constant curvature continuum

robots: A review, Int. J. Robot. Res. **29**(13), 1661–1683 (2010)
20.45 D.B. Camarillo, C.F. Milne, C.R. Carlson, M.R. Zinn, J.K. Salisbury: Mechanics modeling of tendon-driven continuum manipulators, IEEE Trans. Robot. **24**(6), 1262–1273 (2008)
20.46 L. Cowan: *Analysis and experiments for tendril-type robots*, M.S. Thesis (Clemson University, Clemson 2008)
20.47 M.W. Hannan, I.D. Walker: Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots, J. Robot. Syst. **20**(2), 45–63 (2003)
20.48 G. Immega: Tentacle-like manipulators with adjustable tension lines, U.S. Patent 5317952A (1992)
20.49 J.S. Mehling, M.A. Diftler, M. Chu, M. Valvo: A minimally invasive tendril robot for in-space inspection, Proc. Conf. BioRobotics (2006) pp. 690–695
20.50 A. Grzesiak, R. Becker, A. Verl: The bionic handling assistant – A success story of additive manufacturing, Assem. Autom. **31**(4), 329–333 (2011)
20.51 D.M. Lane, J.B.C. Davies, G. Robinson, D.J. O'Brien, J. Sneddon, E. Seaton, A. Elfstrom: The AMADEUS dextrous subsea hand: Design, modeling, and sensor processing, IEEE J. Ocean. Eng. **24**(1), 96–111 (1999)
20.52 M.B. Pritts, C.D. Rahn: Design of an artificial muscle continuum robot, Proc. IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp. 4742–4746
20.53 K. Suzumori, S. Iikura, H. Tanaka: Development of flexible microactuator and its applications to robotic mechanisms, Proc. IEEE Int. Conf. Robot. Autom., Sacramento (1991) pp. 1622–1627
20.54 D.B. Camarillo, C.R. Carlson, J.K. Salisbury: Task-space control of continuum manipulators with coupled tendon drive, 11th Int. Symp. Exp. Robot. (2009) pp. 271–280
20.55 G. Immega, K. Antonelli: The KSI tentacle manipulator, Proc. IEEE Int. Conf. Robot. Autom., Nagoya (1995) pp. 3149–3154
20.56 I. Gravagne, C. Rahn, I.D. Walker: Large deflection dynamics and control for planar continuum robots, IEEE/ASME Trans. Mechatron. **8**(2), 299–307 (2003)
20.57 B.A. Jones, W. McMahan, I.D. Walker: Design and analysis of a novel pneumatic manipulator, Proc. 3rd IFAC Symp. Mechatron. Syst., Sydney (2004) pp. 745–750
20.58 W. McMahan, B.A. Jones, I.D. Walker: Design and implementation of a multi-section continuum robot: Air-Octor, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., Edmonton (2005) pp. 3345–3352
20.59 J. Lock, G. Laing, M. Mahvash, P.E. Dupont: Quasistatic modeling of concentric tube robots with external loads, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., Taipei (2010) pp. 2325–2332
20.60 L.G. Torres, R. Alterovitz: Motion planning for concentric tube robots using mechanics-based models, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 5153–5159
20.61 R.J. Webster III, J.M. Romano, N.J. Cowan: Kinematics and calibration of active cannulas, Proc. IEEE Int. Conf. Robot. Autom., Pasadena (2008) pp. 3888–3895
20.62 R.S. Penning, J. Jung, J.A. Borgstadt, N.J. Ferrier, M.R. Zinn: Towards closed loop control of a continuum robotic manipulator for medical applications, Proc. IEEE Int. Conf. Robot. Autom., Shanghai (2011) pp. 4822–4827
20.63 B. Bardou, P. Zanne, F. Nageotte, M. de Mathelin: Control of multiple sections flexible endoscopic system, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., Taipei (2010) pp. 2345–2350
20.64 G. Chen, P.M. Tu, T.R. Herve, C. Prelle: Design and modeling of a micro-robotic manipulator for colonoscopy, 5th Int. Workshop Res. Educ. Mechatron., Annecy (2005) pp. 109–114
20.65 K. Xu, J. Zhao, J. Geiger, A.J. Shih, M. Zheng: Design of an endoscopic stitching device for surgical obesity treatment using a N.O.T.E.S. approach, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 961–966
20.66 H.-S. Yoon, S.M. Oh, J.H. Jeong, S.H. Lee, K. Tae, K.-C. Koh, B.J. Yi: Active bending robot endoscope system for navigation through sinus area, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 967–972
20.67 L.A. Lyons, R.J. Webster III, R. Alterovitz: Planning active cannula configurations through tubular anatomy, Proc. IEEE Int. Conf. Robot. Autom., Anchorage (2010) pp. 2082–2087
20.68 S. Wakimoto, K. Suzumori: Fabrication and basic experiments of pneumatic multi-chamber rubber tube actuator for assisting colonoscope insertion, Proc. IEEE Int. Conf. Robot. Autom., Anchorage (2010) pp. 3260–3265
20.69 N. Simaan, R. Taylor, P. Flint: A dexterous system for laryngeal surgery, Proc. IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp. 351–357
20.70 H. Watanabe, K. Kanou, Y. Kobayashi, M.G. Fujie: Development of a 'Steerable Drill' for ACL reconstruction to create the arbitrary trajectory of a bone tunnel, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 955–960
20.71 K. Xu, R.E. Goldman, J. Ding, P.K. Allen, D.L. Fowler, N. Simaan: System design of an insertable robotic effector platform for single port access (SPA) surgery, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., St. Louis (2009) pp. 5546–5552
20.72 J. Ding, K. Xu, R. Goldman, P. Allen, D. Fowler, N. Simaan: Design, simulation and evaluation of kinematic alternatives for insertable robotic effectors platforms in single port access surgery, Proc. IEEE Int. Conf. Robot. Autom., Anchorage (2010) pp. 1053–1058
20.73 P. Sears, P.E. Dupont: Inverse kinematics of concentric tube steerable needles, Proc. IEEE Int. Conf. Robot. Autom. (2007) pp. 1887–1892
20.74 P. Sears, P.E. Dupont: A steerable needle technology using curved concentric tubes, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2006) pp. 2850–2856
20.75 R. Buckingham: Snake arm robots, Ind. Robot An Int. J. **29**(3), 242–245 (2002)

20.76 M. Mahvash, P.E. Dupont: Stiffness control of a continuum manipulator in contact with a soft environment, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., Taipei (2010) pp. 863–870

20.77 J. Jung, R.S. Penning, N.J. Ferrier, M.R. Zinn: A modeling approach for continuum robotic manipulators: Effects of nonlinear internal device friction, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 5139–5146

20.78 D.C. Rucker, B.A. Jones, R.J. Webster III: A model for concentric tube continuum robots under applied wrenches, Proc. IEEE Int. Conf. Robot. Autom., Anchorage (2010) pp. 1047–1052

20.79 T. Aoki, A. Ochiai, S. Hirose: Study on slime robot: Development of the mobile robot prototype model using bridle bellows, Proc. IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp. 2808–2813

20.80 H. Ohno, S. Hirose: Design of slim slime robot and its gait of locomotion, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., Maui (2001) pp. 707–715

20.81 B.A. Jones, M. Csencsits, W. McMahan, V. Chitrakaran, M. Grissom, M. Pritts, C.D. Rahn, I.D. Walker: Grasping, manipulation, and exploration tasks with the OctArm continuum manipulator, Proc. Int. Conf. Robot. Autom., Orlando (2006)

20.82 W. McMahan, M. Pritts, V. Chitrakaran, D. Dienno, M. Grissom, B. Jones, M. Csencsits, C.D. Rahn, D. Dawson, I.D. Walker: Field trials and testing of *OCTARM* continuum robots, Proc. IEEE Int. Conf. Robot. Autom. (2006) pp. 2336–2341

20.83 J.K. Salisbury: Whole arm manipulation, 4th Symp. Robot. Res. (1987)

20.84 I.D. Walker: Continuum robot appendages for traversal of uneven terrain in in-situ exploration, IEEE Aerosp. Conf. (2011) pp. 1–8

20.85 R. Kang, A. Kazakidi, E. Guglielmino, D.T. Branson, D.P. Tsakiris, J.A. Ekaterinaris, D.G. Caldwell: Dynamic model of a hyper-redundant, octopus-like manipulator for underwater applications, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 4054–4059

20.86 I.S. Godage, D.T. Branson, E. Guglielmino, G.A. Medrano-Cerda, D.G. Caldwell: Shape function-based kinematics and dynamics for variable-length continuum robotic arms, Proc. IEEE Int. Conf. Robot. Autom., Shanghai (2011) pp. 452–457

20.87 R. Alterovitz, A. Lim, K. Goldberg, G.S. Chirikjian, A.M. Okamura: Steering flexible needles under Markov motion uncertainty, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2005) pp. 120–125

20.88 N.J. Cowan, K. Goldberg, G.S. Chirikjian, G. Fichtinger, R. Alterovitz, K.B. Reed, V. Kallem, W. Park, S. Misra, A.M. Okamura: Robotic needle steering: Design, modeling, planning, and image guidance. In: *Surgical Robotics – Systems, Applications, and Visions*, ed. by J, Rosen, B. Hannaford, R. Satava (Springer New York **pp**, 557–582 (2011)

20.89 W. Park, Y. Wang, G.S. Chirikjian: The path-of-probability algorithm for steering and feedback control of flexible needles, Int. J. Robot. Res. **29**(7), 813–830 (2010)

20.90 W. Park, J.S. Kim, Y. Zhou, N.J. Cowan, A.M. Okamura, G.S. Chirikjian: Diffusion-based motion planning for a nonholonomic flexible needle model, Proc. IEEE Int. Conf. Robot. Autom., Barcelona (2005)

20.91 R.J. Webster III, J.-S. Kim, N.J. Cowan, G.S. Chirikjian, A.M. Okamura: Nonholonomic modeling of needle steering, Int. J. Robot. Res. **25**(5–6), 509–525 (2006)

20.92 C. Rucker, R.J. Webster III, G.S. Chirikjian, N.J. Cowan: Equilibrium conformations of concentric-tube continuum robots, Int. J. Robot. Res. **29**(10), 1263–1280 (2010)

20.93 P.E. Dupont, J. Lock, B. Itkowitz, E. Butler: Design and control of concentric-tube robots, IEEE Trans. Robot. **26**(2), 209–225 (2010)

20.94 J. Lock, G. Laing, M. Mahvash, P.E. Dupont: Quasistatic modeling of concentric tube robots with external loads, 2010 IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2010) pp. 2325–2332

20.95 P.E. Dupont, J. Lock, E. Butler: Torsional kinematic model for concentric tube robots, Proc. IEEE Int. Conf. Robot. Autom. (2009) pp. 3851–3858

20.96 C. Bedell, J. Lock, A. Gosline, P.E. Dupont: Design optimization of concentric tube robots based on task and anatomical constraints, Proc. IEEE Int. Conf. Robot. Autom. (2011) pp. 398–403

20.97 M. Mahvash, P.E. Dupont: Stiffness control of surgical continuum manipulators, IEEE Trans. Robot. **27**(2), 334–345 (2011)

20.98 A. Degani, H. Choset, A. Wolf, M.A. Zenati: Highly articulated robotic probe for minimally invasive surgery, Proc. IEEE Int. Conf. Robot. Autom. (2006) pp. 4167–4172

20.99 A. Degani, H. Choset, A. Wolf, T. Ota, M.A. Zenati: Percutaneous intrapericardial interventions using a highly articulated robotic probe, In The First IEEE/RAS-EMBS Int. Conf. Biomed. Robot. Biomech. (2006) pp. 7–12

20.100 R.J. Murphy, M.S. Moses, M.D. Kutzer, G.S. Chirikjian, M. Armand: Constrained workspace generation for snake-like manipulators with applications to minimally invasive surgery, Proc. IEEE Int. Conf. Robot. Autom. (2013) pp. 5341–5347

20.101 M.S. Moses, M.D. Kutzer, H. Ma, M. Armand: A continuum manipulator made of interlocking fibers, Proc. IEEE Int. Conf. Robot. Autom. (2013) pp. 4008–4015

20.102 S.M. Segreti, M.D.M. Kutzer, R.J. Murphy, M. Armand: Cable length estimation for a compliant surgical manipulator, Proc. IEEE Int. Conf. Robot. Autom. (2012) pp. 701–708

20.103 R.J. Murphy, M.D. Kutzer, S.M. Segreti, B.C. Lucas, M. Armand: Design and kinematic characterization of a surgical manipulator with a focus on treating osteolysis, Robotica **32**(6), 835–850 (2014)

20.104 D. Trivedi, A. Lotfi, C.D. Rahn: Geometrically exact dynamics for soft robotics manipulators, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Diego (2007) pp. 1497–1502

20.105 G.S. Chirikjian, J.W. Burdick: A modal approach to hyper-redundant manipulator kinematics, IEEE Trans. Robot. Autom. **10**(3), 343–354 (1994)

20.106 I.A. Gravagne, I.D. Walker: Manipulability, force, and compliance analysis for planar continuum manipulators, IEEE Trans. Robot. Autom. **18**(3), 263–273 (2002)

20.107 B.A. Jones, I.D. Walker: Kinematics for multisection continuum robots, IEEE Trans. Robot. **22**(1), 43–55 (2006)

20.108 H. Mochiyama, T. Suzuki: Dynamic modeling of a hyper-flexible manipulator, Proc. 41st SICE Annu. Conf., Osaka (2002) pp. 1505–1510

20.109 H. Mochiyama, T. Suzuki: Kinematics and dynamics of a cable-like hyper-flexible manipulator, Proc. IEEE Intl. Conf. Robot. Autom., Taipei (2003) pp. 3672–3677

20.110 G.S. Chirikjian: Variational analysis of snakelike robots. In: *Redundancy in Robot Manipulators and Multi-Robot Systems*, Lecture Notes in Electrical Engineering, Vol. 57, ed. by D. Milutinovic, J. Rosen (Springer, New York 2013) pp. 77–91

20.111 G.S. Chirikjian: Hyper-redundant manipulator dynamics: A continuum approximation, Adv. Robot. **9**(3), 217–243 (1995)

20.112 I.S. Godage, E. Guglielmino, D.T. Branson, G.A. Medrano-Cerda, D.G. Caldwell: Novel modal approach for kinematics of multisection continuum arms, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 1093–1098

20.113 G.S. Chirikjian, J.W. Burdick: Kinematics of hyper-redundant locomotion with applications to grasping, Proc. IEEE Int. Conf. Robot. Autom. (1991) pp. 720–725

20.114 M. Csencsits, B.A. Jones, W. McMahan: User interfaces for continuum robot arms, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., Edmonton (2005) pp. 3011–3018

20.115 E. Tatlicioglu, I.D. Walker, D.M. Dawson: Dynamic modeling for planar extensible continuum robot manipulators, Proc. IEEE Int. Conf. Robot. Autom., Rome (2007) pp. 1357–1362

20.116 W. Khalil, G. Gallot, O. Ibrahim, F. Boyer: Dynamic modeling of a 3-D serial eel-like robot, Proc. IEEE Int. Conf. Robot. Autom., Barcelona (2005) pp. 1282–1287

20.117 N. Giri, I.D. Walker: Three module lumped element model of a continuum arm section, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 4060–4065

20.118 J. Li, J. Xiao: Determining 'grasping' configurations for a spatial continuum manipulator, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 4207–4214

20.119 J. Xiao, R. Vatcha: Real-time adaptive motion planning for a continuum manipulator, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., Taipei (2010) pp. 5919–5926

20.120 M. Ivanescu, N. Bizdoaca, D. Pana: Dynamic control for a tentacle manipulator with SMA actuators, Proc. IEEE Int. Conf. Robot. Autom., Taipei (2003) pp. 2079–2084

20.121 M. Ivanescu, V. Stoian: A variable structure controller for a tentacle manipulator, Proc. IEEE Int. Conf. Robot. Autom., Nagoya (1995) pp. 3155–3160

20.122 A. Bajo, N. Simaan: Finding lost wrenches: Using continuum robots for contact detection and estimation of contact location, Proc. IEEE Int. Conf. Robot. Autom., Anchorage (2010) pp. 3666–3672

20.123 H. Mochiyama: Whole-arm impedance of a serial-chain manipulator, Proc. IEEE Int. Conf. Robot. Autom., Seoul (2001) pp. 2223–2228

20.124 D.C. Rucker, R.J. Webster III: Deflection-based force sensing for continuum robots: A probabilistic approach, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 3764–3769

20.125 D. Braganza, D.M. Dawson, I.D. Walker, N. Nath: Neural network grasping controller for continuum robots, Proc. 45th IEEE Conf. Decis. Control, San Diego (2006)

20.126 A. Kapadia, I.D. Walker: Task space control of extensible continuum manipulators, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst., San Francisco (2011) pp. 1087–1092

20.127 S. Ma: Analysis of creeping locomotion of a snakelike robot, Adv. Robot. **15**(2), 205–224 (2001)

20.128 K.Y. Pettersen, O. Stavdahl, J.T. Gravdahl: *Snake Robots: Modelling, Mechatronics, and Control* (Springer, London 2012)

20.129 A.A. Transeth, K.Y. Pettersen, P. Liljeback: A survey on snake robot modeling and locomotion, Robot. **27**(7), 999–1015 (2009)

20.130 R. Vaidyanathan, H.J. Chiel, R.D. Quinn: A hydrostatic robot for marine applications, Robot. Auton. Syst. **30**(1), 103–113 (2000)

20.131 C. Wright, A. Johnson, A. Peck, Z. McCord, A. Naaktgeboren, P. Gianfortoni, M. Gonzalez-Rivero, R. Hatton, H. Choset: Design of a modular snake robot, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2007) pp. 2609–2614

20.132 M. Tesch, K. Lipkin, I. Brown, R. Hatton, A. Peck, J. Rembisz, H. Choset: Parameterized and scripted gaits for modular snake robots, Adv. Robot. **23**(9), 1131–1158 (2009)

20.133 R.L. Hatton, H. Choset: Generating gaits for snake robots: Annealed chain fitting and keyframe wave extraction, Auton. Robot. **28**(3), 271–281 (2010)

20.134 J.C. McKenna, D.J. Anhalt, F.M. Bronson, H.B. Brown, M. Schwerin, E. Shammas, H. Choset: Toroidal skin drive for snake robot locomotion, Proc. IEEE Int. Conf. Robot. Autom. (2008) pp. 1150–1155

20.135 A. Wolf, H.B. Brown, R. Casciola, A. Costa, M. Schwerin, E. Shamas, H. Choset: A mobile hyper redundant mechanism for search and rescue tasks, Proc. 3rd IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2003) pp. 2889–2895

20.136 C. Wright, A. Buchan, B. Brown, J. Geist, M. Schwerin, D. Rollinson, H. Choset: Design and archi-

tecture of the unified modular snake robot, Proc. IEEE Int. Conf. Robot. Autom. (2012) pp. 4347–4354
20.137 J.M. Snyder, J.F. Wilson: Dynamics of the elastica with end mass and follower loading, J. Appl. Mech. **57**, 203 (1990)
20.138 J.F. Wilson, D. Li, Z. Chen, R.T. George Jr.: Flexible robot manipulators and grippers: Relatives of elephant trunks and squid tentacles. In: *Robots and Biological Systems: Towards a New Bionics?*, (Springer, Berlin, Heidelberg 1993) pp. 475–494
20.139 F. Naccarato, P.C. Hughes: Inverse kinematics of variable-geometry truss manipulators, J. Robot. Syst. **8**(2), 249–266 (1991)
20.140 P.C. Hughes, W.G. Sincarsin, K.A. Carroll: Trussarm – A variable-geometry-truss manipulator, J. Intell. Mater. Syst. Struct. **2**(2), 148–160 (1991)
20.141 R.J. Salerno, C.F. Reinholtz, S.G. Dhande, R. Hall: Kinematics of long-chain variable geometry truss manipulators: An overview of solution techniques, Proc. 2nd Int. Workshop Adv. Robot Kinemat. (1990)
20.142 A.S. Boxerbaum, K.M. Shaw, H.J. Chiel, R.D. Quinn: Continuous wave peristaltic motion in a robot, Int. J. Robot. Res. **31**(3), 302–318 (2012)
20.143 G.S. Chirikjian: Framed curves and knotted DNA, Biochem. Soc. Trans. **41**, 635–638 (2013)
20.144 I. Ebert-Uphoff, G.S. Chirikjian: Discretely actuated manipulator workspace generation by closed-form convolution, ASME J. Mech. Des. **120**(2), 245–251 (1998)
20.145 I. Ebert-Uphoff, G.S. Chirikjian: Inverse kinematics of discretely actuated hyper-redundant manipulators using workspace densities, Proc. IEEE Int. Conf. Robot. Autom. (1996) pp. 139–145
20.146 I. Ebert-Uphoff: On the development of discretely-actuated hybrid-serial-parallel manipulators (Department of Mechanical Engineering, Johns Hopkins University, Baltimore 1997)
20.147 Y. Wang, G.S. Chirikjian: Workspace generation of hyper-redundant manipulators as a diffusion process on SE(N), IEEE Trans. Robot. Autom. **20**(3), 399–408 (2004)
20.148 Y. Zhou, G.S. Chirikjian: Conformational statistics of semi-flexible macromolecular chains with internal joints, Macromolecules **39**(5), 1950–1960 (2006)

Multimedia Contents

21. Actuators for Soft Robotics

Alin Albu-Schäffer, Antonio Bicchi

Although we do not know as yet how robots of the future will look like exactly, most of us are sure that they will not resemble the heavy, bulky, rigid machines dangerously moving around in old-fashioned industrial automation. There is a growing consensus, in the research community as well as in expectations from the public, that robots of the next generation will be physically compliant and adaptable machines, closely interacting with humans and moving safely, smoothly and efficiently – in other terms, robots will be *soft*.

This chapter discusses the design, modeling and control of actuators for the new generation of soft robots, which can replace conventional actuators in applications where rigidity is not the first and foremost concern in performance. The chapter focuses on the technology, modeling, and control of *lumped parameters* of soft robotics, that is, systems of discrete, interconnected, and compliant elements. *Distributed parameters*, snake-like and continuum soft robotics, are presented in Chap. 20, while Chap. 23 discusses in detail the biomimetic motivations that are often behind soft robotics.

21.1 **Background** 500
21.1.1 Motivations 500
21.1.2 Safety 500
21.1.3 Resilience 501
21.1.4 Performance and Energy Efficiency 501
21.1.5 The *Third Way* of Robotic Control and λ 501

21.2 **Soft Robot Design** 502
21.2.1 Natural Muscles 502
21.2.2 Breaking Down Artificial Muscle Systems 502
21.2.3 Sources for Soft Actuators 503
21.2.4 Pneumatic and Hydraulic 503
21.2.5 Electromagnetic and Electrostatic 503
21.2.6 Piezoelectric 504
21.2.7 Electro- and Magneto-Strictive 504
21.2.8 Shape Memory Alloys and Polymers 505
21.2.9 Polymeric Actuators 505
21.2.10 Carbon Nanotubes 505
21.2.11 Energy Storage 505
21.2.12 Energy Dissipation and Dampers 505
21.2.13 Artificial Muscles Arrangements 506
21.2.14 Specifications and Performance 507

21.3 **Modeling Actuators for Soft Robotics** 508
21.3.1 The Mass–Spring–Damper Paradigm 508
21.3.2 Nonlinear Mechanical Systems 508

21.4 **Modeling Soft Robots** 511

21.5 **Stiffness Estimation** 513
21.5.1 Observers for Linear Impedance 514
21.5.2 Variable Stiffness Observers 514
21.5.3 Parametric Estimation 514
21.5.4 Nonparametric Estimation 515

21.6 **Cartesian Stiffness Control** 515
21.6.1 Cartesian Impedance Control 515
21.6.2 Independent Position and Stiffness Control 516

21.7 **Periodic Motion Control** 518
21.7.1 Periodic Motion Tracking and Stiffness Adaptation 518
21.7.2 Controlled Oscillatory Dynamics 519
21.7.3 Exciting Intrinsic Oscillatory Dynamics 520
21.7.4 Model-Based Approach 520
21.7.5 Model-Free Approach 520

21.8 **Optimal Control of Soft Robots** 521
21.8.1 OC Theory 521
21.8.2 SEA Optimization 521
21.8.3 Safe Brachistochrone OC Problem 522
21.8.4 Max Speed OC Problem 522
21.8.5 VSA Optimization 523
21.9 **Conclusions and Open Problems** 524
Video-References 525
References 526

The chapter starts by analyzing the main motivations that are behind the development of actuators for soft robotics: these are mainly safety of operation, especially in interaction with humans; resilience of the robot itself in interaction with harsh, unpredictable environments, and performance enhancement in dynamic tasks.

We then briefly survey the different principles and technologies that can be used to implement actuators for soft robotics. Using natural muscles as a term of comparison, we consider different elements constituting an actuator, such as the prime movers, energy storage and dissipation elements, and their different possible arrangements.

We then move to consider a class of actuators for soft robotics that includes many of the recent developments in this field, that is, the class of variable impedance actuators (VIA) and the subclass of variable stiffness actuators (VSAs), and consider how a mathematical model of these systems can be provided in general but precise terms. With the objective of helping the reader navigate the space of the many existing, and many more conceivable, designs we consider these actuators from a general point of view, discussing how their performance envelope and specifications could be given in a unified, comprehensive way.

In the final part of the chapter, we consider the new problems and opportunities arising from the control of soft robots, including the need to estimate the variable physical compliance of the actuators, to plan its variations to adapt to tasks and environments, and to do real-time control of soft robots exploiting their physical properties.

21.1 Background

21.1.1 Motivations

The intrinsic properties of the biological muscles and their advantageous arrangement within the musculoskeletal system are a key aspect to the motion performance of many animal species. The flexibility, force-to-weight ratio, viscous damping, and fast reacting properties provide animals with an actuator having properties well suited for the tasks they are faced with day to day. Taking the human as an example, our muscles allow us to do very fine manipulation, precisely exerting forces and torques at the millinewton level, but also lift objects at a 10 : 1 force-to-weight ratio. Through the interplay of these physical properties with the neural sensory-motor control, motion is accomplished in a very energy-efficient, safe, and effective way.

Despite the diverse range of actuation techniques developed over the past years, the lack of an actuator which can reach the functional performance of the biological muscle and its neuro-mechanical control system still remains one of the most significant barriers for the development of machines, which can match the motion capabilities of natural systems. This observation has motivated many researchers in recent years to work toward developing and exploiting actuation technologies for a new generation of soft robots, which can co-exist and co-operate with people and reach or even surpass their performance.

21.1.2 Safety

One of the main motivations for revisiting traditional robotic design is the concern about their safety: it is to be expected that interacting with robot assistants or collaborators should not constitute (or even just be perceived as) a higher injury risk to humans than the interaction with another cautious human. Attention toward safe operation of robots near to, and even in contact with, humans is rather recent, although some of the pioneering contributions date back to the 1990s [21.1]. Robot-specific risk assessment methods have been studied in recent years [21.2, 3], and are discussed in Chap. 69.

The inherent danger to humans of conventional arms can be mitigated by retro-fitting their design by increasing their sensorization (using, e.g., proximity-sensitive skins), or by increasing the energy-absorbing properties of protective layers (adding enough soft and compliant coverings or placing airbags around the arm).

Starting in the early 1990s, researchers have gradually abandoned the retro-fitting approach and started designing robots with interaction safety in mind from the beginning: notable examples are MIT's WAM (whole arm manipulator) [21.4] and DLR's LWR (light-weight robot) [21.5]. Arms in this class are primarily characterized by low inertia of their moving parts (links and motors) and backdrivability. Advanced sensing and control can realize a *soft* behavior via software.

Another approach to increase the safety level of robot arms interacting with humans is *soft robotics*, that is, to intentionally introduce mechanical compliance and damping in design. By this researchers tend to replace the sensor-based computation of a behavior and its error-prone realization using active actuator control, by its direct physical embodiment, as in the natural example. Having compliance and damping in the robot structure is by no means sufficient to ensure its safety, as it might indeed be even counterproductive for the elastic energy potentially stored. Just like a human arm, a soft robot arm will need intelligent control to make it behave softly as when caressing a baby, or strongly as when punching.

21.1.3 Resilience

Physical interaction of a robot with its environment can also be dangerous to the robot itself. Indeed, while fortunately the number of accidents involving robots and humans is very limited, the number of times a robot is damaged because of impacts or force overexertion is rather large, and clearly the more so the less structured is the task and/or the environment. A typical cause of breakdown for traditional robots is the failure of reduction gears (especially harmonic drives are quite sensitive to shock loads) and motor bearings, but large accelerations can easily damage sensors and electronics. Although seldom considered in the scientific literature, resilience to shocks is not only instrumental to achieve viable applications of robots in everyday life, but would also be very useful in industrial environments, substantially enlarging the scope of applicability of robot technology. Soft robotics technologies can provide solutions that are extremely effective in absorbing shocks and reducing accelerations: soft materials can be used as coverings or even as structural elements in robot limbs, but the main technological challenge remains with soft actuators and transmissions.

21.1.4 Performance and Energy Efficiency

A further reason to search for a new generation of actuators with controllable compliance is for their dynamic behavior, which can afford high-performance, natural-looking motion while being more energy-efficient than rigid robots.

Indeed, adaptable compliance can be used to adjust the natural dynamics of the robot in tune with those of its environment, so that the intrinsic physical behavior of the resulting system is close to the desired motion. In these circumstances, actuators would only have to inject and extract energy into and out of the system for small corrective actions, thus reducing energy consumption.

The idea of embodying desirable dynamics in the physical properties of soft robots is of particular importance, for example, in walking/running robots and prostheses: indeed, biomechanical studies reveal that compliance adaptation is important for human walking and running. Although most humanoid robots (such as HRP-4 [21.6, 7] and ASIMO [21.8, 9]) use rigid robotics technologies, soft actuators are more and more often adopted in humanoid robots: pneumatic muscles are used, for example, in [21.10] and [21.11], while several legged systems have appeared that use series-elastic actuators [21.12]. What is probably the first prototype of fully physically compliant humanoid has been recently presented in [21.13].

The fact that natural systems change the compliance of their muscular system depending on the gait and environmental conditions, and even during the different phases of the gait, seems to indicate the potential usefulness of variable impedance actuators (VIA) for locomotion – yet, because of the technical and theoretical challenges posed, only early prototypes of bipedal walking systems with variable stiffness actuation have been reported so far [21.14, 15], and no humanoid robot has been reported (to the best of our knowledge) making full use of VIAs.

Besides their role in reducing energy consumption in cyclic motions, soft robotics can provide definite performance advantages in many other tasks. One notable class are those tasks where performance is measured in terms of peak velocity reached at the end of an explosive-type motion, such as for example, in throwing, hitting, jumping, kicking, etc. Compliance in the robot structure indeed allows to store elastic energy in a build-up phase, and restitute it in the subsequent explosive phase. Furthermore, it permits us to have joints that, at least for a short time duration, move at a substantially higher velocity than the motor shaft [21.16, 17].

21.1.5 The *Third Way* of Robotic Control and λ

Novel actuators for soft robotics introduce one further interesting innovation for what concerns their control

philosophy and architecture. Indeed, it can be roughly but fairly said that most existing robotic systems are controlled according to either one of two dominant architectures: position-based controller and torque-based controller. Position-based controllers, which prevail in traditional industrial robotics, are rather closed controller architectures built around an inner servomechanism loop, which basically provide the user with the ability to set joint position references, with very limited access to inner variables such as velocities, gains, or torques. Torque-controlled robots, on the other hand, provide a completely open control architecture, whereby the user can specify the motor currents or joint torques directly as a function of all accessible states. While the position-based control is still by far the most widely used architecture, because it requires less costly sensors, actuators and processors, and is more robust, serviceable, and fool-proof, researchers have always had a preference for the versatility and openness of the torque-based control. The introduction of relatively large physical compliance in soft actuators changes the picture considerably. This is particularly true in VIA systems, where the user is provided with the control of neither the joint positions nor torques directly. Rather, one can specify the neutral equilibrium position of the joint, and the stiffness (and damping) by which the joint is attracted toward it, when external or inertial loads displace it from such equilibrium. To illustrate the difference, consider building a joint for a fully dynamic task, such as jumping on one leg: while position- or torque-control architectures would hardly be able to do the job (unless using very high-performance components), two relatively slow, low-cost servomotor units connected through springs in agonist–antagonist arrangement can already provide a very effective solution.

It should be noted here that such VIA control model is very similar in philosophy to the λ version of the equilibrium-point hypothesis in human motor control [21.18–20]. According to the equilibrium point hypothesis, movements arise as a consequence of shifts in the equilibrium of the motor system. This equilibrium is dependent upon the interaction of control signals, reflexes, muscle properties and loads, but it is under the control of central arm movement [21.21]. In this sense, soft robotics also provide a very interesting arena for exchanging ideas and insights with the research community in motor neuroscience.

21.2 Soft Robot Design

In this section, we provide a concise presentation of the main elements and technologies that may contribute to the development of novel actuators for soft robotics.

21.2.1 Natural Muscles

For comparison and inspiration, it is natural to consider first the characteristics of natural muscles. A typical muscle consists of many thousands of *muscle fibers* working in parallel, organized in *motor units*, that is, one or several fibers with a motor neuron activating them. A single muscle fiber contains several *myofibrils*, made of *sarcomere* separated by *Z disks*. In a sarcomere, *thin (actin) and thick (myosin) filaments* are arranged in a hexagonal lattice, such that one thick filament interacts with six thin filaments. *Cross-bridges* linking actin and myosin pull toward sacromere center from both ends resulting in muscle shortening. Actin and myosin filaments slide against each other byrepeated attachment and detachment of cross-bridges, thus resulting in muscle contraction.

As a consequence of their structure, natural muscles have several mechanical properties which differ from conventional actuators. Muscles (and tendons) have spring-like properties, with stiffness and damping increasing with activation, and are capable of storing and releasing energy, but also of dissipating it. Muscles are equipped with sensors which can provide stretch (e.g., spindles) and force (e.g., Golgi tendon organs) information to the nervous system. Muscle tension increases nonlinearly with activation, and – for fixed activation – depends on both length and velocity. Muscles are most often arranged in antagonistic pairs (such as the bicep and tricep in the elbow) or groups (such as in the shoulder): this fact, together with the nonlinearity of their force–stretch relationship, is used to modulate the effective impedance at the joints.

21.2.2 Breaking Down Artificial Muscle Systems

Even a nonexhaustive but reasonably broad analysis of technologies potentially contributing to the development of novel actuators for soft robotics could not fit this handbook chapter. However, we will attempt here to provide a simple categorization of the subject matter, along whose lines a more complete survey could be organized. To this purpose, we will break down a soft robot system into a network of simpler elements, whose behavior is simpler and more easily described.

Like other systems, a soft actuator is comprised of *storage* elements that keep energy inside the same energetic domain, and of *transducers* that allow a flow of energy from one domain to another. Transducers may dissipate energy via heat production (energy *sinks*), or can instead inject energy inside the system (energy *sources*). Sources can, in turn, be divided into *flow* sources, *effort* sources, and *hybrid*, or *real* sources. A flow source determines the amount of flow (linear or angular velocity, heat flow, volume or mass flow, . . .) going through it, without caring for the effort (force, torque, temperature, pressure, . . .) needed to keep such flow. The converse holds for an effort source. A hybrid source, instead, is a device that offers a generic constraint between the effort/flow pair.

Storage and transducer elements can be connected in different arrangements, which may result in completely different behaviors. VIA, an important class of actuators for soft robotics, have the characteristic that their overall behavior can be changed under active control between that of an effort and a flow source.

21.2.3 Sources for Soft Actuators

Technologies used as sources for soft actuators vary broadly in nature and scope. In this section, we limit ourselves to provide a list with comments on the range of applicability, while we refer the reader to specific literature for more detailed analysis: for instance, the author of [21.22] compares several active materials in a volumetrically normalized fashion, including stress (force-generating capability), strain (stroke length), speed of actuation, and power and energy metrics.

21.2.4 Pneumatic and Hydraulic

Pneumatic and hydraulic actuators [21.23, 24] include classical tie-rod or rod-less cylinders and rotary motors. Pneumatic actuators are usually low-impedance effort sources, while hydraulics tend to be high-impedance sources, but compliant behaviors can be actively controlled, or obtained through combination with other elements in the actuator system arrangement.

Of specific interest for soft robotics are the so-called *pneumatic artificial muscles* (PAMs), a name used to describe contractile linear-motion engines operated by gas pressure. The actuator's core element is a flexible membrane or shell attached at both ends to fittings along which mechanical power is transferred to a load. As the membrane is inflated, it bulges outward and expands radially, and contracts axially thus exerting a pulling force on its load. The force and motion thus generated by this type of actuator is typically unidirectional in tension (different from bellows, which work in compression). Pneumatic artificial muscles (PAM) became popular in robotics through the work of McKibben, who used braided sleeves around an elastic bladder as an orthotic actuator in the late 1950s [21.25], and have been widely used since then in robotics in various forms [21.26–32].

PAMs have an intrinsically nonlinear characteristic (force decreases with contraction) and, due to gas compressibility, they exhibit a natural low impedance. As they act one way, two PAM are needed to generate bidirectional motion in an antagonistic arrangement. By a pressure differential on the two muscles, a net joint torque is applied. On the other hand, due to nonlinearity of the stretch/force characteristic, an increase in the common-mode pressure varies the joint stiffness, while leaving torque unaffected. The above characteristics of PAMs are somehow similar to those natural muscles, although many important differences exist. From a practical viewpoint, the main drawbacks of pneumatic muscles are the encumbrance, noise, and slow response related to use of pressurized gas, and their relatively limited energy density.

21.2.5 Electromagnetic and Electrostatic

Electromagnetic motors use Lorentz-type forces acting on electric charges moving in a magnetic field. Examples are solenoids, that is, electromagnetically inductive coils wound around a movable steel or iron slug (the armature) [21.33], and rotating or linear induction motors [21.34]. Electromagnetic sources are typically effort sources, although often they have to be matched to the system impedance through gears, which may transform their behavior substantially. On the other hand, stepper motors, that is, brushless synchronous electric machines that divide a full rotation into a large number of steps, can be considered ideally as position sources. Stepper motors are mainly of three main types, that is, permanent magnet, hybrid synchronous, and variable reluctance steppers.

Electrostatic actuators are based on the attraction and repulsion of electric charges [21.35]. They typically require high-driving voltages at low currents (dual in a sense to electromagnetic motors). Linear electrostatic motors can be realized using comb-drive arrangements. While electrostatic drives at macroscopic scales are impractical due to large voltages, they have excellent potential at micro or nanoscale, where microelectromechanical systems (MEMSs) with moving, charged plates are far easier to fabricate than coils and iron cores.

21.2.6 Piezoelectric

The *reverse piezoelectric effect*, that is the production of stress and/or strain in a piezoelectric material when an electric field is applied can also be used for actuating robots. Most used inorganic materials for piezoelectric transducres consist of ceramics with perovskite or tungsten-bronze structures, such as barium titanate ($BaTiO_3$), lead titanate ($PbTiO_3$) and lead zirconate titanate ($Pb[Zr_xTi_{1-x}]O_3$, $0 < x < 1$), commonly known as lead zirconate titanate (PZT) [21.36], while the direct effect of piezoelectric polymers such as polyvinylidene fluoride (PVDF) is mostly used for sensors.

Elementary piezoelectric actuators are best suited for microscale applications, where they can reach very high accuracy. Piezoelectric motors are sometimes used in resonant mode with rectification mechanisms that transform oscillations in net displacements (e.g., traveling-wave, inch-worm and stick-slip stepping motors). Multilayer (stacks) actuators, piling up several layers (usually thinner than 100 μm), are used to increase the actuator stroke, while retaining continuous and reversible operation. Stacked actuators are sometimes used within a mechanical amplifier structures, which can reach millimeter strokes (Fig. 21.1) Amplified piezoelectric stack actuators are commercially available from Dynamic Structures and Materials (Franklin, TN, USA) and Cédrat (Meylan, France). In essence, the compliant mechanism accomplishes a force–displacement tradeoff from input to output at the cost of storing energy within the compliant mechanism. This principle has been used in several mechanisms, such as in [21.37].

Ueda et al. [21.38] envision amplified piezoelectric stack actuators as modular units in a *cellular actuator*, drawing a parallel with muscles in that the overall actuation for a given joint is the sum of a number of discrete actionable units connected by compliant material. In the same way, muscles are composed of individual fibers grouped into motor units whose effects are coupled by elastic tendons. In order to function as modular, muscle-cell like units, the compliant mechanism in the amplified piezoelectric stacks need to have very aggressive force–displacement tradeoffs while fitting into a compact volumetric envelope. This is accomplished by using a hierarchical set of strain amplifiers, where the output of each stage in the amplification is the input to the subsequent stage. Each stage fits inside the subsequent stage; hence, this is referred to as a *nested* strain amplification mechanism. *Schultz* and *Ueda* [21.39, 40] present a framework to describe and specify the geometry of a nested configuration that results in a favorable tradeoff that meets design specifications. The work of *Schultz* and *Ueda* [21.41] draws yet another parallel with muscle, the idea that the central nervous system has authority over each motor unit in the muscle and activates each one in an on–off manner. This, in essence makes the control of muscles or muscle-like actuators a switching, rather than analog interface, and circumvents the problem of hysteresis inherent in positioning piezoelectric material with an analog voltage. This actuator type also incorporates this idea of redundancy; a lost motor unit (dead piezoelectric stack) does not result in total failure, but rather a loss of performance.

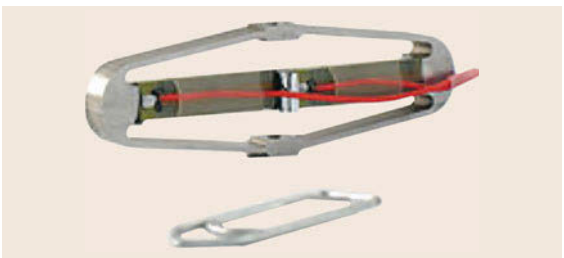


Fig. 21.1 Example of an amplified piezo actuator with multilayer stacks

21.2.7 Electro- and Magneto-Strictive

The electro- and magnetostrictive effects exhibited by some materials, that is, the tendency to change their shape under electric or magnetic fields, can also be exploited to build microscale actuators. Electrostriction differs from piezoelectricity because it is exhibited by all dielectric materials, and is proportional to the square of the polarization (instead of linearly). Reversal of the electric field does not reverse the direction of the deformation. Certain engineered ceramics, such as lead lanthanum zirconate titanate (PLZT) have high-electrostrictive constants and can produce strains of the order of 0.1% at field strengths of the order of 1 MV/m. Electrostrictive actuators are typically structured and used like piezoelectric actuators. Similarly, magnetostrictive materials deform under magnetic fields and in proportion to their magnetostrictive coefficient [21.42]. One of the strongest effects is exhibited by Terfenol-D, with strain of the order of 0.1% in a field of the order of 100 kA/m at room temperature.

The pronounced thermal expansion of some materials (e.g., doped single-crystal silicon or polysilicon) can be used to transform heat in mechanical energy and generate motion at micro scales. Temperature is typically controlled by Joules's effect, or through local heat injection, while mechanical amplification is often introduced with MEMS techniques (e.g., symmetric bent-beam or asymmetric bimorph structures).

21.2.8 Shape Memory Alloys and Polymers

Macro-scale thermal actuators can otherwise be built using shape memory materials, which include alloys (shape memory alloys, SMA) and polymers (shape memory polymers SMP). Alloys like copper–zinc–aluminum–nickel, copper–aluminium–nickel, and nickel–titanium (NiTi) alloys [21.43] exhibit shape memory when transitioning from martensitic to austenitic phase (one-way effect) and, under particular conditions, also on the opposite phase change (two-way effect). SMA can exhibit up to 8% recoverable strain, and can be formed in virtually any shape. Energy inefficiency, slow response times, and large hysteresis are the main limitations. *Torres-Jara* et al. [21.44] have created an SMA muscle-like module in a convenient form factor by combining laser-cut SMA sheets with plastic hubs so that the hubs move away from one another when energized. These modules can be connected together in series or parallel to make an overall actuator with the desired properties.

SMPs are relatively new shape memory materials, which can be triggered by temperature but also by electric or magnetic field, light or solution [21.45, 46]. The shape change happens with transitions to glass phase, crystallization, or melting. While mechanical properties of SMPs are not yet comparable to SMAs, SMPs have an enormous potential for their versatility, as they range from stable to biodegradable, from soft to hard, and from elastic to rigid polymers.

21.2.9 Polymeric Actuators

Shape memory polymers represent only an example of viscoelastic materials which can be used for actuation. Very intense research has been pursued in recent years on actuators based on several different phenomena exhibited by polymeric materials [21.47]. Electroactive polymers (EAPs) are divided into ionic EAPs (based on diffusion of ions and solvents) and electronic EAPs (based electronic charging of the material). Ionic EAPs include polyelectrolyte gels [21.45, 46], ionic polymer metal composites (IPMC, such as Naftion by DuPont), conducting polymers (such as polypyrrole (PPy) and polyaniline (PANi)). Electronic EAPs include piezoelectric [21.48], electrostrictive, and dielectric polymers and elastomers (such as acrylic polymers and silicones [21.49]), and flexoelectric polymers, such as liquid crystal elastomers [21.50].

21.2.10 Carbon Nanotubes

Carbon nanotubes (CNTs) have a nonpolymeric macromolecular structure consisting of a graphite monoatomic sheet rolled to form tubes whose length can be about 1000 times their diameter. CNT actuators can be realized by using sheets of single- or multiwalled nanotubes as electrolyte-filled electrodes of a super capacitor [21.51], developing up to 0.75 MPa (cf. 0.3 MPa of the human muscle tissue) at low voltages, though still with limited stroke. A recent breakthrough in CNT actuators has been the use of highly ordered aerogel sheets, which allowed fabrication of actuators with giant strokes (ca. 180% actuation along the width) with fast response (5 ms delay) [21.52], which is slightly better than the human muscle. The artificial CNT muscle is stronger than steel in one direction, and more flexible than rubber in the other two directions [21.53]. When large strokes will be obtained at low voltages, the potential of CNT actuators for the design of artificial muscles is clearly enormous.

21.2.11 Energy Storage

Although energy storage elements in artificial muscles use a variety of materials and structures, due to the need of having mechanical energy readily available in dynamic phases, storage in the form of elastic energy prevails. This can be done by either pressurized gas systems (mainly in connection with hydraulic or pneumatic actuation systems), of by deformation of solid elements, that is, springs. Materials for storing a large amount of elastic energy in a small volume, and with limited losses, include special metal alloys (e.g., medium- and high-carbon steel, beryllium copper, and phosphor bronze). Composite materials, in the three main types available (polymer matrix composites (PMCs), metal matrix composites (MMCs) and ceramic matrix composites (CMCs)) offer a wider versatility in obtaining different stiffness, which depends not only on the constituent materials but also on the fiber design. Advanced needs of soft robotics applications motivate research in using sophisticated materials for energy storage, such as, for example, carbon nanotubes, elastomers, and superelastic materials.

Mechanical energy can also be stored in kinetic form through the use of inertial elements, similar to the kinetic energy recovery system (KERS) used in F1 racing cars. However, the mismatch with typical slow speeds of robotic joints introduces the need for gears. Conceptual design of CVT (continuous variable transmission)-based VIAs have been proposed, although research in this direction is still in its infancy.

21.2.12 Energy Dissipation and Dampers

Energy *sinks*, that is, dissipation systems and dampers, are another important element in the construction of

a soft robot actuator. Indeed, underdamped behaviors are both nonnatural looking and potentially dangerous. Damping can be obtained through real-time control, or directly in hardware via electrical shunting or with mechanical devices. The latter have been extensively applied to a diverse range of cases, from the control of excessive structural response to transient environmental disturbances in civil structures, to automotive suspension systems, and more recently in robotics.

Dampers are often classified as either passive, semiactive or active systems, depending on the amount of external power required to perform its functionality. Dampers can be realized using different operating principles, which we briefly summarize here.

A friction damper (FD) is essentially composed of an actuator that applies a normal force on the output shaft, and generates friction as a consequence of relative motion ([21.54] for modeling). The control of the compression force between the frictional surfaces controls friction and can simulate damping: this has been effectively used in robotics to implement variable physical damping [21.55]. Drawbacks include the possibility for hysteresis and dead zone effects, due to static friction [21.56].

Electrorheological (ER) and magnetorheological (MR) dampers are based on liquids whose physical behavior depends on the application of electric or magnetic fields, respectively [21.56]. These fluids follow the Bingham models: after a yielding point, they behave as viscous liquids. The property that can be changed is the yield stress itself. The MR operating principle has been used to realize VDAs in vehicles [21.57] and also in robotics [21.58]. A more accurate model and a comparison between MR and FD can be found in [21.54], where it is pointed out that MR dampers, like the FDs, present high hysteresis.

Eddy current dampers (ECDs) are magnetic devices composed of a conductive material moving through a magnetic field. Eddy currents are induced and create a damping force that is proportional to the relative velocity between the material and the magnetic field. These devices can be realized with both permanent magnets and electromagnets. In both cases, there is the possibility of designing a device whose damping can be adjusted [21.59, 60]. In one case, the damping coefficient can be controlled by varying the intensity of the magnetic field, in the other case, by modifying the geometry of the conductor, or the gap between the conductor and the magnets (the effectiveness is shown in [21.60]). Electromagnetic ECDs have the advantage of not requiring mobile parts, but have the disadvantage of consuming power for maintaining a fixed damping value. Being fluid-free and contact-free, ECD dampers are clean and wear-free. However, they typically exert low damping torque at high velocity, thus requiring gearboxes for typical robotic applications.

Fluidic dampers are probably the most widely used in robotics and general machinery and can be divided into two main categories: turbulent flow dampers (high Reynolds number) which produce a damping force proportional to the square of the relative speed; and laminar flow dampers (low Reynolds number), where the damping force is proportional to relative speed. Simple dampers are largely used in the automotive industry using an orifice through which the viscous fluid flow is turbulent. Such device generates, at a given frequency, high damping for high amplitudes, but lower damping for lower amplitudes, and thus, has the drawback of presenting long lasting residual oscillations [21.56].

21.2.13 Artificial Muscles Arrangements

The basic technologies and elements reviewed above can be arranged in many different ways to achieve actuators for soft robotics. While a detailed taxonomy of arrangements is impossible here, the main distinction is between actuators that have a constant mechanical compliance characteristic, and those which can vary stiffness. In the first class are series elastic actuators (SEA), which have been developed since the 1990s for applications in humanoid locomotion and manipulation [21.61]. An SEA may include a conventional, rigid actuator connected to a compliant element, which is in turn connected on the other end to the moving link. The compliant element can exhibit a nonlinear compliance (load-deformation) characteristic, which however, is constant in time. SEA actuators, however, may use torque and position sensing at both ends of the elastic element and modify its natural mechanical impedance by virtue of active control of the motor. The impedance range to which an SEA can be effectively controlled is centered around the natural (passive) impedance of the elastic element and is limited in span by motor torque limits and overall control bandwidth.

As opposed, VSA can vary their natural impedance directly at the physical level, so that active control can be superimposed and ultimately achieve larger ranges of effective impedance than SEA. The idea of varying the mechanical impedance of actuation comes directly from natural musculoskeletal systems, which often exhibit this feature. VSA systems can change independently the equilibrium point of the moving body part, and the stiffness of the elastic force between the equilibrium point and the displaced body position (thus effectively implementing Feldman's *equilibrium point*, or λ-model for human movement control [21.62]). Accordingly, VSAs always use two prime movers. A rough distinction can be made between VSA ar-

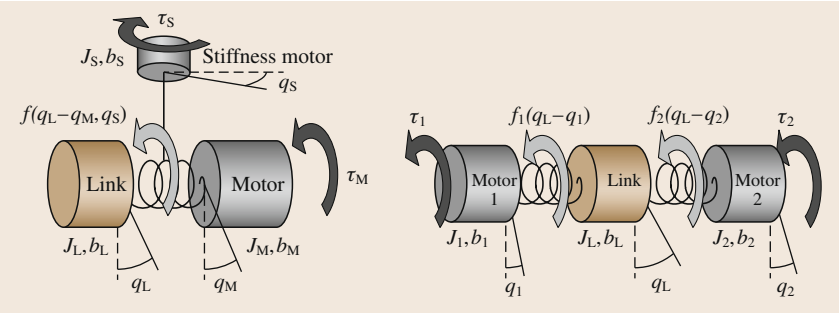


Fig. 21.2 Explicit stiffness variation and agonist–antagonist arrangements

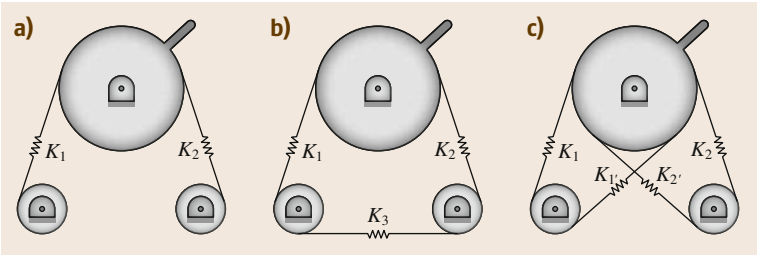


Fig.21.3a–c Three basic antagonist arrangements: **(a)** Simple, **(b)** cross-coupled and **(c)** bidirectional

rangements where the motors are explicitly dedicated to either equilibrium or stiffness control, and arrangements where the roles are mixed (Fig. 21.2). Explicit stiffness variation arrangements include, for example, the MIA (mechanical impedance adjuster) [21.63]), the AMASC (actuator with mechanically adjustable series compliance) [21.64], the MACCEPA (mechanically adjustable compliance and controllable equilibrium position actuator) [21.65]), and the variable stiffness joint (VS-Joint) [21.66]. Other VSA designs use instead agonist–antagonist (AA) arrangements, either in its simplest version, directly inspired to biological models, with two actuators pulling against each other [21.67, 68], or in other variants such as the cross-coupled AA [21.69] and bidirectional AA [21.70] arrangements (Fig. 21.3) It is a matter of simple calculations [21.71] to appreciate that, in an AA arrangement, the elastic elements must exhibit a nonlinear characteristic in order for the joint stiffness to be variable. If each tendon has a quadratic force–deformation relationship, then the overall joint behaves as a linear spring with variable spring constant K. A potential disadvantage of AA arrangements using unilateral (e.g., tendon) couplings between the prime movers and the driven link is that, to be able to exert a maximum torque τ at the link, two motors each capable of τ are needed. To alleviate this problem, the cross-coupled and bidirectional design introduce the possibility for an agonist actuator to also work in favour of the antagonist when required.

Recent work in soft robotics has introduced physically variable damping on the rotor dynamics in conjunction with series elastic or VSAs [21.72]. Physically variable damping based on the ECD effect has been demonstrated in haptic interfaces [21.73]. Some authors introduced physical dampers in robotic design, both in the so-called series damping actuator configuration [21.58], or as a variable friction damper connected to an SEA [21.55], and as a variable diaphragm laminar damper module connected to a variable stiffness actuator [21.74].

21.2.14 Specifications and Performance

The increasing number of available actuators for soft robotics and the widening scope of applications are rapidly creating a need for a simple language for application developers and actuator providers to talk together. Just like the perspective user of conventional hydraulic pistons or DC motors looks at spec sheets for a concise description of the performance guaranteed by the product, a soft robot designer will need to find salient facts about different artificial muscles presented in a clear and homogeneous way. The European project VIACTORS has dedicated a substantial effort to investigate what the crucial parameters of different designs are and has described a data sheet template that organizes these contents both from the designer and the user's points of view [21.75–77]. The interested reader is referred to these publications for details: it is interesting to note here that, besides the characteristics that are shared with conventional actuators (stall and peak motor torque, max velocity, power, electrical and mechanical interfaces, etc.), there are few data and curves

that are specific of soft actuators: for instance, the range of stiffness variability, and the settling time needed to reach from minimum to maximum stiffness, or vice versa.

21.3 Modeling Actuators for Soft Robotics

To approach the analysis and control of novel actuators for soft robotics, it is necessary to describe their characteristics in terms of a mathematical model which accounts for their static and dynamic behaviors. We begin here by reviewing the familiar concept of mechanical impedance in linear time-invariant single degree of freedom (DOF) systems. Hence, in view of the fact that actuator softness control often introduces nonlinearity and time variance of the model, we proceed to generalize the concept. Finally, we move from consideration of a single actuator to modeling of multi-DOF, articulated soft robots.

21.3.1 The Mass–Spring–Damper Paradigm

The simplest example of mechanical impedance is a linear *spring*, that is, a sample of material which exhibits a proportional relation between applied force f and displacement at equilibrium y, $y = cf$. The proportionality constant c is the spring *compliance*. The inverse relation, that is, $f = ky$ introduces the spring *stiffness* $k = c^{-1}$. A spring can store an elastic energy V, whose first derivative with respect to the displacement y is the force

$$f = \frac{\partial V(y)}{y} ,$$

and the second derivative is the spring stiffness constant

$$k = \frac{\partial^2 V(y)}{\partial y^2} .$$

The concepts of compliance and stiffness are generalized from static (equilibrium) to dynamic cases by introducing the notion of *admittance* and *impedance*, respectively.

Consider a *damper* made of viscous material that opposes to displacements proportionally to how quick they are, that is, $f = b\dot{y}$, where b is the damping coefficient, and the inertial effects of accelerating an equivalent *mass* m of material under deformation, $f = m\ddot{y}$. In most materials, the three effects occur at the same time, and we have that the force/displacement relation is described by an ordinary differential equation. In the simple case of a single-DOF mass–spring–damper system, we have

$$f = m\ddot{y} + b\dot{y} + ky , \tag{21.1}$$

that is, a relation between two functions of time, $f(t)$ and $y(t)$. Such a differential equation describes a dynamical system. This can be regarded as a causal system if displacement (appearing with the highest-order derivative) is regarded as the effect (or output), and force as the cause (or input). Indeed, the knowledge of initial conditions on displacement, and the time course of force, determine all subsequent evolution of displacement.

Insofar as the dynamical system (21.1) is linear and time-invariant, its input–output behavior is conveniently studied via Laplace transforms from functions in the domain of time t to functions in the domain of the complex variable s. By denoting $F(s)$, $Y(s)$, $V_y(s)$ the transforms of $f(t)$, $y(t)$, and $v_y(t) = \dot{y}(t)$, respectively, one has immediately

$$\begin{aligned} F(s) &= (ms^2 + bs + k)Y(s) \\ &= (ms^2 + bs + k)s^{-1}V_y(s) . \end{aligned} \tag{21.2}$$

The operator $Z(s) := (ms^2 + bs + k)s^{-1}$ is called (mechanical) *impedance* of the spring–damper–mass system. The reciprocal operator of impedance is called *admittance* $A(s)$. Notice that the admittance operator is causal, while impedance is not.

21.3.2 Nonlinear Mechanical Systems

Real materials and systems deform under applied forces in more complicated ways than (21.1): in general, the dynamics between forces and deformations include nonlinearities, and can be of an order higher than two. How do we extend to these cases the concept of *impedance* (21.2), and specifically the notions of *mass, stiffness* and *damping*?

There is no easy clearcut solution to this problem. A straightforward approach is to find a linear, second-order model of type (21.1), which approximates well enough the real dynamics, and take its parameters to define those of the system. In this section we briefly review some of the implications of this approach.

Consider the relation between a force $f \in F$ (F here is the set of allowed forces) applied at some point in a material sample, and the displacement $y \in Y$ measured at equilibrium at the same point and along the same direction of the force (Y the set of allowed displacements). By ideally collecting all pairs $(f(t), y(t))$ corresponding in an infinite set of experiments (indexed

by t for convenience), one can describe the relation by its graph $G \subset F \times Y$. Notice that this relation does not need to be a function: for hysteretic materials, for example, there are more admissible values of one variable corresponding to the same value of the other (Fig. 21.4).

Considering an analytic description of the graph G given by the locus $G(f, y) = 0$, at regular points of the graph a *force function* $f(y)$ is defined for values of y in a neighborhood of y_0. The stiffness associated with such a nonlinear force function can then be defined as

$$k(y) = \frac{\mathrm{d}f(y)}{\mathrm{d}y} = -\left(\frac{\partial G(f,y)}{\partial f}\right)^{-1} \frac{\partial G(f,y)}{\partial y} .$$

This can be generalized to a second-order, nonlinear dynamic setting by considering the relation between forces, displacements, and first- and second-order derivatives of displacements and its graph $G \subset F \times Y \times DY \times D^2Y$, comprised of 4-tuples $\mathrm{d}(t) := (f(t), y(t), \dot{y}(t), \ddot{y}(t))$ corresponding to an idealized, infinite set of experiments. (If hysteretic nonlinearities are considered, it is necessary to describe the relation in an extended space where hysteresis state variables are introduced [21.78].) If $G(f, y, \dot{y}, \ddot{y}) = 0$ is an analytical description of the graph, and d_0 is a regular point, then a *force function* $f(y, \dot{y}, \ddot{y})$ is defined in a neighborhood of d_0. More generally, when the system includes time-varying parameters $u(t)$ – such as, for example, a stiffening action in VSAs – one has $G(f, y, \dot{y}, \ddot{y}, u(t)) = 0$ and $f(y, \dot{y}, \ddot{y}, u)$. Following the linearization approach set out above, we can define admittance *along a given (nominal) trajectory* as the linear operator mapping small changes of the external force with respect to its nominal course, to changes in the resulting motion. To do this, consider the nonlinear ordinary differential equations (ODE) obtained by solving $G(f, y, \dot{y}, \ddot{y}, t) = 0$

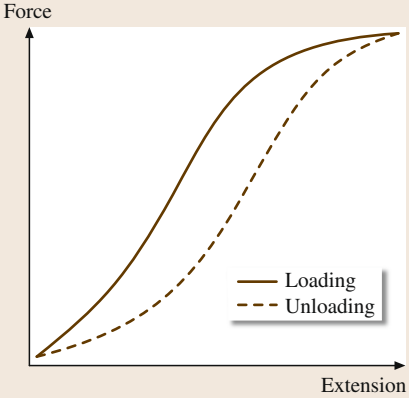


Fig. 21.4 Elastic hysteresis of a rubber-like material

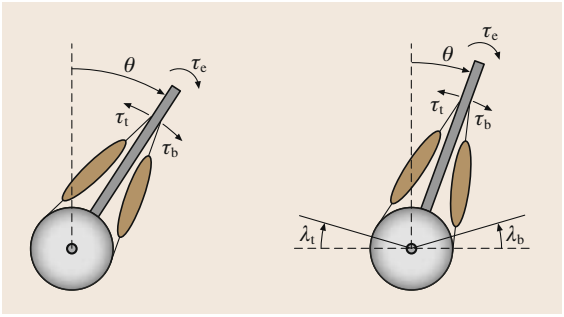


Fig. 21.5 A link subject to an external load and actuated by two antagonistic actuators. Nomenclature refers to actuator models used in (21.3) and (21.5), respectively

at a regular point with respect to $\ddot{y}$ as

$$\ddot{y} = g(y, \dot{y}, f, u) ,$$

and its state space form, with $x \in \mathbb{R}^2$, $x_1 = y$, $x_2 = \dot{y}$, that is,

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_2 \\ g(x_1, x_2, f, u) \end{pmatrix} .$$

For *given initial conditions* $x(0) = \bar{x}_0$ and a *given nominal course in time* of the force $\bar{f}(t)$, let $\bar{x}(t)$ be the nominal solution obtained. The first-order approximation of the dynamics of the perturbed motion $\tilde{x}(t) = x(t) - \bar{x}(t)$ corresponding to a change in force $\tilde{f}(t) = f(t) - \bar{f}(t)$ is the linear time-varying system

$$\dot{\tilde{x}} = \begin{pmatrix} 0 & 1 \\ -\kappa(t) & -\beta(t) \end{pmatrix} \tilde{x} + \begin{pmatrix} 0 \\ \mu(t) \end{pmatrix} \tilde{f} .$$

For the given system *and* the given motion, hence, we can define instantaneous mass, stiffness, and damping coefficients as

$$m(t) = \mu^{-1}(t) ;$$
$$b(t) = m(t)\beta(t) ;$$
$$k(t) = m(t)\kappa(t) .$$

Example 21.1

Consider the link in Fig. 21.5, actuated by two antagonistic actuators (with a role vaguely similar to that of the biceps and triceps muscles at the elbow), with a quadratic damping and subject to gravity and to an external torque load τ_e. The system dynamics are

$$I\ddot{\theta} + \beta\dot{\theta}|\dot{\theta}| - \tau_b + \tau_t - \mathrm{mgl}\sin\theta - \tau_e = 0 .$$

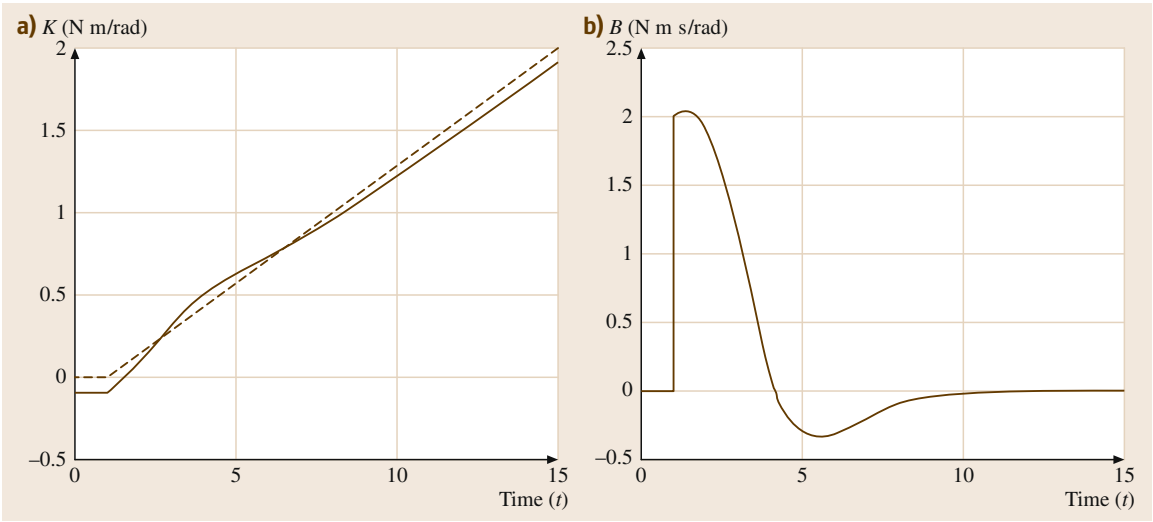


Fig.21.6a,b Generalized stiffness **(a)** (*dashed* is without gravity term) and generalized damping for the example with actuators as in (21.3), subject to a unit step in external torque at $t = 1$ s, and **(b)** with time-varying activation $u_b(t) = u_t(t)$ linearly increasing from 0 at $t = 1$ s to 1 at $t = 15$ s. Numerical values used in simulation: $I = 0.05\,\mathrm{Nm\,s^2}$, $mgl = 0.1\,\mathrm{Nm}$, $\beta = 1\,\mathrm{Nm\,s^2}$, $\alpha = 1\,\mathrm{Nm}$, $\tau_{max} = 2\,\mathrm{Nm}$

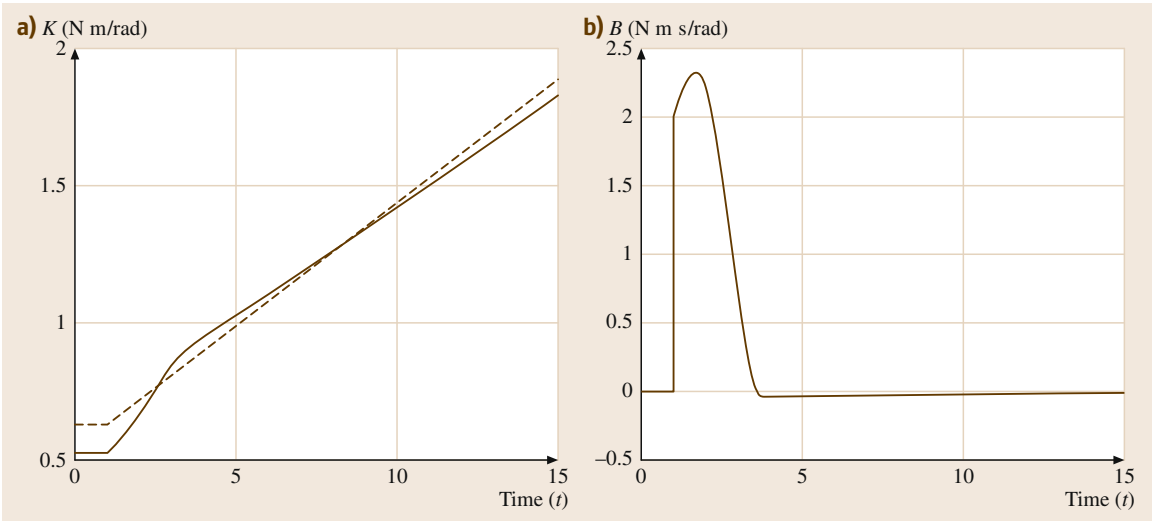


Fig.21.7a,b Generalized stiffness **(a)** (*dashed* is without gravity term) and generalized damping for the example with actuators as in (21.5), subject to a unit step in external torque at $t = 1$ s, and **(b)** with time-varying reference angle $u_b(t) = u_t(t)$ linearly decreasing from $\pi/3$ at $t = 1$ s to 0 at $t = 15$ s. Numerical values used in simulation as in Fig. 21.6, except for $\alpha = 0.3\,\mathrm{Nm}$

Assume first that the two actuators generate torques according to the model [21.67]

$$\tau_b = (\tau_{max} - \alpha\theta_b)\, u_b\,, \qquad \tau_t = (\tau_{max} - \alpha\theta_t)\, u_t\,, \tag{21.3}$$

where $\theta_b = (\pi/2 + \theta)$, $\theta_t = (\pi/2 - \theta)$, τ_{max} is the maximum isometric torque, u_b, u_t are the normalized contraction parameters ($0 \le u \le 1$, $u = u_b$, u_t), and α is a constant assumed to be equal for the two actuators. We easily obtain $m = I$ for the generalized mass, $b(\dot\theta) = 2\beta|\dot\theta|$ for the generalized damping, and

$$k(\theta, u) = \alpha(u_b + u_t) - \mathrm{mgl}\cos(\theta) \tag{21.4}$$

for the generalized stiffness. In the latter expression, the

role of a gravity-induced term and that of a co-activation stiffness term are apparent.

If a different actuator model is adopted, namely [21.79]

$$\tau_b = -\alpha(\theta_b - u_b)^2 ,$$
$$\tau_t = -\alpha(\theta_t - u_t)^2 , \tag{21.5}$$

where u_b, u_t are now interpreted as the rest lengths of the actuators, one has

$$k(\theta) = 2\alpha(\pi - \lambda_b - \lambda_t) - mgl\cos(\theta) .$$

The values of stiffness and damping for the two examples above, corresponding to time-varying values of the control parameters, are reported in Figs. 21.6 and 21.7, respectively. ■

So far, we have considered only impedance for second-order dynamical systems – basically modeling actuators with a single moving element that dominates the system inertia. In practice, one may encounter higher order systems because of either (i) more complex actuator systems, or (ii) the connection of actuators and links in an articulated robot structure. The latter case will be discussed in Sect. 21.4.

Examples of actuators with higher order dynamics are the arrangements in Fig. 21.3: if the agonist and antagonist prime movers as well as the driven link have nonnegligible inertia, the dynamics are at least of sixth order. If the system is linear, admittance can still be defined as the sixth-order transfer function from link torque to link displacement: however, there is no natural way of associating an equivalent single mass, spring or damper coefficient. To recover these notions, an approach could be to apply some form of model reduction of the higher order dynamics to second order [21.80]. However, model reduction is all but an obvious procedure, as the approximate heavily depends on how the similarity of response between dynamics is measured. With nonlinear systems, the problem is clearly even more complicated. Overall, the topic is definitely worth further investigation, which should take into account what are the ultimate practical goals of defining impedance parameters.

21.4 Modeling Soft Robots

In this section, a brief introduction of dynamic models for soft robots is given, starting with a quite general lumped-parameter formulation and successively introducing several simplifying assumptions.

A soft robot with lumped elasticities can be imagined as a set of directly actuated rigid bodies with configuration $\boldsymbol{\theta} \in \mathbb{R}^m$, connected to the indirectly actuated rigid bodies (with configuration vector $\boldsymbol{q} \in \mathbb{R}^n$) through viscoelastic forces, as exemplary sketched in Fig. 21.8. The entire configuration space of the system is denoted by $\boldsymbol{x} = (\boldsymbol{\theta}, \boldsymbol{q})$, $\boldsymbol{x} \in \mathbb{R}^{(n+m)}$. A quite general abstraction of a compliant robot, which can be used for the generic design of controllers, is given by

$$\mathbf{M}(\boldsymbol{x})\ddot{\boldsymbol{x}} + \mathbf{c}(\boldsymbol{x}, \dot{\boldsymbol{x}}) + \left(\frac{\partial V(\boldsymbol{x})}{\partial \boldsymbol{x}}\right)^{\mathrm{T}} + \boldsymbol{d}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \begin{pmatrix} \boldsymbol{\tau_m} \\ \boldsymbol{\tau}_{\mathrm{ext}} \end{pmatrix}, \tag{21.6}$$

with $\mathbf{M}(\boldsymbol{x})$ being the inertia matrix, $\boldsymbol{c}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ the Coriolis and centrifugal vector, V the potential energy of the elastic element and of the gravity forces, $\boldsymbol{\tau}_m \in \mathbb{R}^m$ the actuator generalized forces acting as control inputs, and $\boldsymbol{\tau}_{\mathrm{ext}} \in \mathbb{R}^n$ the external torques acting on the robot as a disturbance. The most relevant property of this structure is its underactuation, meaning that the system has less control inputs ($\boldsymbol{m}$) than its configuration space dimension ($\boldsymbol{n} + \boldsymbol{m}$). However, in contrast to other purely inertially coupled underactuated systems (for example multiple pendulums such as the Acrobot [21.81].), for the considered robots $V(\boldsymbol{x})$ is [21.82] positive definite, implying that a unique equilibrium point exists for each external torque with actuators in a fixed configu-

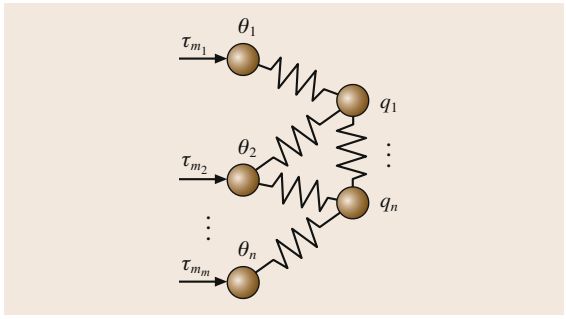


Fig. 21.8 A soft robot can be imagined as a set of directly actuated rigid bodies with configuration $\boldsymbol{\theta}$, connected to the indirectly actuated rigid bodies (with configuration coordinates $\boldsymbol{q}$ through elastic forces). Note that in general, inertial couplings, not depicted in the figure, might exist as well

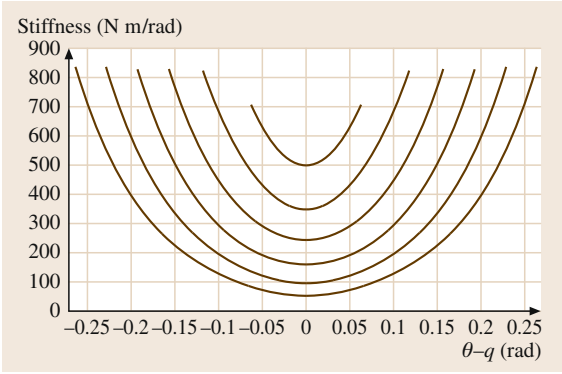


Fig. 21.9 Typical shapes of elastic torque characteristics, denoted by nonlinear deflection dependency and a family of linear curves, parameterized by $\boldsymbol{\sigma}$

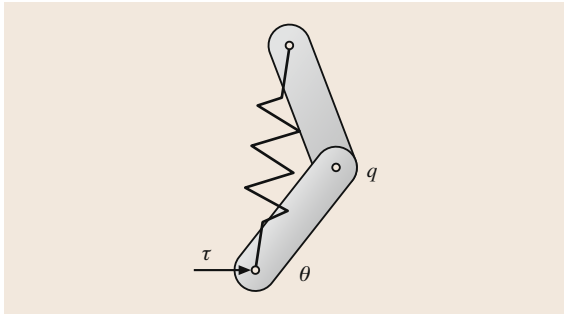


Fig. 21.10 In this example, the directly and the indirectly actuated states are both inertially and elastically coupled. This leads to a fully coupled inertia matrix in the dynamics equations

ration $\boldsymbol{\theta} = \boldsymbol{\theta}_0$ and that the linearization of the system around an equilibrium point $\{\boldsymbol{x} = \boldsymbol{x}_0, \dot{\boldsymbol{x}} = 0\}$ is controllable. Typically, $V(\boldsymbol{x}) = V_G(\boldsymbol{x}) + V_\tau(\boldsymbol{x})$, that is, the potential function is the sum of a gravity potential V_G and an elastic potential V_τ. The elastic potential function $V_\tau(\boldsymbol{x})$ is a convex function (Fig. 21.9), increasing strongly enough to compensate for the destabilizing effects of V_G, such that $V(\boldsymbol{x})$ is convex as well. Furthermore, the system contains in general a dissipative friction force $\boldsymbol{d}(\dot{\boldsymbol{x}}, \boldsymbol{x})$ with $\dot{\boldsymbol{x}}^T \boldsymbol{d}(\dot{\boldsymbol{x}}, \boldsymbol{x}) \leq 0$. Note that between the directly and indirectly actuated states, there can exist also inertial couplings, as depicted in Fig. 21.10. Therefore, $\mathbf{M}(\boldsymbol{x})$ is in general a fully coupled matrix.

While at first sight looking pretty general and simple, model (21.6) is actually quite complex due to the full, parallel inertia and stiffness coupling. This can be seen by rewriting it in the less compact form

$$\begin{pmatrix} \mathbf{M}(\boldsymbol{q},\boldsymbol{\theta}) & \mathbf{Q}(\boldsymbol{q},\boldsymbol{\theta}) \\ \mathbf{Q}^T(\boldsymbol{q},\boldsymbol{\theta}) & \mathbf{B}(\boldsymbol{q},\boldsymbol{\theta}) \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{q}} \\ \ddot{\boldsymbol{\theta}} \end{pmatrix} + \boldsymbol{c}(\boldsymbol{q},\boldsymbol{\theta},\dot{\boldsymbol{q}},\dot{\boldsymbol{\theta}}) + \boldsymbol{g}(\boldsymbol{q},\boldsymbol{\theta}) + \begin{pmatrix} \dfrac{\partial V(\boldsymbol{q},\boldsymbol{\theta})^T}{\partial \boldsymbol{q}} \\ \dfrac{\partial V(\boldsymbol{q},\boldsymbol{\theta})^T}{\partial \boldsymbol{\theta}} \end{pmatrix} + \boldsymbol{d}(\dot{\boldsymbol{q}},\dot{\boldsymbol{\theta}}) = \begin{pmatrix} \boldsymbol{\tau}_{\text{ext}} \\ \boldsymbol{\tau}_m \end{pmatrix}. \tag{21.7}$$

The above formulation emphasized the inertial coupling terms $\mathbf{Q}(\boldsymbol{q},\boldsymbol{\theta})$ as well as possible dependencies of the inertial properties on both motor and link configuration.

Although (21.6) is the only formulation general enough to describe most soft robot designs with lumped elasticities encountered in the literature so far, it might pose substantial difficulties in controller design, that is, for the fact that it is not necessarily feedback-linearizable. (See, for example, the full dynamic model of flexible joint robots from [21.83, 84], which is not static state-linearizable.) Therefore, several simplifying assumptions are encountered in the literature, justified by particular robot design choices:

- *A1*: The system has no inertia coupling between the directly and the indirectly actuated stated, but only elastic couplings. This assumption includes the usual neglecting of motor gyroscopic effects for high-gear actuators [21.85] as well the neglection of small inertial couplings usually appearing in the VIA mechanisms [21.86] and imposes some restrictions related to mechanical design (Fig. 21.10 for a counter-example). The assumption leads to a block-diagonal structure of the mass matrix.
- *A2*: The gravity potential energy as well as the variation of the mass matrix is only due to the link-side variables $\boldsymbol{q}$. This implies that also the Coriolis and centrifugal torques are only depending on $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$.
- *A3*: The dynamics of the stiffness adjusting mechanism is negligible; the system dynamics is mainly described by the states $\boldsymbol{q}$ and a directly actuated state vector $\boldsymbol{\theta} \in \mathbb{R}^n$ of the same dimension as $\boldsymbol{q}$, corresponding to one principal motor per joint. The remaining $m-n$ motor coordinates of the stiffness adjusters are treated as parameters and are usually denoted by $\boldsymbol{\sigma}$. In this case, the elastic potential reduces to $V_\tau(\boldsymbol{q}-\boldsymbol{\theta},\boldsymbol{\sigma})$. This is already a quite restricting assumption, applying mainly for SEA-type actuators with a dedicated stiffness adjuster, as typically used in the design of large joints of arms and legs. The model is practically valid only if stiffness is changed off-line or at low rate. Classical antagonistic actuators including two equally sized motors, as used, for example, in some robotic hand designs [21.87] do not fulfill this assumption.
- *A4*: A further particular choice of $V_\tau(\boldsymbol{q}-\boldsymbol{\theta},\boldsymbol{\sigma})$ is that of a quadratic function in $\boldsymbol{q}-\boldsymbol{\theta}$ such that the

joint torque is

$$\boldsymbol{\tau} = \frac{\partial V(\boldsymbol{\phi}, \boldsymbol{\sigma})^{\mathrm{T}}}{\partial \boldsymbol{\phi}} = \mathbf{K}(\boldsymbol{\sigma})\boldsymbol{\phi}\,, \tag{21.8}$$

with $\boldsymbol{\phi} = \boldsymbol{q} - \boldsymbol{\theta}$. This implies that the stiffness of the actuator depends only on the stiffness adjuster and not on the deflection of the springs [21.82]. This assumption introduces additional simplifications in controller design, applies however only to a restricted subclass of soft actuators.

The model, under assumptions *A1*, *A2* reduces to

$$\begin{pmatrix} \mathbf{M}(\boldsymbol{q}) & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{q}} \\ \ddot{\boldsymbol{\theta}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ \mathbf{0} \end{pmatrix} + \begin{pmatrix} \boldsymbol{g}(\boldsymbol{q}) \\ \mathbf{0} \end{pmatrix} + \begin{pmatrix} \dfrac{\partial V(\boldsymbol{q}, \boldsymbol{\theta})^{\mathrm{T}}}{\partial \boldsymbol{q}} \\ \dfrac{\partial V(\boldsymbol{q}, \boldsymbol{\theta})^{\mathrm{T}}}{\partial \boldsymbol{\theta}} \end{pmatrix} + \boldsymbol{d}(\dot{\boldsymbol{x}}) = \begin{pmatrix} \boldsymbol{\tau}_{\mathrm{ext}} \\ \boldsymbol{\tau}_m \end{pmatrix}\,. \tag{21.9}$$

Note that the model above does not make any further specification regarding the dimensions m and n, thus covering a broad range of robots, including as particular cases:

- The classical, fully antagonistic case with $m = 2n$.
- The quite popular $n + 1$ tendon design for robot fingers which leads to $m = n + 1$. More generally, $2n > m > n$ tendon designs are encountered.
- Synergy hands with $m < n$, where each synergistic actuator acts over springs on several links.
- Multi-articular muscles, as mostly existing in biological systems.

Under assumption *A3* the size of $\boldsymbol{\theta}$ reduces to the size of $\boldsymbol{q}(m = n)$ and the model becomes

$$\begin{pmatrix} \mathbf{M}(\boldsymbol{q}) & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{q}} \\ \ddot{\boldsymbol{\theta}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ \mathbf{0} \end{pmatrix} + \begin{pmatrix} \boldsymbol{g}(\boldsymbol{q}) \\ \mathbf{0} \end{pmatrix} + \begin{pmatrix} \boldsymbol{\tau} \\ -\boldsymbol{\tau} \end{pmatrix} + \boldsymbol{d}(\dot{\boldsymbol{x}}) = \begin{pmatrix} \boldsymbol{\tau}_{\mathrm{ext}} \\ \boldsymbol{\tau}_m \end{pmatrix}\,, \tag{21.10}$$

with $\boldsymbol{\tau}^{\mathrm{T}} = \partial V(\boldsymbol{\phi}, \boldsymbol{\sigma})/\partial \boldsymbol{\phi}$. It is then quite close to the flexible joint models treated in detail in Chap. 11. The similarity becomes even stronger, when the torque is described by (21.8) under assumption *A4*. Note that the dimension of $\boldsymbol{\theta}$ in (21.10) is half of the dimension of the same variable in (21.9) since the dynamics of the stiffness adjuster is neglected.

21.5 Stiffness Estimation

Strictly speaking, impedance is a differential operator relating physical quantities (forces and displacements). Rather than a direct measurement, the process of characterizing impedance of a system is therefore a process of *dynamical system identification*.

In typical procedures for impedance measurement of material samples, such as, for example, organic tissue samples, the measured system is put into mechanical contact with another mechanical system, the instrument. The resulting compound dynamical system is then subject to a dynamical excitation of different nature, and data are collected. Experiments are typically conducted by touching the sample with a small, hard, vibrating probe. On the basis of measurements on the probe motion (displacement y, velocity $\dot{y}$ or acceleration $\ddot{y}$) and the applied force f, the mechanical impedance of the sample is inferred. A vast literature is available on the metrology of impedance of materials, and the reader is referred there for a detailed survey [21.88].

The problem of measuring impedance in articulated systems is obviously more complex than for material samples, but is of great relevance to several fields – including notably human movement science and neuroscience, who strive to understand how humans vary and control impedance in their movements. When the hand is slightly perturbed during arm movements, it tends to return to the undisturbed trajectory [21.20, 89], as if the hand would be connected to a spring along the planned trajectory. This spring-like property stems mainly from muscle elasticity and the stretch reflex, which produce – through the complex leverage of muscles at different limb configurations – a restoring force toward the undisturbed trajectory. Methods to characterize the mechanical impedance of human limbs have been the subject of extensive research [21.90–98]. Current protocols for identifying impedance in human motion typically recur to experiments in which perturbations are purposefully injected in the system, and their effects are measured. In artificial robotic systems, impedance parameters can also be calculated on the basis of a precise description of the model (wherever this is available), or obtained through accurate calibration procedures. In both natural and artificial systems, it would be of great utility to have a method which could identify impedance in real time, without per-

turbing the normal execution of the task, and robustly to inaccurate modeling and to time-varying parameters altering calibration results. Of particular interest in robotics is the application of such methods to VIAs for soft robotics. As discussed, these can be implemented in a variety of ways and arrangements, but all designs share a fundamentally unavoidable nonlinear behavior, which makes stiffness measurement a tough problem.

According to the description of impedance as a differential operator, its characterization can be cast as an identification problem of a dynamical system. To date, the literature on this subject is not abundant, and it can be subdivided in two main approaches, that is, parametric and nonparametric identification. Parametric approaches postulate that a model of the actuator generating variable stiffness is known in its structure, but that some of its parameters are unknown. Nonparametric approaches do not use such an assumption, and are thus more general; however, as we will see, nonmodel-based observers have the disadvantage that they cannot *go to sleep* after reaching convergence, and require persistent excitation to stay tuned to variable impedance.

21.5.1 Observers for Linear Impedance

To introduce the problem, let us first consider the case of constant linear impedance identification in the elementary system (21.1)

$$f = m\ddot{y} + b\dot{y} + ky \,,$$

where we assume that m, b, k are unknown but constant, while y and f are available from position and force measurements at one of the ends of the compliant elements.

Before embarking in building an algorithm to estimate impedance, it is important to establish that the problem is well posed, that is, that a solution exists and is unique. To this purpose, consider an extended state vector

$$z = \begin{pmatrix} y & \dot{y} & -\frac{k}{m} & -\frac{b}{m} & \frac{1}{m} \end{pmatrix} \,,$$

and rewrite the dynamics (21.1) as a nonlinear dynamical system

$$\dot{z} = \begin{pmatrix} z_2 \\ z_1 z_3 + z_2 z_4 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ z_5 \\ 0 \\ 0 \\ 0 \end{pmatrix} f, \qquad (21.11)$$

$$y = h(z) = z_1 \,.$$

The identification of the impedance parameters can thus be cast as a nonlinear observability problem, that is, from the knowledge of the input f and output y, estimate the initial state $z(0)$, and in particular its three last components which completely determine the linear impedance. The observability codistribution for this system is

$$\Omega(z) = \operatorname{span} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ z_3 & z_4 & z_1 & z_2 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ z_3 z_4 & z_3 + z_4^2 & z_2 + z_1 z_4 & z_1 z_3 + 2 z_2 z_4 & 0 \\ 0 & 0 & 0 & z_5 & z_4 \end{pmatrix} .$$

It is easy to check that, for m, b, $k > 0$, the observability codistribution has codimension zero for all states, except if $z_1 = z_2 = 0$. Hence, whenever the system is not in stationary equilibrium, the three linear constant impedance parameters can, in principle, be reconstructed from position and force measurements.

To actually estimate the impedance in this case, different methods can be adopted. These include standard off-line identification techniques (which exploit the linear nature of the regressor for the unknown parameters), such as, for example, in [21.99], or on-line nonlinear state observers (e.g., extended Kalman filters) applied to system (21.11) [21.100].

21.5.2 Variable Stiffness Observers

Unfortunately, the generalization of the above straightforward approach to the case when impedance is nonlinear and/or time-varying is not trivial. To convince oneself, it is sufficient to consider the case that, instead of the linear spring term ky, a nonlinear and time-varying spring function $s(y, u(t))$ is introduced in (21.1), that is

$$f = m\ddot{y} + b\dot{y} + s(y, u) \,. \qquad (21.12)$$

In the $s(y, u(t))$ term, u indicates an exogenous input that commands changes in stiffness. For instance, disregarding gravity in Example 21.1 earlier, u would represent the co-activation term $u_\mathrm{b} + u_\mathrm{t}$ in (21.4).

21.5.3 Parametric Estimation

A first approach to identify stiffness can be based on the assumption that the arrangement of the variable stiffness actuator is known (Fig. 21.2), and that a parametric

description of the spring function is available, for example, in terms of a finite basis expansion. In this case, parametric stiffness observers have been proposed that work *on the motor side*, that is, considering the dynamics of each motor as

$$\tau = J\ddot{\theta} + b\dot{\theta} - f(\phi)\,, \tag{21.13}$$

where τ is the torque of the motor, J and b are the motor inertia and damping, respectively, and $\phi = \theta - q$ is the displacement angle with q being the link angle. These parameters and signals being known or measured, the problem is to estimate the stiffness $\sigma(\phi) = \frac{\partial f}{\partial \phi}$, assuming it can be approximated by the truncated Taylor expansion

$$\sigma_i(\phi_i) \approx \sum_{j=0}^{N} \alpha_j^i \frac{(\phi_i)^j}{j!}\,.$$

In [21.101], a parametric stiffness observer is presented to solve this problem. The idea is to first obtain an estimation of the spring force function $f(\phi)$ by combining a modified kinematic Kalman filter for the estimation of the first derivative of q and a first-order filter. Then $f(\phi)$ is approximated by a power series whose coefficients are estimated by a least-squares algorithm. Finally, σ is obtained by analytically differentiating the Taylor expansion. The method has been demonstrated to work well in simulation, and encouraging experimental results have been reported. The method is somewhat sensitive to the choice of several parameters, which require careful tuning. Subsequently, [21.102, 103] proposed algorithms based on differential algebra and modulating functions, respectively, that improved convergence and robustness avoiding taking any derivatives and are very easy to tune.

21.5.4 Nonparametric Estimation

When no information on the structure of f or on the variable $u(t)$ is available, the only possible approach is *on the link side*, that is trying to estimate the joint stiffness looking only at the link dynamics, without assuming any structure for the spring stiffness function. A nonparametric method was proposed in [21.104] to measure stiffness in a system such as (21.12), which can be easily derived from the idea of differentiating (21.12) once with respect to time

$$\dot{f} = m\dddot{y} + b\ddot{y} + \sigma\dot{y} + s_u\dot{u}\,,$$

where

$$s_u := \frac{\partial s(y,u)}{u}\,.$$

The method is based on the continuous update of an estimate $\hat{\sigma}$ of stiffness. Based on a best effort prediction for the link torque $\dot{f}$ as

$$\dot{\hat{f}} = m\dddot{y} + b\ddot{y} + \hat{\sigma}\dot{y}\,,$$

it was shown that, under mild assumptions, the update law

$$\dot{\hat{\sigma}} = \alpha(\dot{f} - \dot{\hat{f}})\mathrm{sgn}(y)\,, \tag{21.14}$$

with $\alpha > 0$ makes $\hat{\sigma}(t)$ converge to the true stiffness value $\sigma(t)$ within a uniformly ultimately bounded error. The error bound is affected by errors in obtaining filtered approximations of derivatives of the position $y(t)$ and of the applied force $f(t)$, and in estimating mass and damping coefficients m and b, respectively. Conversely, the error can be reduced by increasing the gain factor α. As opposed to parametric methods, no assumptions are made on the function $s(y,u)$, except that it is smooth and bounded in both arguments. The method has been experimentally demonstrated to work well, practically converging to the true stiffness value in all circumstances except when stiffness is varied without moving the link. In this case (where it is indeed theoretically impossible to observe stiffness *from the link side*), the method of [21.104] simply stops updating the stiffness estimate, to resume as soon as a sufficiently rich dynamic excitation is again applied to the link.

21.6 Cartesian Stiffness Control

21.6.1 Cartesian Impedance Control

The Cartesian impedance control paradigm is one of the most widely used manipulation control approaches in modern robotics to handle contact situations [21.90, 105, 106]. It combines the advantages of compliant manipulation with the ease and intuitiveness of a task description in Cartesian coordinates. Indeed, this control technique has been one of the motivations for the research of variable-impedance-actuated robots. Therefore, it is a natural aim to analyze and transfer Cartesian impedance control concepts to soft robots.

The goal addressed in this section is to realize a desired Cartesian impedance behavior at the tool center point (TCP) of a variable- impedance- actuated robot. Following a central paradigm of VIA, the mechanical

robot parameters need to be tuned such that the desired behavior is naturally achieved on a mechanism level to the largest extent possible.

With the model from (21.10), the VIA joints shall be adjusted to realize a Cartesian stiffness. The stiffness behavior is described by a constant stiffness matrix $\mathbf{K}_C = \partial \boldsymbol{f} / \partial \boldsymbol{x} \in \mathbb{R}^{m \times m}$ as the relation between the Cartesian wrench f and the Cartesian displacement x. The n passive and adjustable joint stiffness components provide the diagonal matrix $\mathbf{K}_J = \partial \boldsymbol{\tau} / \partial \boldsymbol{q} \in \mathbb{R}^{n \times n}$ with the joint torques τ and the joint positions q. Decoupled joint compliance is assumed. The mapping from the Cartesian stiffness space to the joint stiffness space is given by $\mathcal{T}: \mathbf{K}_J = \mathbf{T}(\mathbf{K}_C)$. This transformation can be written as

$$\begin{aligned}\mathbf{K}_J &= \frac{\partial \boldsymbol{\tau}}{\partial \boldsymbol{q}} = \frac{\partial (\mathbf{J}(\boldsymbol{q})^T \mathbf{K}_C \Delta \boldsymbol{x})}{\partial \boldsymbol{q}} \\ &= \mathbf{J}(\boldsymbol{q})^T \mathbf{K}_C \mathbf{J}(\boldsymbol{q}) - \frac{\partial \mathbf{J}(\boldsymbol{q})^T \mathbf{K}_C \Delta \boldsymbol{x}}{\partial \boldsymbol{q}} ; \end{aligned} \tag{21.15}$$

$\mathbf{J}(\boldsymbol{q}) = \frac{\partial f(\boldsymbol{q})}{\partial \boldsymbol{q}}$ is the manipulator Jacobian, where $\boldsymbol{f}(\boldsymbol{q})$ is the forward kinematics mapping. $\Delta \boldsymbol{x} = \boldsymbol{x}_d - \boldsymbol{x}$ is the Cartesian position error between the desired and the actual position. The first part of (21.15) reflects the stiffness around the equilibrium point. The second part of (21.15) is due to the change of the Jacobian [21.107].

The Cartesian stiffness at the equilibrium position ($\Delta \boldsymbol{x} = 0$), resulting from a specific joint stiffness, can be obtained by solving the inverse problem of (21.15), $\mathbf{K}_C = \mathcal{T}^{-1}(\mathbf{K}_J)$. Note that at the equilibrium position the second term in (21.15) vanishes. Using compliance matrices $\mathbf{C}_\bullet = \mathbf{K}_\bullet^{-1}$, it results from

$$\mathbf{C}_C = \mathbf{J}(\boldsymbol{q}) \mathbf{C}_J \mathbf{J}(\boldsymbol{q})^T \tag{21.16}$$

that

$$\mathbf{K}_C = (\mathbf{J}(\boldsymbol{q}) \mathbf{K}_J^{-1} \mathbf{J}(\boldsymbol{q})^T)^{-1} . \tag{21.17}$$

In [21.108], it was investigated that an arbitrary Cartesian stiffness matrix with three translational and three rotational stiffnesses ($\mathbf{K}_C \in \mathbb{R}^{6 \times 6}$) can hardly be reached by a 7-DOF robot, even exploiting the null space. Bounds are imposed by the technical realization, namely the diagonal joint stiffness matrix or mechanical VSA stiffness limits. Furthermore, the transformation of the joint stiffness into the Cartesian space is strongly determined by the robot kinematics and pose.

To overcome these restrictions and improve the stiffness tracking performance, one can use the passive stiffness in combination with an active impedance controller [21.109], which widely extends the achievable Cartesian stiffness range. Active control adapts the stiffness in a wider range, and the elastic elements are capable of absorbing impacts and increase the energy efficiency. By combining the two concepts, one can benefit from the individual advantages. The serial interconnection of the active stiffness $\mathbf{K}_{active}$ and the passive one $\mathbf{K}_{passive}$ results in an overall stiffness $\mathbf{K}_{res}$ (Fig. 21.11)

$$\mathbf{K}_{res}^{-1} = \mathbf{K}_{active}^{-1} + \mathbf{K}_{passive}^{-1} . \tag{21.18}$$

To compute the active and passive stiffness components, a two-step optimization algorithm can be used [21.109]. It achieves first a passive stiffness as close as possible to the desired one and secondly designs the active stiffness to minimize the residual.

21.6.2 Independent Position and Stiffness Control

Variable stiffness robots are well suited for manipulation tasks involving the interaction with an environment. Such tasks are often specified by a robot pose trajectory and an associated stiffness trajectory. As VSA robots allow us to vary joint position and stiffness in a mechanical way, a direct approach is to adjust the respective position or stiffness inputs. Depending upon the joint setup, these inputs are reflected by distinct actuators [21.66] or can be achieved by a simple coordinating transformation arising from the coupling of actuators [21.69]. The underlying idea is to embed and thereby achieve the desired robot behavior on a mechanical level. Additional control goals, especially important in the presence of disturbances, are the minimization of deflections and the suppression of vibrations.

The solutions to achieve independent control of link motion and stiffness can be classified in two categories:

- One class of controllers exploits the knowledge of system models.
  In [21.110] and [21.111], feedback linearization approaches are used to transform the robot dynamics

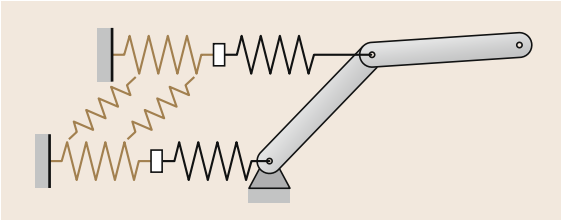


Fig. 21.11 Combination of active impedance controller (*brown* springs) with passive compliance (*black* springs) to achieve a desired Cartesian stiffness

into an equivalent model of simpler form. A decoupled chain of integrators can be achieved, as long as system inversion is possible. The simple structure of the equivalent dynamics allows for simultaneous decoupling and accurate tracking of motion and stiffness reference profiles. However, the abstraction of the robot dynamics hinders the implementation of performance criteria.

Another model-based approach aims to achieve a reduced order model [21.112]. Therefore, separate dynamics are identified in the robot system, namely the arm, the positioning actuators, and the stiffness actuators. The independence of these dynamics is shown by a singular perturbation analysis. A cascaded control structure is based upon this analysis.

The abstraction that this class of controllers provides, allows for theoretical simplicity and design flexibility. On the other hand, robot performance and robustness in the presence of model uncertainty are not guaranteed. High-model accuracy and high derivatives of states are often required. Furthermore, to conform with the idea of embodiment additional effort is necessary, as the controllers often introduce a different robot behavior.

- The second class of controllers are energy-shaping-based controllers.

  One of the first controllers in this category was presented in [21.7]. The controller acts on motor position and uses a transformation to independently control the joint position and stiffness. The controller is validated on a one-DOF VSA joint.

  An extension and a generalization have been presented in [21.113]. The control design formulation is valid for a quite general form of underactuated Euler–Lagrange systems including variable impedance robots. Herein, the controller action can be interpreted as shaping of the potential and kinetic robot energy ensuring system passivity.

  A general task is described as to control k independent output variables given by $\boldsymbol{q} = \boldsymbol{h}(\boldsymbol{x})$ to desired constant values $\boldsymbol{q}_\mathrm{d} \in \mathbb{R}^k$. $\boldsymbol{x} \in \mathbb{R}^n$ is the vector of generalized coordinates, where $n = 2k$ is the usual case for VSA robots. Given the structure of VSA robots (21.10), a new variable $\bar{\boldsymbol{q}}$ can be found. This is a collocated (directly actuated) variable, which is statically equivalent to the noncollocated (indirectly actuated) $\boldsymbol{q}$. Using this collocated variable for a passive feedback ensures stability and can be interpreted as shaping the potential energy of the system.

  The variable $\bar{\boldsymbol{q}}$ is achieved by solving the static solution of the link side equation

$$\boldsymbol{\tau} + \boldsymbol{g}(\boldsymbol{q}) = 0 \tag{21.19}$$

  for q. Except for very simple cases, this equation has to be solved numerically. Due to the convex nature, this is a fast and numerically robust task in practice. Consequently, the controller

$$\boldsymbol{u} = \boldsymbol{g}(\bar{\boldsymbol{q}}) - \mathbf{J}_{\bar{q}}^{\mathrm{T}}(\boldsymbol{\theta})\mathbf{K}_\mathrm{p}(\bar{\boldsymbol{q}} - \boldsymbol{q}_\mathrm{d}) - \mathbf{K}_\mathrm{d}\dot{\boldsymbol{\theta}} \tag{21.20}$$

  stabilizes the desired position $\boldsymbol{q}_\mathrm{d}$; $\boldsymbol{g}(\bar{\boldsymbol{q}})$ is a feed forward term, compensating for the gravity. The use of $\bar{\boldsymbol{q}}$ enables arbitrarily low controller gains even for large displacements from the equilibrium. $\mathbf{J}_{\bar{q}}$ is a Jacobian mapping the collocated variable on the statical equivalent $\bar{\boldsymbol{q}}$. Global asymptotic stability based on LaSalle's invariance theorem can be shown.

  The approach can be extended in order to include also the feedback of torque and torque derivative, with the effect of reducing the apparent actuator inertia and friction. This allows improving the transient performance while remaining within the passivity framework. A torque feedback of the form

$$\boldsymbol{\tau}_m = \mathbf{B}\mathbf{B}_\theta^{-1}\boldsymbol{u} + (\mathbf{I} - \mathbf{B}\mathbf{B}_\theta^{-1})\boldsymbol{\tau}\ , \tag{21.21}$$

  with a new control input u leads to a new subsystem with scaled motor inertia

$$\mathbf{B}_\theta\ddot{\boldsymbol{\theta}} + \boldsymbol{\tau} = \boldsymbol{u}\ . \tag{21.22}$$

  The torque controller can therefore be interpreted as a scaling of the kinetic energy of the rotors in order to reduce the vibrations caused by the joint flexibility. The controlled system allows again a passive representation.

  The energy shaping approach provides excellent performance in the static case and for well damped systems. Some joints show low intrinsic damping to enable joint torque estimation and energy efficiency. In this case it is desirable to add additional damping via control or to include a physical variable damping element. Several damping control structures can be considered. A simple gain scheduling approach for a one-DOF system has been presented in [21.114]. Local linear sub problems are identified on which a linear quadratic regulator (LQR) state feedback controller is designed. The nonlinearity of the robot dynamics requires to adapt the controller poles dependent on the system state, which is especially hard for the multi-joint robots.

  A physically motivated state feedback control approach for multi-DOF VSA robots has been presented in [21.115] using an eigenvalue-based modal decomposition.

21.7 Periodic Motion Control

Robotic tasks, such as walking, jumping, running, and some pick-and-place motions, are essentially periodic motions. During these tasks, the natural oscillatory dynamics of compliant actuators can be ideally exploited to increase performance and energy efficiency. It is well known that fast running animals store and recover a substantial amount of energy during a running gait period in their tendons and muscles. Also human running strongly relies on elastic energy storage.

Exciting periodic motions in a highly nonlinear multibody system by using nonlinear elastic actuators is, however, a nontrivial task. While modal oscillations and their control are very well understood for linear systems, the nonlinear generalization is a very active topic of research. Since we consider that the ability to perform (quasi) periodic motions with high velocity and force at low weight will be one of the major benefits of soft robots, we dedicate an extensive section to this topic.

The topic of exploiting intrinsic elasticity for periodic motions was addressed in robotics by [21.116–120] among others. Two basic approaches can be distinguished:

- Tracking an *arbitrary* desired periodic trajectory, or enforcing an arbitrary oscillatory closed-loop behavior, by control while simultaneously adjusting the compliance of the actuator such that the control effort is minimal.
- Exciting and sustaining *intrinsic* oscillatory dynamics of the compliantly actuated robotic system by control. Completion of a given task implies that this intrinsic oscillation modes correspond to the task by the initial mechanical design and by online adjustment of the robot configuration and of its intrinsic actuator stiffness.

Depending on the design of the compliantly actuated robotic system, the above control approaches have to cope with underactuation, model nonlinearities, and multiple-DOF structures.

21.7.1 Periodic Motion Tracking and Stiffness Adaptation

The idea of tracking a periodic trajectory in the link-side variables $\boldsymbol{q}$ with minimal torque is inspired by classical rigid-robot control. The approach [21.117] considers, for example, robotic systems with elastic elements *in parallel to the main actuators*

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + (\mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \mathbf{D})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \mathbf{K}(\boldsymbol{q}-\boldsymbol{\theta}) = \boldsymbol{\tau}\ . \tag{21.23}$$

Since $\boldsymbol{\theta}$ is assumed to be constant here, the model reduces in this case to a fully actuated robot dynamics with parallel elasticity, being thus simpler than (21.6).

The objectives of the controller are:

1. Tracking a periodic trajectory $\boldsymbol{q}_\mathrm{d}(t+T) = \boldsymbol{q}_\mathrm{d}(t)$ with period $\boldsymbol{T}$ such that $\boldsymbol{q}(t) \to \boldsymbol{q}_\mathrm{d}(t)$
2. Minimizing the actuator torque $\boldsymbol{\tau}$ by optimizing the stiffness $\mathbf{K}$.

Considering the actuator torque required to generate the desired motion

$$\boldsymbol{\tau}_\mathrm{d} = \mathbf{M}(\boldsymbol{q}_\mathrm{d})\ddot{\boldsymbol{q}}_\mathrm{d} + [\mathbf{C}(\boldsymbol{q}_\mathrm{d},\dot{\boldsymbol{q}}_\mathrm{d}) + \mathbf{D}]\,\dot{\boldsymbol{q}}_\mathrm{d} + \boldsymbol{g}(\boldsymbol{q}_\mathrm{d}) + \mathbf{K}(\boldsymbol{q}_\mathrm{d}-\boldsymbol{\theta})\ ,$$

a stiffness matrix $\mathbf{K}_\mathrm{opt}$ is defined to be optimal if it minimizes the cost function

$$\mathbf{J}(\mathbf{K}) = \int_{i\mathrm{T}}^{i\mathrm{T}+\mathrm{T}} \boldsymbol{\tau}_\mathrm{d}(\mathbf{K},t)^\mathrm{T}\boldsymbol{\tau}_\mathrm{d}(\mathbf{K},t)\,\mathrm{d}t\ . \tag{21.24}$$

In the case of the single-DOF system

$$m\ddot{q} + d\dot{q} + kq = \tau\ , \tag{21.25}$$

with constant system parameters m, d, $k>0$, the desired, sinusoidal motion $q_\mathrm{d}(t) = a\sin\omega t$ obviously minimizes the cost function (21.24), when the stiffness is $k_\mathrm{opt} = m\omega^2$. With the optimal stiffness, the sinusoidal motion with an arbitrary amplitude $a>0$ corresponds to a resonance motion, since the angular frequency $\omega = \sqrt{k_\mathrm{opt}/m}$ of the desired trajectory corresponds to the resonance frequency of the system (21.25).

The method proposed in [21.117] extends this simple concept of resonance to the multi-DOF system (21.23). Various online or offline optimization algorithms can be defined to minimize the cost function (21.24). For example, [21.117] uses the parameter adjustment law

$$\dot{\boldsymbol{k}} = \boldsymbol{\Gamma}\,\mathrm{diag}(\boldsymbol{q}-\boldsymbol{\theta})\left[\ddot{\tilde{\boldsymbol{q}}} + \mathbf{K}_\mathrm{B}\boldsymbol{s}(\tilde{\boldsymbol{q}})\right]$$

to adapt the stiffness $\mathbf{K} = \mathrm{diag}(k)$. The controller gain matrix $\mathbf{K}_\mathrm{B}$ is positive definite and diagonal and the adaptation gain $\boldsymbol{\Gamma}$ is a positive definite matrix. The rationale behind this law is to increase the stiffness whenever the elastic deflection $(\boldsymbol{\theta}-\boldsymbol{q})$ has the opposite sign to the tracking error. In this case, a stiffness increase reduces the tracking error. The stiffness is reduced whenever the elastic torque increases the tracking

error. In particular, the control error $\tilde{\boldsymbol{q}} = \boldsymbol{q} - \boldsymbol{q}_\mathrm{d}$ is saturated by functions $s_\mathrm{i}(\tilde{q}_\mathrm{i})$ satisfying certain boundedness constraints. These saturated functions are important for the stability analysis in [21.117].

The paper [21.121] considers dynamic systems with elastic elements either in parallel with the main actuator as in (21.23), or in series as in

$$\boldsymbol{f}(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}, t) = -\mathbf{K}(\boldsymbol{q} - \boldsymbol{\theta}) \tag{21.26}$$

$$\mathbf{B}\ddot{\boldsymbol{\theta}} = \mathbf{K}(\boldsymbol{q} - \boldsymbol{\theta}) + \boldsymbol{\tau}\,, \tag{21.27}$$

where $\mathbf{B}$ represents the constant inertia matrix of the main motor part of the model. The objectives of [21.117] were extended to compute the optimal value of a constant stiffness for minimizing energy consumption in a periodic task for series configuration, and both the optimal stiffness and spring preload values for the parallel case.

Assuming that $\mathbf{K} = \mathrm{diag}(K_\mathrm{j})$ and $\mathbf{B} = \mathrm{diag}(B_\mathrm{j})$ (the second assumption is required in the serial case) it is shown how to derive expressions of the optimal actuation parameters for minimizing energy-related cost functions such as (21.24) and

$$\mathbf{J}(\mathbf{K}) = \int_{\mathrm{iT}}^{\mathrm{iT}+\mathrm{T}} \boldsymbol{w}_\mathrm{d}(\mathbf{K}, t)^\mathrm{T} \boldsymbol{w}_\mathrm{d}(\mathbf{K}, t)\,\mathrm{d}t\,. \tag{21.28}$$

where $\boldsymbol{w}_\mathrm{d}(\mathbf{K}, t) = \boldsymbol{\tau}_\mathrm{d}(\mathbf{K}, t) \cdot^* \dot{\boldsymbol{\theta}}_\mathrm{d}(\mathbf{K}, t)$, with $\cdot^*$ denoting element-wise multiplication. Rather surprisingly, it is shown in [21.121] that it is possible to save lengthy iterations and simulations by computing analytical expressions for the optimal stiffness and preload as integral functions of any given desired joint trajectories: for example, the series stiffness that minimizes the cost (21.24) is given by the formula

$$\mathbf{K}_\mathrm{j}^* = -\frac{\int_{i\mathrm{T}}^{i\mathrm{T}+\mathrm{T}} \mathbf{B}_\mathrm{j}^2 \ddot{f}_\mathrm{j}(\ddot{\boldsymbol{q}}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \boldsymbol{q}_\mathrm{d}, t)\,\mathrm{d}t}{\int_{i\mathrm{T}}^{i\mathrm{T}+\mathrm{T}} \mathbf{B}_j \ddot{f}_j(\ddot{\boldsymbol{q}}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \boldsymbol{q}_\mathrm{d}, t)(\mathbf{B}_j \ddot{q}_{\mathrm{d},j} + f_j(\ddot{\boldsymbol{q}}_\mathrm{d}, \dot{\boldsymbol{q}}_\mathrm{d}, \boldsymbol{q}_\mathrm{d}, t))\,\mathrm{d}t}\,. \tag{21.29}$$

21.7.2 Controlled Oscillatory Dynamics

The basic idea to generate periodic motions is here to control the output of the joint such that it behaves like a desired, oscillatory dynamic system rather than imposing a predefined, time-dependent trajectory [21.118]. The resulting motion will be therefore depending on initial conditions and possible perturbations, but will eventually converge to the periodic orbit. For simplicity, here we describe the concept for the model of a single variable stiffness actuated joint, satisfying

$$m\ddot{q} + \frac{\partial V_\tau(\theta, q)}{\partial q} = 0\,. \tag{21.30}$$

The output position of the joint is denoted $q \in \mathbb{R}$ and the velocities $\dot{\theta} = u \in \mathbb{R}^2$ of the two actuators of the VIA joint are assumed as control inputs. Note that this model corresponds to the first row in (21.9) with $m = 2$ and $n = 1$. Let us assume a generic family of desired oscillatory dynamics

$$\ddot{q} + a(q) = 0\,, \tag{21.31}$$

with $a(q)$ chosen such that a periodic orbit results. In particular, $a(q)$ can be the derivative of a desired potential function $V_\mathrm{d}(q)$. The goal is to control the system (21.30) such that

$$h(\theta, q) = a(q) - \frac{1}{m}\frac{\partial V_\tau(\theta, q)}{\partial q} = 0 \tag{21.32}$$

holds. With initial conditions $(q(0), \dot{q}(0), \theta_1(0), \theta_2(0)) \in h(0)^{-1}$ the constraint (21.32) can be fulfilled, if the control u satisfies the constraint on velocity level, that is, $\dot{h}(\theta, q) = 0$. Since $\dot{\theta} = u$ is the control input, this implies

$$\dot{h}(\theta, q) = -\mathbf{A}(\theta, q)u + b(\theta, q, \dot{q}) = 0\,, \tag{21.33}$$

with

$$\mathbf{A}(\theta, q) := \frac{1}{m}\frac{\partial^2 V_\tau(\theta, q)}{\partial q \partial \theta}\,, \tag{21.34}$$

$$b(\theta, q, \dot{q}) := \left(\frac{\partial a(q)}{\partial q} - \frac{1}{m}\frac{\partial^2 V_\tau(\theta, q)}{\partial q^2}\right)\dot{q}\,. \tag{21.35}$$

Using a generalized right inverse $\mathbf{A}^\dagger$ of the matrix $\mathbf{A}$, the control

$$u = \mathbf{A}(\theta, q)^\dagger b(\theta, q, \dot{q})\,, \tag{21.36}$$

fulfills the velocity constraint (21.33).

The solution of (21.33) with regard to u is not unique, as the problem is under-constrained. By choosing a particular generalized pseudoinverse of $\mathbf{A}$ of the form $\mathbf{A}^\dagger = \mathbf{B}^{-1}\mathbf{A}^\mathrm{T}(\mathbf{A}\mathbf{B}^{-1}\mathbf{A}^\mathrm{T})^{-1}$, the solution (21.36) minimizes the norm $\|u\|_\mathrm{B}^2$ with respect to a certain metric tensor $\mathbf{B}$. If, for example, $\mathbf{B}$ is the matrix containing the motor inertias then the kinetic energy of the actuators is minimized. The under-constrained situation can be interpreted such that the desired motion might be achieved by both adjusting the stiffness and the motor position of the actuators. Choosing a certain metric $\mathbf{B}$ implicitly defines a weighting of the two components.

21.7.3 Exciting Intrinsic Oscillatory Dynamics

Exciting the intrinsic oscillatory behavior of mechanical systems, that consist of kinetic and potential energy storages, is inspired by the concept of resonance from linear oscillation theory. The idea is to move the actuators of the compliant system such that the trajectory of link variables is periodic and thereby the energy input due to the motor motion ideally counteracts only the loss of dissipated energy. Instead of tracking an arbitrary trajectory, here we are interested in exciting intrinsic resonant motions of the system.

Consider the dynamical system of the form (21.10). The objectives in this section are finding a control $\boldsymbol{\tau}_m$ such that for a certain period T, the trajectories of link variables fulfill $\boldsymbol{q}(t+T) = \boldsymbol{q}(t)$, and the intrinsic oscillatory dynamics are exploited to the maximally possible extent. Assuming that the motion $\boldsymbol{q}(t)$ should take place on one of several one-dimensional manifolds $\mathcal{Z}_j \subset \mathbb{R}^n$ (referred to as oscillation modes), the problem can be further refined:

- *C1*: Given the mapping $\boldsymbol{z} = \boldsymbol{z}(\boldsymbol{q})$. If z_j represents the coordinate of the desired oscillation mode, then the control must ensure that oscillations excited in all other modes decay, that is, for $i \neq j$, $\dot{z}_i \to 0$.
- *C2*: The motion of z_j has to be controlled to sustain a periodic motion with a desired oscillation amplitude.

21.7.4 Model-Based Approach

To derive a coordinate transformation $\boldsymbol{z} = \boldsymbol{z}(\boldsymbol{q})$ and to fulfill the above conditions, the free motion ($\boldsymbol{\tau}_{\text{ext}} = \boldsymbol{0}$) link-side dynamics of (21.10) can be modified by first considering $\boldsymbol{\theta}$ as control input, that is,

$$\boldsymbol{\theta}_{\text{d}} = \boldsymbol{q}^{\star} + \mathbf{K}_{\text{P}}^{-1}(\mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) - \mathbf{K}_{\text{D}}\dot{\boldsymbol{q}} + \boldsymbol{\gamma}) \,, \tag{21.37}$$

with $\boldsymbol{\gamma}$ being a decoupling term to be specified later. This leads to the closed-loop dynamics

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{K}_{\text{P}}(\boldsymbol{q} - \boldsymbol{q}^{\star}) + \mathbf{K}_{\text{D}}\dot{\boldsymbol{q}} = 0 \,. \tag{21.38}$$

In order to excite oscillations which are quite close to the initial oscillation modes, consider coordinates $\boldsymbol{z}_{\text{i}}$ which correspond to the instantaneous modal coordinates $\boldsymbol{z} = \mathbf{Q}(\boldsymbol{q})^{-1}(\boldsymbol{q} - \boldsymbol{q}^{\star})$. $\mathbf{Q}(\boldsymbol{q})$ is defined on the basis of double diagonalization of $\mathbf{M}$ and $\mathbf{K}$ such that $\mathbf{Q}(\boldsymbol{q})^{-\text{T}}\mathbf{Q}(\boldsymbol{q})^{-1} = \mathbf{M}(\boldsymbol{q})$ and $\mathbf{Q}(\boldsymbol{q})^{-\text{T}}\text{diag}(\lambda(\boldsymbol{q}))\mathbf{Q}(\boldsymbol{q})^{-1} = \mathbf{K}_{\text{P}}$, where $\boldsymbol{\lambda}(\boldsymbol{q}) \in \mathbb{R}^n_{>0}$ are generalized eigenvalues, as proposed in [21.120]. By considering a damping design

$$\mathbf{K}_{\text{D}} = 2\mathbf{Q}(\boldsymbol{q})^{-\text{T}}\text{diag}\left[\xi_{\text{i}}\sqrt{\lambda_{\text{i}}(\boldsymbol{q})}\right]\mathbf{Q}(\boldsymbol{q})^{-1} \,,$$

and by choosing γ as

$$\boldsymbol{\gamma} = [\mathbf{M}(\boldsymbol{q})\ddot{\mathbf{Q}}(\boldsymbol{q}) + K_{\text{D}}\dot{\mathbf{Q}}(\boldsymbol{q})]\boldsymbol{z} + 2\mathbf{M}(\boldsymbol{q})\dot{\mathbf{Q}}(\boldsymbol{q})\dot{\boldsymbol{z}} \,, \tag{21.39}$$

the resulting closed-loop dynamics

$$\ddot{z}_i + 2\xi_i\sqrt{\lambda_i(\boldsymbol{q})}\dot{z}_i + \lambda_i(\boldsymbol{q})z_i = 0$$

fulfills the condition *C1*, if for $i \neq j$, $\xi_i > 0$. Then, the nonlinear damping term

$$\xi_j = -\frac{\dot{\lambda}_j(\boldsymbol{q})}{4\lambda_j(\boldsymbol{q})^{\frac{3}{2}}} + \frac{1}{2}\sqrt{\lambda_j(\boldsymbol{q})}k_{\text{H}}[H(\boldsymbol{q}, z_j, \dot{z}_j) - H_{\text{d}}]$$

produces a periodic motion $z_{\text{j}}(t+T) = z_{\text{j}}(t)$. In particular, the resulting limit cycle corresponds to a level H_{d} of the energy function $H(\boldsymbol{q}, z_j, \dot{z}_j) = \frac{1}{2}(\frac{1}{\lambda_j(\boldsymbol{q})}\dot{z}_j^2 + z_j^2)$ as proposed in [21.122].

Note that applying the control (21.37) to the system (21.10) requires a controller that ensures a tracking $\boldsymbol{\theta}(t) \to \boldsymbol{\theta}_{\text{d}}(t)$ of the desired motor position $\boldsymbol{\theta}_{\text{d}}(t)$.

21.7.5 Model-Free Approach

Exciting oscillations in multi-DOF mechanical systems without changing at all the original inertia and stiffness properties of the plant requires either an ideally separated excitation on a certain oscillation mode or intrinsic physical damping properties that ensure the decay of oscillations excited in the other modes cf. condition *C1*. Note that in most cases fully decoupled modes do not even exist. On the basis of the latter assumption, which is valid in biological systems and some robotic examples [21.123], the control law

$$\Delta\boldsymbol{\theta}_z = \begin{cases} \text{sign}(\tau_z)\hat{\theta}_z & \text{if } |\tau_z| > \epsilon_{\tau_z} \\ 0 & \text{otherwise} \end{cases} \tag{21.40}$$

excites a modal oscillation in a simple way. Thereby, the linear transformation $z_{\text{j}} = \boldsymbol{w}^{\text{T}}\boldsymbol{q}$ approximates the nonlinear mapping φ: $\mathbb{R}^n \to \mathcal{Z}_j$, such that the generalized modal force can be transformed by the vector of weights $\boldsymbol{w} \in \mathbb{R}^{\text{n}}$, that is,

$$\tau_z = \boldsymbol{w}^{\text{T}}(\boldsymbol{\psi}(\boldsymbol{\theta} - \boldsymbol{q}) - \boldsymbol{\psi}(\boldsymbol{\theta}^{\star} - \boldsymbol{q}^{\star})) \,, \tag{21.41}$$

where $\boldsymbol{\theta}^{\star}$ and $\boldsymbol{q}^{\star}$ satisfy the equilibrium conditions $\boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{\psi}(\boldsymbol{q} - \boldsymbol{\theta}) = 0$. When the modal force τ_{z} exceeds a certain threshold ϵ_{τ_z}, the controller switches by an

amplitude $\hat{\boldsymbol{\theta}}_z$. This results in steps of desired motor positions

$$\boldsymbol{\theta}_d = \boldsymbol{\theta}^\star + \boldsymbol{w}\Delta\boldsymbol{\theta}_z \,.$$

Since the motor positions θ are not the control input of the system (21.10), a motor PD controller should be considered to ensure that $\boldsymbol{\theta}(t) \to \boldsymbol{\theta}_d(t)$.

Note that the oscillation controller requires neither derivatives of measured signals nor model knowledge of the plant. Therefore it is simple and robust, and can be easily applied to different types of compliantly actuated systems. The control law is, at least for the one-DOF case, the optimal solution for maximizing the oscillation amplitude in minimum time. Furthermore, the control law defined by (21.40) and (21.41) is plausible to have a biological equivalent, since it strongly resembles the computation function of neurons.

21.8 Optimal Control of Soft Robots

Robot joints with intrinsic elasticity have some major advantages over their rigid counterparts: they are more robust, versatile, and can store energy. These properties enable robots to realize motions which would not be possible using stiff actuators of the same size. However, the question remains open as to how elastic elements in soft robots can be best controlled to exploit their potential. Optimal control (OC) theory is being regarded as one of the most promising tools to clarify such questions: as far as an appropriate choice of the cost functional is available, optimal control techniques can provide design guidelines for both the physical design and the control of soft robots.

There are at least two main reasons why OC is fundamental to study soft actuators. On one side OC provides an absolute performance reference that factorizes the control design out of the equation, hence it furnishes a principled basis to compare the performance of different system designs. On the other side, OC is a key element in understanding planning and control methodologies for soft actuators. A careful analysis of OC results obtained through either analytic or numerical techniques, allows to distillate laws summarizing control policies that can be applied to classes of tasks.

General analytic solutions could be developed so far only for one-DOF joints. Physical bounds on states and inputs, the strong nonlinear structure of the robots and the high number of DOF allow only numerical solutions for general robotic systems. Nevertheless, choosing an advantageous parametrization for numerical solutions of the OC problem strongly benefits from insights given by the low-dimensional analytical solutions. Therefore, the analytical concepts are explained for some basic examples in this section.

21.8.1 OC Theory

We consider the dynamics of a system described in standard form ($\dot{x} = f(u, x, t)$), with the cost functional J consisting of a terminal cost ϑ and an integral cost L

$$J(u) = \vartheta(x(t_\mathrm{f}), t_\mathrm{f}) + \int_{t_0}^{t_\mathrm{f}} L(x, u, t)\,\mathrm{d}t \,. \tag{21.42}$$

According to Pontryagin's minimum principle [21.124], the optimal control u^* is known to minimize the Hamiltonian function $\mathcal{H} = \lambda^\mathrm{T} f(u, x, t) + L$, along the optimal trajectory, where λ are the costates. For an OC problem with constraints on only the control $u \in \mathbb{U}$, differential equations of these costates are given by $\dot{\lambda} = -\partial\mathbb{H}/\partial x$. Furthermore, if the final state is not explicitly stated, Minimum Principle also provides boundary conditions for $\boldsymbol{\lambda}$ at the final time

$$\lambda(t_\mathrm{f}) = \left.\frac{\partial\vartheta}{\partial x}\right|_{t_\mathrm{f}} \,.$$

In order to make use of the Minimum Principle, one has to first solve these differential equations for x and λ and then find the control u^*, which minimizes the Hamiltonian: $\mathbb{H}(x^*, \lambda^*, u^*) \leq \mathbb{H}(x^*, \lambda^*, u)$ for all $u \in \mathbb{U}$, where x^* and λ^* denote the optimal states and costates.

21.8.2 SEA Optimization

Naturally, the performance achievable with a given soft actuator under optimal control depends on the physics of the actuator itself. It is therefore very interesting to study the nested optimization problem of how to design soft robotics actuators which can produce the best performance under optimal control.

From the analysis of a one-DOF SEA joint, it can be observed [21.17] that there exists an optimal value of the linear stiffness that maximizes the (optimally controllable) peak speed. The optimal stiffness value depends on the motor (e.g., maximum acceleration and speed) and on the task (e.g., terminal time). This result is illustrated in Fig. 21.12.

21.8.3 Safe Brachistochrone OC Problem

One of the first instances of use of OC for soft (variable stiffness) robots is the study of the so-called *safe brachistochrone* problem [21.125]. The goal was to find control laws for the equilibrium position and variables transmission stiffness such that a point-to-point link motion is achieved in minimum time subject to a limit on the worst-case impact force that the link could exchange with the environment during the trajectory. Limiting the impact force during motion can be useful both for increasing the safety of a robot operating in proximity of humans, and for reducing the risk of damaging the robot itself in accidental collisions with a stiff environment. The main results were to show that VSAs can perform better (that is, faster within bounds on a measure of impact risk) than both conventional (rigid) actuators and SEAs, and that the optimal stiffness law is inversely related to the link velocity, in a fashion roughly described by the easy-to-remember rule *fast and soft, stiff and slow*.

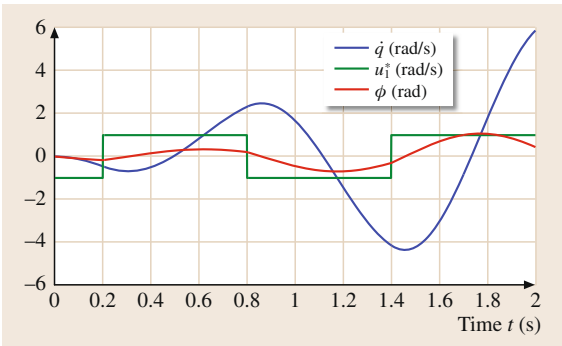


Fig. 21.12 Optimal motor velocity ($\omega = 5\,\mathrm{rad/s}$, $|u_{1\,\min}| = u_{1\,\max} = 1\,\mathrm{rad/s}$)

21.8.4 Max Speed OC Problem

A complementary example of application of OC theory concerns the problem of maximizing the velocity of an elastic robot joint at some desired final time or position. This goal is relevant to impulsive tasks, such as jumping on feet, hitting a ball or hammering a nail. Conversely to the previous example, in such tasks it might be desirable that impact is maximized: indeed, it turns out that elastic actuators are neither intrinsically more or less safe than rigid joints, but can be both, under suitable control laws.

The differential equations for the states and costates of this OC problem are summarized in Table 21.1. Note that the functional J to be minimized consists merely of a terminal cost $J(u_1) = -\dot{q}(t_\mathrm{f})$. For this particular problem, studied in [21.126, 127], the condition of minimizing the Hamiltonian $\mathbb{H}$ on the optimal trajectory becomes $u_1^*\lambda_1^* \le u_1\lambda_1^*$, for all $u_1 \in [u_{1\,\min}, u_{1\,\max}]$. Consequently, u_1^* will have its minimum or maximum value depending on the sign of the first costate λ_1^*. Using Table 21.1, this first costate can be obtained as $\lambda_1^* = -\omega \sin[\omega(t_\mathrm{f} - t)]$, where ω denotes the natural frequency of the system. The optimal control u_1^* will thus be periodic with the eigenfrequency of the system ω and take its maximum (minimum) value, whenever $\sin[\omega(t_\mathrm{f} - t)]$ is positive (negative). Note that the switching of the optimal control occurs with the eigenfrequency starting from the final time backward. For the particular case

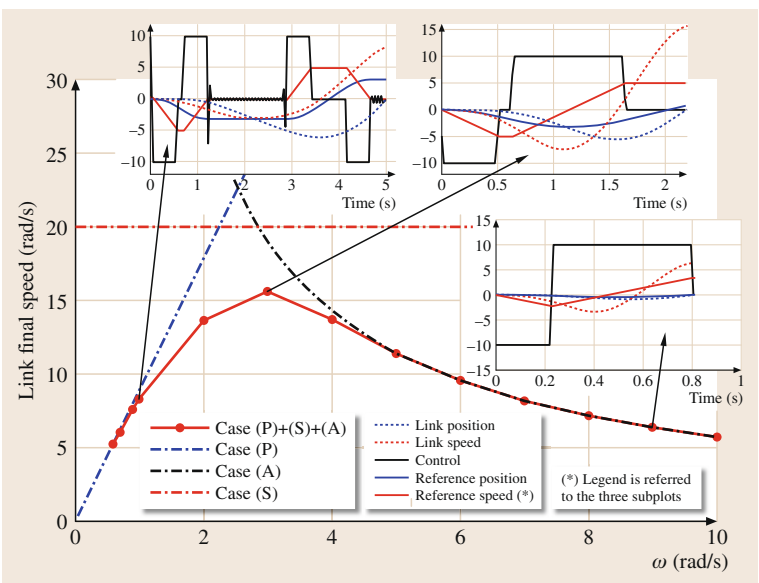


Fig. 21.13 Diagram (after [21.17]) representing the maximum speed reached by a one-DOF SEA after a single swing at varying the spring stiffness ($\omega = \sqrt{k/m}$). *Dashed curves* show terminal speed $\dot{q}(T)$ under ideal motor position control (case P), speed control (case S) and acceleration control (case A). *The continuous red curve* shows terminal speed in the realistic condition of motor acceleration control ($|\ddot{\theta}(t)| \le 10\,\mathrm{rad\,s^{-2}}$), with bounds on position ($|\theta(t)| \le \pi$ rad) and velocity ($|\dot{\theta}(t)| \le 5\,\mathrm{rad\,s^{-1}}$). The *small charts* show optimal control and state patterns in time

Table 21.1 OC problem for the elastic joint ($\omega = \sqrt{\frac{k}{m}}$)

Excitation	States x	Cost J	$\dot{x} = f(x, u_1)$	$\dot{\lambda} = -\frac{\partial \mathbb{H}}{\partial x}$	$\lambda(t_f)$
$u_1 = \dot{\theta}$	$\begin{pmatrix}\phi\\ \dot{q}\end{pmatrix}$	$-x_2(t_f)$	$\begin{pmatrix}u_1 - x_2\\ \omega^2 x_1\end{pmatrix}$	$\begin{pmatrix}-\omega^2\lambda_2\\ \lambda_1\end{pmatrix}$	$\frac{\partial \vartheta}{\partial x}\|_{t_f} = \begin{pmatrix}0\\ -1\end{pmatrix}$

when the final time is a multiple of the oscillation period $t_f = nT = 2\pi n/\omega$, switching can be equivalently done in feed-forward after every half period. For an elastic joint starting from the rest position, this particular control law states then to switch the motor velocity when the angular deflection ϕ or similarly the acceleration of the system changes its sign

$$\dot{\theta}^* = \text{sign}(\ddot{q}) u_{1\,\max} \,. \tag{21.43}$$

If an additional quadratic integral cost in the control, $L = cu_1^2$ with a positive scalar c, is considered in (21.42), a continuous control law will follow. For $t_f = nT$ a classical resonance excitation with a sinusoidal input then follows, corresponding to intuition and knowledge of linear systems. It is possible to extend these analytical solutions for nonlinear stiffness characteristics and damped systems [21.127]. For example, for an underdamped robotic link a similar optimal control is found, which is periodic with the damped eigenfrequency. Different motor models can also be investigated analytically.

21.8.5 VSA Optimization

The very fact that for different tasks the optimal value of stiffness is different implies that the adaptability of the physical stiffness of actuators can be very useful in improving performance over a set of tasks.

One further interesting question is raised by the possibility offered by VSA technologies to vary the actuator stiffness in time. The problem becomes that of finding the optimal time course of both the actuator position $\theta(t)$ and stiffness $k(t)$ so as to maximize performance. This problem can be again investigated using OC theory [21.17, 128]. Without going in the details of OC computations, results show that under the simplifying assumption that the actuator position and stiffness are bounded but can be changed instantaneously, the optimal control for a single swing and free terminal time (within this swing) is bang–bang: the optimal reference position is

$$\theta^* = \text{sign}(\dot{q})\theta_{\max} \,,$$

while the optimal stiffness $k^*(t)$ takes its minimum value $k_{\min}$ or maximum value $k_{\max}$ depending on the switching function

$$k^* = \begin{cases} k_{\min} & \text{if } \dot{q}\ddot{q} < 0 \\ k_{\max} & \text{if } \dot{q}\ddot{q} > 0 \end{cases} , \tag{21.44}$$

while for $\dot{q}\ddot{q} = 0$ the stiffness value can be set arbitrarily. This indicates the rather intuitive notion between link velocity and accelerations summarized by

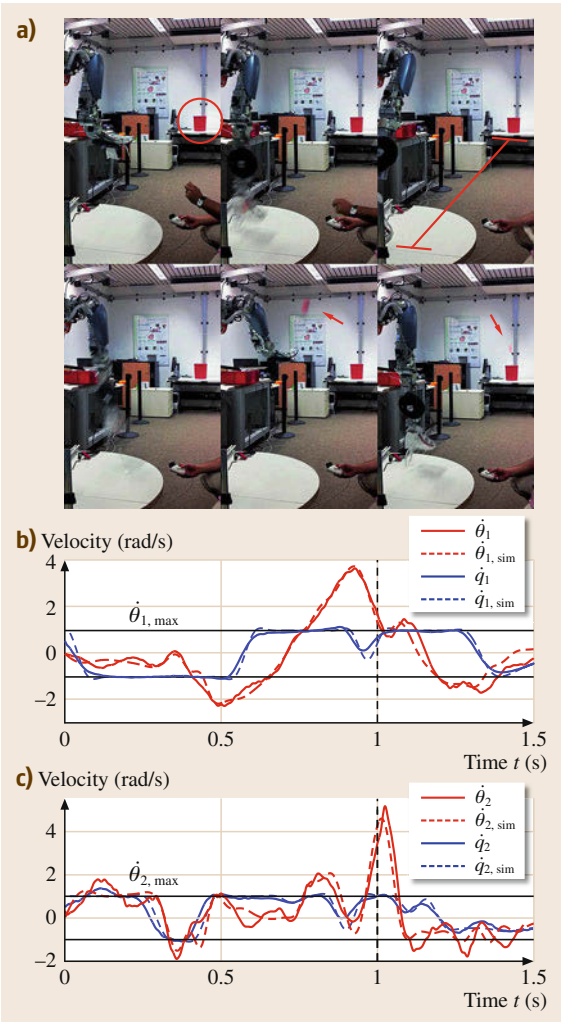


Fig.21.14a–c Throwing experiment with the DLR hand-arm system (after [21.87]). **(a)** Experimental setup, **(b)** velocity of joint 1, **(c)** velocity of joint 2

the rule *stiff speed-up, soft slow-down*. More general cases of VSA optimization are discussed in [21.17, 128] and [21.126]. It is important to notice that even for one-link robots analytical solutions are not always straightforward to find, especially if state constraints exist or system properties have nonlinearities.

Under some conditions, it may be possible to apply efficient numerical solutions to such problems. For instance, if the soft actuator dynamics system are linear (as e.g., in the SEA case) but there exist state and control constraints (e.g., the torque-speed characteristic of a dc motor), it is possible to translate the optimal control problem into a convex optimization problem [21.129], for which there exist fast and reliable methods to obtain the global solution. For robots with multiple-links, differential equations resulting from OC theory are nonlinear and cannot in general be solved analytically. Consequently, numerical methods such as pseudospectral methods [21.130] or iterative linear quadratic regulator (ILQR) methods [21.131, 132] are employed for these systems. One example of how to use these methods to outperform rigid robots is described in [21.133], where a throwing motion is optimized using the gauss pseudospectral method [21.134]. The reported experiments verify the system's ability to use the elastic energy for explosive motions. Figure 21.14a illustrates the experimental setup, while the motor and link velocity are depicted on Fig. 21.14b,c. Note that the link velocities $\dot{q}$ reach more than double the maximum motor velocities $\dot{\theta}_{\max}$ during the motion, which would be clearly not possible with stiff actuators.

21.9 Conclusions and Open Problems

In this chapter, we have reviewed some of the technologies and methods for analysis and control underpinning the use of novel actuators for soft robotics. The field has rapidly expanded in recent years, and it is very difficult to provide an exhaustive up-to-date account of all innovations being continuously brought forward: hopefully, this chapter provides at least a faithful picture of some of the most important current research trends in soft actuation.

When comparing current robots with biological systems in terms of actuator energy density, energy efficiency of motion, resilience, and sensitivity during interaction, the importance of intrinsic compliance becomes striking. We are still far from having found so far the optimal technical realization of compliant, muscle-like actuators regarding design simplicity, integration, and energy density. Nevertheless, in this chapter foundations have been described both by properly defining basic notions such as nonlinear impedance and addressing parametric identification, and in analyzing and standardizing the main physical properties to be used for the description of elastic actuators. A widely agreed set of characteristic parameters summarized in a standardized data sheet format [21.75–77] help comparing the very heterogeneous design approaches and choosing the best concept for a given application. We proposed a quite general dynamic model for flexible robots with lumped elasticities, covering all designs proposed to our knowledge so far. The model is obviously related to the models of flexible robots from Chap. 11 and can be regarded as a generalization thereof. The control of flexible robots was focused on two main challenges. First, the simultaneous control of the position and intrinsic impedance of the devices has to be solved. This is a challenge for systems with highly nonlinear and very compliant behavior. This aspect was addressed on joint and the Cartesian level.

Second, control to push flexible robots to their limit and maximally exploit their potential benefits. If robustness comes mainly by the design process, that is not true for energy efficiency and performances maximization (even under safety constraints). The achievement of these features requires optimization and optimal control algorithms, such as those exposed in the present chapter. Moreover, OC algorithms can be used as a tool for comparison among different actuator designs extracting their inherent (i. e., control-independent) performance boundaries, and to derive easy-to-understand control rules for peculiar DOFs (e.g., stiffness) of soft robots.

Important new directions are open in front of the research community. On one side, new active materials could provide a boost to the development in soft robotics actuators, especially in the small scale; on the other hand, it can be expected that some of the promising advanced material research in the past will eventually make their way in practically viable technologies. New functional architectures for actuators that can adapt their physical behavior to the task are also potentially very fruitful. An important future trend is the extension from *variable stiffness* to VIAs, capable of more closely mimicking the functionalities of muscular structures where not just the elasticity, but also the damping (and perhaps the effective inertia) can be physically tuned. Much work remains also to be done in the understanding of the theoretical underpinnings and in practical methods for the identification and

control of nonlinear, time varying, multidimensional impedance.

The very fact that research in novel actuators for soft robotics has produced a wealth of ideas and solutions poses a challenge of different nature. Indeed, a huge potential for innovation is already available to researchers and practitioners, which has been only superficially tapped so far in applications. Soft actuators could change the landscape not only in robust and safe interaction or in locomotion: there are probably many, more specific application niches where the new technologies could have an impact – possibly also outside the scope of *robotics* narrowly defined, in the larger domain if *automation*. To foster technological awareness of soft robotics potential in the community of those interested in machines that can move and interact more naturally, several initiatives are being undertaken, including an IEEE technical committee on soft robotics [21.135] and several workshops, special issues and even specialized journals on the topic. Recently, a natural machine motion initiative (NMMI) [21.136] has been launched with the aim of diffusing the ideas of soft actuation to achieve *naturally moving* robots with human-like grace, power, and dexterity. One of the objectives of the NMMI is to make the technology of soft actuators widely available to the general public, by publishing as open-source the design and all instructions needed to build and use low-cost actuators, thus building a community of developers interested in sharing advances toward simple, affordable, yet smooth, strong, and accurate muscles for robots.

Video-References

VIDEO 456 Variable impedance actuators: Moving the robots of tomorrow
available from http://handbookofrobotics.org/view-chapter/21/videodetails/456

VIDEO 457 Petman tests camo
available from http://handbookofrobotics.org/view-chapter/21/videodetails/457

VIDEO 458 Introducing WildCat
available from http://handbookofrobotics.org/view-chapter/21/videodetails/458

VIDEO 460 VSA CubeBot – peg in hole
available from http://handbookofrobotics.org/view-chapter/21/videodetails/460

VIDEO 461 DLR Hand Arm System smashed with baseball bat
available from http://handbookofrobotics.org/view-chapter/21/videodetails/461

VIDEO 463 Safety evaluation of lightweight robots
available from http://handbookofrobotics.org/view-chapter/21/videodetails/463

VIDEO 464 Hammering task with the DLR Hand Arm System
available from http://handbookofrobotics.org/view-chapter/21/videodetails/464

VIDEO 465 Dynamic walking of whole-body compliant humanoid COMAN
available from http://handbookofrobotics.org/view-chapter/21/videodetails/465

VIDEO 466 Dynamic walking of whole-body compliant humanoid COMAN
available from http://handbookofrobotics.org/view-chapter/21/videodetails/466

VIDEO 467 Maccepa System
available from http://handbookofrobotics.org/view-chapter/21/videodetails/467

VIDEO 468 AMASC – changing stiffness
available from http://handbookofrobotics.org/view-chapter/21/videodetails/468

VIDEO 470 VSA-Cube: Arm with high and low stiffness preset
available from http://handbookofrobotics.org/view-chapter/21/videodetails/470

VIDEO 471 CompAct™ robotics technology
available from http://handbookofrobotics.org/view-chapter/21/videodetails/471

VIDEO 472 VSA-Cube arm: Drawing on a wavy surface (high stiffness)
available from http://handbookofrobotics.org/view-chapter/21/videodetails/472

VIDEO 473 Arm drawing on a wavy surface (low stiffness)
available from http://handbookofrobotics.org/view-chapter/21/videodetails/473

VIDEO 474 Arm drawing on a wavy surface (selective stiffness)
available from http://handbookofrobotics.org/view-chapter/21/videodetails/474

VIDEO 475 Intrinsically elastic robots: The key to human like performance (Best Video Award)
available from http://handbookofrobotics.org/view-chapter/21/videodetails/475

VIDEO 546 DLR Hand Arm System: Punching holes
available from http://handbookofrobotics.org/view-chapter/21/videodetails/546

VIDEO 547 DLR Hand Arm System throwing a ball and Justin catching it
available from http://handbookofrobotics.org/view-chapter/21/videodetails/547

VIDEO 548 Active damping control on the DLR Hand Arm System
available from http://handbookofrobotics.org/view-chapter/21/videodetails/548
VIDEO 549 Throwing a ball with the DLR VS-Joint
available from http://handbookofrobotics.org/view-chapter/21/videodetails/549
VIDEO 550 DLR Hand Arm System: Two arm manipulation
available from http://handbookofrobotics.org/view-chapter/21/videodetails/550
VIDEO 698 Full body compliant humanoid COMAN
available from http://handbookofrobotics.org/view-chapter/21/videodetails/698
VIDEO 699 AWAS-II
available from http://handbookofrobotics.org/view-chapter/21/videodetails/699

References

21.1 K.T. Ulrich, T.T. Tuttle, J.P. Donoghue, W.T. Townsend: *Intrinsically Safer Robots*, *Tech. Rep.* (Barrett Technology Inc., Cambridge 1995), Final Report for NASA Contract NAS10-12178

21.2 B. Lacevic, P. Rocco: Safety-oriented control of robotic manipulators – A kinematic approach, Proc. 18th World Congr. Int. Fed. Autom. Control (IFAC) (2011)

21.3 S. Haddadin, S. Haddadin, A. Khoury, T. Rokahr, S. Parusel, R. Burgkart, A. Bicchi, A. Albu-Schaeffer: On making robots understand safety: Embedding injury knowledge into control, Int. J. Robotics Res. **31**, 1578–1602 (2012)

21.4 K. Salisbury, W. Townsend, B. Eberman, D. DiPietro: Preliminary design of a whole-arm manipulation system (WAMS), Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Philadelphia (1988) pp. 254–260

21.5 G. Hirzinger, A. Albu-Schäffer, M. Hähnle, I. Schaefer, N. Sporer: On a new generation of torque controlled light-weight robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 3356–3363

21.6 K. Yokoi, F. Kanehiro, K. Kaneko, S. Kajita, K. Fujiwara, H. Hirukawa: Experimental study of humanoid robot HRP-1S, Int. J. Robotics Res. **23**(4/5), 351–362 (2004)

21.7 K. Kaneko, F. Kanehiro, M. Morisawa, K. Miura, S. Nakaoka, S. Kajita: Cybernetic human HRP-4C, Proc. IEEE/RAS Int. Conf. Humanoid Robots (2009) pp. 7–14

21.8 Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, K. Fujimura: The intelligent asimo: System overview and integration, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 3 (2002) pp. 2478–2483

21.9 M. Hirose, K. Ogawa: Honda humanoid robots development, Phil. Trans. Roy. Soc. A **365**(1850), 11–19 (2007)

21.10 B. Vanderborght, B. Verrelst, R. Van Ham, M. Van Damme, D. Lefeber: A pneumatic biped: Experimental walking results and compliance adaptation experiments, Proc. Int. Conf. Humanoid Robots, Tsukuba (2006)

21.11 T. Takuma, K. Hosoda, M. Ogino, M. Asada: Stabilization of quasi-passive pneumatic muscle walker, Proc. IEEE/RAS Int. Conf. Humanoid Robots, Vol. 2 (2004) pp. 627–639

21.12 S.H. Collins, A. Ruina: A bipedal walking robot with efficient and human-like gait, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1983–1988

21.13 N.G. Tsagarakis, S. Morfey, G. Medrano-Cerda, Z. Li, D.G. Caldwell: Compliant humanoid coman: Optimal joint stiffness tuning for modal frequency control, Proc. IEEE Int. Conf. Robotics Autom. (2013) pp. 665–670

21.14 J.W. Hurst, J.E. Chestnutt, A.A. Rizzi: *An Actuator with Mechanically Adjustable Series Compliance*, CMU-RI-TR-04-24 (Robotics Inst./Carnegie Mellon University, Pittsburgh 2004)

21.15 L.C. Visser, S. Stramigioli, R. Carloni: Robust bipedal walking with variable leg stiffness, Proc. 4th IEEE/RAS/EMBS Int. Conf. Biomed. Robotics Biomechatron. (BioRob) (2012) pp. 1626–1631

21.16 S. Haddadin, M.C. Ozparpucu, A. Albu-Schäffer: Optimal control for maximizing potential energy in a variable stiffness joint, Proc. 51st IEEE Annu. Conf. Decis. Control (CDC) (2012) pp. 1199–1206

21.17 M. Garabini, A. Passaglia, F.A.W. Belo, P. Salaris, A. Bicchi: Optimality principles in variable stiffness control: The VSA hammer, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), San Francisco (2011) pp. 1843–1851

21.18 A.G. Feldman: Once more on the equilibrium-point hypothesis (λ model) for motor control, J. Mot. Behav. **18**, 17–54 (1986)

21.19 J.R. Flanagan, A.M. Wing: Modulation of grip force with load force during point-to-point arm movements, Exp. Brain Res. **95**, 131–143 (1993)

21.20 J. Won, N. Hogan: Stability properties of human reaching movements, Exp. Brain Res. **107**(1), 125–136 (1995)

21.21 P.L. Gribble, D.J. Ostry, V. Sanguineti, R. Laboissière: Are complex control signals required for human arm movement?, J. Neurophysiol. **79**, 1409–1424 (1998)

21.22 J.D.W. Madden, N.A. Vandesteeg, P.A. Anquetil, P.G.A. Madden, A. Takshi, R.Z. Pytel, S.R. Lafontaine, P.A. Wieringa, I.W. Hunter: Artificial muscle technology: Physical principles and naval prospects, IEEE J. Ocean. Eng. **29**(3), 706–728 (2004)

21.23 A. Parr: *Hydraulics and Pneumatics: A Technician's and Engineer's Guide* (Butterworth-Heinemann, Oxford 1998)

21.24 I.L. Krivts, G.V. Krejnin: *Pneumatic Actuating Systems for Automatic Equipment: Structure and Design* (CRC, Boca Raton 2006)

21.25 H.A. Baldwin: Realizable models of muscle function, Proc. 1st Rock Biomech. Symp. (1969) pp. 139–148

21.26 K. Inoue: Rubbertuators and applications for robots, Proc. 4th Int. Symp. Robotics Res. (1987) pp. 57–63

21.27 B. Hannaford, J.M. Winters, C.-P. Chou, P.H. Marbot: The anthroform biorobotic arm: A system for the study of spinal circuits, Ann. Biomed. Eng. **23**, 399–408 (1995)

21.28 D.G. Caldwell, G.A. Medrano-Cerda, M. Goodwin: Control of pneumatic muscle actuators, IEEE Control Syst. Mag. **15**(1), 40–48 (1995)

21.29 T. Hesselroth, K. Sarkar, P.P. van der Smagt, K. Schulten: Neural network control of a pneumatic robot arm, IEEE Trans. Syst. Man Cybern. **24**(1), 28–38 (1994)

21.30 A. Bicchi, S.L. Rizzini, G. Tonietti: Compliant design for intrinsic safety: General issues and preliminary design, Proc. IEEE Int. Symp. Intell. Robots Syst. (IROS), Maui (2001) pp. 1864–1869

21.31 F. Daerden, D. Lefeber: The concept and design of pleated pneumatic artificial muscles, Int. J. Fluid Power **2**(3), 41–50 (2001)

21.32 B. Verrelst, R. Van Ham, B. Vanderborght, D. Lefeber, F. Daerden, M. Van Damme: Second generation pleated pneumatic artificial muscle and its robotic applications, Adv. Robotics **20**(7), 783–805 (2006)

21.33 I.M. Gottlieb: *Practical Electric Motor Handbook* (Butterworth-Heinemann, Oxford 1997)

21.34 J.F. Gieras, M. Wing: *Permanent Magnet Motor Technology: Design and Applications* (CRC, Boca Raton 2002)

21.35 M. Gad-el Hak: *The MEMS Handbook: MEMS, Design and Fabrication* (CRC, Boca Raton 2006)

21.36 K. Uchino: *Piezoelectric Actuators and Ultrasonic Motors* (Kluwer, Boston 1997)

21.37 S.-C. Huang, W.-L. Chen: Design of topologically optimal micrograpper, IEEE Int. Conf. Syst. Man Cybern. (2008)

21.38 J. Ueda, T. Secord, H. Asada: Large effective-strain piezoelectric actuators using nested cellular architecture with exponential strain amplification mechanisms, IEEE/ASME Trans. Mechatron. **15**, 770–782 (2010)

21.39 J. Schultz, J. Ueda: Two-port network models for compliant rhomboidal strain amplifiers, IEEE Trans. Robotics **29**(1), 42–54 (2013)

21.40 J. Schultz, J. Ueda: Nested piezoelectric cellular actuators for a biologically inspired camera positioning mechanism, IEEE Trans. Robotics **29**(5), 1–14 (2013)

21.41 J. Schultz, J. Ueda: Experimental verification of discrete switching vibration suppression, IEEE/ASME Trans. Mechatron. **17**(2), 298–308 (2012)

21.42 G. Engdahl: *Handbook of Giant Magnetostrictive Materials* (Academic, San Diego 2000)

21.43 D.C. Lagoudas: *Shape Memory Alloys: Modeling and Engineering Applications* (Springer, New York 2008)

21.44 E. Torres-Jara, K. Gilpin, J. Karges, R.J. Wood, D. Rus: Composable flexible small actuators built from thin shape memory alloy sheets, IEEE Robotics Autom. Mag. **17**(4), 78–87 (2010)

21.45 L. Jinsong: *Shape-Memory Polymers and Multifunctional Composites* (CRC, Boca Raton 2009)

21.46 F. Carpi, E. Smela: *Biomedical Applications of Electroactive Polymer Actuators* (Wiley, Chichester 2009)

21.47 P. Brochu, Q. Pei: Advances in dielectric elastomers for actuators and artificial muscles, Macromol. Rapid Commun. **31**(1), 10–36 (2010)

21.48 H.S. Nalwa: *Ferroelectric Polymers: Chemistry, Physics, and Applications* (CRC, Boca Raton 1995)

21.49 F. Carpi, D. De Rossi, R. Kornbluh, R. Pelrine, P. Sommer-Larsen: *Dielectric Elastomers as Electromechanical Transducers: Fundamentals, Materials, Devices, Models and Applications of an Emerging Electroactive Polymer Technology* (Elsevier, Amsterdan 2008)

21.50 G. Fernandez: Liquid-crystal polymers: Exotic actuators, Nat. Mater. **12**(1), 12–14 (2013)

21.51 R.H. Baughman, C. Cui, A.A. Zakhidov, Z. Iqbal, J.N. Barisci, G.M. Spinks, G.G. Wallace, A. Mazzoldi, D.D. Rossi, A.G. Rinzler, O. Jaschinski, S. Roth, M. Kertesz: Carbon nanotube actuators, Science **284**(5418), 1340–1344 (1999)

21.52 A.E. Aliev, J. Oh, M.E. Kozlov, A.A. Kuznetsov, S. Fang, A.F. Fonseca, R. Ovalle, M.D. Lima, M.H. Haque, Y.N. Gartstein, M. Zhang, A.A. Zakhidov, R.H. Baughman: Giant-stroke, superelastic carbon nanotube aerogel muscles, Science **323**(5921), 1575–1578 (2009)

21.53 J.D.W. Madden: Stiffer than steel, Science **323**(5921), 1571–1572 (2009)

21.54 E. Guglielmino, C.W. Stammers, K.A. Edge, T. Sireteanu, D. Stancioiu: Damp-by-wire: Magnetorheological vs friction dampers, Proc. 16th IFAC World Cong. (2005) pp. 340–345

21.55 M. Laffranchi, N.G. Tsagarakis, D.G. Caldwell: A variable physical damping actuator (VPDA) for compliant robotic joints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 1668–1674

21.56 J.C. Dixon: *The Shock Absorber Handbook*, Senior Lect. Eng. Mech. (Wiley, Chichester 2007)

21.57 J. Li, D. Jin, X. Zhang, J. Zhang, W.A. Gruver: An electrorheological fluid damper for robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 2631–2636

21.58 C.-M. Chew, G.-S. Hong, W. Zhou: Series damper actuator: A novel force control actuator, Proc. 4th IEEE/RAS Int. Conf. Humanoid Robots (2004) pp. 533–546

21.59 B. Ebrahimi, M.B. Khamesee, F. Golnaraghi: A novel eddy current damper: Theory and experiment, J. Phys. D Appl. Phys. **42**(7), 075001 (2009)

21.60 H.A. Sodano, J.-S. Bae, D. Inman, K. Belvin: Improved concept and model of eddy current damper, J. Vib. Acoust. **128**(3), 294–302 (2006)

21.61 G.A. Pratt, M.M. Williamson: Series elastic actuators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. Hum. Robot Interact. Coop. Robots, Vol. 1 (1995) pp. 399–406

21.62 M.L. Latash: Motor synergies and the equilibrium-point, Mot. Control **14**(3), 294–322 (2010)

21.63 T. Morita, S. Sugano: Development and evaluation of seven-d.o.f. mia arm, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 462–467

21.64 J.W. Hurst, J. Chestnutt, A. Rizzi: An actuator with physically variable stiffness for highly dynamic legged locomotion, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4662–4667

21.65 R. Van Ham, B. Vanderborght, M. Van Damme, B. Verrelst, D. Lefeber: MACCEPA, the mechanically adjustable compliance and controllable equilibrium position actuator: Design and implementation in a biped robot, Robotics Auton. Syst. **55**(10), 761–768 (2007)

21.66 S. Wolf, G. Hirzinger: A new variable stiffness design: Matching requirements of the next robot generation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 1741–1746

21.67 N. Hogan: Adaptive control of mechanical impedance by coactivation of antagonist muscles, IEEE Trans. Autom. Control **AC-29**(8), 681–690 (1984)

21.68 S.A. Migliore, E.A. Brown, S.P. DeWeerth: Biologically inspired joint stiffness control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 4519–4524

21.69 G. Tonietti, R. Schiavi, A. Bicchi: Design and control of a variable stiffness actuator for safe and fast physical human/robot interaction, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 528–533

21.70 M.G. Catalano, G. Grioli, M. Garabini, F. Bonomo, M. Mancinit, N. Tsagarakis, A. Bicchi: Vsa-cubebot: A modular variable stiffness platform for multiple degrees of freedom robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5090–5095

21.71 A. Bicchi, G. Tonietti: Fast and *soft-arm* tactics, IEEE Robotics Autom. Mag. **11**(2), 22–33 (2004)

21.72 A. Albu-Schäffer, S. Wolf, O. Eiberger, S. Haddadin, F. Petit, M. Chalon: Dynamic modelling and control of variable stiffness actuators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 2155–2162

21.73 A.H.C. Gosline, V. Hayward: Eddy current brakes for haptic interfaces: Design, identification, and control, IEEE/ASME Trans, Mechatron. **13**(6), 669–677 (2008)

21.74 M. Catalano, G. Grioli, M. Garabini, F.W. Belo, A. di Basco, N. Tsagarakis, A. Bicchi: A variable damping module for variable impedance actuation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 2666–2672

21.75 VIACTORS (Deutsches Zentrum für Luft- und Raumfahrt, Università di Pisa, Universiteit Twente, Imperial College London, Instituto Italiano di Tecnologia, Vrije Universiteit Brussel): Variable Impedance Actuators data sheets, http://www.viactors.org/VSA%20data%20sheets.htm (2011)

21.76 G. Grioli, S. Wolf, O. Eiberger, W. Friedl, M. Grebenstein, H. Höppner, E. Burdet, D. Caldwell, R. Carloni, M. Catalano, D. Lefeber, S. Stramigioli, N. Tsagaratkis, M. Van Damme, R. Van Ham, B. Vanderborght, L.C. Visser, A. Bicchi, A. Albu-Schaeffer: Variable stiffness actuators: The user's point of view, Int. J. Robotics Res. **34**(6), 727–743 (2015)

21.77 S. Wolf, G. Grioli, W. Friedl, M. Grebenstein, H. Hoeppner, E. Burdet, D. Caldwell, A. Bicchi, S. Stramigioli, B. Vanderborght: Variable stiffness actuators: Review on design and components, IEEE/ASME Trans. Mechatron. **PP**(99), 1 (2015)

21.78 M. Krasnosel'skii, A. Pokrovskii: *Systems with Hysteresis* (Springer, Berlin, Heidelberg 1989)

21.79 A.G. Feldman: Functional tuning of the nervous system with control of movement or maintenance of a steady posture. II: Controllable parameters of the muscle, Biophysics **11**, 565–578 (1966)

21.80 A.C. Antoulas: *Approximation of Large-Scale Dynamical Systems* (SIAM, Philadelphia 2005)

21.81 I. Fantoni, R. Lozano, M.W. Spong: Energy based control of the pendubot, IEEE Trans. Autom. Control **45**(4), 725–729 (2000)

21.82 A. Jafari, N. Tsagarakis, D. Caldwell: A novel intrinsically energy efficient development of a novel actuator with adjustable stiffness (awas), IEEE Trans. Mechatron. **18**(1), 355–365 (2013)

21.83 A. De Luca: Decoupling and feedback linearization of robots with mixed rigid/elastic joints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 816–821

21.84 A. De Luca, P. Lucibello: A general algorithm for dynamic feedback linearization of robots with elastic joints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 504–510

21.85 M. Spong: Modeling and control of elastic joint robots, ASME J. Dyn. Syst. Meas. Control **109**, 310–319 (1987)

21.86 A. Albu-Schffäer, S. Wolf, O. Eiberger, S. Haddadin, F. Petit, M. Chalon: Dynamic modeling and control of variable stiffness actuators, Proc. IEEE Conf. Robotics Autom. ICRA (2010) pp. 2155–2162

21.87 M. Grebenstein, A. Albu-Schäffer, T. Bahls, M. Chalon, O. Eiberger, W. Friedl, R. Gruber, S. Haddadin, U. Hagn, R. Haslinger, H. Hoppner, S. Jörg, M. Nickl, A. Nothhelfer, F. Petit, J. Reill, N. Seitz, T. Wimböck, S. Wolf, T. Wusthoff, G. Hirzinger: The DLR hand arm system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Shanghai (2011) pp. 3175–3182

21.88 H.P. Olesen, R.B. Randall: *A Guide to Mechanical Impedance and Structural Response Techniques*, Tech. Rep. (Bruel and Kjaer, Noerum 1977)

21.89 H. Gomi, M. Kawato: Human arm stiffness and equilibrium-point trajectory during multi-joint movement, Biol. Cybern. **76**(3), 163–171 (1997)

21.90 N. Hogan: Impedance Control: An approach to Manipulation – Parts I–III, J. Dyn. Syst. Meas. Control **107**, 1–24 (1985)

21.91 F.A. Mussa-Ivaldi, N. Hogan, E. Bizzi: Neural, mechanical, and geometric factors subserving

arm posture in humans, J. Neurosci. **5**(10), 2732 (1985)
21.92 H. Gomi, R. Osu: Task-dependent viscoelasticity of human multijoint arm and its spatial characteristics for interaction with environments, J. Neurosci. **18**(21), 8965 (1998)
21.93 E. Burdet, R. Osu, D.W. Franklin, T.E. Milner, M. Kawato: Measuring stiffness during arm movements in various dynamic environments, Proc. 1999 ASME Annu. Symp. Haptic Interfac. Virtual Environ. Teleoper. Syst. (1999) pp. 421–428
21.94 E. Burdet, R. Osu, D.W. Franklin, T. Yoshioka, T.E. Milner, M. Kawato: A method for measuring endpoint stiffness during multi-joint arm movements, J. Biomech. **33**(12), 1705–1709 (2000)
21.95 D.W. Franklin, E. Burdet, R. Osu, M. Kawato, T.E. Milner: Functional significance of stiffness in adaptation of multijoint arm movements to stable and unstable dynamics, Exp. Brain Res. **151**(2), 145–157 (2003)
21.96 K.P. Tee, E. Burdet, C.M. Chew, T.E. Milner: A model of force and impedance in human arm movements, Biol. Cybern. **90**(5), 368–375 (2004)
21.97 E.J. Perreault, R.F. Kirsch, P.E. Crago: Multijoint dynamics and postural stability of the human arm, Exp. Brain Res. **157**(4), 507–517 (2004)
21.98 R.D. Trumbower, M.A. Krutky, B.S. Yang, E.J. Perreault: Use of self-selected postures to regulate multi-joint stiffness during unconstrained tasks, PLoS ONE **4**(5), e5411 (2009)
21.99 K. Hashimoto, T. Kureha, Y. Nishimura, K. Okumura, S. Muraoka: Measurement of mechanical impedance using quartz resonator force sensor during the process of grasping, SICE Annu. Conf., Vol. 1 (2004) pp. 722–726
21.100 A. Serio, G. Grioli, I. Sardellitti, N.G. Tsagarakis, A. Bicchi: A decoupled impedance observer for a variable stiffness robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5548–5553
21.101 F. Flacco, A. De Luca, I. Sardellitti, N. Tsagarakis: On-line estimation of variable stiffness in flexible robot joints, Int. J. Robotics Res. **31**(13), 1556–1577 (2012)
21.102 T. Ménard, G. Grioli, A. Bicchi: A real time observer for an agonist-antagonist variable stiffness actuator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)
21.103 T. Ménard, G. Grioli, A. Bicchi: A stiffness estimator for agonistic-antagonistic variable-stiffness-actuator devices, IEEE Trans. Robotics **30**(5), 1269–1278 (2014)
21.104 G. Grioli, A. Bicchi: A non-invasive real-time method for measuring variable stiffness, Proc. 10th Int. Conf. Robotics Sci. Syst., Zaragoza (2010)
21.105 J.K. Salisbury: Active stiffness control of a manipulator in cartesian coordinates, Proc. 19th IEEE Conf. Decis. Control (1980) pp. 83–88
21.106 C. Ott: *Cartesian Impedance Control of Redundant and Flexible-Joint Robots*, Springer Tracts. Adv. Robotics (Springer, Berlin, Heidelburg 2008)
21.107 N. Hogan: Mechanical impedance of single- and multi-articular systems. In: *Multiple Muscle Systems*, ed. by J.M. Winters, S.L.-Y. Woo (Springer, New York 1990) pp. 149–164
21.108 A. Albu-Schäffer, M. Fischer, G. Schreiber, F. Schoeppe, G. Hirzinger: Soft robotics: What cartesian stiffness can we obtain with passively compliant, uncoupled joints?, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004)
21.109 F. Petit, A. Albu-Schäffer: Cartesian impedance control for a variable stiffness robot arm, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2011)
21.110 G. Palli, C. Melchiorri, A. De Luca: On the feedback linearization of robots with variable joint stiffness, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008)
21.111 A. De Luca, F. Flacco, A. Bicchi, R. Schiavi: Nonlinear decoupled motion-stiffness control and collision detection/reaction for the VSA-II variable stiffness device, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009)
21.112 G. Palli, C. Melchiorri: Output-based control of robots with variable stiffness actuation, J. Robotics (2011) doi:dx.doi.org/10.1155/2011/735407
21.113 A. Albu-Schäffer, C. Ott, F. Petit: Energy shaping control for a class of underactuated Euler-Lagrange systems, Proc. IFAC Symp. Robot Control (2012)
21.114 I. Sardellitti, G. Medrano-Cerda, N.G. Tsagarakis, A. Jafari, D.G. Caldwell: A position and stiffness control strategy for variable stiffness actuators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)
21.115 F. Petit, A. Albu-Schäffer: State feedback damping control for a multi dof variable stiffness robot arm, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)
21.116 B. Vanderborght: Dynamic Stabilisation of the Biped Lucy Powered by Actuators with Controllable Stiffness, Ph.D. Thesis (Vrije Universiteit Brussel, Brussel 2007)
21.117 M. Uemura, S. Kawamura: Resonance-based motion control method for mulit-joint robot through combining stiffness adaptation and iterative learning control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009)
21.118 L.C. Visser, S. Stramigioli, A. Bicchi: Embodying desired behavior in variable stiffness actuators, Proc. IFAC Congr. Int. Fed. Autom. Control (2011)
21.119 D. Lakatos, F. Petit, A. Albu-Schäffer: Nonlinear oscillations for cyclic movements in variable impedance actuated robotic arms, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 503–513
21.120 D. Lakatos, G. Garofalo, F. Petit, C. Ott, A. Albu-Schäffer: Modal limit cycle control for variable stiffness actuated robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)
21.121 A. Velasco, G.M. Gasparri, M. Garabini, L. Malagia, P. Salaris, A. Bicchi: Soft-actuators in cyclic motion: Analytical optimization of stiffness and preload, Proc. IEEE/RAS Int. Conf. Humanoid Robots, Atlanta (2013)
21.122 G. Garofalo, C. Ott, A. Albu-Schäffer: Orbital stabilization of mechanical systems through semidef-

inite Lyapunov functions, Proc. Am. Control Conf. (ACC) (2013)
21.123 D. Lakatos, F. Petit, A. Albu-Schäffer: Nonlinear oscillations for cyclic movements in human and robotic arms, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 508–511
21.124 M. Papageorgiou: *Optimierung: Statische, Dynamische, Stochastische Verfahren* (Oldenbourg, Munich 1996)
21.125 A. Bicchi, G. Tonietti: Fast and soft arm tactics: Dealing with the safety-performance trade-off in robot arms design and control, IEEE Robotics Autom. Mag. **11**(2), 22–33 (2004)
21.126 S. Haddadin, M.C. Özparpucu, A.A. Schäffer: Optimal control for maximizing potential energy in variable stiffness joints, Proc. 51st IEEE Conf. Decis. Control (CDC), Maui (2012)
21.127 S. Haddadin, M. C. Özparpucu, F. Huber, N. Mansfeld, A. Albu-Schäffer: Exploiting the natural dynamics of elastic joints for peak velocity, Int. J. Robotics Res. (2012)
21.128 M. Garabini, A. Passaglia, F. Belo, P. Salaris, A. Bicchi: Optimality principles in stiffness control: The vsa kick, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 3341–3346
21.129 R. Incaini, L. Sestini, M. Garabini, M.G. Catalano, G. Grioli, A. Bicchi: Optimal control and design guidelines for soft jumping robots: Series elastic actuation and parallel elastic actuation in comparison, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 2477–2484
21.130 D. Garg, M.A. Patterson, W.W. Hager, A.V. Rao, D.A. Benson, G.T. Huntington: A unified framework for the numerical solution of optimal control problems using pseudospectral methods, Automatica **46**(11), 1843–1851 (2010)
21.131 W. Li, E. Todorov: Iterative linearization methods for approximately optimal control and estimation of non-linear stochastic system, Int. J. Control **80**(9), 1439–1453 (2007)
21.132 W. Li, E. Todorov: Iterative linear quadratic regulator design for nonlinear biological movement systems, Proc. Int. Conf. Inform. Control, Autom. Robotics (2004) pp. 222–229
21.133 S. Haddadin, F. Huber, A. Albu-Schaffer: Optimal control for exploiting the natural dynamics of variable stiffness robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 3347–3354
21.134 D. Benson: A Gauss Pseudospectral Transcription for Optimal Control, Ph.D. Thesis (MIT, Cambridge 2005)
21.135 IEEE Robotics and Automation Society Technical Committee on Soft Robotics: http://www.ieee-ras.org/soft-robotics
21.136 Natural Machine Motion Initiative: http://naturalmachinemotion.com

Multimedia Contents

22. Modular Robots

I-Ming Chen, Mark Yim

This chapter presents a discussion of modular robots from both an industrial and a research point of view. The chapter is divided into four sections, one focusing on existing reconfigurable modular manipulators typically in an industry setting (Sect. 22.2) and another focusing on self-reconfigurable modular robots typically in a research setting (Sect. 22.4). Both sections are sandwiched between the introduction and conclusion sections.

This chapter is focused on design issues. Rather than a survey of existing systems, it presents some of the existing systems in the context of a discussion of the issues and elements in industrial modular robotics and modular robotics research. The reader is encouraged to look at the references for further discussion on any of the presented topics.

22.1 **Concepts and Definitions** 531
22.1.1 Concept of Modularity 532
22.1.2 Definition and Classification of Modular Robots 532
22.2 **Reconfigurable Modular Manipulators** 533
22.2.1 Background of Modular Manipulator Systems 533
22.2.2 Module Design Issues 534
22.2.3 Interface Design Issues 534
22.2.4 Modeling of Modular Manipulators 534
22.2.5 Configuration Optimization 535
22.3 **Self-Reconfigurable Modular Robots** 535
22.3.1 Types of Self-Reconfigurable Modular Robots 536
22.3.2 System and Module Design Issues 536
22.3.3 Interface Design Issues 537
22.4 **Conclusion and Further Reading** 539
Video-References 540
References 540

Modularity in design engineering refers to a compartmentalization of elements. Most often modularity in complex systems occurs as a result of taking a complex system and dividing it into pieces in order to better understand the simpler elements and parallelize the design efforts. Modularity also facilitates the replacing of elements either for repair or upgrading new functionality. The alternative to a modular approach is an integrated approach where systems are designed as a whole. While integrated approaches tend not to be as easy to repair, upgrade or reconfigure, they do have fewer constraints on element design and therefore can be made more optimal. Integrated approaches can focus on lowering cost or having higher performance. In mechanical devices, the choice between modular or integrated architectures can have a large impact on the range of application as well as cost or performance [22.1].

22.1 Concepts and Definitions

For example, a hand drill with modular attachments can expand itsrange of functionality from drilling holes to screwing bolts or buffing surfaces. For robotics, the same impact applies. However, robots have an inher-

ent complexity that lends itself to modularity,such as actuator modules, sensor modules, and sometimes computation modules. In the following sections, definitions will be given for different kinds of modularity frequently seen in robotics.

22.1.1 Concept of Modularity

The product design literature can be looked at as encompassing robotics – which can be considered as an industrial or research product. The architecture of product modularity can be categorized into three subtypes: slot, bus, and sectional modularity [22.1]:

- *Slot architecture*: Each of the interfaces between components is of a different type from the others, so that the various components in the product cannot be interchanged.
- *Bus architecture*: There is a common bus to which the other physical components connect via the same type of interface.
- *Sectional architecture*: All interfaces are of the same type and there is no single element to which all the other components attach, i. e., there is no base component. The assembly is built up by connecting the components to each other via identical interfaces (Fig. 22.1).

Such classifications provide a good definition of modular robots as follows:

- If a complex robotic system adopts a slot- and bus-modularity design approach for its internal structure and architecture, not the external configuration, it can be called a *modularly designed robotic system* benefiting from design parallelization. Such a robot may have a unified and integrated configuration that cannot be changed from outside.
- If a robot adopts a bus- and sectional-modularity design approach for both internal structure and external configuration, it can be called a *modular robot*. The users can reconfigure the compartmentalization and interchange functional modules with some level of effort.

22.1.2 Definition and Classification of Modular Robots

Any system can be reconfigured by destructing and reconstructing it, for example in the worst case, using a blowtorch and milling machine. The key element we must define is the level of effort required to reconfigure. We propose here three levels from the lowest to the highest level of effort of the user:

1. The system reconfigures itself. It is *self-reconfigurable*.
2. The system is reconfigured by a lay user with or without special tools typically in matter of seconds to minutes.
3. The system is reconfigured only by an expert with specialized tools.

This chapter will focus on modular robots that give rise to *plug-and-play* reconfiguration of the system for task and function changes, levels 1 and 2. Level 1 systems are self-reconfigurable systems with sectional modularity. Level 2 systems in this chapter focuses on reconfigurable modular manipulators with a finite set of modules of different functions.

The modular manipulator type of robotic systems are natural evolution of industrial robot manipulators that consist of a number of specific functional modules, such as actuator modules, link modules, and end-effector modules. Subsequently, robots with the serial and branching topology, such as humanoid robots, legged robots, mobile manipulators adopt a similar approach for modularity as these functional modules form the basis of a robotic system.

The self-reconfigurable modular robots grew out of the concept of self-evolution and self-configuration of biological cells with identical units. Such a robot normally consists of a large number of a small set of types of mechatronic units that possess actuation, connection, communication, and computing capability that can be assembled together in arbitrary forms and also reconfigure itself.

Although the two types of modular robots originated from different fundamental concepts, the goals to provide a large number of possible robot configurations for different tasks with the same set of basic robot modules are the same.

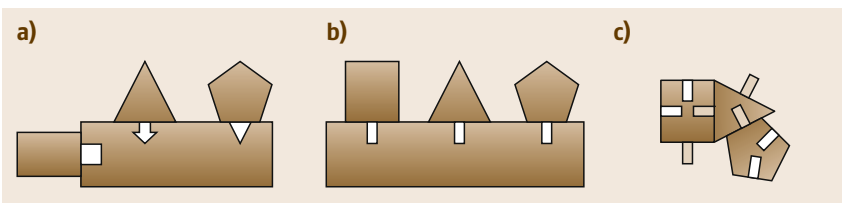


Fig.22.1a–c Modular architecture types: **(a)** slot architecture **(b)** bus architecture **(c)** sectional architecture

22.2 Reconfigurable Modular Manipulators

Reconfigurable modular manipulators are robot arms that have elements that can be rearranged.

22.2.1 Background of Modular Manipulator Systems

The simplest form of modular manipulator comes in the form of automatic tool changers also called quick-change end-effectors. These are optional tools that can be attached to the end of a robotic arm. Automatic tool changers are standard equipment for many CNC (computer numerically controlled) milling machines and lathes since the 1960s. They enable the machines to drill holes of different sizes, or cut different shapes. Although CNC machines are not often considered industrial robots, they share the same elements (actuators, sensors, and computation) and their function is more limited. Most industrial robot arms can be equipped with automatic changing end-effectors by adding a wrist that has the compatible interface for a variety of grippers and end-effectors. These devices are available commercially from Schunk (Germany), ATI industrial automation (USA), Destaco, Amatrol, RE2 (USA), RAD, and others.

In the modularization of industrial robots, the granularity of the components is usually based on their basic functions, i. e., motion actuation and tooling. Thus, the design of modules is highly differentiated into actuator modules, passive joint modules, and tooling modules, etc. Several prototype modular robotic systems have been built and demonstrated, including the reconfigurable modular manipulator system (RMMS) [22.2], several generations of the cellular robotic system (CEBOT) [22.3], and modular manipulator systems developed by University of Toronto [22.4], University of Stuttgart [22.5], University of Texas at Austin [22.6], and Toshiba Corp. [22.7].

Basically, these systems have serial-type (or open-chain) geometry with large working envelopes. These serial-type modular robots are well suited for assembly, trajectory tracking, welding, and hazardous material handling. Parallel modular robots have also been developed for light-machining tasks [22.8]. As indicated in [22.8], modular design can reduce the development cycle of the parallel robots significantly. Furthermore, it allows a trial-and-error approach for the construction of parallel robots that is impossible with the integrated design approach.

With globalization of world manufacturing, the concept of modular manipulators has quickly gained industrial attention. A full-scale reconfigurable robotic system workcell consisting of three modular robots with a total 15 axes was successfully showcased in 1999 [22.9] (Fig. 22.2). The modular robot workcell has a serial type 6-degrees-of-freedom (DOF) robot for the pick-and-place action of the work piece, a 2-DOF robot for work piece transfer, and a 6-DOF parallel robot for milling operations on the work piece. All the robots are built from the same set of modular components, including actuator modules, link modules, and tool modules.

The German company Amtec, later acquired by Schunk, developed the first commercially available modular manipulator system called PowerCube. Schunk has subsequently developed its industrial manipulators and automation systems based on PowerCube with some success [22.10]. Today there are many robotic systems with a wide spectrum of applications that are built around modular robot components.

The modular robot concept also proliferated in the hobby and educational robot sectors around the year 2000 by the introduction of well-packaged self-contained servo motor modules into inexpensive robotic devices, such as Robotis (Korea) and Kondo (Japan), as well as Lego (Denmark) and other toy companies making educational robots.

Mobile robots with legs, wheels, and tracks also belong to this class of modularity where they are configured for different task requirements such as those needed for disaster relief, rescue, and surveillance purposes. Two tracked modular mobile robots designed with multiple track segments [22.11] and reconfigurable tracks allowing serial and parallel connections [22.12] have been demonstrated. The work in [22.13] contains an in-depth review of the development in modular mobile robots.

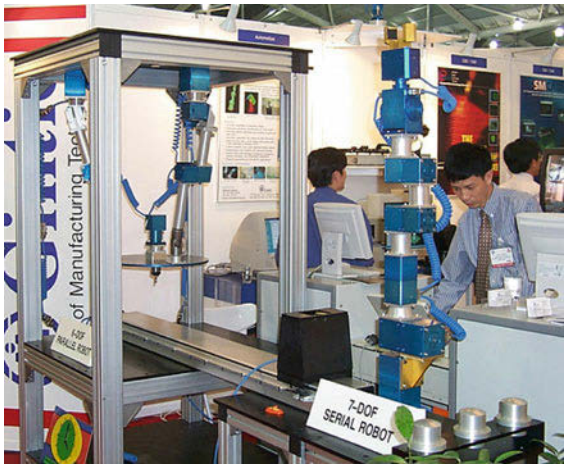


Fig. 22.2 15-axis Reconfigurable Robotic Workcell (after [22.9])

22.2.2 Module Design Issues

A modular robot consists of the two main features found in a modular product: 1) a one-to-one mapping from functional elements to the physical components of the product and 2) decoupled interfaces between the components of different modules [22.1]. For modular manipulators, the essential components are the base, positioning, and orienting mechanisms composed by actuator modules and link modules of different dimensions and geometry, and the end-effector module. For legged and wheeled mobile robotic systems, the motion generation mechanism modules are essential.

The actuator modules normally use DC or AC motors as a 1-DOF *rotate* or *pivot* joint module typically with compact high reduction ratio transmission mechanisms [22.2, 4, 6, 7, 9]. Some modular systems also adopt 1-DOF linear modules for larger workspace envelope [22.4, 9] and 2-DOF joint modules for compact dexterous motions [22.10]. The actuator module shown in Fig. 22.3 has two independent linear and rotary motion capabilities suitable for compact assembly tasks [22.9]. The actuator modules are typically designed with similar geometry but with different dimensions and power ratings for different application requirements.

The link modules connecting units in between the actuators function as reachable workspace extenders. Some systems adopt a standard fixed-dimension connection module [22.3, 4, 6] and some use variable dimension modules that can be customized to satisfy arbitrary design constraints [22.9]. In some systems [22.2], the link module becomes part of the actuator module so that the module acts as an actuator as well as the connecting structure.

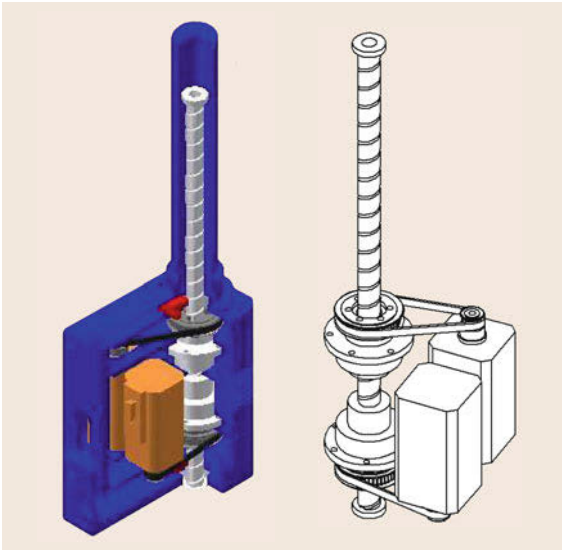


Fig. 22.3 A 2-DOF translate-turn module with ball-screw and ball-spline mechanism

22.2.3 Interface Design Issues

The mechanical connecting interface between modules in a modular manipulator needs to satisfy three basic requirements:

1. Stiffness
2. Fast reconfiguration
3. Interchangeability.

Thus, the design of mechanical connections or docking mechanisms is a critical issue. In a fully or semisupervised robotic system, like some modular manipulator systems [22.2–9], the connecting mechanism is designed to be manually operated for reliability and safety reasons. In a fully autonomous system, the connecting mechanism needs to be designed typically with an extra actuator and locking mechanism for carrying out the connection automatically. This is the case for most of the self-reconfigurable modular robots.

In order to meet the requirement of interchangeability, the electronic and communication interface for the modular system normally adopts common communication network architecture with plug-and-play capability similar to local area network (LAN). There are a number of existing industrial standard network protocols for real-time robot control suitable for such applications, like CAN-bus, RS485, and IEEE 1394. The progressive development of industrial automation protocols will facilitate the implementation of modular robot communications. While many of the early systems use wired multidrop bus architectures for communications, multirobot systems have used fast local message forwarding [22.14] and wireless networking.

22.2.4 Modeling of Modular Manipulators

Challenges to model the modular manipulator systems come from the lack of uniform formalisms of the unfixed robot configuration and geometry and the errors accumulated from assembly and dis-assembly of the modules. Hence, the first effort in modular robot modeling was the introduction of a graph-based technique with additional module assembly information for the representation of a modular robot configuration [22.15]. This work introduced a modular robot representation scheme, termed assembly incidence matrix (AIM) for distinct modular robot configurations. There are several subsequent extensions and variations of AIM for broader categories of modular robots including modular mobile manipulators [22.16–18].

Once the modular robot configuration can be distinctly defined with the type of modules, the connection sequence, module orientation, kinematic, and dynamic models of the robot can be obtained through an automatic generation algorithm [22.19]. Kinematic model generation can be achieved through the conventional Denavit–Hartenberg (DH) parameterization [22.2–4, 17] or the coordinate-free local product-of-exponential (POE) approach [22.8, 9, 19]. However, the DH method does not provide a clear distinction between the arranging sequence of modules in a robot chain, and it is an initial position-dependent representation. The local POE formulation of the kinematics and dynamics based on the theory of Lie groups and Lie algebras for rigid motion in $se(3)$ and $SO(3)$ can avoid this problem. Furthermore, the POE representation can avoid the singularity conditions that frequently occur in the kinematic calibration formulated by the DH method [22.20]. Thus, POE representations provide a uniform and well-behaved method for handling the inverse kinematics of both calibrated and un-calibrated robot systems. In local POE modeling, the joint axes are described in the local module (body) coordinate systems, it is progressive in constructing the kinematic models, so it conveniently resembles the assembling action of the physical modular robot components.

The machining tolerance, compliance, and wear of the connecting mechanism due to frequent module reconfiguration may introduce errors in positioning the end effector. Hence, kinematic calibration is a must for modular robots. In the POE calibration model, the robot errors are assumed to be in the initial positions of the consecutive modules because the local POE model is a zero reference method. Based on linear superposition and differential transformation, a 6-parameter error model can be established for serial-type robots [22.19]. This model can be obtained through the automatic generation process. An iterative least-square algorithm employed to find the error parameters to be corrected. The corrected kinematic model is then updated in the robot controller for operation. The simulation and experiment have shown that the proposed method can improve the position accuracy up to two orders of magnitude, or to the nominal repeatability of the robot after calibration with measurement noise. A typical 6-DOF articulate-type modular robot can reach a position accuracy of 0.1 mm compared to an accuracy of 1 mm before the calibration [22.20]

A formulation of the dynamic model of modular manipulators starts from a recursive Newton–Euler algorithm [22.21, 22]. The generalized velocity, acceleration, and forces can be expressed in terms of linear operations on $se(3)$ [22.23]. Based on the relationship between the recursive formulation and the closed-form Lagrangian formulation for serial-robot dynamics discussed in [22.24, 25], the AIM can assist in the construction of the closed-form equation of motion of a modular robot in any generic topology with redundant and non-redundant configurations [22.19].

22.2.5 Configuration Optimization

Due to the modular design, the modular manipulator can be optimal at the component level, but may not obtain optimal performance at the system level. Task-driven robot configuration optimization becomes necessary to establish locally optimal performance for the overall robotic system. Typically, the problem of robot configuration optimization can be stated as finding an assembly of robot modules that can achieve a certain task requirement based on an inventory of modules. The configuration of a modular robot can be treated as a compound entity with finite number of constituents. Finding the most suitable task-oriented robot configuration then becomes a discrete design optimization problem using a task performance related objective function. Discrete optimization techniques, such as genetic algorithms (GAs), the simulated annealing (SA) method, and other artificial intelligence techniques have been employed to find solutions [22.26–28].

The criteria used in selecting the optimal configuration depend largely on the task requirements, which mostly describe the necessary robot trajectories or key postures. *Yang* and *Chen* proposed a reduced DOF approach to minimize the total number of actuator modules employed in a serial-type modular robot for a given task [22.29]. With fewer modules, the robot can carry more payloads instead of carrying distal modules. Furthermore, the robot can be operated at higher speed with better dynamic response.

22.3 Self-Reconfigurable Modular Robots

Self-configurable systems can rearrange their own topology. An example is shown in VIDEO 2. There are dozens of research groups who have constructed many versions of self-reconfigurable robots [22.3, 30–46], with many approaches for programming them [22.36, 47–61]. As of 2012, over 800 papers and a book [22.31] have been written.

These systems are characterized by many identical modules that can be rearranged into a variety of shapes and configurations and by being highly scal-

able with simulated systems having hundreds, thousands, or millions of modules. These systems have three promises:

1. Low cost from batch fabrication of repeated modules
2. High robustness from redundancy and the ability to self-repair
3. High versatility from the ability to reconfigure and adapt to changing situations [22.62].

Practically speaking, none of the promises have been proven, though the promise of versatility is getting close. These systems have exhibited a wide variety of locomotion and manipulation including: legged walking with between 2 and 14 legs; riding a tricycle [22.63]; rolling like a tread [22.14]; snake-like locomotion (lateral and rectilinear undulation, concertina, sidewinding) [22.64]; manipulation of large objects with multiple arms/fingers [22.65]; manipulation of small objects; climbing stairs, fences, poles, in pipes; self-reconfiguration between dozens of shapes and many others [22.31]. Figure 22.4 and VIDEO 1 show a self-reconfigurable modular robot SMORES [22.66]. Other self-reconfigurable robots like ATRON [22.40] is shown in VIDEO 5 and M-blocks is shown in VIDEO 3.

22.3.1 Types of Self-Reconfigurable Modular Robots

Self-reconfiguring systems can be classified into three types based on the style of reconfiguration: chain, lattice, and mobile [22.62]. The chain systems reconfigure by using chains of modules that form and break loops [22.36, 62]. They tend to be well suited for work on the environment, as they can form articulated limbs. The lattice systems have modules which have nominal positions sitting on a regular lattice and tend to be better at self-reconfiguration as moving to neighboring lattice positions makes collision checking easy [22.33, 34, 39, 43]. The mobile systems have modules that individually maneuver on terrain and reconfigure by moving on the environment to relocate themselves in a conglomerate [22.38, 67].

Of the systems that have been implemented to date, some that have been shown to be most capable (judging by number of demonstrations) are the hybrid chain-lattice systems: Superbot [22.14], MTRAN III [22.68] and CKbot [22.69]. Recent additions to this group are Johns Hopkins University [22.70], iMobot [22.71], and SMORES [22.66] which are hybrid in all three areas, chain-lattice-mobile systems.

22.3.2 System and Module Design Issues

There is an interesting phenomenon called second system syndrome [22.72], where designers include many features in the second version of a system they design. They often include many more features than the system may need. This is especially problematic in modular reconfigurable systems with repeated modules; any feature added to one module has the possibility of its effect multiplied by n, the number of modules. For example, an increase of d in computational processing power (e.g., microprocessor without interlocked pipeline stages, MIPS) in one module will result in a system increase of dn.

Another similar phenomenon for designers is called *feature creep*, where more and more features are added as the system is being designed. Often this has a cumulative linear effect on cost, but worse, there is an exponential effect on reliability.

The simplest form of robustness analysis assumes independent probabilities that a module may fail during a specified function. If one module succeeds during that function with probability p then the system with $n$identical module has a probability of success of p^n, under the assumption that all modules must be functioning to succeed. Clearly this is problematic for large n.

As systems scale up in number, a key to make things work is to be sure that the system does not depend on every module working perfectly. Indeed, if only X numbers are needed, one would wonder why one would use more than X, especially if the only impact is reduced reliability. Here, one strategy is to ensure that the solution is devised in a way where the performance improves with the number of modules. If modules fail, then the system can gracefully degrade. In systems with tightly coupled actions between modules, this can be difficult

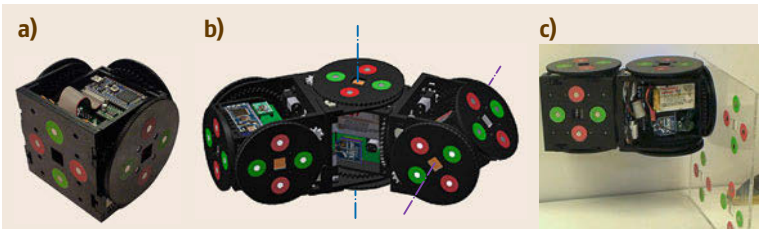


Fig. 22.4 (a) One SMORES module with four main actuators and four docking faces. (b) Three SMORES modules attached together. (c) Two modules moving on a lattice

to implement. In tasks where only binary metrics exist, (success or failure) using the optimal X modules, setting aside the extras may be the best solution.

One of the aspects that researchers find most interesting about self-reconfigurable robots is examining what happens when modules scale up in number (and also typically, scaled down in size). As the numbers increase, the number of shapes/configurations increases and concomitantly, the types of activities that are possible increase as well.

Control and planning get very complicated quickly though. Simply enumerating the number of isomorphic configurations has resulted in a PhD thesis without completely solving the problem [22.15].

While researchers have simulated hundreds, thousands, and even millions of modules, no physical system has been demonstrated with more than a few hundred to date. The largest single system so far is Kilobot [22.73], which has very simple mobile modules that swarm together and actually do not connect rigidly. The modules communicate wirelessly in a one-to-many (broadcast) fashion. For rigidly connected systems the largest number of modules demonstrated in one connected system was a 14-legged 48-module Polybot system [22.74].

Main Actuation

Every module has some form of actuation that enables the modules to move from one position to another or to do some work on the environment. We call it the *main actuator*. By far the most common main actuator is a DC electric motor as it is the lowest cost and easiest to implement.

In chain systems, the main actuator typically drives a revolute joint so that a chain of modules forms an articulated arm. In lattice systems, the main actuator typically moves the module (or a neighboring module) along a constrained 1-DOF path that can be translation or rotation. In mobile reconfiguration, the main actuator enables locomotion, usually through wheels or treads.

The main actuator is typically the largest component in a module and so has been the focus when trying to make modules smaller. To date, the smallest module with onboard actuators, that both attaches and detaches uses shape memory actuated module that was 2 cm [22.75]. An even smaller 1 mm module was created at Carnegie Mellon University [22.76] using electrostatics as an actuator, but it did not attach to other modules.

On the other extreme, the largest module is GHC [22.77], an 8 m^3 helium-filled cube with shape memory actuators on the edges to rotate these floating balloons about edges, attaching electrostatically. DARPA is also sponsoring a project to look at reconfigurable maritime craft constructed in a 20′ ISO container form factor. University of Pennsylvania has demonstrated scale models at 1/12 scale, with main actuators being thrusters to maneuver the parallelepipeds in water [22.78].

22.3.3 Interface Design Issues

The main component of modular systems differentiating it from an integrated system is the interface between modules. When talking about modular systems, the amount of *modularity* can be measured by the number of these interfaces. For self-reconfigurable modular robotic systems in this chapter the number of interfaces in one connected component can be as small as six or as large as millions.

In the most general case, every interface must do two things: 1) attach and 2) detach. When they attach they must do two things: 1) form a physical coupling and 2) allow a flow through the interface for supply power and information (often done electrically).

The interfaces can be *gendered* [22.36], that is two mating interfaces are not identical, one has male features sticking out and the other female features to receive the male features. They can be *ungendered* with no protruding or receding mating features. Or they can be *hermaphroditic* [22.39] where interfaces contain both male and female features. Ungendered and hermaphroditic interfaces can have identical interfaces on both sides. This increases the number of possible arrangements over gendered modules; however, ungendered and hermaphroditic components are typically more complex than gendered interfaces.

Each module can have multiple interfaces. If we consider the number of configurations possible with m interfaces per module assuming identical modules and ignoring any physical constraints such as self-collision, we obtain m^n possible configurations. The number of non-isomorphic configurations is much less, but is very difficult to enumerate in general. However, one special case to consider is two interfaces per module. In this case, topologically, there are only two different configurations, a single chain, and a loop. For this reason, most systems have three or more interfaces.

In addition, each interface can have multiple symmetries that allow different kinematic relationships for the same adjacency relationship. For example many systems have either a 2-way connector (modules may be optionally be rotated 180° to each other) or a 4-way connector (90° rotations.) This p-way symmetry leads to $(pm)^n$ total possible configurations.

Mechanical Interfaces

Mechanical interfaces rigidly attach two modules together with the ability to detach as well. Strategies to

enable this capability include using nuts and bolts, magnetic bonding [22.39], electrostatic bonding [22.77], probe and drogue [22.79], and a variety of physical hook-type mechanisms. Each mechanical interface has three performance metrics: bond strength, acceptance range, and interface precision/stiffness.

Bond strength can be characterized by the amount of force required to separate two modules in its weakest direction [22.80]. The acceptance range is the tolerance to position errors as two docks approach to dock [22.81]. Interface precision/stiffness indicates the amount of position error (deflection) that can occur contributed by the interface under load. This can come from elastic deflection, *slop* at a joint in a mate, or misalignments in non-self-guiding docks.

For self-reconfiguring systems, attaching and detaching is automatic and the area of acceptance is required to be much larger than typical manually assembled modular systems. This is in large part because the cost of precision is high and bond precision/stiffness is typically small. The ideal mechanical interface has high bond strength, high acceptance range and high bond stiffness, all of which comes with increased weight, size, and cost. However, different styles of reconfiguration emphasize different aspects.

Lattice reconfiguration systems do not need high acceptance range as modules move in highly constrained manner when reconfiguring, usually with one degree of freedom from one lattice position to neighboring one. However, high bond stiffness is emphasized, as these modules do not often have actuated degrees of freedom to compensate for errors in position. For example, a sequence of modules in a lattice that are forming a loop might not close properly because one section of modules may sag under gravity. This sag was noticeable in the 3-D lattice module [22.82].

Chain reconfiguration modules can deal with lower bond stiffness since they usually have degrees of freedom that can compensate for errors in deflection. However, docking even with this compensation is not trivial and most systems require as large an acceptance range as possible.

Bond strength for systems that reconfigure tend to be much lower than the one would find in a manually assembled system. For example, magnetic bonding methods are easy to implement, typically have good precision and wide range of acceptance, but are weak. Manual systems that are bolted together have high bond strength, but are very complicated if they are made automatic with a very small acceptance range. The variability in bond strength versus the stiffness in joints can be used to vary the overall stiffness of a conglomerate system [22.80]. This variable stiffness can be used to make a compliant material that can conform to surfaces or a stiff material that will not bend under load.

Depending on the task, bond strength may not be required to be high. For example, in mobile reconfiguration systems that primarily move on the ground, e.g., forming trains such as *millibot* [22.67] or *swarms* [22.38], the worst case situation is when the conglomerate tries to cross a gap with modules cantilevered over the gap. Otherwise, the conglomerate is nominally supported at all times and the bond only sees dynamic friction and inertial forces. But in general, higher bond strength does not hurt and keeps a system from falling apart from static or dynamic loads.

Power and Communications Interface

Power and communications interfaces are typically electrical, though in the case of the related work in quick-change end effectors, pneumatics is often supplied too. In any case, the primary parameters of concern are the number of lines that must be transmitted between the interface, and the type of lines (e.g., pneumatic, high voltage, high current, fiber optic, etc.). The type of line will determine the size (which usually limits the number of lines) as well as the precision alignment required at the interface.

In self-reconfigurable systems, the typical interface passes electric power and a separate electronics communications bus on which all modules talk, though they are sometimes combined [22.77]. A key consideration for electronic lines is not only that the correct lines have good contact when mated, but also that wrong contacts do not touch during the docking process due to position error. For most self-reconfiguring systems maximizing the tolerance to position errors is the most important [22.56].

Communication between modules is essential for any self-reconfigurable system, whether the control is distributed or centralized. There are primarily two forms of communication: a global communications bus (any module talks to any other module) or a local communications medium (modules only talk to their neighbors through the modular interface.)

These two forms have different implications. Global buses are typically much faster than local methods of equivalent cost. There are no issues of varied latency between modules as with local methods. One problem with global buses is that there is no mechanism for modules communicating with each other to know the relative physical position of each module. Local methods get this for free. Local methods, can also emulate global architectures by message passing and routing. Local methods are also more robust to physical errors (such as shorting a communications line to power) containing failures to locally. However, they are just as

Table 22.1 Characteristics of used communication protocols

Method	Speed (kbs)	Range (m)	Power (mW)	Address space	Rel. cost	Notes
Ethernet/CAT	1k–1m	100's	500	≈4 billion	$$$$	Complex stack (IPV4)
CANbus	10–1000	100	≈ 200	≈ 2000	$$$	Robust (base frame)
Bluetooth	1k–3k	10	≈ 100	≈1 million	$$$	3 s wake-up
BLE (wibree)	1k	10	10	≈1 million	$$	Low power Bluetooth
Zigbee	20–250	10–75	≈ 30	64 bit	$$	Wakeup 15 ms
RS485	100–10 k	1200	5	256	$	Robust elec. protoc
SPI	1–12 k	≈ 10	5	Out of band	$	Simple sync 5 wire

vulnerable to software errors such as flooding a network with garbage messages.

There are many choices for global communication protocols falling into two categories: wired and wireless. Wired is the most common and has many possible protocols, mostly requiring that the bus is multidrop. Popular wired protocols include Ethernet, EtherCAT, RS-485, SPI, and CANbus. Popular wireless protocols include Zigbee, Bluetooth, and 802.11. Important characteristics of communication protocols include the speed, real-time aspects, address space, and cost. A comparison is shown in Table 22.1 with typical values of common implementation.

The importance of the speed depends (or often dictates) the architecture of distributed control. Communications to individual modules can range from 100 times a second (for direct remote control) to several times per minute for higher level behavior control (e.g., hormone control [22.50]). Many protocols have automatic recovery from bad packets, which has obvious importance. Ethernet uses random backoff retransmission after a collision, which makes the protocol non-deterministic. This makes it difficult to have guarantees of real-time performance. The protocol is very fast so it is possible to ignore this as messages will get through with small latency with high probability. EtherCAT is Ethernet for Control Automation Technology that is better suited for real-time control. CANbus (Control Area Network bus) is a well-established bus used in the automotive technology that can be real-time, and robust, though it is typically slower than Ethernet or EtherCAT.

22.4 Conclusion and Further Reading

After more than a quarter century of research and development in modular robotics, a number of modular robotic systems have successfully entered the industry automation, education and entertainment markets. These successful modular robotic systems indicate that modular design offers advantages in the areas of product variety, application variety and creativity. However, the cost of such systems, no matter at the module level or at the system level, has room to be reduced to help mass adoption.

The cost structure of a modular robot is closely linked with the design of the individual modules and the systemic architecture to be conceived as well as the market demand and expectation. From the development history of LEGO bricks to the Schunk Powercube modules, it is clear that module design functions are becoming simpler (reducing cost) and increasing variability (increasing the user base). Hence, for reconfigurable modular manipulators, mobile systems, and self-reconfigurable modular robots, focusing on system designs to meet the expectation of the end user is the current trend.

Besides system design, standardization of mechanical and communication interfaces is critical. Like other electronic and industrial products, electronic and communication interfaces for modular robots could adopt existing industry standards depending on the form factors, connecting performance, reliability, etc. The mechanical interface design normally does not have industry standard to follow due to variety in mechanical specifications on the form factor, loading capacity, rigidity of the joints. Hence, many novel mechanical connection designs can be explored for modular robots.

Future research continues to explore increasing the number of modules. As they approach hundreds and thousands, there are interesting questions that arise as modules become more tightly coupled than current efforts [22.73]. Issues include how to deal with the increased likelihood that some modular elements are not functioning completely correctly. Biologically inspired mechanisms such as intentional cell-death and cell-replacement may become a required part of very large systems.

As the modular approach has increasingly large numbers of modules, there are many more configurations and resulting capabilities from those configurations. Future research will need to address the problem of determining appropriate or optimal configurations

for arbitrary tasks. This may lead to a better understanding of robotic tasks in general.

The most recent review article on modular mobile robots can be found in [22.13]. In [22.30], the technical challenges and future of self-reconfigurable modular robots are reviewed. Interested readers may find further information on self-reconfigurable robots in [22.31].

Video-References

VIDEO 1 SMORES
available from http://handbookofrobotics.org/view-chapter/22/videodetails/1

VIDEO 2 4x4ht4a
available from http://handbookofrobotics.org/view-chapter/22/videodetails/2

VIDEO 3 M-Blocks: Momentum-driven, magnetic modular robots self-reconfiguring
available from http://handbookofrobotics.org/view-chapter/22/videodetails/3

VIDEO 5 ATRON robot showing robust and reversible execution of self-reconfiguration sequences
available from http://handbookofrobotics.org/view-chapter/22/videodetails/5

References

22.1 K. Ulrich: The role of product architecture on the manufacturing firm, Res. Policy **24**, 419–440 (1995)

22.2 C. Paredis, H.B. Brown, P. Khosla: A rapidly deployable manipulator system, Proc. IEEE ICRA (1996) pp. 1434–1439

22.3 T. Fukuda, S. Nakagawa: Dynamically reconfigurable robot system, Proc. IEEE ICRA (1988) pp. 1581–1586

22.4 R. Cohen, M. Lipton, M. Dai, B. Benhabib: Conceptual design of a modular robot, ASME J Mech. Des. **25**, 114–117 (1992)

22.5 K.H. Wurst: The conception and construction of a modular robot system, Proc. 16th Int. Symp. Ind. Robots (ISIR) (1986) pp. 37–44

22.6 D. Tesar, M.S. Butler: A generalized modular architecture for robot structures, ASME J. Manuf. Rev. **2**(2), 91–117 (1989)

22.7 T. Matsumaru: Design and control of the modular robot system: TOMMS, Proc. IEEE ICRA (1995) pp. 2125–2131

22.8 G. Yang, I.-M. Chen, W.K. Lim, S.H. Yeo: Kinematic design of modular reconfigurable in-parallel robots, Auton. Robot. **10**(1), 83–89 (2001)

22.9 I.-M. Chen: Rapid response manufacturing through a rapidly reconfigurable robotic workcell, Robotics Comput. Integr. Manuf. **17**(3), 199–213 (2001)

22.10 Schunk GmbH, Lauffen, Germany, http://www.schunk.com/

22.11 W. Wang, W. Yu, H. Zhang: JL-2: A mobile multi-robot system with docking and manipulating capabilities, Int. J. Adv. Robotic Syst. **7**(1), 9–18 (2010)

22.12 B. Li, S. Ma, J. Liu, M. Wang, T. Liu, Y. Wang: AMOEBA-I: A shape-shifting modular robot for urban search and rescue, Adv. Robotics **23**(9), 1057–1083 (2009)

22.13 P. Moubarak, P. Ben-Tzvi: Modular and reconfigurable mobile robotics, Robot. Auton. Syst. **60**, 1648–1663 (2012)

22.14 H.C.H. Chiu, M. Rubenstein, W.M. Shen: Multifunctional superbot with rolling track configuration, Proc. IEEE/RSJ IROS (2007)

22.15 I.-M. Chen, J.W. Burdick: Enumerating non-isomorphic assembly configurations of a modular robotic system, Int. J. Robotics Res. **17**(7), 702–719 (1998)

22.16 J. Liu, Y. Wang, S. Ma, Y. Li: Enumeration of the non-isomorphic configurations for a reconfigurable modular robot with square-cubic-cell modules, Int. J. Adv. Robotic Syst. **7**(4), 58–68 (2010)

22.17 H. Liu, H. Wang, S. Li, L. He: Research on the topology description and modeling method for reconfigurable modular robots, Proc. ASME/IFToMM Int. Conf. Reconfig. Mech. Robot. (ReMAR) (2009) pp. 129–135

22.18 E. Meister, A. Gutenkunst: Self-adaptive framework for modular and self-reconfigurable robotic systems, Proc. 4th Int. Conf. Adapt. Self-Adapt. Syst. Appl. (ADAPTIVE) (2012) pp. 30–37

22.19 I.-M. Chen, G. Yang, S.H. Yeo, G. Chen: Kernel for modular robot applications – Automatic modeling techniques, Int. J. Robotics Res. **18**(2), 225–242 (1999)

22.20 I.-M. Chen, C.T. Tan, G. Yang, S.H. Yeo: A local POE model for robot kinematic calibration, Mech. Mach. Theory **36**(11), 1215–1239 (2001)

22.21 J.M. Hollerbach: A recursive lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity, IEEE Trans. Syst. Man Cybern. **10**, 730–736 (1980)

22.22 G. Rodriguze, A. Jain, K. Kreutz-Delgado: A spatial operator algebra for manipulator modeling and control, Int. J. Robotics Res. **10**(4), 371–381 (1991)

22.23 R. Murray, Z. Li, S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)

22.24 R. Featherstone: *Robot Dynamics Algorithms* (Kluwer, Dordrecht 1987)
22.25 F.C. Park, J.E. Bobrow, S.R. Ploen: A lie group formulation of robot dynamics, Int. J. Robotics Res. **14**(6), 609–618 (1995)
22.26 I.-M. Chen, J.W. Burdick: Determining task optimal modular robot assembly configurations, Proc. IEEE ICRA (1995) pp. 132–137
22.27 J. Han, W.K. Chung, Y. Youm, S.H. Kim: Task based design of modular robot manipulator using efficient genetic algorithm, Proc. IEEE ICRA (1997) pp. 507–512
22.28 C. Paredis, P.K. Khosla: Agent based design of fault tolerant manipulators for satellite docking, Proc. IEEE ICRA (1997) pp. 3473–3480
22.29 G. Yang, I.-M. Chen: Task-based optimization of modular robot configurations: Minimized degree-of-freedom approach, Mech. Mach. Theory **35**(4), 517–540 (2000)
22.30 M. Yim, W.M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, G. Chirikjian: Modular self-reconfigurable robot systems – Challenges and opportunities for the future, IEEE Robotics Autom. Mag. **14**(1), 43–52 (2007)
22.31 K. Stoy, D. Brandt, D.J. Christensen: *Self-Reconfigurable Robots – An Introduction* (MIT, Cambridge 2010)
22.32 D. Rus, M. Vona: Self-reconfiguration planning with compressible unit modules, Proc. IEEE ICRA (1999)
22.33 S. Murata, E. Yoshida, K. Tomita, H. Kurokawa, A. Kamimura, S. Kokaji: Hardware design of modular robotic system, Proc. IEEE/RSJ IROS (2000)
22.34 K. Kotay, D. Rus: Self-reconfigurable robots for navigation and manipulation, Proc. IEEE/RSJ IROS (1997)
22.35 C. Ünsal, H. Kiliççöte, P.K. Khosla: I(CES)-cubes: A modular self-reconfigurable bipartite robotic system, Proc. SPIE, Sens. Fusion Decent. Control Robotic Syst., Bellingham (1999) pp. 258–269
22.36 A. Castano, W.M. Shen, P. Will: CONRO: Towards deployable robots with inter-robots metamorphic capabilities, Auton. Robots **8**(3), 309–324 (2000)
22.37 W.H. Lee, A.C. Sanderson: Dynamic analysis and distributed control of tetrobot modular reconfigurable robotic system, Auton. Robots **10**(1), 67–82 (2001)
22.38 E. Tuci, R. Gross, V. Trianni, F. Mondada, M. Bonani, M. Dorigo: Cooperation through self-assembly in multi-robot systems, ACM Trans. Auton. Adapt. Syst. (TAAS) **1**(2), 115–150 (2006)
22.39 J.W. Suh, S.B. Homans, M. Yim: Telecubes: Mechanical design of a module for self-reconfigurable robotics, Proc. IEEE ICRA (2002)
22.40 M.W. Jorgensen, E.H. Ostergaard, H.H. Lund: Modular Atron: Modules for a self-reconfigurable robot, Proc. IEEE/RSJ IROS (2004) pp. 2068–2073
22.41 M. Hafez, M.D. Lichter, S. Dubowsky: Optimized binary modular reconfigurable robotic devices, IEEE/ASME Trans. Mechatron. **8**(1), 18–25 (2003)
22.42 P. White, V. Zykov, J. Bongard, H. Lipson: Three dimensional stochastic reconfiguration of modular robots, Proc. Robotics Sci. Syst. **1**, 161–168 (2005)
22.43 B. Kirby, J. Campbell, B. Aksak, P. Pillai, J. Hoburg, T. Mowry, S.C. Goldstein: Catoms: Moving robots without moving parts, Proc. Natl. Conf. Artif. Intell. **20**, 1730–1731 (2005)
22.44 S.C. Goldstein, J.D. Campbell, T.C. Mowry: Programmable matter, Computer **38**(6), 99–101 (2005)
22.45 A.A. Goldenberg, N.M. Kircanski, P. Kuzan, J.A. Wiercienski, R. Hui, C. Zhou: Modular, expandable and reconfigurable robot, US Patent 552 3662 (1996)
22.46 H. Lipson, J.B. Pollack: Towards continuously reconfigurable self-designing robotics, Proc. IEEE ICRA (2000)
22.47 J.E. Walter, E.M. Tsai, N. Amato: Choosing good paths for fast distributed reconfiguration of hexagonal metamorphic robots, Proc. IEEE ICRA (2002)
22.48 A. Castano, P. Will: Representing and discovering the configuration of CONRO robots, Proc. IEEE ICRA (2001) pp. 3503–3509
22.49 K. Støy, W.M. Shen, P. Will: Global locomotion from local interaction in self-reconfigurable robots, Proc. 7th Int. Conf. Intell. Auton. Syst. (IAS-7), Marina del Rey (2002)
22.50 P. Will, B. Salemi, W.M. Shen: Hormone-controlled metamorphic robots, Proc. IEEE ICRA (2001) pp. 4194–4199
22.51 A. Kamimura, S. Murata, E. Yoshida, H. Kurokawa, K. Tomita, S. Kokaji: Self-reconfigurable modular robot – Experiments on reconfiguration and locomotion, Proc. IEEE/RSJ IROS (2001) pp. 606–612
22.52 R. Fitch, D. Rus, M. Vona: Basis for self-repair robots using self-reconfiguring crystal modules, Proc. IEEE Int. Auton. Syst. (2000) pp. 903–910
22.53 G.S. Chirikjian, A. Pamecha, I. Ebert-Uphoff: Evaluating efficiency of self-reconfiguration in a class of modular robots, J. Robotic Syst. **13**(5), 317–338 (1996)
22.54 F.H. Bennett, E.G. Rieffel: Design of decentralized controllers for self-reconfigurable modular robots using genetic programming, Proc. 2nd NASA/DoD Workshop Evol. Hardw. (2000)
22.55 Z. Butler, R. Fitch, D. Rus: Distributed control for unit-compressible robots: Goal-recognition, locomotion, and splitting IEEE/ASME Trans. Mechatron. **7**(4), 418–430 (2002)
22.56 M. Yim, Y. Zhang, K. Roufas, D. Duff, C. Eldershaw: Connecting and disconnecting for chain self-reconfiguration with PolyBot, IEEE/ASME Trans. Mechatron. **7**(4), 442–451 (2002)
22.57 Y. Zhang, M.P.J. Fromherz, L.S. Crawford, Y. Shang: A general constraint-based control framework with examples in modular self-reconfigurable robots, Proc. IEEE/RSJ IROS (2002) pp. 2163–2168
22.58 H. Bojinov, A. Casal, T. Hogg: Emergent structures in modular self-reconfigurable robots, robotics and automation, Proc. IEEE ICRA (2000)
22.59 P. Pirjanian, C. Leger, E. Mumm, B. Kennedy, M. Garrett, H. Aghazarian, S. Farritor, P. Schenker: Distributed control for a modular, reconfigurable cliff robot, Proc. IEEE ICRA (2002)
22.60 K.C. Prevas, C. Unsal, M.O. Efe, P.K. Khosla: A hierarchical motion planning strategy for a uniform

self-reconfigurable modular robotic system, Proc. IEEE ICRA (2002)
22.61 W.W. Melek, A.A. Goldenberg: Neurofuzzy control of modular and reconfigurable robots, IEEE/ASME Trans. Mechatron. **8**(3), 381–389 (2003)
22.62 M. Yim, D.G. Duff, K.D. Roufas: Polybot: A modular reconfigurable robot, Proc. IEEE ICRA (2000) pp. 514–520
22.63 M. Yim, Y. Zhang, D. Duff: Modular Reconfigurable Robots, Machines that shift their shape to suit the task at hand, IEEE Spectrum **39**(2), 30–34 (2002)
22.64 M. Yim, C. Eldershaw, Y. Zhang, D. Duff: Limbless conforming gaits with modular robots. In: *Springer Tract. Adv. Robot.*, Vol. 21, 2006) pp. 459–468
22.65 M. Yim, J. Reich, A.A. Berlin: Two approaches to distributed manipulation. In: *Distributed Manipulation*, ed. by K. Bohringer, H.H. Choset (Kluwer, Dordrecht 2000) pp. 237–261
22.66 J. Davey, N. Kwok, M. Yim: Emulating self-reconfigurable Robots – Design of the SMORES system, Proc. IEEE/RSJ IROS (2012)
22.67 H.B. Brown, J.M. Vande Weghe, C.A. Bererton, P.K. Khosla: Millibot trains for enhanced mobility, IEEE/ASME Trans. Mechatron. **7**(4), 452–461 (2002)
22.68 S. Murata, K. Kakomura, H. Kurokawa: Toward a scalable modular robotic system, IEEE Robotics Autom. Mag. **14**(4), 56–63 (2007)
22.69 M. Yim, B. Shirmohammadi, J. Sastra, M. Park, M. Dugan, C.J. Taylor: Towards robotic self-reassembly after explosion, Proc. IEEE/RSJ IROS (2007) pp. 2767–2772
22.70 M. Kutzer, M. Moses, C. Brown, M. Armand, D. Scheidt, G. Chirikjian: Design of a new independently-mobile reconfigurable modular robot, Proc. IEEE ICRA (2010)
22.71 H. Cheng, G. Ryland: Design of iMobot, an intelligent reconfigurable mobile robot with novel locomotion, Proc. IEEE ICRA (2010)
22.72 F.P. Brooks: *The Mythical Man-Month* (Addison-Wesley, Reading 1975)
22.73 M. Rubenstein, R. Nagpal: Kilobot: A robotic module for demonstrating behaviors in a large scale (2^{10} units) collective, Proc. IEEE ICRA (2010) pp. 47–51
22.74 Y. Zhang, M. Yim, C. Eldershaw, D. Duff, K. Roufas: Scalable and reconfigurable configurations and locomotion gaits for chain-type modular reconfigurable robots, Proc. IEEE Symp. Comput. Intell. Robotics Autom. (CIRA) (2003)
22.75 E. Yoshida, S. Murata, S. Kokaji, K. Tomita, H. Kurokawa: Micro self-reconfigurable modular robot using shape memory alloy, J. Robotics Mechatron. **13**(2), 212–218 (2001)
22.76 M.E. Karagozler, S.C. Goldstein, J.R. Reid: Stress-driven MEMS assembly + electrostatic forces = 1mm diameter robot, Proc. IEEE/RSJ IROS (2009)
22.77 M.E. Karagozler, J.D. Campbell, G.K. Feddore, S.C. Goldstein, H.P. Weller, B.W. Yoon: Electrostatic latching for inter-module adhesion, power transfer, and communication in modular robots, Proc. IEEE/RSJ IROS (2007)
22.78 I. O'Hara, J. Paulos, J. Davey, N. Eckenstein, N. Doshi, T. Tosun, J. Greco, Jungwon. Seo, M. Turpin, V. Kumar, M. Yim: Self-assembly of a swarm of autonomous boats into floating structures, Proc. IEEE ICRA (2014) pp. 1234–1240
22.79 L. Vasilescu, P. Varshavshaya, K. Kotay, D. Rus: Autonomous modular optical underwater robot (amour) design, prototype and feasibility study, Proc. IEEE ICRA (2005)
22.80 P.J. White, S. Revzen, C.E. Thorne, M. Yim: A general mechanical strength model for programmable matter and modular robotic structures, Robotica **29**(1), 103–121 (2011)
22.81 N. Eckenstein, M. Yim: The X-Face: An improved planar passive mechanical connector for modular self-reconfigurable robots, Proc. IEEE/RSJ IROS (2012) pp. 3073–3078
22.82 S. Murata, H. Kurokawa, E. Yoshida, K. Tomita, S. Kokaji: A 3-D self-reconfigurable structure, Proc. IEEE ICRA (1998) pp. 432–439

Multimedia Contents

23. Biomimetic Robots

Kyu-Jin Cho, Robert Wood

Biomimetic robot designs attempt to translate biological principles into engineered systems, replacing more classical engineering solutions in order to achieve a function observed in the natural system. This chapter will focus on mechanism design for bio-inspired robots that replicate key principles from nature with novel engineering solutions. The challenges of biomimetic design include developing a deep understanding of the relevant natural system and translating this understanding into engineering design rules. This often entails the development of novel fabrication and actuation to realize the biomimetic design.

This chapter consists of four sections. In Sect. 23.1, we will define what biomimetic design entails, and contrast biomimetic robots with bio-inspired robots. In Sect. 23.2, we will discuss the fundamental components for developing a biomimetic robot. In Sect. 23.3, we will review detailed biomimetic designs that have been developed for canonical robot locomotion behaviors including flapping-wing flight, jumping, crawling, wall climbing, and swimming. In Sect. 23.4, we will discuss the enabling technologies for these biomimetic designs including material and fabrication.

23.1 **Overview** 544
23.2 **Components of Biomimetic Robot Design** 544
23.3 **Mechanisms** 545
23.3.1 Legged Crawling 545
23.3.2 Worm-Like Crawling 547
23.3.3 Snake Robots 548
23.3.4 Flapping-Wing Flight 550
23.3.5 Wall Climbing 551
23.3.6 Swimming 553
23.3.7 Jumping 555
23.3.8 Gripping and Perching 558
23.4 **Material and Fabrication** 561
23.4.1 Shape Deposition Manufacturing 561
23.4.2 Smart Composite Microstructures 562
23.4.3 Pop-Up Book MEMS 564
23.4.4 Other Fabrication Methods 565
23.5 **Conclusion** 567
Video-References 568
References 570

Biomimetics is a broad field that covers all ranges of robotics including robot structure and mechanics, actuation, perception, and autonomy. This section will focus on robots that mimic structure and movement principles found in nature to perform desired tasks in unstructured environments.

Nature frequently inspires engineers to adopt solutions from biology for application to human challenges. Robots are built to perform certain tasks, and many tasks include moving from one place to another. The various modes of locomotion in nature have inspired robots to mimic these locomotion with a goal to overcome various obstacles in the environment, and move around with the extreme agility similar to that found in nature. Mankind has engineered various modes of transportation on ground, air, and water. On ground, wheeled vehicles are the most popular choice. In air, fixed wing airplanes and helicopters with rotating blades dominate, and in water, ships, and submarines propelled by similar rotating elements are most common. In contrast, nature

has different solutions to locomotion involving moving legs, undulating fins, and flapping wings. Many of these biological locomotion mechanisms have analogies to engineered systems. Instead of using wheels, nature uses legs of different sizes, numbers, and impedances; humans and birds are bipedal, many mammals and reptiles move on four legs, and insects have six legs, and other arthropods have eight or more legs. These legs are coordinated to move on different surfaces in a stable manner. Snakes and worms move around without legs by creating waves with their body. Instead of fixed wings with jets and rotary propellers, birds, bats, and flying insects flap their wings. Fish create undulatory motions with their bodies to swim with agility far beyond conventional boats and submarines. Each of these biological locomotion modes are based on the nature's fundamental actuator: muscle. Muscles create linear motion that is coupled to the structures that generate locomotion. Therefore, these structures are closely related to the morphology of the muscles that move the structures.

23.1 Overview

The main purpose of locomotion in manmade machines is to deliver large payloads across long distances in the shortest possible time or with the minimal energy expenditure. There are roads, airports, shipyards, and other foundations that support this transportation, enabling the structural design of cars, trains, planes, and ships to focus on transportation tasks. On the other hand, locomotion in nature is primarily for survival. Animals and insects have evolved to survive in various environments. Each species uses their mode of locomotion to hunt for food, find mates, and escape danger all in unstructured natural environments. Therefore, the requirements for biological locomotion are much more complicated than human transportation systems.

Biomimetic robots try to mimic the structural characteristics and the principles of movement to be able to move around places where conventional machines or robots are unable to perform as needed. Animals can crawl on a rugged terrain at a high speed, can climb walls without a tether, can fly in cluttered environments and hover and perch as needed.

Depending on the size of the species the optimal mode of locomotion and the underlying structure is different. For example, jumping is found frequently in small insects to escape danger, since their small size makes it hard for them to escape quickly using other forms of locomotion. Jumping by a large animal is different from jumping by small insects like a flea. Larger insects or animals tend to use their legs to run or crawl to escape the danger. Large birds flap their wings at a much lower frequency, and use gliding mode whereas bees and flies beat their wings continuously at a high frequency during flight. The structure is very different, where the wings of a bird have bones, muscle, and feathers versus the muscle-less and lightweight wings of insects. Therefore, when developing biomimetic robots, the size of the target species should be taken into account.

Since nature has evolved to survive in extreme environments, there are many examples of extreme locomotion that are typically not possible with conventional engineering designs. For example, the wall climbing robot Stickybot uses the directional adhesion principle of a gecko to climb up smooth vertical walls. However, directional adhesion alone is not enough. The design has to enable even pressure distribution on the pad to make sure the pads are well in contact with the wall. These small details can be important to performance of biomimetic robots and should be considered carefully.

Since robots are commonly built to perform tasks too tedious or dangerous for humans, it is natural to adopt designs found in nature for functionality where nature has already found a solution. In fact, many robot designs in this handbook, such as legged robots and robot hands and many examples of flying robots, underwater robots, and micro robots, are biomimetic. This chapter will focus on areas of biomimetic robot designs that have been recently expanding. Legged robots and robot hands have already formed large communities and are covered in separate chapters. There are various other forms of locomotion, e.g., crawling, jumping, swimming, and flapping, that are being researched to mimic the principle of movement and the biological structure. The structure and the components of these robots will be reviewed and discussed in this chapter.

23.2 Components of Biomimetic Robot Design

Biomimetic robot design begins with an understanding of the underlying principles at work in a natural system. For the design of a biomimetic robot, the components used in building the robot are important; the structural

design is constrained or enabled by the actuator system, the materials and the fabrication methods used to build the robot.

Understanding of the principles of movement found in nature initiates the design of biomimetic robots. This can be studied by observing the kinematics and measuring forces in moving animals, or by understanding and modeling the dynamics. Substantial research on biological locomotion analyzes the kinematics of the animal. However, it is important to understand the dynamics behind the kinematics to be able to implement the dynamic principles into the robot design, which will enable the robot to perform like an animal, instead of just looking like one. *Full* et al. proposed clock-driven, mechanically self-stabilizing, compliant sprawled-posture mechanics to explain a cockroach racing seemingly effortlessly over a rough surface [23.1, 2]. *Ellington* built a system that can measure the force generated by flapping wings to understand the thrust generated by insect wings [23.3]. These studies in biology have influenced robotics researchers, and inspired them to build hexapod and flapping-wing robots.

To implement biological principles in engineered systems, components that can create performance similar to that of nature is required. Nature's biological structures are composed of various materials, e.g., tissues, bones, cuticles, flesh, and feathers. These materials are replaced with engineered materials, e.g., metal, plastics, composites, and polymers. New engineered materials are being developed that exhibit similar properties to natural materials, which will enable life-like robots. However, the performance of the biomimetic robot can be similar to nature without actually mimicking the exact structure. Therefore, it is important to decide to what level biomimicry is required. Large robots tend to be built with conventional mechanical components such as motors, joints, and linkages made of metal. The challenge is in building meso-scale robots, where the conventional mechanical components become ineffective, due to friction and other inefficiencies. As the robot becomes smaller, it is beneficial to mimic not just the kinematics seen in nature, but also the structure at the component level. Another challenge for small scales is actuation – the actuator must be as effective as the muscle even at small scales.

A number of new manufacturing methods have become available which also enable new designs that were not previously possible by more classical machining methods and *nuts-and-bolts* assembly techniques. Examples include shape deposition manufacturing and smart composite microstructures. In many cases, these new fabrication processes facilitate the construction of novel biomimetic robots beyond what was possible by conventional means.

Levels of mimicry can vary depending on the components used in the design. The robot can simply look like an animal, but not perform similarly to the mimicked animal. On the other hand, once the principle is mimicked the robot does not have to look like the animal, but it can still perform a task in a similar manner as the animal. The different ways of mimicking allows each robot to have its own characteristic. For example, a flea uses what is called a torque reversal for jumping, which is a unique method of storing and releasing large amount of energy. In nature, this method is possible with muscles and a lightweight-legged structure. To mimic the flea's locomotion, we need an actuator that is comparable to muscles and a fabrication method that allows us to build a small-scale rigid structure.

23.3 Mechanisms

Developing multilegged robot has been achieved by using specific design which enables some function or biomimetic model inspired by multilegged insects. Especially, the ability of a cockroach enabling to run on a rough surface at a high speed has inspired to develop a series of multilegged robots. They are capable of maintaining stability during locomotion at a high speed (relative to their body length).

23.3.1 Legged Crawling

Some types of cockroaches can achieve speed up to 50 body length per second and can crawl on uneven terrain, overcoming obstacles far higher than their height [23.2]. RHex (Fig 23.1a) [23.4] is one of the first robots to implement cockroach-like characteristics. It is a hexapod crawling robot with *C*-shaped legs, so it is suitable for walking on uneven terrain or large obstacles, as shown in VIDEO 400. Mini-Whegs (Fig 23.1b) [23.5] also have unusual wheel with three spokes. Because of the wheel-spoke structure, the gait passively adapts to the terrain similar to climbing cockroaches, as shown in VIDEO 401 [23.15]. Sprawlita (Fig 23.1c) [23.6] is a hexapod crawling robot that uses pneumatic actuators on each leg and passive rotary joints so that it can achieve dynamic stability. It weighs 27 kg and its speed is over 3 body length/s or 550 mm/s. iSprawl (Fig 23.1d) [23.7] is hexapod crawling robot that uses extension of each legs during crawling. This robot is driven by electric motors and

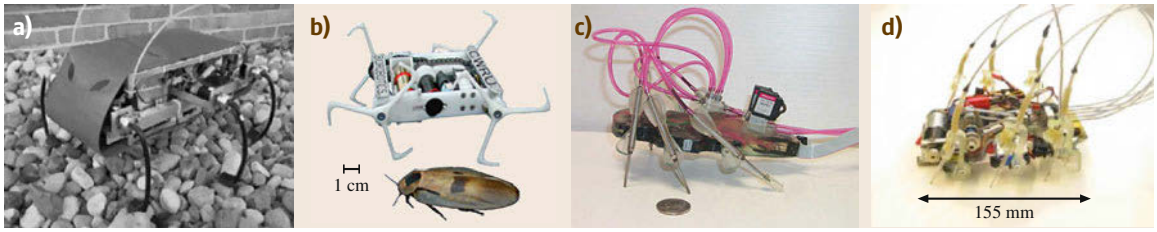


Fig.23.1a–d Crawling robots inspired by cockroach **(a)** RHex (after [23.4]), **(b)** Mini-Whegs (after [23.5]), **(c)** Sprawlita (after [23.6]), **(d)** iSprawl (after [23.7])

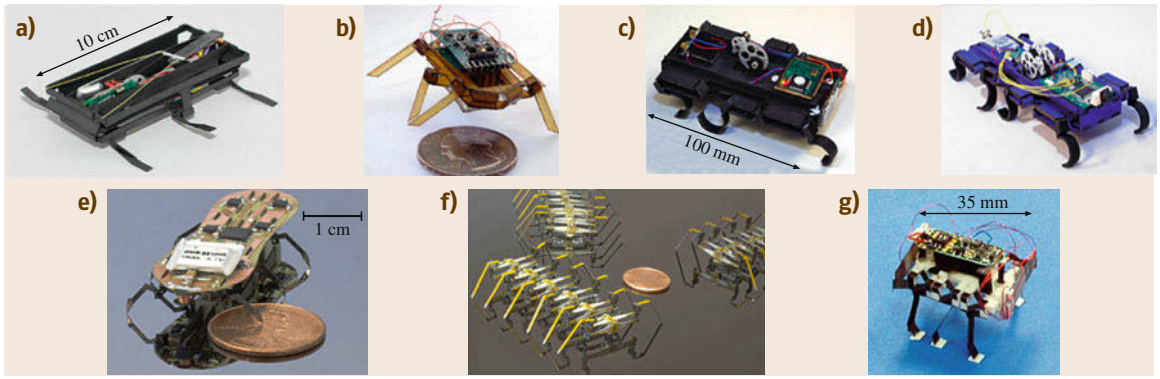


Fig.23.2a–g Crawling robots inspired by cockroach. **(a)** DASH (after [23.8]), **(b)** RoACH (after [23.9]), **(c)** DynaRoACH (after [23.10]), **(d)** OctoRoACH (after [23.11]), **(e)** HAMR3 (after [23.12]), **(f)** centipede-like modular robot (after [23.13]), **(g)** crawling robot through the integration of microrobot technologies (after [23.14])

flexible push–pull cables. Rotation of the motor causes the legs to extend or contract. It weighs 300 g and is able to crawl 15 body length/s or 2.3 m/s, as can be seen in VIDEO 403.

Cockroach-based designs would improve the performance of millimeter or centimeter scale crawling robots which face inefficiency with conventional mechanisms. DASH (Fig 23.2a) [23.8] is a hexapedal crawling robot fabricated using SCM (Smart composite manufacturing) process. This robot has only one electric motor coupled to a four-bar linkage pushing the legs in an elliptical crawling motion, as shown in VIDEO 405. DASH weighs 16.2 g and achieves speed up to 15 body length/s or 1.5 m/s. RoACH (Fig 23.2b) [23.9] is a hexapod crawling robot that imitate cockroach's alternating tripod gait. A typical cockroach gait involves the ipsilateral front leg, hind leg, and contralateral middle leg moving simultaneously. Two sets of three legs tread on a surface in turn – generating the *alternating tripod gait*. RoACH uses two shape memory alloy (SMA) wire actuators to contract the body in two orthogonal directions. Contraction of the two actuators results in motion of legs through a four-bar linkage. The legs repeatedly operate swing and stance motions according to the sequential stimulation of the two actuators and it can be seen in VIDEO 286. It weighs 2.4 g and is able to crawl 1 body length/s or 3 cm/s. DynaRoACH (Fig 23.2c) [23.10] has six legs driven by one DC motor. It has passive dynamics similar to RoACH to achieve better locomotion performance. Lift motion is achieved by a slider crank mechanism and swing motion by a four-bar mechanism. Like RHex, it uses *C*-shape legs so that the robot has lower vertical stiffness, lateral collapsibility for obstacle climbing, and more distributed ground contact. It weigh 24 g and is 100 mm long and is capable of speeds up to 14 body length/s or 1.4 m/s. OctoRoACH (Fig 23.2d) [23.11] is quite similar to DynaRoACH, but it has two motors driv-

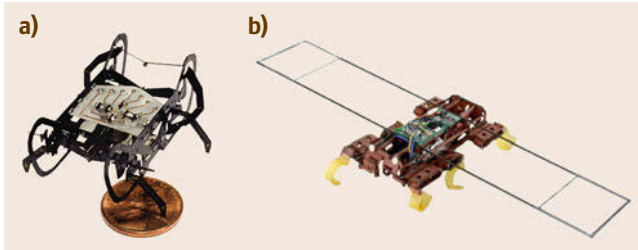


Fig.23.3a,b The latest version of crawling robots which shows improved performance **(a)** HAMRV (after [23.16]), **(b)** VelociRoACH (after [23.17])

ing the legs on each side. It has eight legs so that it maximizes pitch stability using two motors. OctoRoACH weigh 35 g and is 100 mm long. HAMR3 (Fig 23.2e) [23.12] uses nine piezoelectric actuators. Each leg performs swing and lift motion through two decoupled piezoelectric actuator through four-bar slider-crank mechanisms and spherical five-bar mechanisms (VIDEO 406). This robot weighs 1.7 g and has speeds up to 0.9 body length/s or 4.2 cm/s (4.7 cm long robot). Especially, the HAMR is manufactured using a method inspired by pop-up books that enables fast and repeatable assembly. Multilegged robots inspired by centipede using mechanism similar to alternative tripod gaits is also developed (Fig 23.2f) [23.13]. Its several gait patterns that differ in gait frequency and phase are described in VIDEO 407. Using microrobot technologies developed until 2006, integrated structure was developed although the structure was not yet tested (Fig 23.2g) [23.14].

With more experiments and research about motion of cockroaches, the cockroaches like robots can extremely improve their performance by revision using biomimetic model. HAMRV, which is the most recent version of HAMR [23.16], can move 10.1 body length/s (44.2 cm/s) which is remarkably improved compared to preceded versions are capable of running at 0.9 body length/s. It is even capable of maneuverability and control at both low and high speeds. VelociRoACH (Fig 23.3b) [23.17] which is the latest version of RoACH can run 2.7 m/s extremely higher speed relative to previous version (VIDEO 408).

23.3.2 Worm-Like Crawling

Worm-like crawling motion can be separated into two categories: peristaltic crawling and two-anchor crawling. Worms exhibiting peristaltic crawling locomotion – such as earth worms – can move through small tunnels with limited space. Therefore, mimicry of this locomotion imposes similar characteristics to the robot and it has the potential to be used in small and harsh environment such as collapsed disaster site or inside a pipe line. which A schematic of peristaltic locomotion is shown in Fig. 23.4. By sequentially changing the volume of the body, the whole structure generates the moving motion.

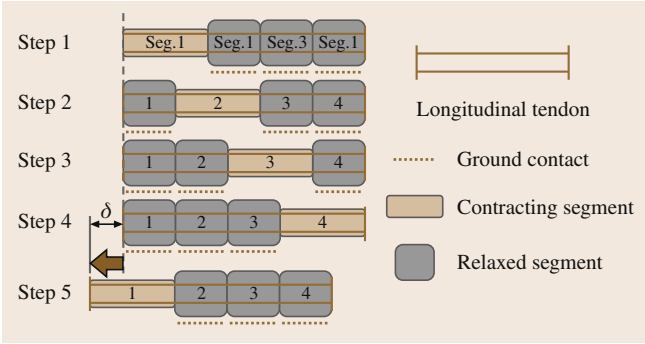


Fig. 23.4 Peristaltic locomotion (after [23.19])

The key design issue in mimicking peristaltic motion is how to create sequential volume change. Many researchers have tried various creative methods to solve this problem. *Boxerbaum* et al. built a robot with a mesh structure, and using a single motor and wire, made a partial volume change of the robot which realized a crawling motion (Fig. 23.5a) [23.18]. *Seok* et al. used a SMA coil spring actuator to change the segmented volume, and also used a mesh tube as the body structure (Fig. 23.5b) [23.19]. *Menciassi* et al. also used a SMA coil spring actuator, but they used a micro hook to enhance the friction force (Fig. 23.5c) [23.20].

Two-anchor crawling is a locomotion method used by inchworms. This locomotion mode is not fast, but it can overcome nearly any complicated topology. With an appropriate gripping method, it not only can climb vertical walls, but also can cross gaps. There are two key design issues for generating a two-anchor crawling motion: the first is how to change the shape of the waist and the second is how to anchor and unanchor to the surface. *Kotay* and *Rus* simply used an electric motor to articulate the waist motion and used an electromagnetic pad as the anchoring method (Fig. 23.6a) [23.21]. Using an electromagnetic pad, the robot can climb a steel structure. *Cheng* et al. used a tendon-driven mechanism with a compressible body and an anisotropic friction pad to generate motion (Fig. 23.6b) [23.22]. Using symmetrical or asymmetrical winding of the wire attached on the both side, the robot can make forward or steering movements. *Koh* and *Cho* used SMA coil spring actuators to control the waist motion (Fig. 23.6d) [23.23].

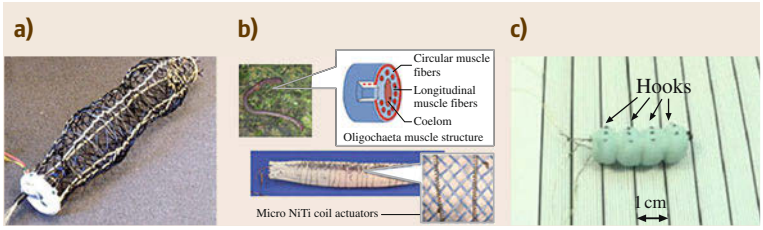


Fig. 23.5 **(a)** Robot with peristaltic motion (after [23.18]). **(b)** Meshworm robot (after [23.19]). **(c)** Biomimetic miniature robotic crawler (after [23.20])

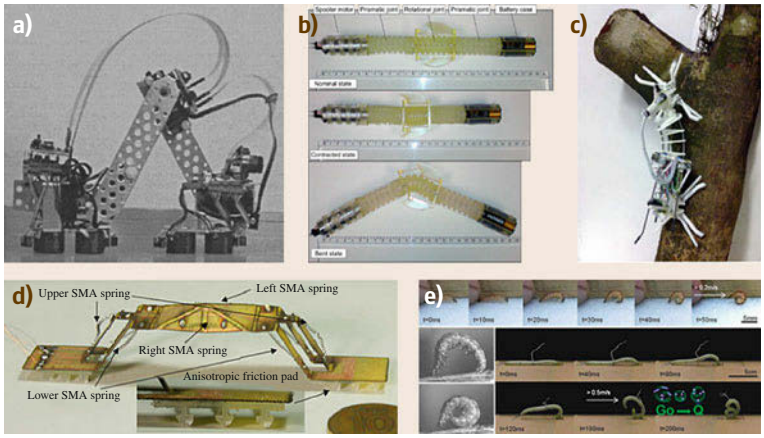


Fig.23.6a–e Two anchor crawling robot. **(a)** The inchworm robot (after [23.21]). **(b)** The soft mobile robot with thermally activated joint (after [23.22]). **(c)** Treebot (after [23.24]). **(d)** Omega shaped inchworm inspired crawling robot (after [23.23]) **(e)** GoQBot (after [23.25])

The body of the robot is made by a single sheet design with glass fiber composite, and a folding pattern designed to enable steering motion even though the robot was built by a single sheet. *Lin* et al. realized a robot with two-anchoring motion, but he also added a rolling locomotion to solve the speed limitation of previous two-anchor motion demonstrations (Fig. 23.6e) [23.25]. *Lam* and *Xu* realized another method to generate the waist motion. The robot used a backbone rod coupled to an electrical motor. By controlling the length of the backbone using the motor, the robot can control the position of the anchoring point (Fig. 23.6c) [23.24].

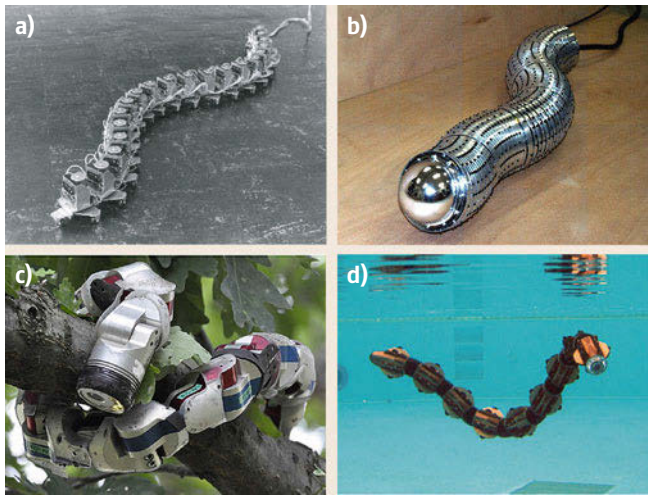


Fig.23.7a–d Three snake-like robots. **(a)** AMC-III, Shigeo Hirose, Fukushima Robotics Lab, Tokyo Institute of Technology (after [23.26]), **(b)** Slim Slime Robot II (SSR-2), Shigeo Hirose, Fukushima Robotics Lab, Tokyo Institute of Technology (after [23.27]), **(c)** Modular snake robot, Howie Choset, Biorobotics Lab, Carnegie Mellon University (after [23.28]), **(d)** AMC-R5, Shigeo Hirose, Fukushima Robotics Lab, Tokyo Institute of Technology (after [23.29])

23.3.3 Snake Robots

Studies on the locomotion of snakes began in middle of the 20th century [23.30–32]. Snakes are limbless, slender, and flexible [23.33]. Their locomotion gives them adaptability and mobility through land, uneven ground, narrow channel, pipes, and even water, and even flying between trees [23.34, 35]. An advantage of snake-like locomotion is the great versatility and freedom of movement with numerous degrees of freedom [23.36]. Additionally, snake locomotion could be efficient compared to legged animals, because there is no lifting of the center of gravity or limb acceleration [23.37]. In 1970s, *Hirose* developed a continuous locomotion model and a snake-like robot called the *Active Cord Mechanism* (ACM) [23.38]. After the Hirose's ACM robot, snake-like robots have been widely studied. In 1972, ACM-III (Fig. 23.6) was developed and it was the first robot that mimics the serpentine motion of real snakes [23.38]. The recent versions of ACM are in VIDEO 397.

Locomotion of snake-like robots can be categorized into the following different types: serpentine motion, sinus lifting, pedal wave, side-winding, spiral swimming, lateral rolling, lateral walking, mixture lean serpentine, and lift rolling motions.

The first generation of snake-like robots could only achieve motion on planar surfaces. These designs quickly evolved and current snake-like robots can go upward within narrow pipes and can climb and hold on to trees like VIDEO 393 [23.39]. To facilitate traversing large obstacles, some robots have added actuated articulation between each joint [23.40]. In addition, there are some snake-like robots that can swim in water. With a firmly waterproofed body, these robots can swim with spiral and sinusoidal locomotion patterns [23.38].

Today, mechanism design of snake-like robots can be classified with following five different types: active

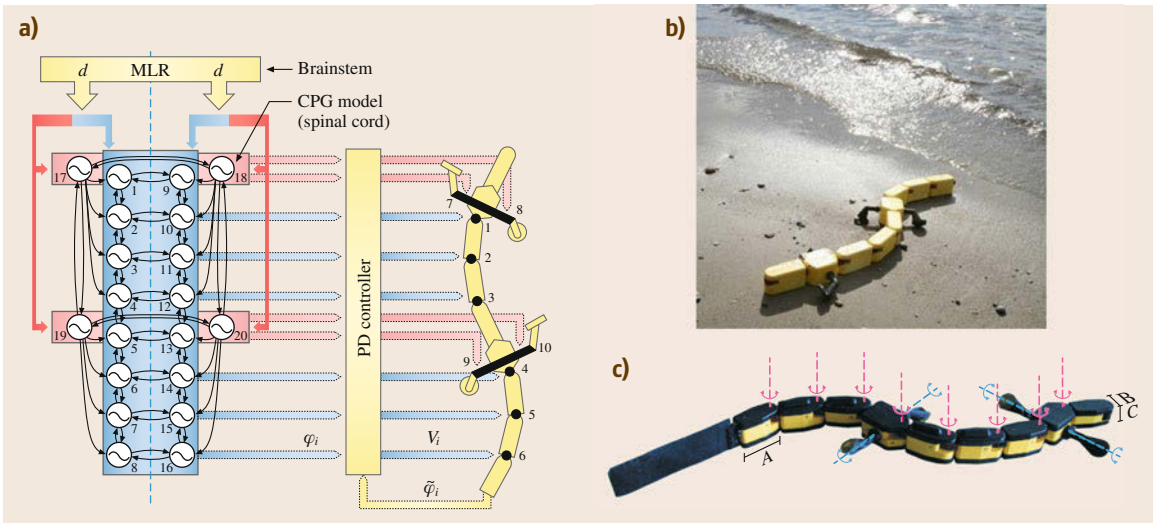


Fig. 23.8 **(a)** Configuration of the CPG model (after [23.41]). **(b)** Salamandra robotica I, Auke Jan Ijspeert, Biorobotics Lab, Ecole Polytechnique Federale de Lausanne (after [23.41]). **(c)** Salamandra robotica II, Auke Jan Ijspeert, Biorobotics Lab, Ecole Polytechnique Federale de Lausanne (after [23.42])

bending joint type, active bending and elongation joint type, active bending joint and active wheel type, passive bending joint and active wheel type, and active bending joint, and active crawler type [23.43]. Each of these types of snake-like robots consist of a number of serially connected joints. Therefore, snake-like robots are easy to modularize with their joints [23.43].

Most snake-like robots are equipped wheels that are actively or passively driven. Recently, wheelless snake-like robots have been studied [23.32]. These robots move with undulatory motion, especially lateral undulation that can be observed in real snakes. Some snake-like robots are actuated with smart materials, such as shape memory alloys and IPMCs, rather than motors [23.44]. Snake-like robots roll to avoid obstacles and interact with environment. They make waveforms with their body for propulsion. Therefore, it is important to define the configuration of the body of the robots. Measuring yaw (pitch) and roll angles are important for controlling the snake robots. Tilt sensors, accelerometers, gyroscope, and joint angle sensors are typically leveraged to control the robot [23.37]. Tactile sensors attached to the contact area or outside of the body of the snake-like robots have been used to measure surface contact forces at each joint, providing more information to the controller. Additionally, measuring contact forces could be useful for active and adaptive grasping of the snake-like robot [23.45].

However, due to the high degrees of freedom of the snake-like robots, designing controllers is not easy even for flat surface locomotion. Efforts without complex sensors and controllers are in VIDEO 392. Because of this, contrasting with the advantages of locomotion of real snakes – which can move in uneven environments – most existing snake-like robots are developed based on flat surface movement [23.46].

A major motivating application for snake-like robots is the exploration of hazardous environment that are inaccessible to humans. In particular, industrial inspection of pipes and ventilation tubes, and chemical channels are key operating environments. There is also the potential for snake-like robots as medical devices, such as minimal invasive surgery device and laparoscopy and endoscopy [23.37]. For these applications, snake-like robots require their outer skin to hermetically seal the internal components [23.40].

There are numerous challenges for snake-like robots. For greater reliability, robustness and controllability, the mechanisms and configuration of the snake-like robots could be simplified. To use the snake-like robots for exploration and inspection applications, routing of external wires for electronics and power to the robot is an important consideration. Finally, with a large number of degrees of freedom typical of snake-like robots, designing efficient control strategy is large issue [23.37].

Similar to undulatory locomotion of snakes, body of salamanders makes *S*-shape standing waves. They are capable of rapidly switching between swimming and walking locomotion. Their locomotion in aquatic and terrestrial environments is generated by a central pattern generation (CPG) and stimulation of a mesencephalic locomotor region (MLR) located in the midbrain (Fig. 23.7 and VIDEO 395) [23.41]. There were

some efforts to produce similar swimming and walking gaits to real salamander with robotic salamanders. The salamander robots with the mathematical CPG model, DC motors, and oscillators could produce similar kinematics to real salamanders [23.42] (Fig. 23.8).

23.3.4 Flapping-Wing Flight

Flapping-wing flight is a common inspiration for biomimetic aerial robots. This is due to the agility of natural flyers such as birds, bats, and insects. Advances in the understanding of the aerodynamics of flapping-wing based on hydrodynamic theories and experimental results have provided insights into thrust production in flapping-wing animals [23.47]. From observation of the flapping motion exhibited by insect flight, the translational, and rotational motions of the wing produce lift forces at high angles of attack – beyond what is typical for fixed wing aircraft. The formation of a large vortex at the leading edge of the wing and the recapturing of shed vortices by properly timing of the swing enhance the resultant lift force [23.48]. Figure 23.9 shows the representative motion of an insect wing during flight and the vector formation of hydrodynamic forces. Such characteristics learned from nature inspires the design of wing-driving systems in flapping-wing flying robots.

Most flapping mechanisms are constructed from electromagnetic rotary motors driving crank-rocker linkages to flap the wings. Examples include the DelFly II [23.49], Robot dragonfly [23.50] and Nano Hummingbird [23.51] as shown in Fig. 23.10. As can be seen in VIDEO 402, DelFly II employs four-wing morphology where two wings on each side perform a *clapping* motion during each period. This contributes to lower power consumption and the low rocking amplitude of the fuselage. The DelFly II model used a crank mechanism such that the gear axis is perpendicular to the flying direction, for overcoming phase differences between the two sets of wings and this difference induced rotational movement on the fuselage [23.49]. In related work, the Robot Dragonfly [23.50] is inspired from a dragonfly's hovering capabilities and utilizes a tandem wing model. Each transmission link is attached to gear mechanisms that can make wing motions similar to those of a dragonfly. The Nano Hummingbird uses single pair of flapping wings as with real hummingbirds. The flapping mechanism is a dual lever, string-based flapping mechanism. Wing rotation modulation for control of the wing attack angle is achieved by the two adjustable *stops* which limit how far each wing can rotate [23.51]. Alternative mechanisms use oscillating actuators and flexure-based transmissions systems (these mechanisms are discussed in Sect. 23.4.2).

As the robot scale decreases, previous flapping mechanisms have been difficult to adapt due to manufacturing challenges and the physics of scaling. The Harvard robotic fly shows promising fabrication and design processes to build the small scale flapping air vehicle as shown in Fig. 23.11 [23.52]. Previous versions of the Harvard robotic fly had three degrees of freedom, only one of which was actuated. The angle of attach of two the wings are passively controlled by a compliant flexure joint connected to the transmission. The wing beat frequency is tuned at 110 Hz in resonance. Through advances in meso-scale manufacturing methods, the capabilities of the robotic fly have been dramatically extended and unconstrained flight

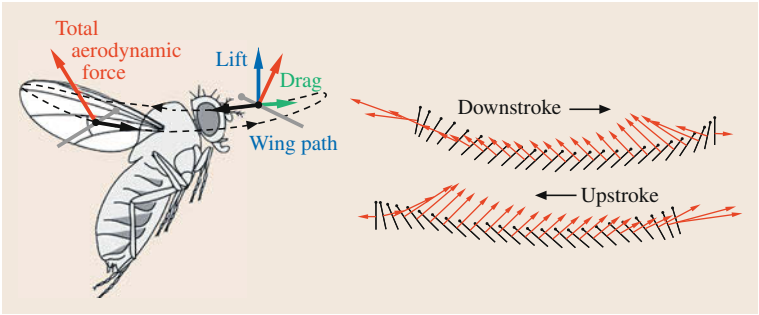


Fig. 23.9 Diagram of wing motion indicating magnitude and orientation of the aerodynamic force (after [23.48])

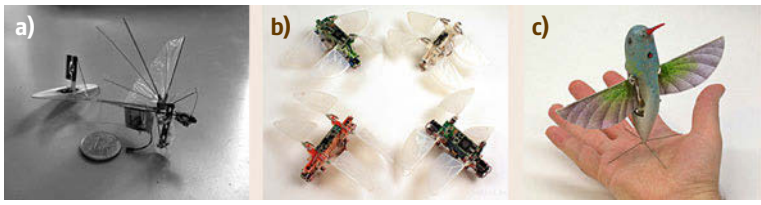


Fig. 23.10 **(a)** Delfy II (after [23.49]), **(b)** Robot Dragonfly [23.50], **(c)** Nano Hummingbird (after [23.51])

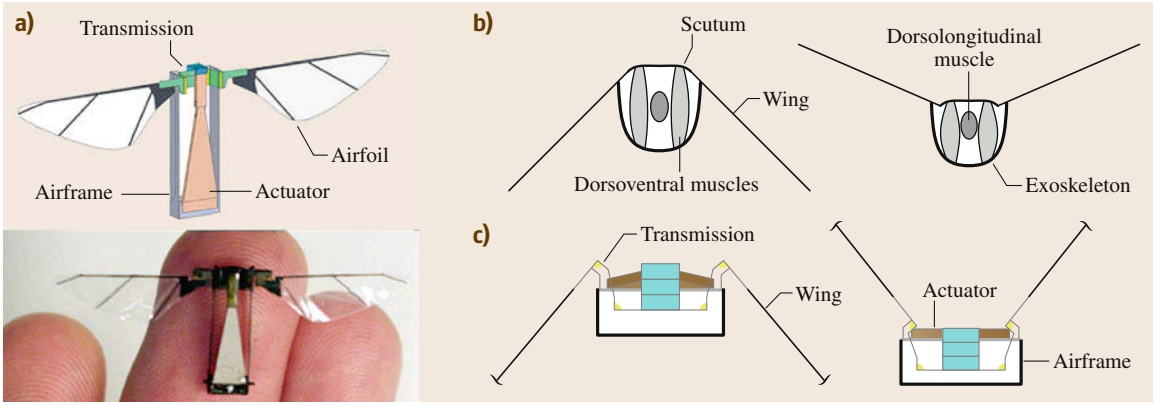


Fig. 23.11 (a) Harvard robotic fly, (b) Illustrations of thorax mechanisms, (c) and transmission (after [23.52])

has been demonstrated like VIDEO 399 in an 80 mg robot as discussed in Sect. 23.4.2.

Flapping-wing locomotion can also be extended to multimodal locomotion in combination with other mechanisms. In the DASH+Wings shown in Fig. 23.12 [23.53], the combination of wing flapping and crawling compliments each other for improving agility and stability. This hybrid robot improves performance of the maximum horizontal running speed in a factor of two and the maximum climbing incline angle by a factor of three.

23.3.5 Wall Climbing

Climbing and maneuvering on vertical surfaces present a difficult challenge. However, this locomotion mode is needed in many areas such as shipping, construction, and terrestrial locomotion in natural environment. Early attempts involved the use of suction cups, magnets, or pressure-sensitive adhesives to implement climbing. More recently, claw, spines and sticky pads inspired by nature have been used. Climbing insects and animals inspired many researchers. Insects and reptiles employ small spines that catch on fine asperities. Geckos and some spiders employ large numbers of very fine hairs that achieve adhesion based on van der Waals interaction.

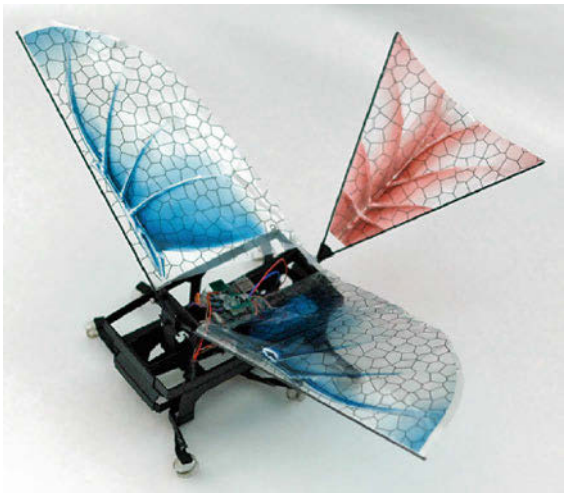


Fig. 23.12 DASH+Wings (after [23.53])

Early in the 1990s, nonbiomimetic wall climbing robots, i. e., the Ninja-1, RAMR, and Alicia were developed using suction cup. Ninja-1 attaches to a wall making use of a suction mechanism (Fig. 23.13a). The main mechanism consists of a three-dimensional (3-D) parallel link, a conduit-wire-driven parallelogram, and valve-regulated multiple sucker that enabled the robot to attach the surface with grooves [23.54]. RAMR used underactuation to remove the redundant actuators to drive the small two-legged robot [23.59]. The Alicia robot was developed for a variety of applications such as maintenance, building inspection, and safety in process and construction industries. An aspirator is used to depressurize a suction cup, so the whole robot can adhere to the wall like a standard suction cup. The Alicia3 robot use three of the Alicia II modules, allowing the whole system to better deal with obstacles on the target surface [23.60]. REST is an exceptional case that applies electromagnets instead of suction cup. It climbed only ferromagnetic wall using electromagnetic four legs with 12-DOF [23.61].

The effective wall climbing mechanisms of animals and insects have inspired development of biomimetic wall climbing robots. Typical robots utilizing bio-inspired spines found in climbing insects and cockroach are Spinybot and RiSE. Spinybot in VIDEO 388 climbs hard vertical surfaces including concrete, brick, stucco, and masonry with compliant microspine arrays (Fig. 23.13b). It can exploit small asperities (bumps or pits) on the surface. The sequence of motions is

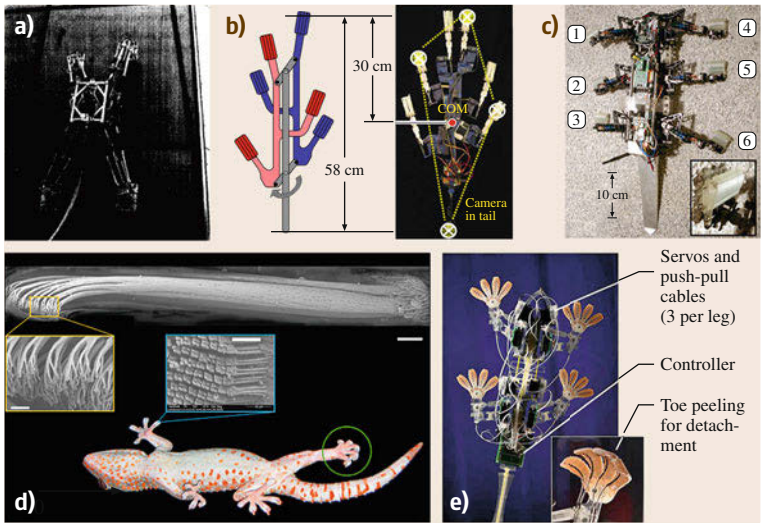


Fig. 23.13 **(a)** Ninja-1 (after [23.54]), **(b)** Spinybot II (after [23.55]), **(c)** RiSE (after [23.56]), **(d)** Gecko adhesive system (after [23.57]), **(e)** Stickybot (after [23.58])

accomplished using an under-actuated mechanism consisting of a single rotary RC servo motor and several spines independently engaging asperities on the concrete surface [23.55]. RiSE is a hexapod robot capable of locomotion on ground and vertical surfaces such as brick, stucco, crushed stone, and wood, as shown in VIDEO 390 (Fig. 23.13c) [23.56]. To climb a vertical wall, it uses microspines inspired by cockroach's tarsus structure. In addition, it held its center of mass close to surface to minimize the pitch-back moment. RiSE also employed a static tail to reduce disparity from the pull-in forces experienced by the different legs [23.56, 62].

Insects and geckos can provide inspiration for novel adhesive technology and for the locomotion mechanisms employed during climbing. Geckos are able to climb rapidly up smooth vertical surfaces and biologists reveal that a gecko's foot has nearly five hundred thousand keratinous hairs or setae. And measured adhesive force values show that individual seta operate by van der Waals forces. The gecko's toe uncurling and peeling suggests that two aspects of setal function increase effectiveness [23.63]. The subsequent study shows that the linear relation between adhesion and shear force is consistent with a critical angle of release in live gecko toes and isolated setal arrays (Fig. 23.13d). And the frictional adhesion model provides an explanation for the very low detachment forces observed in climbing geckos that does not depend on toe peeling [23.57].

Stickybot, Mini-Whegs, Geckobot, and Waalbot are prototypical robots that leverage biomimetic dry adhesives. Stickybot climbs smooth vertical surfaces such as glass, plastic, and ceramic tile at 4 cm/s, as shown in VIDEO 389 (Fig. 23.13e). The robot employs several design principles adapted from the gecko including a hierarchy of compliant structures, directional adhesion. The undersides of Stickybot's toes are covered with arrays of small, angled polymer stalks. They readily adhere when pulled tangentially from the tips of the toes toward the ankles. When pulled in the opposite direction, they release [23.58]. Mini-Whegs uses wheel-legs with compliant, adhesive feet for climbing. The foot motion mimics the foot kinematics of insects, in order to test new bio-inspired adhesive technologies and novel, reusable insect-inspired polymer (polyvinylsiloxane) [23.67]. Geckobot has kinematics similar to a gecko's climbing gait. It uses a novel peeling mechanism of the elastomer adhesive pads, steering mechanisms and an active tail for robust and agile climbing [23.68, 69]. Waalbot used two actuated legs with rotary motion and two passive revolute joints at each foot. The robot has ability to climb on nonsmooth surfaces as well as on inverted smooth surfaces using gecko-like fibrillar adhesives and passive peeling. It is also capable of plane-to-plane transitions and steering to avoid obstacles [23.70].

Different approaches for climbing include electroadhesion. Electroadhesives use a novel clamping technology called compliant electroadhesion – a form of electrically controllable adhesion. This involves inducing electrostatic charges on a wall substrate using a power supply connected to compliant pads situated on the moving robot. This generates high clamping forces that are around 0.2–1.4 N supported for a one square centimeter clamp area, depending on the substrate. Regarding power considerations for electroadhesion, assuming 50% conversion efficiency, in the worst-case scenario, two AAA batteries weighing 7.6 g each can hold up a robot in *perch* mode for almost one year. Electroadhesion combined with a conventional wheeled robot results in inchworm-style wall climbing [23.79].

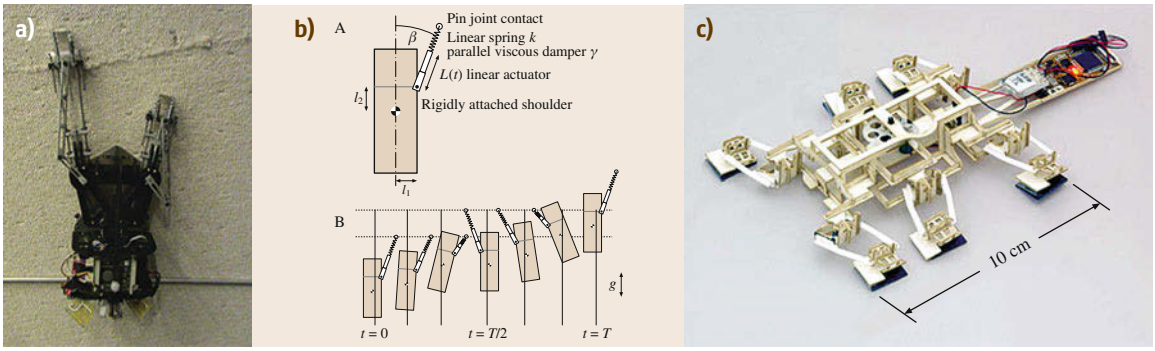


Fig. 23.14 **(a)** DynoClimber (after [23.64]). **(b)** A dynamic template for climbing (after [23.65]). **(c)** CLASH with gecko pad (after [23.66])

Dynamic wall climbing is the next challenge for wall climbing robots because previous robots were slow and in most instances restricted to targeted surfaces. For dynamical climbing originating in biology, pendulous climbing model was proposed (Fig. 23.14b). This model abstracts remarkable similarities in dynamic wall scaling behaviour exhibited by radically different animal species such as cockroaches and geckos [23.65]. The findings suggest that animals employ large lateral in-pulling forces and body rotations to achieve fast, self-stabilizing gaits. DynoClimber displays the feasibility of adapting the dynamics to robot that runs vertically upward (Fig. 23.14a). A novel bi-pedal dynamic climber can scale a vertical wall fast accompanying while achieving dynamic stability. For dynamic climbing, this robot consists of a DC motor, a crank slider mechanism and passive-wrist springs so its climbing at speeds 0.67 m/s (1.5 body lengths/s) [23.64]. The climbing robot CLASH has modified DASH platform but it actuates in horizontal direction to reduce height (7 mm from robot bottom). One of the key points is passive foot mechanism. When climbing upward, the foot hangs its spines on the surface and then retracts passively. This increases the shear and normal forces and enables climbing on loose cloth at 15 cm/s speed, as shown in VIDEO 391 [23.80]. The next version of CLASH has a foot that consists of an 18×15 mm pad of microfabricated PDMS (polydimethylsiloxane) ridges inspired by gecko feet (Fig. 23.14c). The ankle is an isosceles-trapezoid four-bar that creates a remote center-of-motion. This mechanism allows the foot to make coplanar contact with the surface and reduces roll peeling moments [23.66].

23.3.6 Swimming

Underwater vehicles have been made to achieve marine explorations, surveillance, and environmental monitoring. Most underwater vehicles employ propellers for propulsion and these vehicles have shown great performance with respect to the cost of transport. However, efficiency and manoeuvrability in confined areas is problematic for most surface or underwater vehicles. Moreover, propeller-driven vehicles risk tangling when moving thourgh environments with debris and vegetation. To resolve these issues, researchers have tried to replace the conventional rotary propellers with undulatory movement inspired by fish (Fig. 23.15).

The undulatory movement of fish provides two main advantages – manoeuvrability in confined areas and high propulsive efficiency. The main difference between existing propeller and undulatory movement is turning radius and speed. Fish can turn with a radius 1/10 of their body length, while propeller-driven ships require a much larger radius. Accordingly, the turning speed of fish is much faster than ships. Beyond manoeuvrability, the driving efficiency in biological swimmers also show more improvement over man-made systems [23.81].

To achieve fish-like swimming motion, various mechanisms have been employed such as linkage mechanisms and compliant mechanisms. *Barrett* first proposed a RoboTuna by using six servo motors and eight linkages [23.71]. *Morikawa* et al. built a robotic fish mimicking the caudal musculo-skeletal structure of a tuna with two rubber pneumatic artificial muscles and a multijoint bending mechanism [23.82]. A robotic dolphin was designed with four links and six servo motors to mimic the dorsoventral movement of a real dolphin [23.83]. Low developed a fish robot to generate arbitrary undulating waveforms, by connecting ten servo motors in series by linking them with sliders [23.74]. *Liu* and *Hu* developed a robotic fish mimicking the body motion of carangiform fish by using three servo motors on each joint, as shown in VIDEO 431 [23.72]. Yang et al. presented Ichthus V5.5 by using 3-DOF serial link-mechanism with servo motors on each joint for propulsion, as shown in VIDEO 432. Ichthus V5.5 has

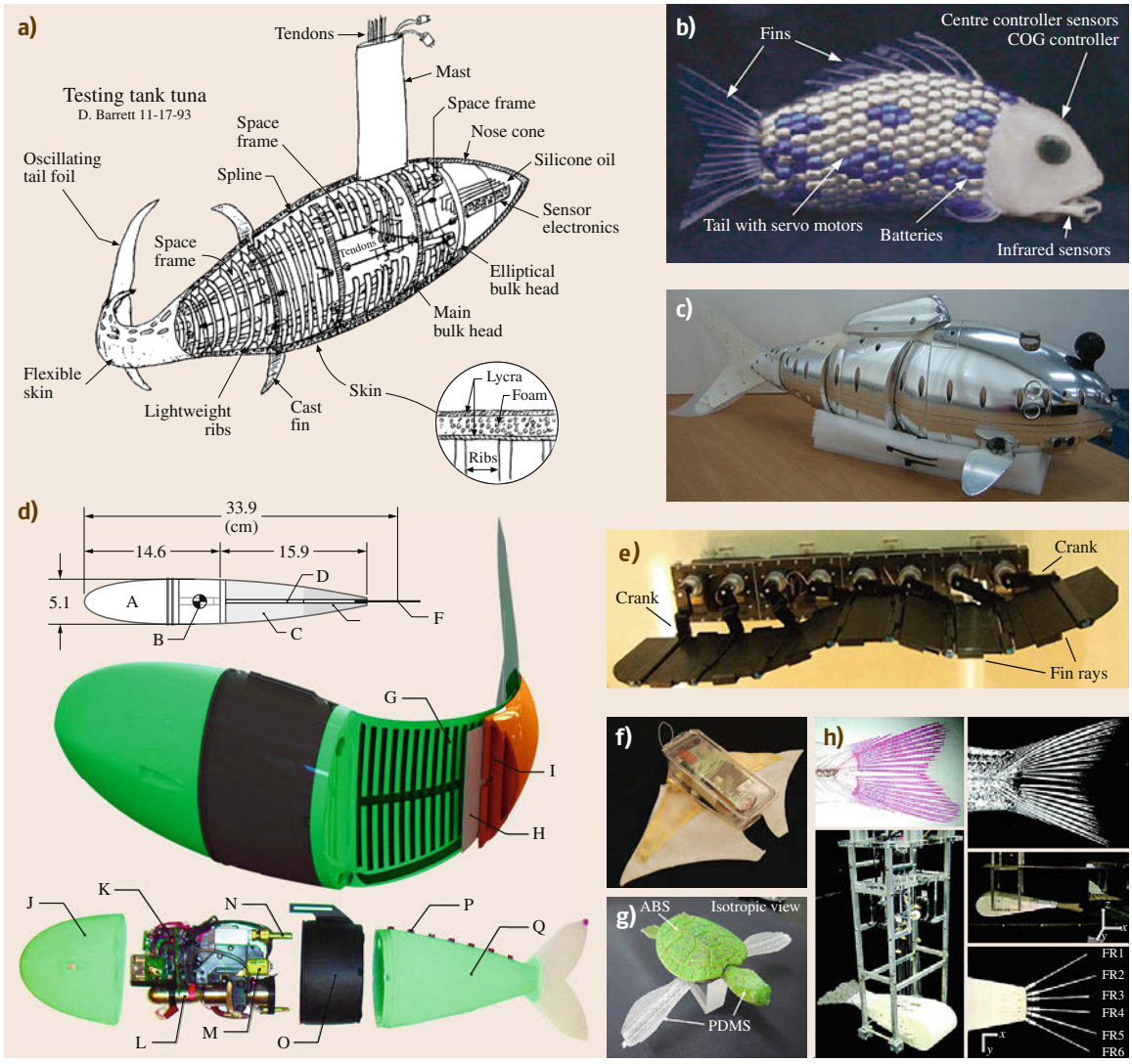


Fig.23.15a–h Design of robotic fin **(a)** Robo Tuna (after [23.71]), **(b)** G9 serise robotic fish (after [23.72]), **(c)** Ichthus (after [23.73]), **(d)** Undulating robotic fin rays (after [23.74]), **(e)** Soft robotic fish (after [23.75]), **(f)** Robotic mata ray (after [23.76]), **(g)** Turtle-like swimming robot (after [23.77]), **(h)** Robotic fin rays (after [23.78])

several sensors to navigate autonomously in the real environment such as rivers [23.73].

Beyond linkage mechanisms, several researchers have employed compliant materials in their designs to make the undulatory motion without complicated linkage structures. *Salumäe* and *Kruusmaa* implemented swimming kinematics of a trout by simply adjusting the compliance of a flexible fin with a single actuator [23.84]. *Alvarado* and *Youcef-Toumi* designed a robotic fish with a simple and robust mechanism, using a compliant body that was approximated by a continuous cantilever beam to generate a fish-like oscillating motions [23.85]. This simple design achieved biomimetic locomotion using only one servomotor, while most other robotic fish use several motors to achieve biomimetic modes of swimming. *Marchese* et al. employed a compliant body with embedded actuators and used a novel fluidic actuation system that drives body motion, as shown in VIDEO 433 [23.75]. Park et al. presented a guideline for optimizing the fin to maximize the thrust generated by a compliant fin. The *half-pi phase delay* condition describes the condition that the thrust is maximized regardless of the shape of the fin, driving frequency, and amplitude. They also presented a variable-stiffness flapping mechanism to improve the performance of a compliant fin while

Fig.23.16a,b Escapement cam mechanism **(a)** Grillo (Ver.1) (after [23.86]) and **(b)** 7 g jumping robot (after [23.87])

the operating conditions change [23.88]. Tendons were used to vary the stiffness and the attachment point is determined based on the anatomy of a dolphin's fluke. Several fish robots that use smart actuators to create undulating motion have also been investigated [23.89]. *Wang* et al. embedded shape memory alloy (SMA) wire actuator to create flexible bending and investigated the musculature of a cuttlefish fin to aid the design of the biomimetic fin [23.90]. *Chen* et al. mimicked manta ray by using ionic polymer-metal composite (IPMC) as artificial muscles. They embedded an IPMC muscle in each pectoral fin and a passive PDMS membrane to lead to an undulatory flapping motion on the fin, as shown in VIDEO 434 [23.76]. *Kim* et al. [23.77] used a smart soft composite (SSC) structure to generate bending and twisting motions in a simple, lightweight structure. *Lauder* et al. designed a robotic fish caudal fin with six independently moving fin rays based on the anatomy of bluegill sunfish and presented that the cupping motion produced greater thrust than others such as *W*-shaped, undulation, and rolling [23.78]. They used five different sets of fin rays and measured thrust by varying the motion program. In addition, *Lauder* et al. used a flexible plastic foil to explore the effects of changing swimming speed, foil length, and shape of the foil-trailing edge on undulatory locomotion [23.91].

23.3.7 Jumping

In nature, many animals use jumping as a locomotion strategy. Jumping has the advantages of overcoming large obstacles and avoiding predators quickly and increasing the chances of survival. Robots also experience challenges in overcoming obstacles larger than the characteristic dimension of the robot. To find solutions for this, many researchers have developed jumping robots inspired by nature.

The jumping process requires large amounts of energy to be released instantaneously. However, muscle has limited reaction speed – achieving a maximum acceleration of 15 m s^{-2}. Therefore, many small creatures, such as insects, have adapted special elastomers for energy storage to generate large accelerations instead of using muscles. On the other hand, most large creatures, such as human that have relatively long legs, primarily use large muscles that can generate sufficient large force to swing long legs quickly.

In small-scale jumping, to achieve large instantaneous acceleration, the jumping process has two steps: 1) slow energy storage and 2) rapid release of the stored energy. Escapement cam mechanism is widely used to achieve these two steps. It consists of a spring, a radius varying cam, a motor, and a gearbox for torque amplification. The motor rotates the cam slowly, but powerfully, toward the direction of compressing or extending the spring. At the final portion of the cycle, the cam's radius returns to the initial state instantly, releasing the spring causing the robot to jump. Grillo (Ver.1) (in VIDEO 278) [23.86] and a similar 7 g jumping robot (in VIDEO 279) [23.87] use this escapement cam mechanism (Fig. 23.16).

A toothless gear mechanism is similar to the escapement cam mechanism. The main differences are that the toothless gear mechanism uses an incomplete gear instead of a change in the cam shape. The motor actuates the incomplete gear, and the gear actuates a transmission to compress or extend a spring. When the

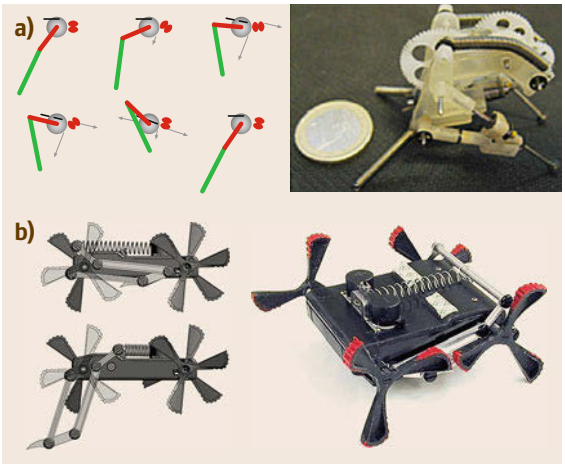


Fig.23.17a,b Toothless gear mechanism **(a)** Grillo (Ver.2) (after [23.92]), **(b)** Mini-Wheg (after [23.93])

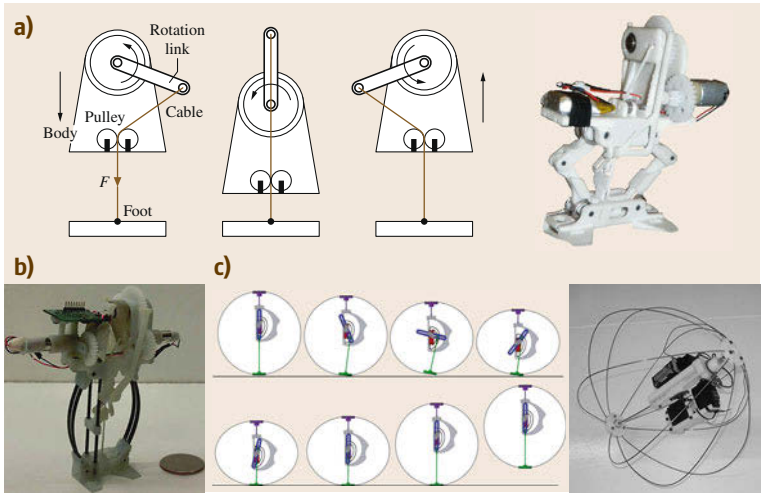


Fig.23.18a–c Other escapement mechanism **(a)** MSU Jumper (after [23.94]), **(b)** MSU Jump-runner (after [23.95]), **(c)** Jollbot (after [23.96])

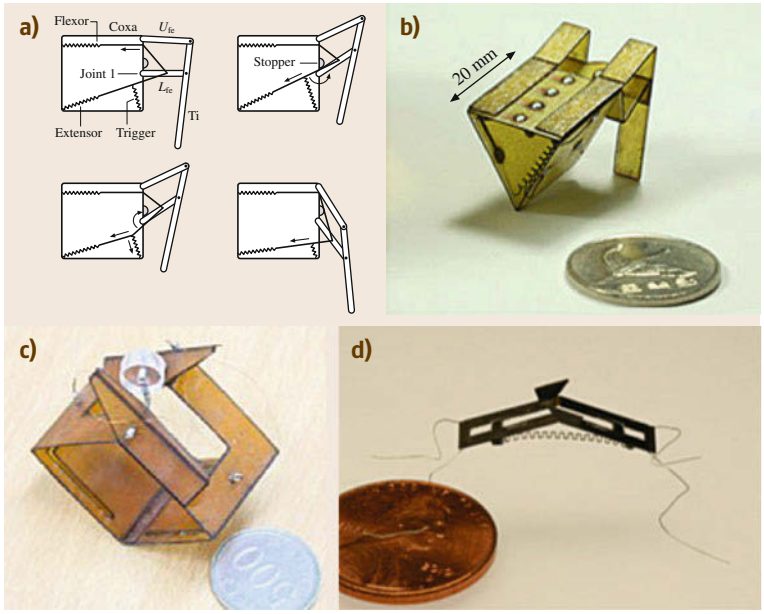


Fig. 23.19 **(a)** Torque-reversal mechanism, **(b)** Flea-inspired catapult mechanism (after [23.97]), **(c)** Simplified flea-inspired catapult mechanism (after [23.98]), **(d)** Jumping robotic insect (after [23.99])

transmission passes the toothless gear part, the transmission returns to the initial position and the stored energy is released instantly. Examples of robots using toothless gear mechanisms include Mini-Whegs [23.93] and Grillo (Ver.2) [23.92] (Fig. 23.17).

These two click mechanisms, the escapement cam mechanism and the toothless gear mechanism, are commonly used for jumping mechanisms, but other methods have also been developed (Fig. 23.18 and VIDEO 280). The catapult mechanism of MSU Jumper [23.94] and MSU Jump-Runner [23.95] is similar to the escapement cam mechanism, except for the absence of a cam. Instead of using a cam, it uses a one-way bearing. This mechanism can be separated in two parts based on the critical point in the jump cycle. Before passing the critical point, the one-way bearing cannot rotate freely, so it rotates to the direction of energy storage. On the other hand, after passing that point, it can rotate freely and release the stored energy for jumping. The catapult mechanism in Jollbot [23.96] is similar to the mechanism in the MSU Jumper. Its structure's slit acts like a one-way bearing.

A flea-inspired catapult mechanism [23.97] is different from aforementioned catapult mechanisms (Fig. 23.19 and VIDEO 281). It consists of three SMA coil springs: (1) Flexor, (2) Extensor, and (3) Trigger. The SMA is activated by heat induced from applied current. The flea catapult mechanism begins by activating

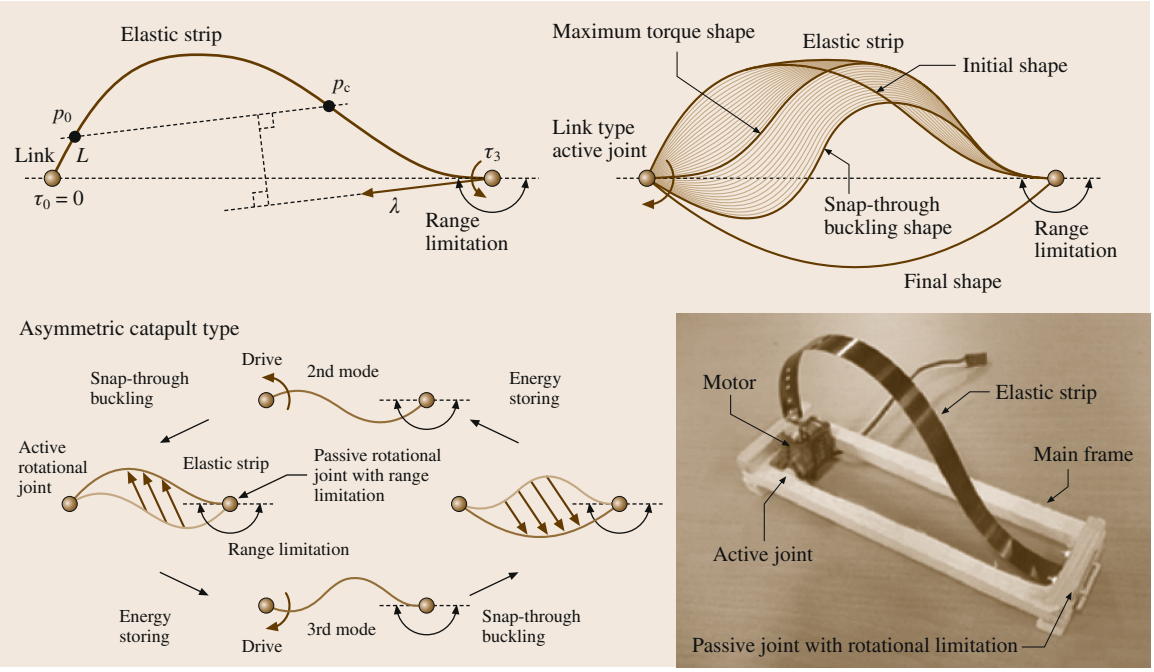


Fig. 23.20 An asymmetric robotic catapult jumping robot (after [23.100])

the flexor that folds the leg. Then, the extensor is activated. Because the direction of the torque generated by the extensor force is in the folding direction, the leg does not move and energy is stored in the extensor SMA coil spring. After energy is storage, activation of the trigger attached to the extensor pulls the extensor until it passes the joint. As a result, the direction of the torque generated by the extensor is reversed and the robot starts to jump. This mechanism uses muscle-like actuators to create the torque reversal mechanism, which enables simple design. Variation of the torque reversal mechanisms have been developed with lesser number of actuators but maintaining the same biological principle: the simplified flea-inspired catapult mechanism [23.98] and the jumping robotic insect [23.99].

An asymmetric robotic catapult jumping robot [23.100] also has a unique catapult mechanism (Fig. 23.20). It utilizes buckling in a compliant beam to jump. It consists of a main frame, an elastic strip and a motor. The elastic strip is connected to the main frame with a free rotational joint that can rotate from 0° to 180° and is connected to a motor that can control the rotation angle. When one of the elastic strip's ends is fixed to the main frame, adjusting the angle of the other end can be in a snap-through buckling shape. This can produce bidirectional jumps, but the buckling modes are different on either side.

Jumping mechanisms for microrobots [23.101] have been built using microelectromechanical systems (MEMS) manufacturing methods to create a silicon body, silicon leg, and a series of PDMS springs (Fig. 23.21a). This mechanism consists of two rigid bodies connected by PDMS springs and is activated by an external force. The robot includes only the mechanisms required to demonstrate a jump. The actuation combined elastomer mechanism is shown in Fig. 23.21b [23.101]. It used chevron actuators. These actuators are used to linearly pull and release the PDMS spring embedded into an etched silicon structure for jumping. The mechanism's PDMS springs are designed

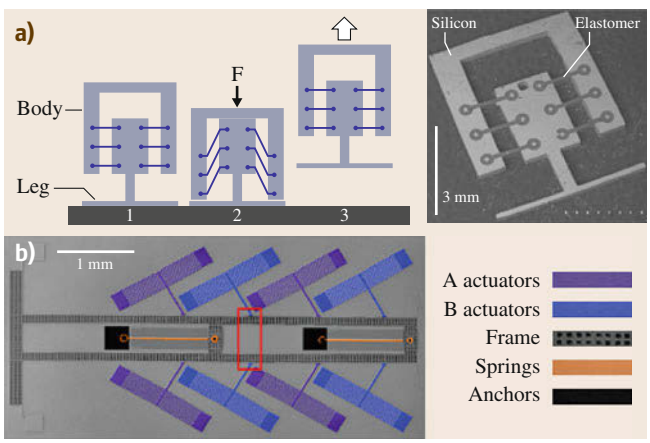


Fig. 23.21 **(a)** Microrobot (after [23.101]), **(b)** a colored SEM image of the actuated mechanism (after [23.101])

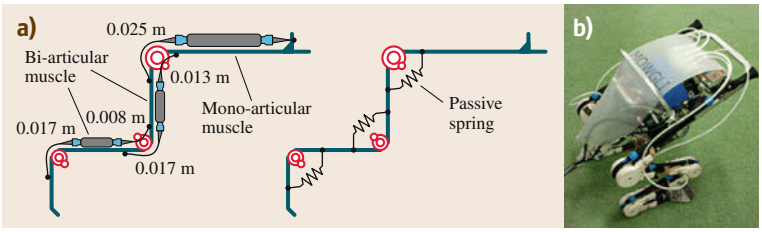


Fig. 23.22 (a) Pneumatic artificial muscles arranged like bi-articular muscle, (b) Mowgli (after [23.102])

to be similar to resilin – the elastomer that appears in insects [23.101].

In large-scale jumping, to overcome muscle's limited speed, large animals use their long legs or special arrangements of muscles and bones such as *bi-articular muscles*. They are muscles that work on two joints. In a mechanical linkage that is composed of bi-articular muscle and bones, the two joints affect each other. During the jumping process, these conditions can be helpful for generating the optimum force. The large jumping robot *Mowgli* uses long legs and pneumatic artificial muscles that are arranged like bi-articular muscle and it can be seen in Fig. 23.22 and in VIDEO 285 [23.102].

Specifications of the jumping robots are summarized in Table 23.1.

23.3.8 Gripping and Perching

In nature, insects and animals climb various kinds of terrains – from flat and smooth to wavy and rugged surfaces. Some animals evolved in a way that passively adapts to unstructured environments to reduce the energy consumption and control complexity. From the view point of robotics, these properties have the potential to increase energy efficiency and reduce system complexity. Therefore, many researchers have employed such mechanism to gripping and perching devices.

Hawkes et al. developed a mechanism that allows large patches of directional dry adhesives to conform to the topology of the surfaces they are in contact with [23.103]. The mechanism uses a rigid tile supported by a compliant material loaded by an inextensible tendon – inspired by the tendon system and the fluid-filled sinus in gecko toes. This mechanism permits the adhesive to make full contact with the surface and have uniform loading despite significant errors in alignment. *Hawkes* et al. also developed a gasper for landing of microair vehicles and grappling objects in space using gecko-inspired directional adhesives, as shown in VIDEO 413 [23.104] (Fig. 23.23).

There are several devices and robots that employ microspines for grasping rough surfaces easily seen in nature. *Kim* et al. proposed arrays of miniature spines that catch opportunistically on surface asperities [23.55]. *Desbiens* et al. proposed a small and unmanned aircraft that can land, perch and take off from vertical surfaces, as shown in VIDEO 412 [23.105] inspired by squirrels that reduce their horizontal velocity up to 60% prior to impact to distribute impact over all four limbs. *Spenko* et al. developed a hexapedal climbing robot using rows of toes having microspine [23.56]. *Parness* et al. also employed 16 carriages, each of which contains 16 microspines that conform to mm-scale and below, as shown in VIDEO 414 [23.106] (Fig. 23.24).

Trimmer et al. employed a passive gripping method found in caterpillars [23.107]. Caterpillars use their re-

Table 23.1 Specifications of the jumping robots

Robot	Actuator	Length	Weight	Jumping height	Initial velocity
7 g jumping robot [23.87]	Motor	5 cm	7 g	1.4 m	5.9 m/s
Grillo(Ver.1) [23.86]	Motor	5 cm	15 g	–	1.5 m/s
Grillo(Ver.2) [23.92]	Motor	3 cm	10 g	–	3.6 m/s
Mini-Wheg [23.93]	Motor	9–10 cm	90–190 g	0.18 m	–
MSU Jumper [23.94]	Motor	6.5 cm	23.5 g	0.87 m	–
MSU Jump-runner [23.95]	Motor	9 cm	25 g	1.43 m	–
Jollbot [23.96]	Motor	30 cm	465 g	0.218 m	–
Flea-inspired catapult mechanism [23.97]	SMA	2 cm	1.1 g	0.64 m	4.4 m/s
Simplified flea-inspired jumping mechanism [23.98]	SMA	3 cm	2.3 g	1.2 m	7 m/s
Jumping robotic insect [23.99]	SMA	2 cm	0.034 g	0.3 m	2.7 m/s
An asymmetric robotic catapult jumping robot [23.100]	Motor	17 cm	30 g	0.2 m	–
Microrobot [23.101]	None	0.4 cm	0.008 g	0.32 m	3 m/s
Mowgli [23.102]	Pneumatic	1 m	3 kg	0.4 m	–

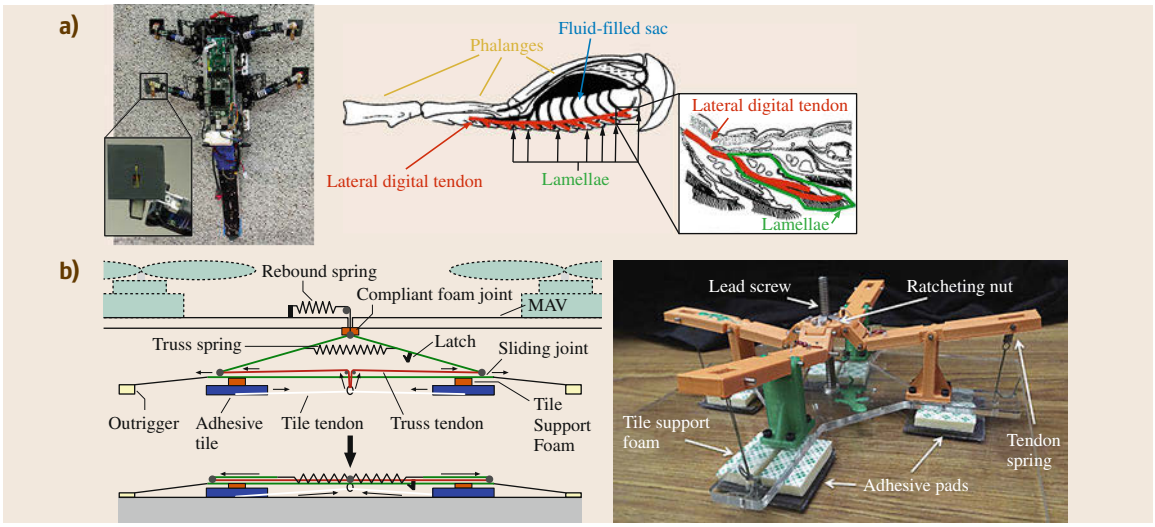


Fig.23.23a,b Graspers based on Gecko-inspired adhesives. **(a)** Gecko-inspired toe used in RiSE robot (*left*) and cross section of the gecko foot (*right*) (after [23.103]), **(b)** Collapsing truss grasper (*left*) and pivot linkage grasper (*right*) (after [23.104])

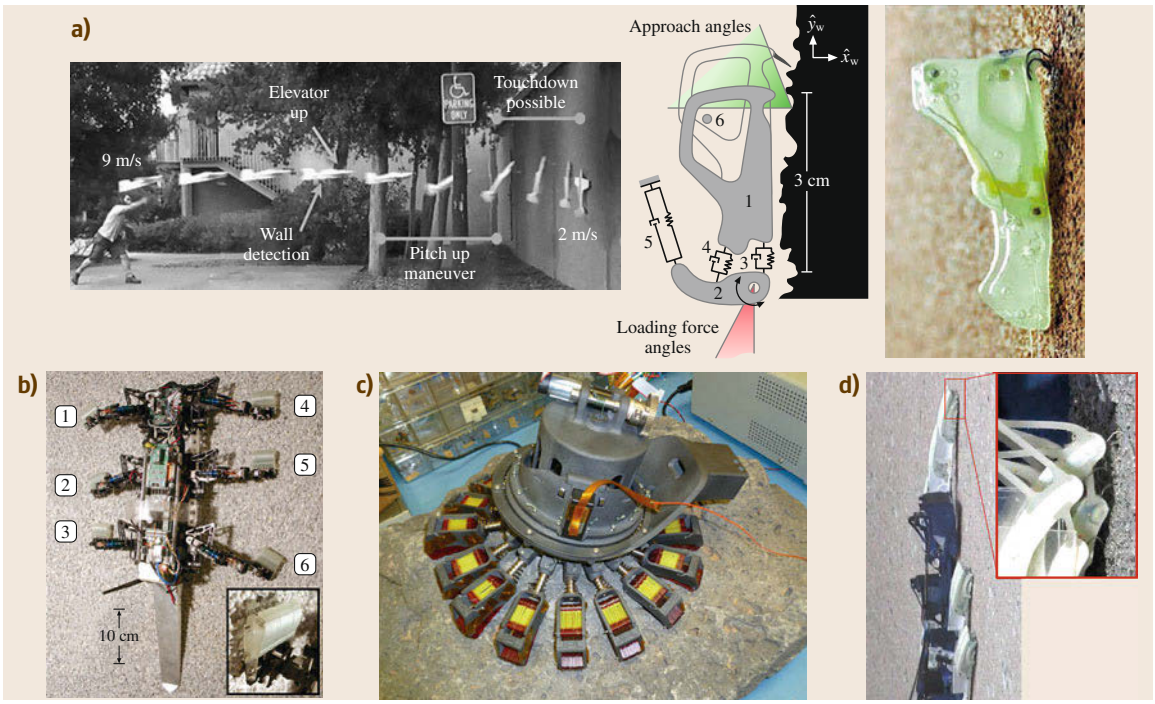


Fig.23.24a–d Microspine-based robots. **(a)** Landing and perching UAV (after [23.105]), **(b)** RiSE robot (after [23.56]), **(c)** sample acquisition tool (after [23.106]), **(d)** Spinybot (after [23.55])

tractor muscles to release the grip, which means that they do not consume any energy during gripping. Like the caterpillar, Trimmer et al. designed the grippers so that gripping is released when the SMA spring actuator is activated. *Jung* et al. proposed an underactuated mechanism based on flexural buckling [23.108]. The flexural buckling mechanism is inspired by the soft cuticle of a caterpillar's feet, which largely deforms depending on the shape of the contacting surface. The large deformation in the engineered device, flexural

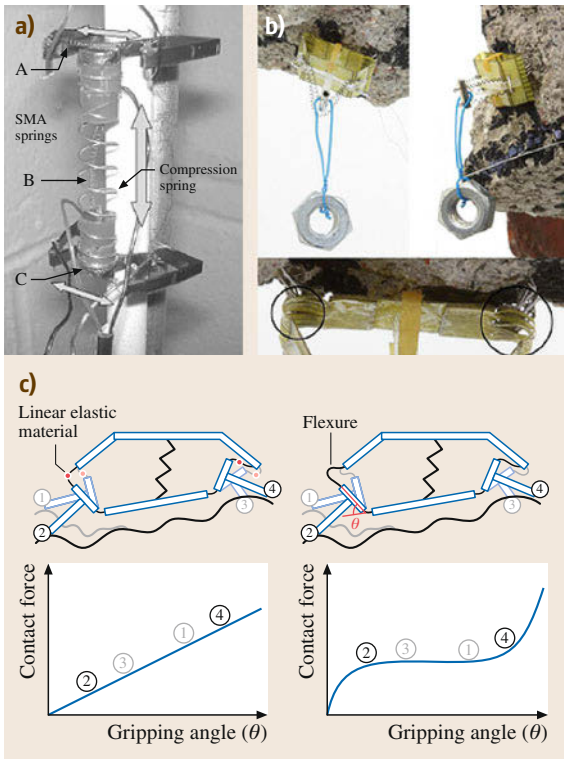


Fig. 23.25 **(a)** Passive gripping system (after [23.107]), **(b)** caterpillar-inspired underactuated gripper and **(c)** constant force region by flexural buckling (after [23.108]) ◄

buckling with an adequately selected length, provides wide gripping range with a narrow range of force variation. This provides a sufficient number of contacts with even contact forces, enabling adaptive gripping on various surfaces. In addition, design of the gripper can easily be scaled up or down depending on required scale. VIDEO 409 shows the small and large scale gripper that can achieve adaptive grasping (Fig. 23.25).

The octopus performs crawling movements with the same limbs used for grasping and manipulation, as shown in VIDEO 411. *Calisti* et al. proposed an octopus-inspired solution for movement and manipulation [23.109]. To implement octopus-like motion, they employed a steel cable for elongating and shortening and fiber cables for bending, which is inspired by the longitudinal muscles found in an octopus shown in Fig. 23.26.

Kim et al. developed flytrap-inspired high-speed gripper, as shown in VIDEO 410 [23.110]. Flytraps achieve fast capturing by using the bistable structural characteristic of its leaf. To achieve similar bistability, Kim et al. used asymmetrically laminated carbon fiber reinforced prepregs (CFRP). They also utilized a developable surface having kinematic constraints, which constrain the curvature of the artificial leaf. Therefore, the curved leaf can be actuated by bending the straight edge orthogonal to the curve, a process called bending propagation.

Doyle et al. developed an avian-inspired passive perching mechanism for quadrotors for perch-and-stare, as shown in VIDEO 415 [23.111]. Songbirds had evolved to sleep while perching. When they perch on a branch, the tendon connected from the ankle and

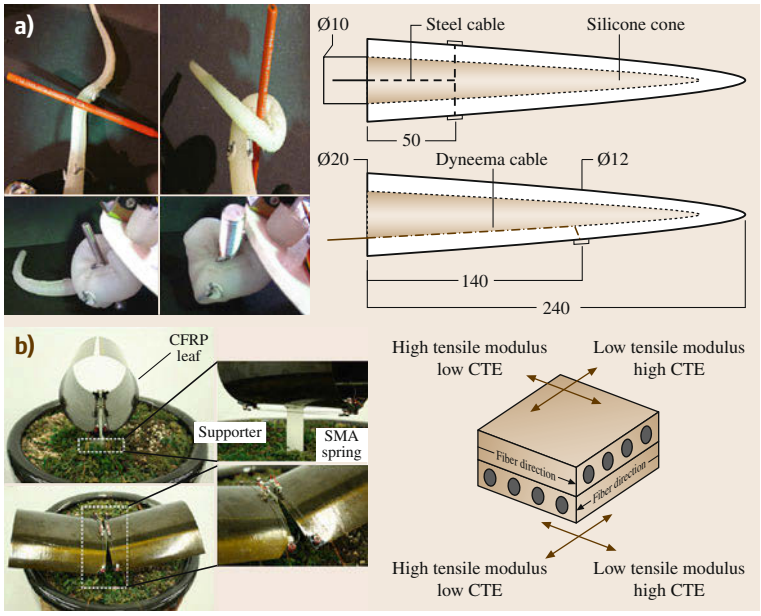


Fig. 23.26 **(a)** Octopus-inspired manipulation (*left*) and tendon-driven mechanism (*right*) (after [23.109]), **(b)** Flytrap-inspired gripper (*left*) and Orthogonally laminated CFRP (*right*) (after [23.110])

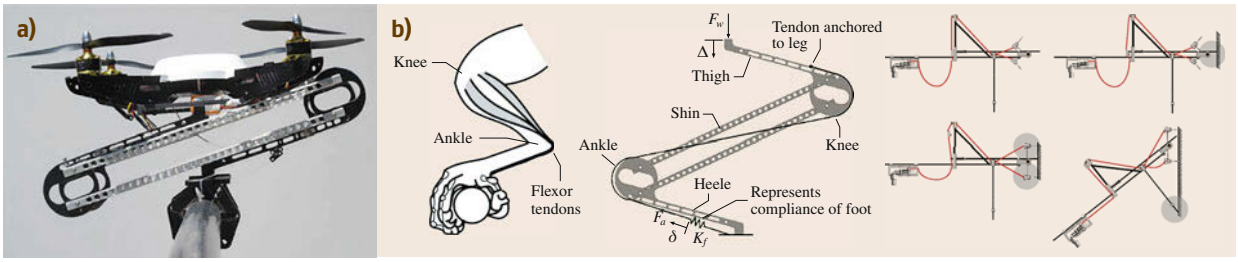


Fig. 23.27 **(a)** Avian-inspired perching mechanism with UAV (*left*) anatomy and mechanism design (*right*) (after [23.111]), **(b)** perching mechanism and process (after [23.112])

the rear side of toe automatically cause the toe to grip the branch. This allows the songbirds to tightly grip the branch without any muscular effort. Inspired by this, Doyle et al. used a four-bar mechanism and a tendon connected from the knee to the ankle and foot to couple landing motion with grasping. *Kovač* et al. presented a 4.6 g perching mechanism for microaerial vehicles (MAVs) to make them perch on various walls such as tree and concrete buildings [23.112], as shown in VIDEO 416. To achieve high impact force, the needles snap through as the trigger collides with the target surface (Fig. 23.27).

23.4 Material and Fabrication

23.4.1 Shape Deposition Manufacturing

The fundamental concept of shape deposition manufacturing (SDM) is layered molding manufacturing with CNC machining process. It not only create complex 3-D shapes rapidly, but also enables high precision finishing and large design flexibility. This concept is initially proposed by *Weiss* et al. [23.113]. Figure 23.28a shows the steps of the SDM process and Fig. 23.28b shows fabrication result [23.114]. After deposing support material, the support was fabricated by CNC machining to make high precision surface. *Li* et al. shows that it is possible to embed a various functional material such as sensors to the structure [23.115] and *Marra* et al. show that this process can be used in the fabrication of scaffolds for bone tissue engineering [23.116].

In 1999, the SDM process was first used for robot design by *Bailey* et al. [23.114]. Robot design methods based on the SDM process have advantages that it does not need complex assembly or connecting methods, and it can embed sensors and actuators directly into the body structure. In other words, the structure is build and assembled at the same time, and this

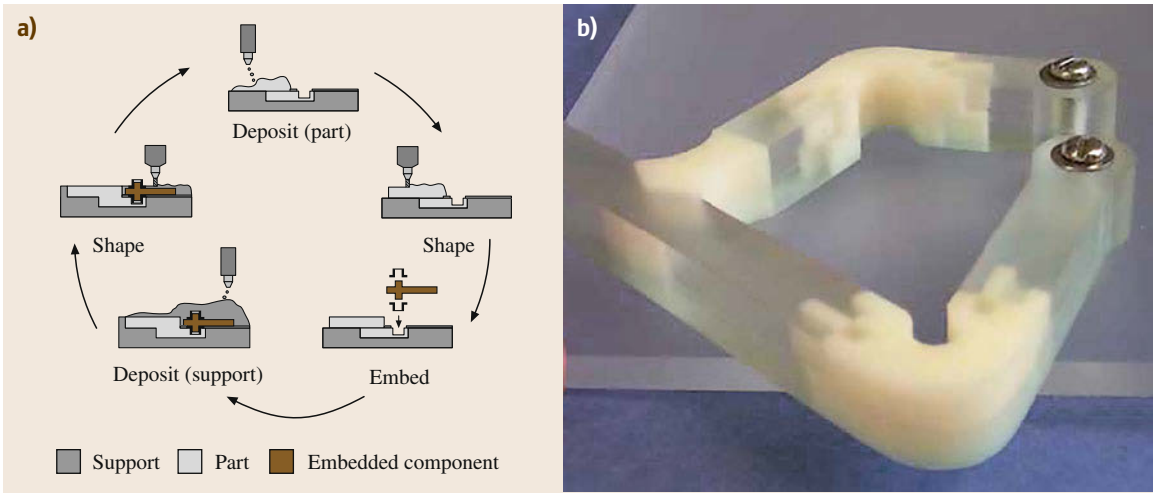


Fig.23.28a,b Shape deposition manufacturing **(a)** Process, **(b)** result of SDM process for robot mechanism design (after [23.114])

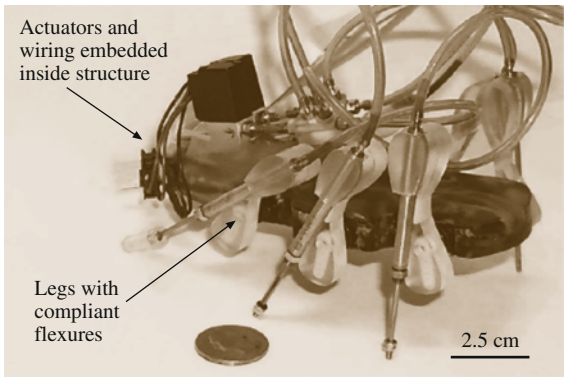


Fig. 23.29 Hexapedal robot built by SDM process (after [23.117])

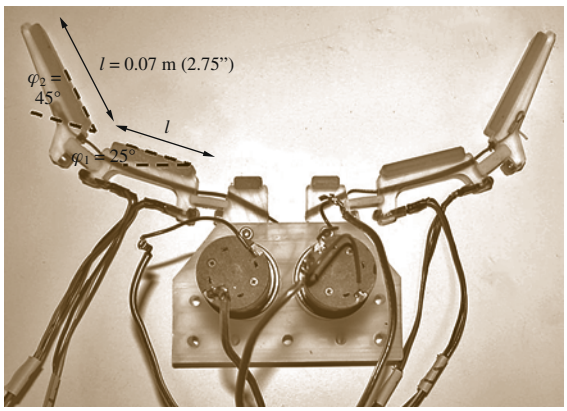


Fig. 23.30 Robotic gripper built by SDM process (after [23.118])

characteristic leads to enhanced manufacturability in small-scale applications. To build a desired mechanism, the author used multiple cycles of material deposition and shaping as shown in Fig. 23.28a. Multiple materials were used in fabrication to create variable characteristic to each functional part. More flexible material was used for joints and rigid material was used for links. The flexible components are not only used as articulated joints, but also as dampers and springs to control the impedance of a mechanism. *Binnard* et al. suggested a design framework for the SDM process for mechanism design, making more multifarious applications possible [23.119]. Figure 23.29 shows Sprawlita, a hexapedal robot fabricated based on the SDM process. Using the SDM process, actuators and wires are embedded in the body structure, resulting in robust performance and minimal manual assembly operations [23.117]. Robotic grippers can also be made by the SDM process (Fig. 23.30) [23.118]. All of joints, links, and sheath for the actuating wire were fabricated and assembled simultaneously. Embedded sensors are also possible. Force sensors embedded in the fingertips of a robotic SDM gripper have been demonstrated [23.120]. Using a soft base material for the robot, it is also possible to design a human-friendly SDM robot [23.121].

23.4.2 Smart Composite Microstructures

In the late 1990s, researchers at the University of California, Berkeley began a project to create a robotic insect capable of sustained autonomous flight – the micromechanical flying insect was born [23.122]. Among the many challenges for this project, how to manufacture and what materials should be used for structures, mechanisms, and actuators were primary concerns. Recognizing the lack of a viable *meso-scale* manufacturing method, the team lead by Fearing attempted multiple techniques including folded triangular stainless steel beams [23.123] and eventually settled on multilayer composites [23.124]. In this paradigm, later called smart composite microstructures (SCM) [23.50], layers of materials are machined, aligned, and laminated to form a quasi-two-dimensional (quasi-2-D) laminate. The choice of materials, 2-D layer geometries, and order of the layup allows the user to create an array of rigid components separated by compliant flexures. This composite laminate, later called a *standard linkage layer* can then be folded into 3-D shapes and mechanisms.

The SCM process presents a new paradigm of design and fabrication for developing small-scale robots. Planar fiber-reinforced prepreg (FRP) and flexure hinges replace conventional links and joints in the robot mechanism. The composite laminating process is

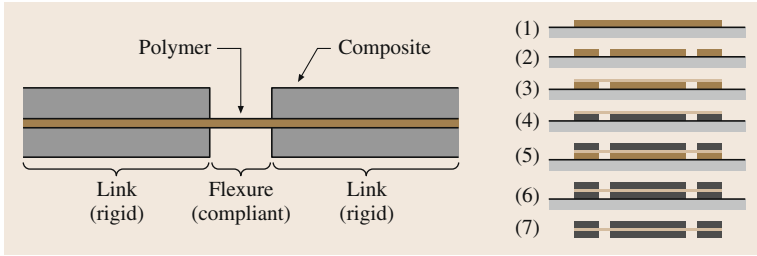


Fig. 23.31 Schematic illustration of a single joint unit in SCM and the layup of laminating process (after [23.50])

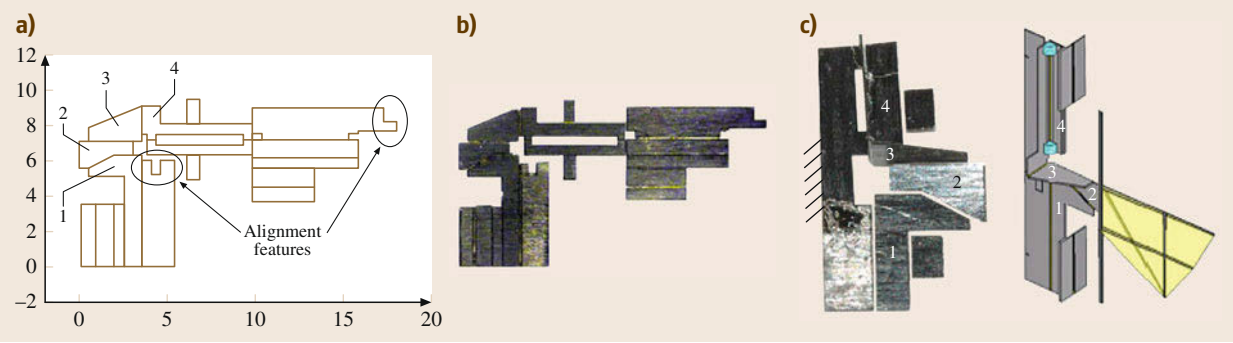


Fig.23.32a–c Spherical five-bar linkage structure created by the SCM fabrication process (after [23.52]). **(a)** The pattern design of a rigid face sheet, **(b)** laser-cut and cured microcomposite sheet before folding, **(c)** SCM spherical five-bar linkage for transmission of MFI

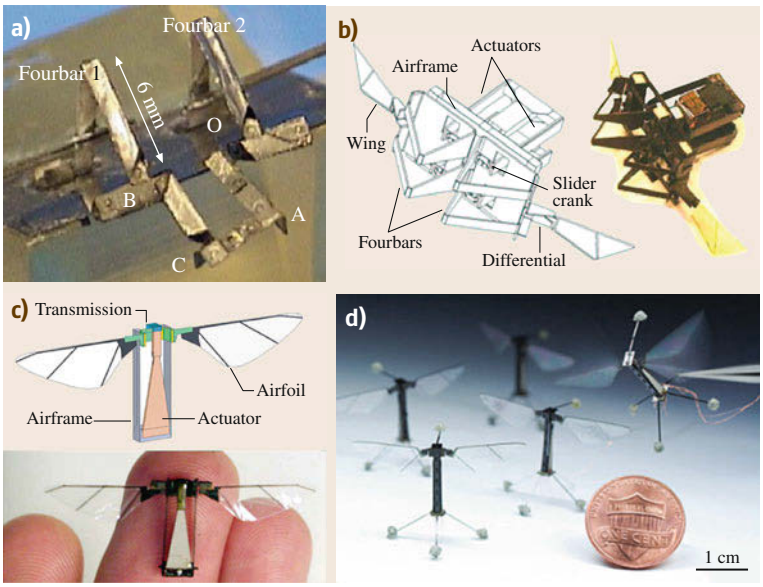


Fig.23.33a–d Various generations of MFI. **(a)** Wing transmission made of steed face sheet (after [23.123]). **(b)** Thorax mechanism with five-bar spherical linkage SCM (after [23.124]). **(c)** The Harvard robotic fly is capable of flapping-wing liftoff (after [23.50]). **(d)** The Harvard robotic fly is the first robotic insect capable of unconstrained controlled flight (after [23.125])

adopted instead of conventional machining and assembly processes which are difficult to apply to building a small-scale robot mechanism. Figure 23.31 shows a single unit of the links and the flexure hinge joints which are the key building blocks of the SCM process [23.50]. The flexure joint is a polymer film that can be bent easily and eliminating friction losses that are the dominant cause of reduced efficiency in small-scale robot mechanisms.

Rigid face sheets of fiber-reinforced composites sandwich polymers and joints are created at the gap between face sheets. The resulting quasi-2-D sheets are then folded into 3-D structures. The face sheets of rigid composite materials are carefully patterned to create robot mechanisms. For example, Fig. 23.32a is a 2-D pattern designed for creating a spherical five-bar linkages as shown in Fig. 23.32c. This is the wing transmission linkage in the early versions of micromechanical flying insect (MFI) [23.52].

Figure 23.33 shows the various versions of the MFI developed using the SCM fabrication process. Figure 23.33a is the early version of a transmission that uses a steel face sheet [23.123]. Figure 23.33b is the next generation of the MFI fabricated using a carbon fiber composite (CF) sheet and polyester film. It has 26 joints, 4-DOF, four actuators, and two wings. CF composites materials improve the performance of MFI thorax structure by reducing the inertia by a factor of three and increasing the resonant frequency by 20% [23.124]. A later version with a 3-DOF transmission (two passive DOF) and a single bimorph PZT actuator as shown in Fig. 23.33c was able to produce sufficient thrust to achieve liftoff [23.52]. Passive dynamics in the thorax structures simpli-

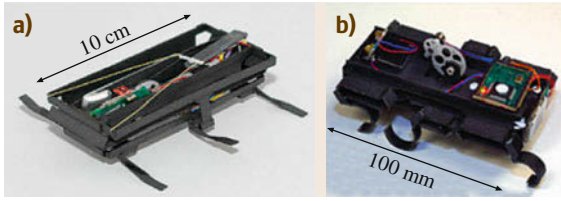


Fig. 23.34 DASH (after [23.8])

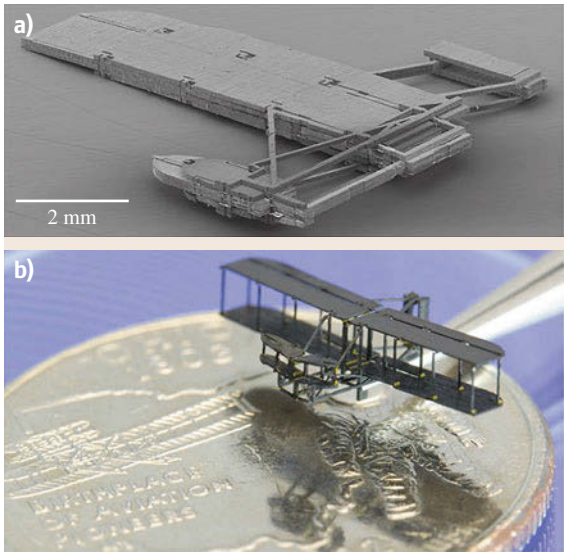


Fig.23.35a,b 1:900 scale 1903 Wright Flyer model 14 mm in wing span (after [23.126]). After laminating layup **(a)**, model after folding **(b)**

fies the design and reduces undesired coupling between the degrees of freedom. Controlled flight is accomplished by separated actuator design shown in Fig. 23.33d. Tethered but unconstrained stable hovering and basic controlled manoeuvres are demonstrated [23.125].

The SCM fabrication process can be extended to larger scales by using various sheet materials. Figure 23.34 shows centimetre scale crawling robots that are built using the SCM fabrication process with cardboard and adhesive films in place of the composites used in the MFI. This new paradigm is fast and inexpensive – both in the materials used and the required infrastructure. Furthermore, novel bio-inspired robot mechanism that produce high performance can be created easily as shown in Fig. 23.34. Figure 23.34a is a small, lightweight, power autonomous running robot, DASH [23.8]. It is capable of running at speeds up to 15 body lengths per second and surviving falls from large heights, due to the unique compliant nature of its structures. The design can be modified easily and achieve high performance with regard to stability, speed, and maneuverability as shown in Fig. 23.34b [23.10].

23.4.3 Pop-Up Book MEMS

In order to streamline the development of high performance and economical robots, many assembly tools have been developed to assist with robot construction at various scales. However, in millimetre-scale robots, many challenges arise in the fabrication process and handling many individual parts for assembly. Pop-up books and paper folding inspire the solution for eliminating the onerous assembly process with small individual parts in the fabrication process of the millimetre-scale robotic structures. Monolithic fabrication using pop-up book-inspired designs enables efficient batch processing starting from multiple layered composites similar to the basic elements used in SCM. The carefully designed layers are interconnected and allow folding mechanisms of high complexity. Once folding ensues, the complex 3-D structures are created by parallel mechanisms created in the thickness

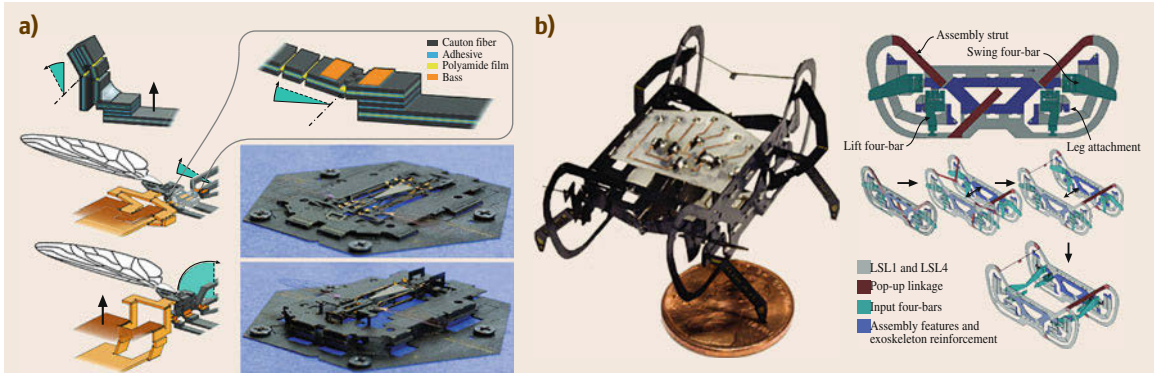


Fig. 23.36 **(a)** Mobee, monolithic design of MFI [23.127], **(b)** HAMR-VP, a 1.27 g quadrupedal microrobot manufactured using the PC-MEMS and pop-up assembly techniques (after [23.16])

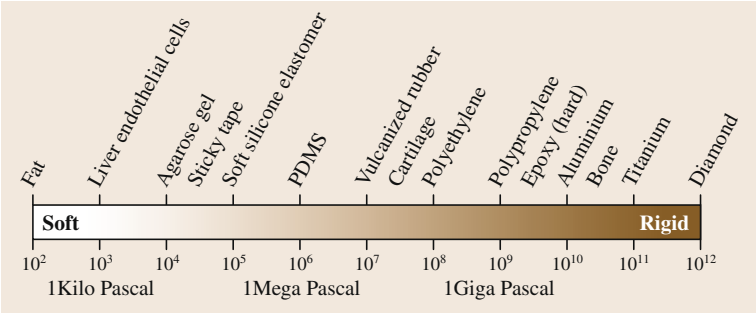


Fig. 23.37 Young's modulus for various materials (after [23.128] which was also adapted by *Autumn* et al. in [23.63])

of the laminate. Figure 23.35 presents a pop-up structure of *1:900 scale 1903 Wright Flyer model*, 14 mm in wing span [23.126]. The model consists of six rigid CF composite layers, seven adhesive layers, and two polymer flexure layers. Multiple rigid-flex folding layers are stacked and selectively bonded. The idea is previously applied to microelectric mechanical systems (MEMS) process. Combining this idea and pop-up book designs, the fabrication process is developed for building micro robotic systems including robot structures and actuators.

Figure 23.36 shows two examples that are designed and fabricated by the pop-up book MEMS process. Figure 23.36a is the Mobee (Harvard robotic fly using the monolithic popup book MEMS design methods) [23.127]. In VIDEO 398, it demonstrates how the pop-up book MEMS process enables mass production by parallel manufacturing on a single sheet and reduces entire fabrication time by eliminating onerous assembly tasks. Figure 23.36b is a small-scale quadruped crawling robot designed by pop-up book MEMS techniques. This device demonstrates how complex millimeter-scale robot structures are capable of pop-up assembly [23.16]. Twenty-three material layers are cut by precision laser machining and laminated with selective adhesion. After popup, the body frame is created and other components such as a circuit board and actuators are bonded on the frame.

23.4.4 Other Fabrication Methods

In nature, animals use soft parts of their body for generating locomotion, morphing configuration, and adapting to the environment. In order to maximize the utility of their compliance, some bio-inspired robots are primarily composed of soft materials such as fluids, gels, granules, and soft polymers. With respect to compliance, soft materials vary with a wide range of elastic (Young's) modulus [23.128] (Fig. 23.37). Therefore, bio-inspired robots should be built by different fabrication methods depending on their constituent materials.

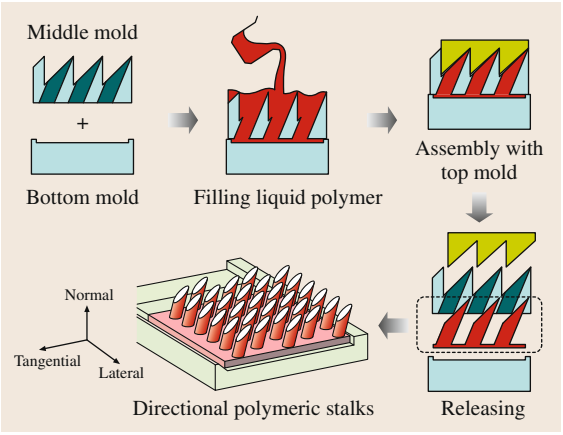


Fig. 23.38 Manufacturing process of directional adhesion pad (after [23.58])

Soft Lithography

Soft lithography was originally proposed for micro- and nanostructure manufacturing in 1998 [23.129]. Early soft lithography used an elastomeric stamp with patterned relief structures on its surface to generate desired patterns and structures. In bio-inspired robotics, for example, a directional adhesion pad inspired by a gecko foot's microscale adhesion spines was developed by a similar fabrication method to soft lithography [23.58] (Fig. 23.38).

Advances of convenient, effective and low-cost soft lithography allowed it to be utilized not only in the micro and nano scale in bio-chemical fields, but also in macro-scale robotics. Moreover, development and popularization of additive manufacturing process using 3-D printers which can design molds with various shapes easily and rapidly have helped soft lithography to be applied in robotics. Soft lithography has become a representative manufacturing method for bio-inspired soft robots since soft lithography was first applied to manufacturing bio-inspired soft robots [23.130–133] (Fig. 23.39). In soft lithography for bio-inspired soft robots, uncured elastomeric polymers, such as PDMS or EcoFlex, are poured into a mold designed with

configuration of the structure. After curing, the soft materials form a structure containing multiple air chambers and pneumatic channels. Such kind of structure is so-called a pneumatic network (PneuNet) or bending fluidic actuator (BFA).

This structure allows the robots to generate sophisticated locomotion such as gripping, walking, and crawling as shown in Fig. 23.40. In addition, extra embedded channels can control the flow of a dyeing solution so that a soft robot can camouflage by changing body color to match the color of the surroundings [23.135]. The actuation or locomotion of robots manufactured by soft lithography depends on the elastic modulus of materials and the geometry and position of air chambers [23.136].

Furthermore, soft lithography in robotics has advanced by mixing different kind of materials. By embedding sheets or fibers into elastomers, an actuator has asymmetric compliance which allows the structure to be flexible but not extensible, so that the actuator can generate a wide range of motions such as bending, extension, contraction, twisting, and others [23.137] (Fig. 23.41). By embedding magnets into elastomers, a soft robot can attach, detach, and easily align modules that have a unique function per each module depending on the task [23.138].

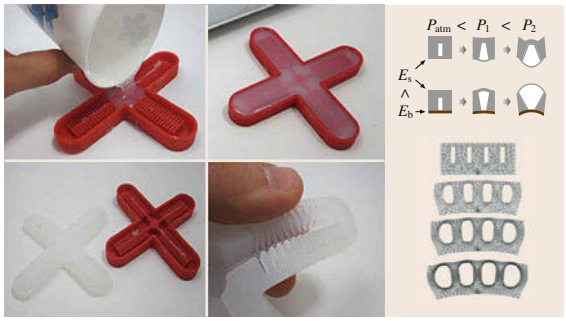


Fig. 23.39 Soft lithography of pneumatic network (PneuNet) or bending fluidic actuator (BFA) (after [23.130, 133, 134])

Fig. 23.40 Bio-inspired soft robots and actuators manufactured by soft lithography (after [23.130, 131, 135, 136])

To overcome a slow actuation, which is a typical issue for soft robots, the design of segmented air chambers divided by slits was introduced as an alternative soft actuator [23.139]. As a result, a fast PneuNet structure had high rate of actuation, improved 25 times relative to the slow PneuNet actuators. Also, a reduced change of volume minimizes fatiguing the materials, and thus the durability improves to a level that the actuator does not fail within a million cycles of full bending.

Actuator Embedded Molding

Bio-inspired soft robots potentially have infinite degrees of freedom and the nonlinearity of soft materials creates difficulty in generating desired postures and motion. Actuator-embedded molding is commonly used to manufacture the soft structure for bio-inspired robots. In actuator-embedded molding, the design considerations are the type of the actuator and the actuator's location and direction in the soft structure. For an example of a bio-inspired robot using actuator-embedded molding, a turtle-like swimming robot was created using a smart soft composite (SSC) structure to create bending and twisting deformation [23.77] (Fig. 23.42). The SSC structure consists of an actuator-embedded layer as an active component, a patterned layer as a passive component, and a soft matrix as the body as shown in Fig. 23.42. The angle of the patterned layer determines the bending direction passively, and the actuator-embedded layer generates deformation. The soft matrix helps to deform the structure continuously. Widely used

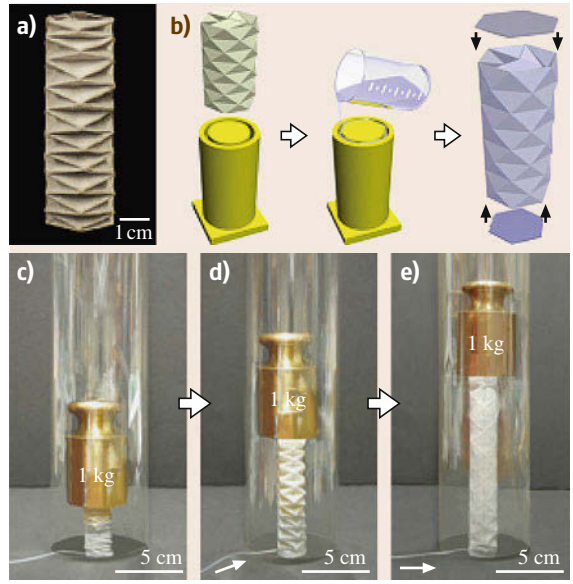


Fig.23.41a–e Programmable paper–elastomer composites-based pneumatic actuators by using soft lithography (after [23.137])

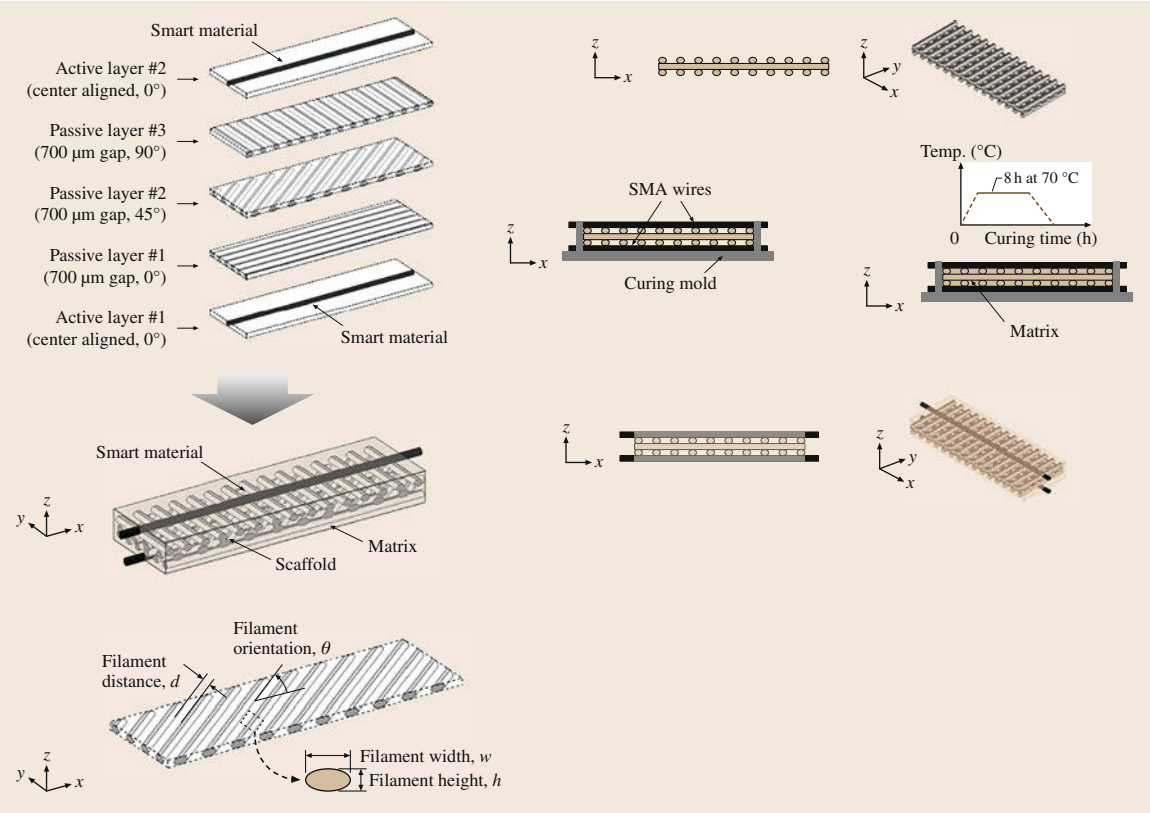


Fig. 23.42 Actuator-embedded molding for the SSC structure actuator (after [23.77])

actuators in the actuator embedded molding are wires including common wires connected with servo motors and shape memory alloys (SMA) [23.25, 77, 140].

Additive Manufacturing

Additive manufacturing or 3-D printing is a rapid prototyping (RP) process of manufacturing a 3-D solid structure of any configuration through sequential layering from a digital CAD model. Since the 1980s, various types of additive manufacturing have been developed: solid-based processes such as fused deposition modeling (FDM), photo curable liquid polymer-based processes such as stereolithography (SLA), and powder-based processes such as selective laser sintering (SLS).

Evolving the technology of 3-D printers, soft material deposition and even hybrid deposition of different materials with different stiffnesses are available. Thus, soft structures can be built at once, and products using hybrid deposition can have features of rigidity and softness at the same time. For example, a highly deformable 3-D printed soft robot was developed [23.141]. This robot body was printed by using a multimaterial printable 3-D printer (Objet Connex 500TM 3-D printer) with two materials: one is a soft rubber-like Objet TangoBlackPlusTM and another is a hard polypropylene-like Objet VeroWhitePlusTM. These two materials have different friction coefficients; thus, the robot can switch friction with the ground on its edge by bending the body.

23.5 Conclusion

Biomimetic robotics attempt to create devices that are capable of various types of effective interaction with natural environment, e.g., locomotion and manipulation, by using the principles of nature. Nature is full of surprising types of movements that enables insects and animals to survive by escaping from danger or hunting for food. In this chapter, we have presented robots that attempt to recreate these feats by understanding the un-

derlying principles, deriving an engineering design, and fabricating them with novel methods. In recent years, engineers have succeeded in recreating insects and animals that show amazing capabilities such as climbing walls like a gecko, hovering like a fly and climbing trees like a snake. These robots have been developed to understand nature, and also to be used as tools for surveillance, information gathering, and rescue operations. However, many of these robots are still in the basic research phase and are not yet ready for everyday use.

The key open research issues remain in the broad areas of materials, fabrication, actuation, and power. Composites and polymers, together with novel fabrication methods, have enabled various novel biomimetic robots. Development of these new material and fabrication has been one of the key enabling technologies, and further development of these technologies will certainly contribute to more mature biomimetic robots. Actuation and power still remain the bottleneck of many biomimetic robots. DC motors are the actuators of choice for many biomimetic robots, and with novel transmission design, DC motors can create motions required by the robots. However, DC motors are inefficient for small-scale biomimetic robots. Although artificial muscle actuators that are based on shrinkage or expansion of material, e.g., shape memory alloys, IPMC, electro-active polymers and shape memory polymers, promise to give robots capabilities similar to creatures that use biological muscles even at small scales. However, many issues related to robustness, efficiency, and power limit the capabilities of robots that use these actuators. Limitations of these actuators should be carefully considered to match the desired application. Development of new artificial muscle actuator that can emulate biological muscles – without the drawbacks of current actuators – is needed to open up a new era for biomimetic robots. Batteries also limit the capabilities of current biomimetic robots compared to their biological counterparts in terms of size and operation time. Development of energy-harvesting technologies together with new battery chemistries and manufacturing methods will enable longer operation time, which will contribute to realizing wider applications for biomimetic robots. Overall, biomimetic robot design is one of the most challenging areas of robot design since it requires development of various technologies to mimic the structure and function of natural systems. Therefore, development of biomimetic robots can have broader impact in many areas of engineering and science, and should be seen as a platform for various convergent technologies.

Video-References

VIDEO 278 The long-jumping robot 'Grillo'
available from http://handbookofrobotics.org/view-chapter/23/videodetails/278

VIDEO 279 A miniature 7 g jumping robot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/279

VIDEO 280 A single motor actuated miniature steerable jumping robot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/280

VIDEO 281 The Flea: Flea-inpired light jumping robot using elastic catapult with active storage and release mechanism
available from http://handbookofrobotics.org/view-chapter/23/videodetails/281

VIDEO 285 Jumping & landing robot 'MOWGLI'
available from http://handbookofrobotics.org/view-chapter/23/videodetails/285

VIDEO 286 RoACH: A 2.4 g, untethered crawling hexapod robot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/286

VIDEO 287 A new form of peristaltic locomotion in a robot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/287

VIDEO 288 Meshworm
available from http://handbookofrobotics.org/view-chapter/23/videodetails/288

VIDEO 289 Treebot: Autonomous tree climbing by tactile sensing
available from http://handbookofrobotics.org/view-chapter/23/videodetails/289

VIDEO 290 Omegabot : Inchworm inspired robot climbing
available from http://handbookofrobotics.org/view-chapter/23/videodetails/290

VIDEO 291 GoQBot: Insanely fast robot caterpillar
available from http://handbookofrobotics.org/view-chapter/23/videodetails/291

VIDEO 388 SpinybotII: Climbing hard walls with compliant microspines
available from http://handbookofrobotics.org/view-chapter/23/videodetails/388

VIDEO 389 Smooth vertical surface climbing with directional adhesion
available from http://handbookofrobotics.org/view-chapter/23/videodetails/389

VIDEO 390 Biologically inspired climbing with a hexapedal robot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/390
VIDEO 391 CLASH: Climbing vertical loose cloth
available from http://handbookofrobotics.org/view-chapter/23/videodetails/391
VIDEO 392 Torque control strategies for snake robots
available from http://handbookofrobotics.org/view-chapter/23/videodetails/392
VIDEO 393 Snake robot climbs a tree
available from http://handbookofrobotics.org/view-chapter/23/videodetails/393
VIDEO 394 Snake robot in the water
available from http://handbookofrobotics.org/view-chapter/23/videodetails/394
VIDEO 395 Salamandra Robotica II robot walking and swimming
available from http://handbookofrobotics.org/view-chapter/23/videodetails/395
VIDEO 397 ACM-R5H
available from http://handbookofrobotics.org/view-chapter/23/videodetails/397
VIDEO 398 Pop-up fabrication of the Harvard monolithic bee (Mobee)
available from http://handbookofrobotics.org/view-chapter/23/videodetails/398
VIDEO 399 Controlled flight of a biologically-inspired, insect-scale robot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/399
VIDEO 400 Rhex the parkour robot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/400
VIDEO 401 Mini whegs
available from http://handbookofrobotics.org/view-chapter/23/videodetails/401
VIDEO 402 Robot dragonfly DelFly explorer flies autonomously
available from http://handbookofrobotics.org/view-chapter/23/videodetails/402
VIDEO 403 Standford Sprawl and iSprawl
available from http://handbookofrobotics.org/view-chapter/23/videodetails/403
VIDEO 405 DASH: Resilient high-speed 16 g hexapedal robot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/405
VIDEO 406 HAMR3: An autonomous 1.7 g ambulatory robot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/406
VIDEO 407 Undulatory gaits in a centipede millirobot
available from http://handbookofrobotics.org/view-chapter/23/videodetails/407
VIDEO 408 VelociRoACH
available from http://handbookofrobotics.org/view-chapter/23/videodetails/408
VIDEO 409 Underactuated adaptive gripper using flexural buckling
available from http://handbookofrobotics.org/view-chapter/23/videodetails/409
VIDEO 410 Flytrap-inspired bi-stable gripper
available from http://handbookofrobotics.org/view-chapter/23/videodetails/410
VIDEO 411 An octopus-bioinspired solution to movement and manipulation for soft robots
available from http://handbookofrobotics.org/view-chapter/23/videodetails/411
VIDEO 412 Landing and perching UAV
available from http://handbookofrobotics.org/view-chapter/23/videodetails/412
VIDEO 413 Dynamic surface grasping with directional adhesion
available from http://handbookofrobotics.org/view-chapter/23/videodetails/413
VIDEO 414 Gravity-independent rock-climbing robot and a sample acquisition tool with microspine grippers
available from http://handbookofrobotics.org/view-chapter/23/videodetails/414
VIDEO 415 Avian-inspired perching mechanism with UAV
available from http://handbookofrobotics.org/view-chapter/23/videodetails/415
VIDEO 416 A perching mechanism for micro aerial vehicles
available from http://handbookofrobotics.org/view-chapter/23/videodetails/416
VIDEO 431 G9 series robotic fish
available from http://handbookofrobotics.org/view-chapter/23/videodetails/431
VIDEO 432 Ichthus
available from http://handbookofrobotics.org/view-chapter/23/videodetails/432
VIDEO 433 Autonomous, self-contained soft robotic fish
available from http://handbookofrobotics.org/view-chapter/23/videodetails/433
VIDEO 434 Robotic Ray takes a swim
available from http://handbookofrobotics.org/view-chapter/23/videodetails/434

References

23.1 R.J. Full, K. Autumn, J. Chung, A. Ahn: Rapid negotiation of rough terrain by the death-head cockroach, Am. Zool. **38**(5), 81A (1998)

23.2 R.J. Full, M.S. Tu: Mechanics of a rapid running insect: Two-, four- and six-legged locomotion, J. Exp. Biol. **156**, 215–231 (1991)

23.3 C.P. Ellington: The novel aerodynamics of insect flight: Applications to micro-air vehicles, J. Exp. Biol. **202**, 3439–3448 (1999)

23.4 U. Saranli, M. Buehler, D.E. Koditschek: RHex: A simple and highly mobile hexapod robot, Int. J. Robotics Res. **20**, 616–631 (2001)

23.5 J.M. Morrey, B. Lambrecht, A.D. Horchler, R.E. Ritzmann, R.D. Quinn: Highly mobile and robust small quadruped robots, Proc IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 1 (2003) pp. 82–87

23.6 J.E. Clark, J.G. Cham, S.A. Bailey, E.M. Froehlich, P.K. Nahata, R.J. Full, M.R. Cutkosky: Biomimetic design and fabrication of a hexapedal running robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (2001) pp. 3643–3649

23.7 S. Kim, J.E. Clark, M.R. Cutkosky: iSprawl: design and tuning for high-speed autonomous open-loop running, Int. J. Robotics Res. **25**, 903–912 (2006)

23.8 P. Birkmeyer, K. Peterson, R.S. Fearing: DASH: A dynamic 16g hexapedal robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 2683–2689

23.9 A.M. Hoover, E. Steltz, R.S. Fearing: RoACH: An autonomous 2.4g crawling hexapod robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 26–33

23.10 A.M. Hoover, S. Burden, X.-Y. Fu, S.S. Sastry, R.S. Fearing: Bio-inspired design and dynamic maneuverability of a minimally actuated six-legged robot, Proc. IEEE/RAS Biomed. Robotics Biomech. (BioRob) (2010) pp. 869–876

23.11 A.O. Pullin, N.J. Kohut, D. Zarrouk, R.S. Fearing: Dynamic turning of 13 cm robot comparing tail and differential drive, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 5086–5093

23.12 A.T. Baisch, C. Heimlich, M. Karpelson, R.J. Wood: HAMR3: An autonomous 1.7g ambulatory robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 5073–5079

23.13 K.L. Hoffman, R.J. Wood: Turning gaits and optimal undulatory gaits for a modular centipede-inspired millirobot, Proc. IEEE/RAS Biomed. Robotics Biomech. (BioRob) (2012) pp. 1052–1059

23.14 R. Sahai, S. Avadhanula, R. Groff, E. Steltz, R. Wood, R.S. Fearing: Towards a 3g crawling robot through the integration of microrobot technologies, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 296–302

23.15 J.T. Watson, R.E. Ritzmann, S.N. Zill, A.J. Pollack: Control of obstacle climbing in the cockroach, Blaberus discoidalis. I. Kinematics, J. Comp. Physiol. A **188**, 39–53 (2002)

23.16 A.T. Baisch, O. Ozcan, B. Goldberg, D. Ithier, R.J. Wood: High speed locomotion for a quadrupedal microrobot, Int. J. Robotics Res. **33**, 1063–1082 (2014)

23.17 D.W. Haldane, K.C. Peterson, F.L. Garcia Bermudez, R.S. Fearing: Animal-inspired design and aerodynamic stabilization of a hexapedal millirobot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 3279–3286

23.18 A.S. Boxerbaum, H.J. Chiel, R.D. Quinn: A new theory and methods for creating peristaltic motion in a robotic platform, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 1221–1227

23.19 S. Seok, C.D. Onal, K.-J. Cho, R.J. Wood, D. Rus, S. Kim: Meshworm: A peristaltic soft robot with antagonistic nickel titanium coil actuators, IEEE/ASME Trans. Mechatron. **18**, 1–13 (2012)

23.20 A. Menciassi, D. Accoto, S. Gorini, P. Dario: Development of a biomimetic miniature robotic crawler, Auton. Robotics **21**, 155–163 (2006)

23.21 K. Kotay, D. Rus: The inchworm robot: A multi-functional system, Auton. Robotics **8**, 53–69 (2000)

23.22 N. Cheng, G. Ishigami, S. Hawthorne, H. Chen, M. Hansen, M. Telleria, R. Playter, K. Iagnemma: Design and analysis of a soft mobile robot composed of multiple thermally activated joints driven by a single actuator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 5207–5212

23.23 J.-S. Koh, K.-J. Cho: Omega-shaped inchworm-inspired crawling robot with large-index-and-pitch (LIP) SMA spring actuators, IEEE/ASME Trans. Mechatron. **18**, 419–429 (2013)

23.24 T.L. Lam, Y. Xu: Climbing strategy for a flexible tree climbing robot – Treebot, IEEE Trans. Robotics **27**, 1107–1117 (2011)

23.25 H.-T. Lin, G.G. Leisk, B. Trimmer: GoQBot: A caterpillar-inspired soft-bodied rolling robot, Bioinsp. Biomimet. **6**, 026007 (2011)

23.26 S. Hirose, Y. Umetani: The development of soft gripper for the versatile robot hand, Mech. Mach. Theory **13**, 351–359 (1978)

23.27 H. Ohno, S. Hirose: Design of slim slime robot and its gait of locomotion, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 707–715

23.28 C. Wright, A. Johnson, A. Peck, Z. McCord, A. Naaktgeboren, P. Gianfortoni, M. Gonzalez-Rivero, R. Hatton, H. Choset: Design of a modular snake robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007) pp. 2609–2614

23.29 H. Yamada, S. Chigisaki, M. Mori, K. Takita, K. Ogami, S. Hirose: Development of amphibious snake-like robot ACM-R5, Proc. ISR (2005)

23.30 J. Gray: The mechanism of locomotion in snakes, J. Exp. Biol. **23**, 101–120 (1946)

23.31 G.S. Miller: The motion dynamics of snakes and worms, ACM Siggraph Comput. Graph. **22**, 169–173 (1988)

23.32 Z. Bayraktaroglu: Snake-like locomotion: Experimentations with a biologically inspired wheelless snake robot, Mech. Mach. Theory **44**, 591–602 (2009)

23.33 D.L. Hu, J. Nirody, T. Scott, M.J. Shelley: The mechanics of slithering locomotion, Proc. Natl. Acad. Sci. **106**, 10081–10085 (2009)

23.34 Z. Wang, S. Ma, B. Li, Y. Wang: Experimental study of passive creeping for a snake-like robot, Proc. IEEE/ICME Int. Conf. Complex Med. Eng. (CME) (2011) pp. 382–387

23.35 J.J. Socha, T. O'Dempsey, M. LaBarbera: A 3-D kinematic analysis of gliding in a flying snake, Chrysopelea paradisi J. Exp. Biol. **208**, 1817–1833 (2005)

23.36 R.L. Hatton, H. Choset: Generating gaits for snake robots: Annealed chain fitting and keyframe wave extraction, Auton. Robotics **28**, 271–281 (2010)

23.37 K.J. Dowling: *Limbless Locomotion: Learning to Crawl with a Snake Robot* (NASA, Pittsburgh 1996)

23.38 S. Hirose, M. Mori: Biologically inspired snake-like robots, IEEE Int. Conf. Robotics Biomimet. (ROBIO) (2004) pp. 1–7

23.39 C. Wright, A. Buchan, B. Brown, J. Geist, M. Schwerin, D. Rollinson, M. Tesch, H. Choset: Design and architecture of the unified modular snake robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 4347–4354

23.40 K.-H. Low: *Industrial Robotics: Programming, Simulation and Applications* (InTech, Rijeka 2007)

23.41 A.J. Ijspeert, A. Crespi, D. Ryczko, J.M. Cabelguen: From swimming to walking with a salamander robot driven by a spinal cord model, Science **315**, 1416–1420 (2007)

23.42 R. Crespi, K. Karakasiliotis, A. Guignard, A.J. Ijspeert: 1 Salamandra robotica II: An amphibious robot to study salamander-like swimming and walking gaits, IEEE Trans. Robotics **29**, 308–320 (2013)

23.43 S. Hirose, H. Yamada: Snake-like robots [Tutorial], IEEE Robotics Autom. Mag. **16**, 88–98 (2009)

23.44 N. Kamamichi, M. Yamakita, K. Asaka, Z.-W. Luo: A snake-like swimming robot using IPMC actuator/sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1812–1817

23.45 P. Liljebäck, K.Y. Pettersen, O. Stavdahl, J.T. Gravdahl: Snake robot locomotion in environments with obstacles, IEEE/ASME Trans. Mechatron. **17**, 1158–1169 (2012)

23.46 P. Liljebäck, K.Y. Pettersen, Ø. Stavdahl, J.T. Gravdahl: A review on modelling, implementation, and control of snake robots, Robotics Auton. Syst. **60**, 29–40 (2012)

23.47 M.H. Dickinson: Wing rotation and the aerodynamic basis of insect flight, Science **284**, 1954–1960 (1999)

23.48 M.H. Dickinson: How animals move: An integrative view, Science **288**, 100–106 (2000)

23.49 G.C.H.E. de Croon, K.M.E. de Clercq, R. Ruijsink, B. Remes, C. de Wagter: Design, aerodynamics, and vision-based control of the DelFly, Int. J. Micro Air Veh. **1**(2), 71–97 (2009)

23.50 R.J. Wood, S. Avadhanula, R. Sahai, E. Steltz, R.S. Fearing: Microrobot design using fiber reinforced composites, J. Mech. Des. **130**, 052304 (2008)

23.51 M. Keennon, K. Klingebiel, H. Won, A. Andriukov: Development of the nano hummingbird: A tailless flapping wing micro air vehicle, AIAA Aerospace Sci. Meet. (2012)

23.52 R.J. Wood: The first takeoff of a biologically inspired at-scale robotic insect, IEEE Trans. Robotics **24**, 341–347 (2008)

23.53 K. Peterson, P. Birkmeyer, R. Dudley, R.S. Fearing: A wing-assisted running robot and implications for avian flight evolution, Bioinsp. Biomimet. **6**, 046008 (2011)

23.54 S. Hirose, A. Nagakubo, R. Toyama: Machine that can walk and climb on floors, walls and ceilings, Adv. Robotics ICAR '05. Proc. (1991) pp. 753–758

23.55 S. Kim, A.T. Asbeck, M.R. Cutkosky, W.R. Provancher: SpinybotII: Climbing hard walls with compliant microspines, Adv. Robotics ICAR '05. Proc. (2005) pp. 601–606

23.56 M. Spenko, G.C. Haynes, J. Saunders, M.R. Cutkosky, A.A. Rizzi, R.J. Full, D.E. Ko-ditschek: Biologically inspired climbing with a hexapedal robot, J. Field Robotics **25**, 223–242 (2008)

23.57 K. Autumn, A. Dittmore, D. Santos, M. Spenko, M. Cutkosky: Frictional adhesion: A new angle on gecko attachment, J. Exp. Biol. **209**, 3569–3579 (2006)

23.58 S. Kim, M. Spenko, S. Trujillo, B. Heyneman, D. Santos, M.R. Cutkosky: Smooth vertical surface climbing with directional adhesion, IEEE Trans. Robotics **24**, 65–74 (2008)

23.59 M. Minor, H. Dulimarta, G. Danghi, R. Mukherjee, R.L. Tummala, D. Aslam: Design, implementation, and evaluation of an under-actuated miniature biped climbing robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 1999–2005

23.60 D. Longo, G. Muscato: The Alicia 3 climbing robot: A three-module robot for automatic wall inspection, IEEE Robotics Autom. Mag. **13**, 42–50 (2006)

23.61 M. Armada, M. Prieto, T. Akinfiev, R. Fernández, P. González, E. García, H. Montes, S. Nabulsi, R. Ponticelli, J. Sarriá, J. Estremera, S. Ros, J. Grieco, G. Fernández: On the design and development of climbing and walking robots for the maritime industries, J. Marit. Res. **2**, 9–32 (2005)

23.62 G.C. Haynes, A. Khripin, G. Lynch, J. Amory, A. Saunders, A.A. Rizzi, D.E. Koditschek: Rapid pole climbing with a quadrupedal robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 2767–2772

23.63 K. Autumn, Y.A. Liang, S.T. Hsieh, W. Zesch, W.P. Chan, T.W. Kenny, R. Fearing, R.J. Full: Adhesive force of a single gecko foot-hair, Nature **405**, 681–685 (2000)

23.64 G.A. Lynch, J.E. Clark, P.-C. Lin, D.E. Koditschek: A bioinspired dynamical vertical climbing robot, Int. J. Robotics Res. **31**, 974–996 (2012)

23.65 J. Clark, D. Goldman, P.-C. Lin, G. Lynch, T. Chen, H. Komsuoglu, R.J. Full, D. Koditschek: Design of a bio-inspired dynamical vertical climbing robot, Robotics Sci. Syst. (2007)

23.66 P. Birkmeyer, A.G. Gillies, R.S. Fearing: Dynamic climbing of near-vertical smooth surfaces, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 286–292

23.67 K.A. Daltorio, T.E. Wei, S.N. Gorb, R.E. Ritzmann, R.D. Quinn: Passive foot design and contact area analysis for climbing mini-whegs, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 1274–1279

23.68 O. Unver, A. Uneri, A. Aydemir, M. Sitti: Geckobot: A gecko inspired climbing robot using elastomer adhesives, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2329–2335

23.69 S.A. Bailey, J.G. Cham, M.R. Cutkosky, R.J. Full: A biomimetic climbing robot based on the gecko, J. Bionic Eng. **3**, 115–125 (2006)

23.70 M.P. Murphy, C. Kute, Y. Mengüç, M. Sitti: Waalbot II: Adhesion recovery and improved performance of a climbing robot using fibrillar adhesives, Int. J. Robotics Res. **30**, 118–133 (2011)

23.71 D.S. Barrett: Propulsive Efficiency of a Flexible Hull Underwater Vehicle, Ph.D. Thesis (MIT, Cambridge 1996)

23.72 J. Liu, H. Hu: Biological inspiration: From carangiform fish to multi-joint robotic fish, J. Bionic Eng. **7**, 35–48 (2010)

23.73 G.-H. Yang, K.-S. Kim, S.-H. Lee, C. Cho, Y. Ryuh: Design and control of 3-DOF robotic fish 'ICHTHUS V5, Lect. Not. Comp. Sci. **8103**, 310–319 (2013)

23.74 K. Low: Modelling and parametric study of modular undulating fin rays for fish robots, Mech. Mach. Theory **44**, 615–632 (2009)

23.75 A.D. Marchese, C.D. Onal, D. Rus: Autonomous soft robotic fish capable of escape maneuvers using fluidic elastomer actuators, Soft Robotics **1**, 75–87 (2014)

23.76 Z. Chen, T.I. Um, H. Bart-Smith: Bio-inspired robotic manta ray powered by ionic polymer–metal composite artificial muscles, Int. J. Smart Nano Mater. **3**, 296–308 (2012)

23.77 H.-J. Kim, S.-H. Song, S.-H. Ahn: A turtle-like swimming robot using a smart soft composite (SSC) structure, Smart Mater. Struct. **22**, 014007 (2013)

23.78 C.J. Esposito, J.L. Tangorra, B.E. Flammang, G.V. Lauder: A robotic fish caudal fin: Effects of stiffness and motor program on locomotor performance, J. Exp. Biol. **215**, 56–67 (2012)

23.79 H. Prahlad, R. Pelrine, S. Stanford, J. Marlow, R. Kornbluh: Electroadhesive robots—wall climbing robots enabled by a novel, robust, and electrically controllable adhesion technology, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 3028–3033

23.80 P. Birkmeyer, A.G. Gillies, R.S. Fearing: CLASH: Climbing vertical loose cloth, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 5087–5093

23.81 K. Streitlien, G.S. Triantafyllou, M.S. Triantafyllou: Efficient foil propulsion through vortex control, AIAA J. **34**, 2315–2319 (1996)

23.82 H. Morikawa, S. Nakao, S.-I. Kobayashi: Experimental study on oscillating wing for propulsor with bending mechanism modeled on caudal muscle-skeletal structure of tuna, Jap. Soc. Mech. Eng. C **44**, 1117–1124 (2001)

23.83 R. Fan, J. Yu, L. Wang, G. Xie, Y. Fang, Y. Hu: Optimized design and implementation of biomimetic robotic dolphin, IEEE Int. Conf. Robotics Biomimet. (ROBIO) (2005) pp. 484–489

23.84 T. Salumäe, M. Kruusmaa: A flexible fin with bio-inspired stiffness profile and geometry, J. Bionic Eng. **8**, 418–428 (2011)

23.85 P.V. y Alvarado, K. Youcef-Toumi: Design of machines with compliant bodies for biomimetic locomotion in liquid environments, J. Dyn. Syst. Meas. Contr. **128**, 3–13 (2006)

23.86 U. Scarfogliero, C. Stefanini, P. Dario: Design and development of the long-jumping, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 467–472

23.87 M. Kovac, M. Fuchs, A. Guignard, J.-C. Zufferey, D. Floreano: A miniature 7g jumping robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 373–378

23.88 Y.-J. Park, T.M. Huh, D. Park, K.-J. Cho: Design of a variable-stiffness flapping mechanism for maximizing the thrust of a bio-inspired underwater robot, Bioinsp. Biomimet. **9**, 036002 (2014)

23.89 W.-S. Chu, K.-T. Lee, S.-H. Song, M.-W. Han, J.-Y. Lee, H.-S. Kim, M.S. Kim, Y.J. Park, K.J. Cho, S.H. Anh: Review of biomimetic underwater robots using smart actuators, Int. J. Prec. Eng. Manuf. **13**, 1281–1292 (2012)

23.90 Z. Wang, G. Hang, Y. Wang, J. Li, W. Du: Embedded SMA wire actuated biomimetic fin: A module for biomimetic underwater propulsion, Smart Mater. Struc. **17**, 025039 (2008)

23.91 G.V. Lauder, J. Lim, R. Shelton, C. Witt, E. Anderson, J.L. Tangorra: Robotic models for studying undulatory locomotion in fishes, Mar. Technol. Soc. J. **45**, 41–55 (2011)

23.92 F. Li, G. Bonsignori, U. Scarfogliero, D. Chen, C. Stefanini, W. Liu, P. Dario, F. Xin: Jumping mini-robot with bio-inspired legs, IEEE Int. Conf. Robotics Biomimet. (ROBIO) (2009) pp. 933–938

23.93 B.G.A. Lambrecht, A.D. Horchler, R.D. Quinn: A small, insect-inspired robot that runs and jumps, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1240–1245

23.94 J. Zhao, J. Xu, B. Gao, N. Xi, F.J. Cintrón, M.W. Mutka, X. Li: MSU Jumper: A single-motor-actuated miniature steerable jumping robot, IEEE Trans. Robotics **29**, 602–614 (2013)

23.95 J. Zhao, W. Yan, N. Xi, M.W. Mutka, L. Xiao: A miniature 25 grams running and jumping robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

23.96 R. Armour, K. Paskins, A. Bowyer, J. Vincent, W. Megill: Jumping robots: A biomimetic solution to locomotion across rough terrain, Bioinsp. Biomimet. **2**, S65–S82 (2007)

23.97 M. Noh, S.-W. Kim, S. An, J.-S. Koh, K.-J. Cho: Flea-inspired catapult mechanism for miniature jumping robots, IEEE Trans. Robotics **28**, 1007–1018 (2012)

23.98 J.-S. Koh, S.-P. Jung, M. Noh, S.-W. Kim, K.-J. Cho: Flea inspired catapult mechanism with active energy storage and release for small scale jumping robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 26–31

23.99 J.-S. Koh, S.-P. Jung, R.J. Wood, K.-J. Cho: A jumping robotic insect based on a torque reversal catapult mechanism, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 3796–3801

23.100 A. Yamada, M. Watari, H. Mochiyama, H. Fujimoto: An asymmetric robotic catapult based on the closed elastica for jumping robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 232–237

23.101 A.P. Gerratt, S. Bergbreiter: Incorporating compliant elastomers for jumping locomotion in microrobots, Smart Mater. Struct. **22**, 014010 (2013)

23.102 R. Niiyama, A. Nagakubo, Y. Kuniyoshi: Mowgli: A bipedal jumping and landing robot with an artificial musculoskeletal system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2546–2551

23.103 E.W. Hawkes, E.V. Eason, A.T. Asbeck, M.R. Cutkosky: The gecko's toe: Scaling directional adhesives for climbing applications, IEEE/ASME Trans. Mechatron. **18**, 518–526 (2013)

23.104 E.W. Hawkes, D.L. Christensen, E.V. Eason, M.A. Estrada, M. Heverly, E. Hilgemann, J. Hao, M.T. Pope, A. Parness, M.R. Cutkosky: Dynamic surface grasping with directional adhesion, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 5487–5493

23.105 A.L. Desbiens, A.T. Asbeck, M.R. Cutkosky: Landing, perching and taking off from vertical surfaces, Int. J. Robotics Res. **30**, 355–370 (2011)

23.106 A. Parness, M. Frost, N. Thatte, J.P. King, K. Witkoe, M. Nevarez, M. Garrett, H. Aghazarian, B. Kennedy: Gravity-independent rock-climbing robot and a sample acquisition tool with microspine grippers, J. Field Robotics **30**, 897–915 (2013)

23.107 B.A. Trimmer, A.E. Takesian, B.M. Sweet, C.B. Rogers, D.C. Hake, D.J. Rogers: Caterpillar locomotion: A new model for soft-bodied climbing and burrowing robots, 7th Int. Symp. Technol. Mine Problem (2006) pp. 1–10

23.108 G.-P. Jung, J.-S. Koh, K.-J. Cho: Underactuated adaptive gripper using flexural buckling, IEEE Trans. Robotics **29**(6), 1396 (2013)

23.109 M. Calisti, M. Giorelli, G. Levy, B. Mazzolai, B. Hochner, C. Laschi, P. Dario: An octopus-bioinspired solution to movement and manipulation for soft robots, Bioinsp. Biomimet. **6**, 036002 (2011)

23.110 S.-W. Kim, J.-S. Koh, J.-G. Lee, J. Ryu, M. Cho, K.-J. Cho: Flytrap-inspired robot using structurally integrated actuation based on bistability and a developable surface, Bioinsp. Biomimet. **9**, 036004 (2014)

23.111 C.E. Doyle, J.J. Bird, T.A. Isom, J.C. Kallman, D.F. Bareiss, D.J. Dunlop, R.J. King, J.J. Abbott, M.A. Minor: An avian-inspired passive mechanism for quadrotor perching, IEEE/ASME Trans. Mechatron. **18**, 506–517 (2013)

23.112 M. Kovač, J. Germann, C. Hürzeler, R.Y. Siegwart, D. Floreano: A perching mechanism for micro aerial vehicles, J. Micro-Nano Mechatron. **5**, 77–91 (2009)

23.113 R. Merz, F. Prinz, K. Ramaswami, M. Terk, L. Weiss: Shape deposition manufacturing, Proc. Solid Freeform Fabric. Symp., University of Texas at Austin (1994) pp. 1–8

23.114 S.A. Bailey, J.G. Cham, M.R. Cutkosky, R.J. Full: Biomimetic robotic mechanisms via shape deposition manufacturing, Robotics Res. Int. Symp. (2000) pp. 403–410

23.115 X. Li, A. Golnas, F.B. Prinz: Shape deposition manufacturing of smart metallic structures with embedded sensors, SPIE Proc. 7th Annu. Int. Symp. Smart Struct. Mater. (International Society for Optics and Photonics, Bellingham 2000) pp. 160–171

23.116 K.G. Marra, J.W. Szem, P.N. Kumta, P.A. DiMilla, L.E. Weiss: In vitro analysis of biodegradable polymer blend/hydroxyapatite composites for bone tissue engineering, J. Biomed. Mater. Res. **47**, 324–335 (1999)

23.117 J.G. Cham, S.A. Bailey, J.E. Clark, R.J. Full, M.R. Cutkosky: Fast and robust: Hexapedal robots via shape deposition manufacturing, Int. J. Robotics Res. **21**, 869–882 (2002)

23.118 A.M. Dollar, R.D. Howe: A robust compliant grasper via shape deposition manufacturing, IEEE/ASME Trans. Mechatron. **11**, 154–161 (2006)

23.119 M. Binnard, M.R. Cutkosky: Design by composition for layered manufacturing, J. Mech. Des. **122**, 91–101 (2000)

23.120 Y.-L. Park, K. Chau, R.J. Black, M.R. Cutkosky: Force sensing robot fingers using embedded fiber Bragg grating sensors and shape deposition manufacturing, IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 1510–1516

23.121 D. Shin, I. Sardellitti, Y.-L. Park, O. Khatib, M. Cutkosky: Design and control of a bio-inspired human-friendly robot, Int. J. Robotics Res. **29**, 571–584 (2010)

23.122 R.S. Fearing, K.H. Chiang, M.H. Dickinson, D.L. Pick, M. Sitti, J. Yan: Wing transmission for a micromechanical flying insect, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2000) pp. 1509–1516

23.123 J. Yan, R.J. Wood, S. Avadhanula, M. Sitti, R.S. Fearing: Towards flapping wing control for a micromechanical flying insect, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 3901–3908

23.124 R.J. Wood, S. Avadhanula, M. Menon, R.S. Fearing: Microrobotics using composite materials: The micromechanical flying insect thorax, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2003) pp. 1842–1849

23.125 K.Y. Ma, P. Chirarattananon, S.B. Fuller, R.J. Wood: Controlled flight of a biologically inspired, insect-scale robot, Science **340**, 603–607 (2013)

23.126 J. Whitney, P. Sreetharan, K. Ma, R. Wood: Pop-up book MEMS, J. Micromech. Microeng. **21**, 115021 (2011)

23.127 P.S. Sreetharan, J.P. Whitney, M.D. Strauss, R.J. Wood: Monolithic fabrication of millimeter-scale machines, J. Micromech. Microeng. **22**, 055027 (2012)

23.128 C. Majidi: Soft robotics: A perspective – Current trends and prospects for the future, Soft Robotics **1**, 5–11 (2013)

23.129 Y. Xia, G.M. Whitesides: Soft lithography, Annu. Rev. Mater. Sci. **28**, 153–184 (1998)

23.130 F. Ilievski, A.D. Mazzeo, R.F. Shepherd, X. Chen, G.M. Whitesides: Soft robotics for chemists, Angew. Chem. **123**, 1930–1935 (2011)

23.131 R.F. Shepherd, F. Ilievski, W. Choi, S.A. Morin, A.A. Stokes, A.D. Mazzeo, X. Chen, M. Wang, G.M. Whitesides: Multigait soft robot, Proc. Natl. Acad. Sci. **108**, 20400–20403 (2011)

23.132 B.C.-M. Chang, J. Berring, M. Venkataram, C. Menon, M. Parameswaran: Bending fluidic actuator for smart structures, Smart Mater. Struct. **20**, 035012 (2011)

23.133 B. Chang, A. Chew, N. Naghshineh, C. Menon: A spatial bending fluidic actuator: Fabrication and quasi-static characteristics, Smart Mater. Struct. **21**, 045008 (2012)

23.134 B. Finio, R. Shepherd, H. Lipson: Air-Powered Soft Robots for K-12 Classrooms, IEEE Proc. Integr. STEM Edu. Conf. (ISEC) (2013) pp. 1–6

23.135 S.A. Morin, R.F. Shepherd, S.W. Kwok, A.A. Stokes, A. Nemiroski, G.M. Whitesides: Camouflage and display for soft machines, Science **337**, 828–832 (2012)

23.136 R.V. Martinez, J.L. Branch, C.R. Fish, L. Jin, R.F. Shepherd, R. Nunes, Z. Suo, G.M. Whitesides: Robotic tentacles with three-dimensional mobility based on flexible elastomers, Adv. Mater. **25**, 205–212 (2013)

23.137 R.V. Martinez, C.R. Fish, X. Chen, G.M. Whitesides: Elastomeric origami: Programmable paper-elastomer composites as pneumatic actuators, Adv. Funct. Mater. **22**, 1376–1384 (2012)

23.138 S.W. Kwok, S.A. Morin, B. Mosadegh, J.H. So, R.F. Shepherd, R.V. Martinez, B. Smith, F.C. Simeone, A.A. Stokes, G.M. Whitesides: Magnetic assembly of soft robots with hard components, Adv. Funct. Mater. **24**, 2180–2187 (2013)

23.139 B. Mosadegh, P. Polygerinos, C. Keplinger, S. Wennstedt, R.F. Shepherd, U. Gupta, J. Shim, K. Bertoldi, C.J. Walsh, G.M. Whitesides: Pneumatic networks for soft robotics that actuate rapidly, Adv. Funct. Mater. **24**, 2163–2170 (2013)

23.140 M. Cianchetti, A. Arienti, M. Follador, B. Mazzolai, P. Dario, C. Laschi: Design concept and validation of a robotic arm inspired by the octopus, Mater. Sci. Eng. C **31**, 1230–1239 (2011)

23.141 T. Umedachi, V. Vikas, B.A. Trimmer: Highly deformable 3-D printed soft robot generating inching and crawling locomotions with variable friction legs, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 4590–4595

Multimedia Contents

24. Wheeled Robots

Woojin Chung, Karl Iagnemma

The purpose of this chapter is to introduce, analyze, and compare various wheeled mobile robots (WMRs) and to present several realizations and commonly encountered designs. The mobility of WMR is discussed on the basis of the kinematic constraints resulting from the pure rolling conditions at the contact points between the wheels and the ground. Practical robot structures are classified according to the number of wheels, and features are introduced focusing on commonly adopted designs. Omnimobile robot and articulated robots realizations are described. Wheel–terrain interaction models are presented in order to compute forces at the contact interface. Four possible wheel–terrain interaction cases are shown on the basis of relative stiffness of the wheel and terrain. A suspension system is required to move on uneven surfaces. Structures, dynamics, and important features of commonly used suspensions are explained.

24.1 **Overview** 575
24.2 **Mobility of Wheeled Robots** 576
24.2.1 Types of Wheels 576
24.2.2 Kinematic Constraints 578
24.2.3 Robot Configuration Variables 580
24.2.4 Restriction on Robot Mobility 580
24.2.5 Characterization of Robot Mobility 581
24.2.6 Five Classes of Wheeled Mobile Robots 581
24.3 **Wheeled Robot Structures** 582
24.3.1 Robots with One Wheel 582
24.3.2 Robots with Two Wheels 582
24.3.3 Robots with Three Wheels 583
24.3.4 Robots with Four Wheels 585
24.3.5 Special Applications of Wheeled Robots 585
24.4 **Wheel–Terrain Interaction Models** 586
24.4.1 Rigid Wheels on Rigid Terrain 586
24.4.2 Rigid Wheels on Deformable Terrain 586
24.4.3 Deformable Tires on Deformable Terrain 587
24.4.4 Deformable Tires on Rigid Terrain 589
24.5 **Wheeled Robot Suspensions** 589
24.5.1 Passive Suspension Dynamics 590
24.5.2 Semiactive Suspension Dynamics 591
24.6 **Conclusions** 592
Video-References 592
References 593

24.1 Overview

Wheeled robots have been widely used to achieve mobility because there are many advantages including the simple structure, energy efficiency, fast speed, low fabrication cost, and so forth. The purpose of this chapter is to provide a general description of wheeled mobile robots, to discuss their properties from the mobility point of view, to describe the most commonly encountered realizations of such robots, and to explain wheel–terrain interaction models and suspension systems.

The chapter is organized as follows: Sect. 24.2 is devoted to the characterization of the restriction of robot motion induced by these pure rolling conditions. We first describe different types of wheels used in the construction of mobile robots and derive the corresponding kinematic constraints. This allows us to characterize the mobility of a robot equipped with several wheels of these different types, and we show that these robots can be classified only into five categories, corresponding to two mobility indices.

In Sect. 24.3, we present several realizations of wheeled mobile robots by the use of various types of wheels. Structures and features of widely used wheel mechanisms are explained. Design issues of wheeled robots include capability of omnidirectional movement, fabrication cost, difficulty of control, and terrain traversability.

Section 24.4 explains wheel–terrain interaction models. It is shown that four possible wheel–terrain interaction cases can be obtained according to the relative stiffness of the wheel and terrain. The presented analytical models are broadly suitable for analysis, simulation, design, and control purposes.

Section 24.5 presents suspensions for wheeled robots. A suspension is a system of linkages, springs, dampers, and actuators that governs the relative motion between a robot's wheels and body. Suspension mechanisms are useful when driving over uneven surfaces. Common methods for modeling passive and semiactive suspensions are presented. The analysis of these models allows for principled selection of springs, dampers, and active or semiactive actuators that form the basis of nearly all wheeled robot suspensions.

Finally, some concluding remarks are given in Sect. 24.6.

24.2 Mobility of Wheeled Robots

In this section, we describe a variety of wheels and wheel implementations in mobile robots. We discuss the restriction of robot mobility implied by the use of these wheels and deduce a classification of robot mobility allowing one to characterize robot mobility fully, whatever the number and type of the wheels.

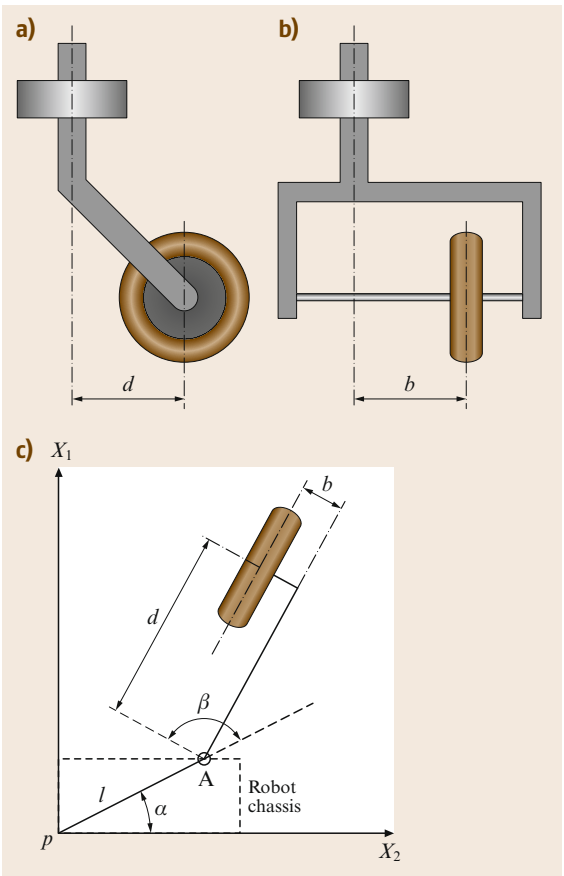


Fig.24.1a–c The general design of a standard wheel. **(a)** Side view, **(b)** front view, and **(c)** top view

24.2.1 Types of Wheels

In order to achieve robot locomotion, wheeled mobile robots are widely used in many applications. In general, wheeled robots consume less energy and move faster than other locomotion mechanisms (e.g., legged robots or tracked vehicles). From the viewpoint of control, less control effort is required, owing to their simple mechanisms and reduced stability problems. Although it is difficult to overcome rough terrain or uneven ground conditions, wheeled mobile robots are suitable for a large class of target environments in practical applications. When we think of a single-wheel design, there are two candidates: a standard wheel or a special wheel. A standard wheel can be understood as a conventional tire. Special wheels possess unique mechanical structures including rollers or spheres. Figure 24.1 shows the general design of a standard wheel. Three conditions should be defined for a standard wheel design:

1. Determination of the two offsets d and b
2. Mechanical design that allows steering motion or not (i. e., to fix the wheel orientation or not)
3. Determination of steering and driving actuation (i. e., active or passive drive).

Condition 1 is the kinematic parameter design problem for a single standard wheel. The parameter d can be either 0 or some positive constant. Parameter b is the lateral offset of the wheel and is usually set to zero. In a special design, a nonzero b may be selected to obtain pure rolling contact between the wheel and ground without causing rotational slip at the contact point. However, this is rarely used and we mainly consider the case of zero lateral offset b.

Condition 2 is a design problem for whether the wheel orientation can be changed or not. If the steering axis is fixed, the wheel provides a velocity constraint on the driving direction. Condition 3 is the design problem of whether to actuate steering or driving motion by actuators or to drive steering or motion passively.

If steering motion is allowed, the offset d plays a significant role in the kinematic modeling. For a conventional caster wheel (i. e., an off-centered orientable wheel), there is a nonzero offset d. Point A in Fig. 24.1 indicates the location of the joint connecting the wheel module to the robot chassis. Two orthogonal linear velocity components at point A are obtained by a caster wheel, which results from the steering and driving motions of the wheel module. This implies that a passive caster wheel does not provide an additional velocity constraint on the robot's motion. If a caster wheel is equipped with two actuators that drive steering and driving motions independently, holonomic omnidirectional movement can be achieved because any desired velocity at point A can be generated by solving the inverse kinematics problem.

If the offset d is set to zero, the allowable velocity direction at point A is limited to the wheel orientation. In such a case, the steering motion should not be passive because the wheel orientation cannot be changed passively. However, the driving velocity can be determined passively by the actuation of other wheels. Wheel orientation should be actively steered to the desired velocity direction due to the nonholonomic velocity constraint. This implies that the wheel orientation should be aligned before movement.

In summary, four types of standard wheels are commonly used. First is a passively driven wheel with a fixed steering axis. Second is a passive caster wheel with offset d. Third is an active caster wheel with offset d, where the steering and driving motions are controlled by actuators. An example of caster wheels is shown in VIDEO 325. The fourth is an active orientable wheel with zero offset d, where steering and driving motions are driven by actuators. The structures of each wheel type are shown in Fig. 24.2. The kinematics and constraints of those wheels will be explained in detail in Sect. 24.2.2.

Although standard wheels are advantageous because of their simple structure and good reliability, the nonholonomic velocity constraint (i. e., no sideslip condition) limits robot motion. On the other hand, special wheels can be employed in order to obtain omnidirectional motion of a mobile robot (omnimobile robot), i. e., to ensure three degrees of freedom for plane motion. We consider two typical designs of special wheels: the Swedish wheel and the spherical wheel. Figure 24.3a and VIDEO 328 show the

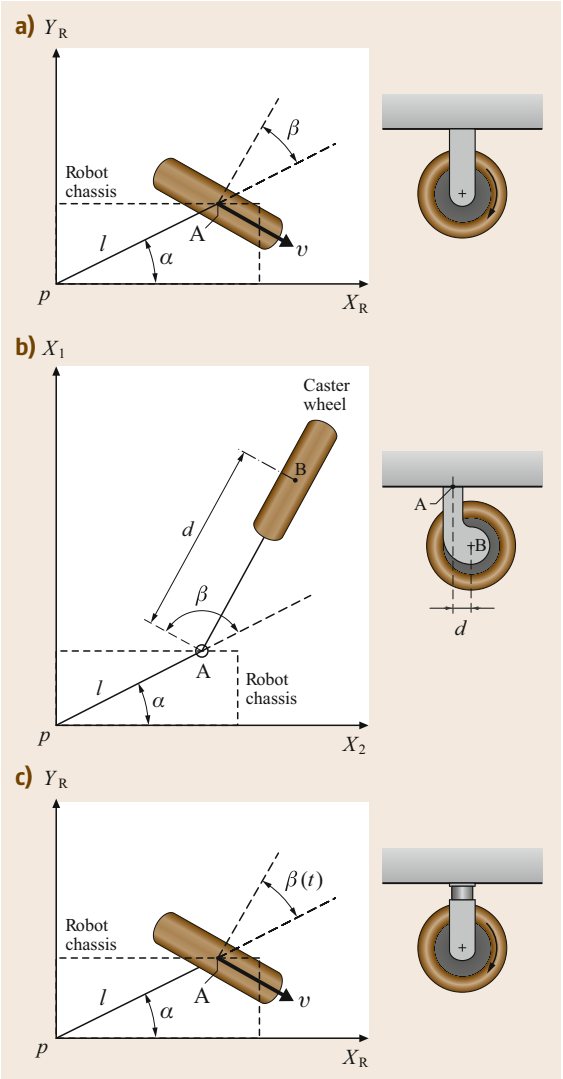


Fig.24.2a–c Structures of standard wheels. **(a)** Passive fixed wheel, **(b)** passive or active, off-centered orientable wheel, and **(c)** active orientable wheel without offsets

Swedish wheel. Small passive free rollers are located along the outer rim of the wheel. Free rollers are employed in order to eliminate the nonholonomic velocity constraint. Passive rollers are free to rotate around the axis of rotation, which results in lateral motion of the wheel. As a result, a driving velocity should be controlled, while the lateral velocity is passively determined by the actuation of the other wheels. A similar design can be seen in VIDEO 327.

A spherical wheel is shown in Fig. 24.3c. The rotation of the sphere is constrained by rollers that make rolling contact with the sphere. The rollers can be divided into driving and supporting rollers. The sphere is

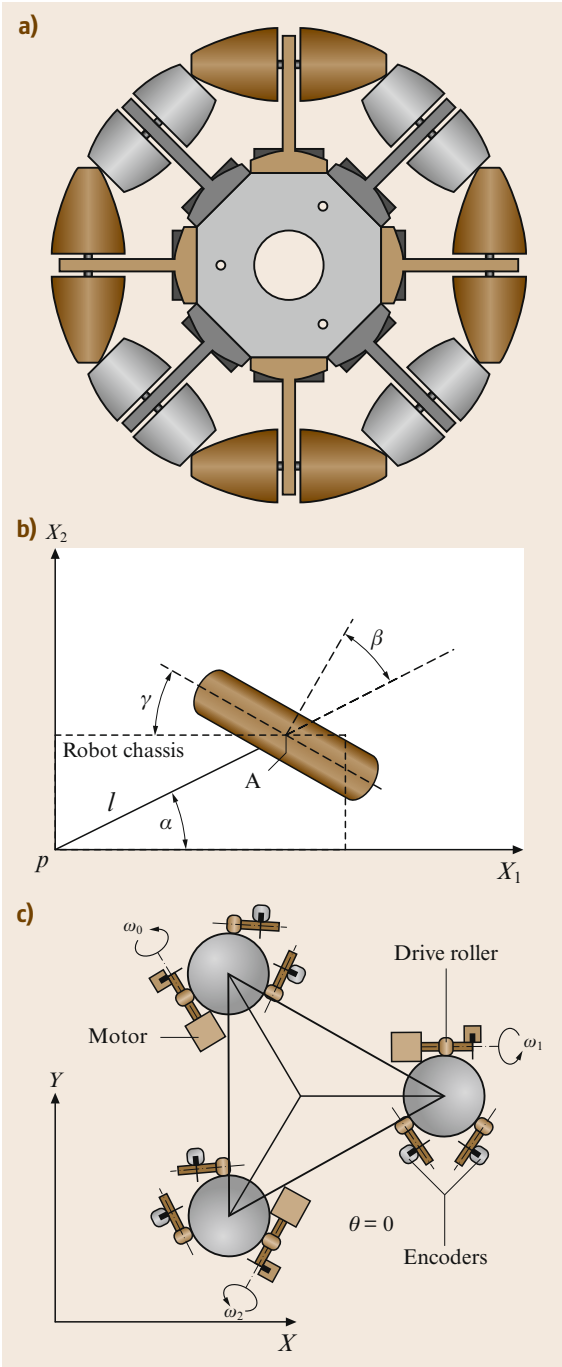


Fig. 24.3 (a) Swedish wheel, (b) attachment of a Swedish wheel, and (c) spherical wheel (after [24.1])

driven by the actuation of the driving rollers, whereas the rolling contacts provide nonholonomic constraints, and the resultant motion of the sphere module becomes holonomic. This implies that the robot can be moved with any desired linear/angular velocities at any time. By using the spherical wheel, a holonomic omnidirectional mobile robot can be developed and the robot achieves smooth and continuous contact between the sphere and the ground. However, the design of the sphere-supporting mechanism is difficult and the payload must be quite low due to the point contact. Another drawback is that the surface of the sphere can be polluted when traveling over dirty ground and it is difficult to overcome irregular ground conditions. These drawbacks limit the practical application of the spherical wheel. An example of the use of spherical wheels can be found in [24.1] and [24.2]. The spherical structure can also be applied to special robotic transmissions; examples include the nonholonomic manipulator in [24.3] and the passive haptic system in [24.4].

24.2.2 Kinematic Constraints

We assume, as a first step, that the mobile robot under study is made up of a rigid cart equipped with nondeformable wheels, and that it is moving on a horizontal plane. The position of the robot on the plane is described, with respect to an arbitrary inertial frame, by the *posture* vector $\boldsymbol{\xi} = (x \quad y \quad \theta)^{\mathrm{T}}$, where x and y are the coordinates of a reference point P of the robot cart, while θ describes the orientation of a mobile frame attached to the robot, with respect to the inertial frame (Fig. 24.4).

We assume that, during motion, the plane of each wheel remains vertical and the wheel rotates around its horizontal axle, whose orientation with respect to the cart can be fixed or varying. We distinguish between the two basic classes of idealized wheels, namely conventional and the Swedish wheels. In each case,

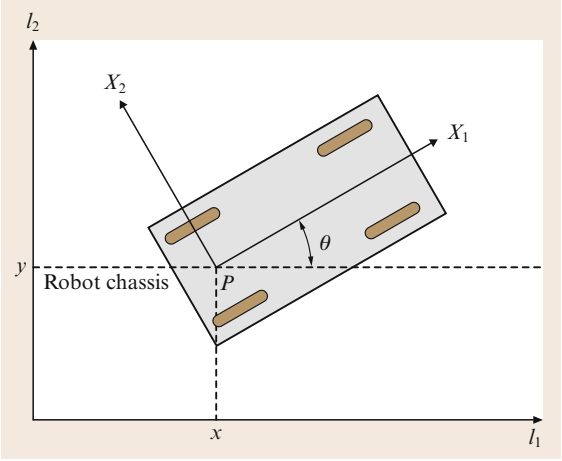


Fig. 24.4 The posture definition of a mobile robot on a plane

it is assumed that the contact between the wheel and the ground is reduced to a single point. The kinematic constraints result from the fact that the velocity of the material point of the wheel in contact with the ground is equal to zero.

For a conventional wheel, the kinematic constraints imply that the velocity of the center of the wheel is parallel to the wheel plane (*nonslip condition*) and is proportional to the wheel rotation velocity (*pure rolling condition*). For each wheel, the kinematic constraints therefore result in two independent conditions. For a Swedish wheel, due to the relative rotation of the rollers with respect to the wheel, only one of the velocity components of the wheel contact point is zero. The direction of this zero component is fixed with respect to the wheel plane and depends on the wheel construction. For such wheels, the kinematic constraints result in only one condition.

Conventional Wheels

We now derive the general form of the kinematic constraints for a conventional wheel.

As shown in Fig. 24.2, there are several variations of the conventional wheel design. First, we focus on the off-centered orientable wheel in Fig. 24.2b. The center of the wheel, B, is connected to the cart by a rigid rod from A (a fixed point on the cart) to B, aligned with the wheel plane. The rod, whose length is denoted by d, can rotate around a fixed vertical axle at point A. The position of A is specified by two constant polar coordinates, l and α, with respect to the reference point P. The rotation of the rod with respect to the cart is represented by the angle β. The radius of the wheel is denoted by r, and its angle of rotation around its horizontal axle is denoted by φ. The description therefore involves four constant parameters: α, l, r, and d, and two variables: $\varphi(t)$ and $\beta(t)$.

With these notations the kinematic constraints are derived as follows.

We make the derivation explicit for the general situation corresponding to a caster wheel (Fig. 24.2b). For fixed or steering wheels one just has to consider either the case $d = 0$ and constant β (fixed wheels), or $d = 0$ and variable β (steering wheels).

First, we evaluate the velocity of the center of the wheel, which results from the following vector expression $\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{OB} = \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{OP} + \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{PA} + \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{AB}$. The two components of this vector in the robot frame are expressed as

$$\dot{x}\cos\theta + \dot{y}\sin\theta - l\dot{\theta}\sin\alpha + (\dot{\theta}+\dot{\beta})\mathrm{d}\cos(\alpha+\beta)$$

and

$$-\dot{x}\sin\theta + \dot{y}\cos\theta - l\,\dot{\theta}\cos\alpha + (\dot{\theta}+\dot{\beta})\mathrm{d}\sin(\alpha+\beta)\,.$$

The projections of this vector onto the direction of the wheel plane, i. e., onto the vector $(\cos(\alpha+\beta-\pi/2), \sin(\alpha+\beta-\pi/2))$ and the vector of the wheel axle $(\cos(\alpha+\beta), \sin(\alpha+\beta))$, are $r\dot{\boldsymbol{\varphi}}$ and 0, respectively, corresponding to the pure rolling and nonslip conditions.

After some manipulations, these conditions can be rewritten in the following compact form.

Pure Rolling Condition.

$$(-\sin(\alpha+\beta)\ \ \cos(\alpha+\beta)\ \ l\cos\beta)\mathbf{R}(\theta)\dot{\boldsymbol{\xi}} + r\dot{\boldsymbol{\varphi}} = 0\,. \tag{24.1}$$

Nonslip Condition.

$$(-\cos(\alpha+\beta)\ \ \sin(\alpha+\beta)\mathrm{d} + l\sin\beta)\mathbf{R}(\theta)\dot{\boldsymbol{\xi}} + d\dot{\boldsymbol{\beta}} = 0\,. \tag{24.2}$$

In the earlier given expressions, $\mathbf{R}(\theta)$ is the orthogonal rotation matrix expressing the orientation of the robot with respect to the inertial frame, i. e.,

$$\mathbf{R}(\theta) = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{24.3}$$

As said before, these general expressions can be simplified for different types of conventional wheels.

For *fixed wheels*, the center of the wheel is fixed with respect to the cart and the wheel orientation is constant. This corresponds to a constant value of β and $d = 0$ (Fig. 24.2a). The nonslip equation (24.2) then reduces to

$$(\cos(\alpha+\beta)\ \ \sin(\alpha+\beta)\ \ l\sin\beta)\mathbf{R}(\theta)\dot{\boldsymbol{\xi}} = 0\,. \tag{24.4}$$

For *steering wheels*, the center of the wheel is also fixed with respect to the cart (i. e., $d = 0$), with β time-varying, so the nonslip equation takes the form (24.2). This structure was already introduced in Fig. 24.2c.

The situation described by (24.1) and (24.2), with a nonzero-length rod AB and time-varying orientation angle β corresponds to *caster wheels*.

Swedish Wheels. The position of a Swedish wheel with respect to the cart is described, as for fixed wheels, by three constant parameters: α, β, and l. An additional parameter is required to characterize the direction, with respect to the wheel plane, of the zero component of the velocity at the contact point of the wheel. This parameter is γ, which is the angle between the axle of the rollers and the wheel plane (Fig. 24.3b).

The kinematic constraints now impose only one condition

$$[-\sin(\alpha+\beta+\gamma) \quad \cos(\alpha+\beta+\gamma) \quad l\cos(\beta+\gamma)] \times \mathbf{R}(\theta)\dot{\boldsymbol{\xi}} + r\cos\gamma\,\dot{\boldsymbol{\varphi}} = 0 \tag{24.5}$$

24.2.3 Robot Configuration Variables

We now consider a wheeled robot equipped with N wheels of the earlier described types. We use the following subscripts to identify quantities related to these four types: f for fixed wheels, s for steering wheels, c for caster wheels, and sw for Swedish wheels. The number of wheels of each type are denoted by N_f, N_s, N_c, and N_sw, with $N = N_\mathrm{f} + N_\mathrm{s} + N_\mathrm{c} + N_\mathrm{sw}$.

The configuration of the robot is fully described by the following generalized coordinate vector:

- *Posture coordinates*: the posture vector $\boldsymbol{\xi}(t) = (x(t)\ \ y(t)\ \ \theta(t))^\mathrm{T}$
- *Orientation coordinates*: the $N_\mathrm{s}+N_\mathrm{c}$ orientation angles of the steering and caster wheels, i. e., $\boldsymbol{\beta}(t) = (\beta_\mathrm{s}(t)\ \beta_\mathrm{c}(t))^\mathrm{T}$
- *Rotation coordinates*: the N rotation angles of the wheels, i. e., $\boldsymbol{\varphi}(t) = (\varphi_\mathrm{f}(t)\ \varphi_\mathrm{s}(t)\ \varphi_\mathrm{c}(t)\ \varphi_\mathrm{sw}(t))^\mathrm{T}$.

This whole set of coordinates is termed as the set of configuration coordinates. The total number of configuration coordinates is $N_\mathrm{f} + 2N_\mathrm{s} + 2N_\mathrm{c} + N_\mathrm{sw} + 3$.

24.2.4 Restriction on Robot Mobility

The *pure rolling conditions* for fixed, steering, and caster wheels, as well as the constraints relative to the Swedish wheels, can be written in the following compact form

$$\mathbf{J}_1(\beta_\mathrm{s}, \beta_\mathrm{c})\mathbf{R}(\theta)\dot{\boldsymbol{\xi}} + \mathbf{J}_2\dot{\boldsymbol{\varphi}} = 0\,, \tag{24.6}$$

with

$$\mathbf{J}_1(\boldsymbol{\beta}_\mathrm{s}, \boldsymbol{\beta}_\mathrm{c}) = \begin{pmatrix} \mathbf{J}_{1\mathrm{f}} \\ \mathbf{J}_{1\mathrm{s}}(\beta_\mathrm{s}) \\ \mathbf{J}_{1\mathrm{c}}(\beta_\mathrm{c}) \\ \mathbf{J}_{1\mathrm{sw}} \end{pmatrix}.$$

In this expression, $\mathbf{J}_{1\mathrm{f}}$, $\mathbf{J}_{1\mathrm{s}}(\beta_\mathrm{s})$, $\mathbf{J}_{1\mathrm{c}}(\beta_\mathrm{c})$, and $\mathbf{J}_{1\mathrm{sw}}$ are, respectively, $(N_\mathrm{f}\times 3)$, $(N_\mathrm{s}\times 3)$, $(N_\mathrm{c}\times 3)$, and $(N_\mathrm{sw}\times 3)$ matrices, whose forms derive directly from the kinematic constraints, while $\mathbf{J}_2$ is a constant $(N\times N)$ diagonal matrix whose entries are the radii of the wheels, except for the radii of the Swedish wheels which are multiplied by $\cos\gamma$.

The value $\gamma = \pi/2$ would correspond to the direction of the zero component of the velocity being orthogonal to the plane of the Swedish wheel. Such a wheel would be subject to a constraint identical to the nonslip condition for a conventional wheel; hence, losing the benefit of implementing a Swedish wheel. This implies that $\gamma \neq \frac{\pi}{2}$ and that $\mathbf{J}_2$ is a nonsingular matrix.

The *nonslip conditions* for caster wheels can be summarize as

$$\mathbf{C}_{1\mathrm{c}}(\beta_\mathrm{c})\mathbf{R}(\theta)\dot{\boldsymbol{\xi}} + \mathbf{C}_{2\mathrm{c}}\dot{\beta}_\mathrm{c} = 0\,, \tag{24.7}$$

where $\mathbf{C}_{1\mathrm{c}}(\beta_\mathrm{c})$ is a $(N_\mathrm{c}\times 3)$ matrix, whose entries derive from the nonslip constraints (24.2), while $\mathbf{C}_{2\mathrm{c}}$ is a constant diagonal nonsingular matrix, whose entries are equal to d.

The last constraints relate to the *nonslip conditions for fixed and steering wheels*. They can be summarized as

$$\mathbf{C}_1^*(\beta_\mathrm{s})\mathbf{R}(\theta)\dot{\boldsymbol{\xi}} = 0\,, \tag{24.8}$$

where

$$\mathbf{C}_1^*(\beta_\mathrm{s}) = \begin{pmatrix} \mathbf{C}_{1\mathrm{f}} \\ \mathbf{C}_{1\mathrm{s}}(\beta_\mathrm{s}) \end{pmatrix},$$

where $\mathbf{C}_{1\mathrm{f}}$ and $\mathbf{C}_{1\mathrm{s}}(\beta_\mathrm{s})$ are, respectively, $(N_\mathrm{f}\times 3)$ and $(N_\mathrm{s}\times 3)$ matrices.

It is important to point out that the restrictions on robot mobility result only from the conditions (24.8) involving the fixed and the steering wheels. These conditions imply that the vector $\mathbf{R}(\theta)\dot{\boldsymbol{\xi}}$ belongs to $N[\mathbf{C}_1^*(\beta_\mathrm{s})]$, the null space of the matrix $\mathbf{C}_1^*(\beta_\mathrm{s})$. For any $\mathbf{R}(\theta)\dot{\boldsymbol{\xi}}$ satisfying this condition, there exists a vector $\dot{\boldsymbol{\varphi}}$ and a vector $\dot{\boldsymbol{\beta}}_\mathrm{c}$ satisfying, respectively, conditions (24.6) and (24.7), because $\mathbf{J}_2$ and $\mathbf{C}_{2\mathrm{c}}$ are nonsingular matrices.

Obviously $\mathrm{rank}[\mathbf{C}_1^*(\beta_\mathrm{s})] \leq 3$. If it is equal to 3 then $\mathbf{R}(\theta)\dot{\boldsymbol{\xi}} = 0$, which means that any motion in the plane is impossible. More generally, restrictions on robot mobility are related to the rank of $\mathbf{C}_1^*(\beta_\mathrm{s})$, as will be discussed as follows in detail.

It is worth noticing that condition (24.8) has a direct geometrical interpretation. At each time instant the motion of the robot can be viewed as an instantaneous rotation about the instantaneous center of rotation (ICR), whose position with respect to the cart can be time varying. At each instant the velocity of any point of the cart is orthogonal to the straight line joining this point and the ICR. This is true, in particular, for the centers of the fixed and steering wheels, which are fixed points of the cart. On the other hand, the nonslip condition implies that the velocity of the wheel center is aligned with the wheel plane. These two facts imply that

the horizontal rotation axles of the fixed and steering wheels intersect at the ICR (Fig. 24.5). This is equivalent to the condition that $\text{rank}[\mathbf{C}_1^*(\beta_s)] \leq 3$.

24.2.5 Characterization of Robot Mobility

As said earlier, the mobility of the robot is directly related to the rank of $\mathbf{C}_1^*(\boldsymbol{\beta}_s)$, which depends on the design of the robot. We define the *degree of mobility* δ_m as

$$\delta_m = 3 - \text{rank}[\mathbf{C}_1^*(\boldsymbol{\beta}_s)] \,. \tag{24.9}$$

Let us first examine the case $\text{rank}(\mathbf{C}_{1f}) = 2$, which implies that the robot has at least two fixed wheels. If there are more than two fixed wheels, their axles intersect at the ICR, whose position with respect to the cart is then fixed in such a way that the only possible motion is a rotation of the cart about this fixed ICR. Obviously, from the user's point of view, such a design is not acceptable. We therefore assume that $\text{rank}(\mathbf{C}_{1f}) \leq 1$.

Moreover, we assume that

$$\text{rank}[\mathbf{C}_1^*(\boldsymbol{\beta}_s)] = \text{rank}(\mathbf{C}_{1f}) + \text{rank}[\mathbf{C}_{1s}(\boldsymbol{\beta}_s)] \leq 2 \,.$$

These two assumptions are equivalent to the following set of conditions:

1. If the robot has more than one fixed wheel, they are all on a single common axle.
2. The centers of the steering wheels do not belong to this common axle of the fixed wheels.
3. The number $\text{rank}[\mathbf{C}_{1s}(\boldsymbol{\beta}_s)]$ is equal to the number of steering wheels that can be oriented independently in order to steer the robot.

We call this number the *degree of steerability*

$$\delta_s = \text{rank}[\mathbf{C}_{1s}(\boldsymbol{\beta}_s)] \,. \tag{24.10}$$

If a robot is equipped with more than δ_s steering wheels, the motion of the extra wheels must be coordinated in order to guarantee the existence of the ICR at each instant.

We conclude that, for wheeled mobile robot of practical interest, the two defined indices, δ_m and δ_s, satisfy the following conditions:

1. The degree of mobility satisfies $1 \leq \delta_m \leq 3$. The upper bound is obvious, while the lower bound means that we consider only cases where motion is possible.
2. The degree of steerability satisfies $0 \leq \delta_s \leq 2$. The upper bound can be reached only for robots without fixed wheels, while the lower bound corresponds to robots without steering wheels.
3. The following is satisfied: $2 \leq \delta_m + \delta_s \leq 3$.

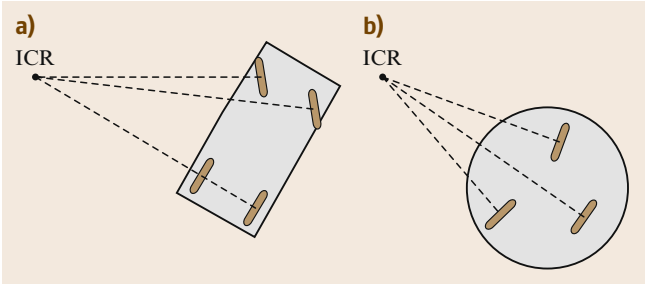


Fig.24.5a,b The instantaneous center of rotation. **(a)** A car-like robot; **(b)** a three-steering-wheels robot

The case $\delta_m + \delta_s = 1$ is not acceptable because it corresponds to the rotation of the robot about a fixed ICR. The cases $\delta_m \geq 2$ and $\delta_s = 2$ are excluded because according to the assumptions, $\delta_s = 2$ implies $\delta_m = 1$. These conditions imply that only five structures are of practical interest, corresponding to the five pairs (δ_m, δ_s) satisfying the aforementioned inequalities, according to the following array

δ_m	3	2	2	1	1
δ_s	0	0	1	1	2

In the following, each type of structure will be designated by using a denomination of the form type (δ_m, δ_s) robot.

24.2.6 Five Classes of Wheeled Mobile Robots

We now briefly describe the five classes of wheeled robot structures, pointing out the mobility restriction inherent to each class. Details and examples can be found in Sect. 24.3 and in [24.5].

Type (3,0) Robots

These robots have no fixed and no steering wheels and are equipped only with Swedish or caster wheels. Such robots are called *omnimobile*, because they have full mobility in the plane, which means that they are able to move in any direction without any reorientation.

Type (2,0) Robots

These robots have no steering wheels, but either one or several fixed wheels with a common axle. Mobility is restricted in the sense that, at a given posture $\xi(t)$, the velocity $\dot{\boldsymbol{\xi}}(t)$ is constrained to belong to a two-dimensional distribution spanned by the vector fields $\mathbf{R}^T(\theta)\boldsymbol{s}_1$ and $\mathbf{R}^T(\theta)\boldsymbol{s}_2$, where $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ are two constant vectors spanning $N(\mathbf{C}_{1f})$. A typical example of such a robot is the wheelchair.

Type (2,1) Robots

These robots have no fixed wheels and at least one steering wheel. If there is more than one steering wheel, their orientations must be coordinated in such a way that $\text{rank}[\mathbf{C}_{1s}(\boldsymbol{\beta}_s)] = \delta_s = 1$. The velocity $\dot{\boldsymbol{\xi}}(t)$ is constrained to belong to a two-dimensional distribution spanned by the vector fields $\mathbf{R}^T(\theta)\boldsymbol{s}_1(\boldsymbol{\beta}_s)$ and $\mathbf{R}^T(\theta)\boldsymbol{s}_2(\boldsymbol{\beta}_s)$, where $\boldsymbol{s}_1(\beta_s)$ and $\boldsymbol{s}_2(\beta_s)$ are two vectors spanning $N(\mathbf{C}_{1s}(\beta_s))$.

Type (1,1) Robots

These robots have one or several fixed wheels on a single common axle, and also one or several steering wheels, with the conditions that their centers are not located on the common axle of the fixed wheels, and that their orientations are coordinated. The velocity $\dot{\boldsymbol{\xi}}(t)$ is constrained to belong to a one-dimensional distribution parameterized by the orientation angle of one arbitrarily chosen steering wheel. Mobile robots built on the model of a conventional car (often called *car-like* robots) belong to this class.

Type (1,2) Robots

These robots have no fixed wheels, but at least two steering wheels. If there are more than two steering wheels, then their orientation must be coordinated in order to satisfy the condition $\text{rank}[\mathbf{C}_{1s}(\boldsymbol{\beta}_s)] = \delta_s = 2$. The velocity $\dot{\boldsymbol{\xi}}(t)$ is constrained to belong to a one-dimensional distribution parameterized by the orientation angles of two arbitrarily chosen steering wheels.

24.3 Wheeled Robot Structures

There are many design alternatives for wheeled mobile robots. Design problems of a single-body mobile robot include the selection of wheel types, the placement of wheels, and the determination of the kinematic parameters. Design objectives should be specified according to the target environments and tasks, as well as the initial and operational costs of a robot. In this section, robot structures are classified according to the number of wheels, and then features will be introduced focusing on commonly adopted designs.

24.3.1 Robots with One Wheel

A robot with a single wheel is basically unstable without dynamic control in order to maintain its balance of the body. A typical example is a unicycle. As a variation of a unicycle, a robot with a rugby-ball-shaped wheel can be used in order to improve stability in the lateral direction, as studied in [24.6].

A spherical robot can also be considered as a single-wheel robot. A balancing mechanism such as a spinning wheel is employed to achieve dynamic stability. This approach has advantages including high maneuverability and low rolling resistance. However, single-wheel robots are rarely used in practical applications, because additional balancing mechanisms are required, control is difficult, and pose estimation by pure dead reckoning is not available. An example of a spherical robot can be found in [24.7].

24.3.2 Robots with Two Wheels

In general, there are two types of two-wheel robots, as shown in Fig. 24.6. Figure 24.6a shows a bicycle-type robot. It is common to steer a front wheel and to drive a rear wheel. Since the dynamic stability of a bicycle-type robot increases with its speed, a balancing mechanism is not necessarily required. The advantage of this approach is that the robot width can be reduced. However, a bicycle type is rarely used because it cannot maintain its pose when the robot stands still. Figure 24.6b shows an inverted-pendulum-type robot. It is a two-wheel differential drive robot.

It is possible to achieve static stability by accurately placing the center of gravity on the wheel axle. However, it is common to apply dynamic balancing control, which is similar to the conventional control problem for an inverted pendulum. The size of a robot can be re-

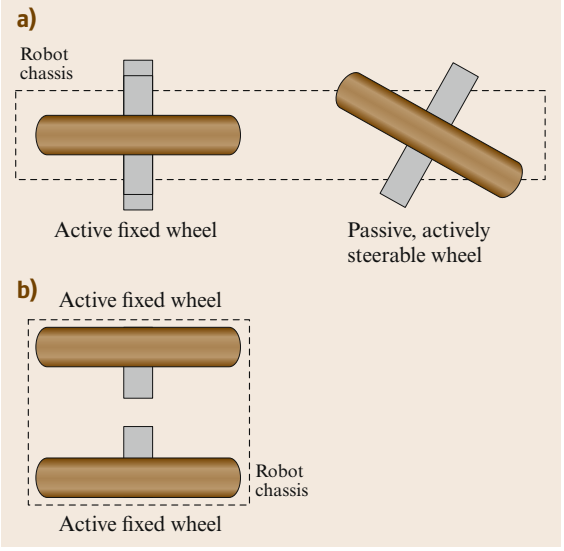


Fig. 24.6 (a) Bicycle-type robot and (b) inverted-pendulum-type robot

duced by using two-wheel robots, when compared with robots with more than three wheels. A typical application of a pendulum-type robot is to design a structure as a four-wheel robot, consisting of two pendulum robots connected. Then, the robot can climb stairs by lifting its front wheels while the robot reaches the stair. A major disadvantage is that control effort is always required for dynamic balancing. Examples of inverted-pendulum-type robots can be found in [24.8] and [24.9].

24.3.3 Robots with Three Wheels

Since a robot with three wheels is statically stable and has a simple structure, it is one of the most widely used structures for wheeled robots. There are a large number of designs according to the choice of individual wheel types. Every wheel introduced in Sect. 24.2.1 can be used to construct three-wheel robots. In this section, five popular design examples are described (Fig. 24.7):

1. Two-wheel differential drive
2. Synchronous drive
3. An omnimobile robot with Swedish wheels
4. An omnimobile robot with active caster wheels
5. An omnidirectional robot with steerable wheels.

Two-Wheel Differential-Drive Robot

A two-wheel differential-drive robot is one of the most popular designs and is composed of two active fixed wheels and one passive caster wheel. The robot can be classified as a type (2,0) robot in the nomenclature of Sect. 24.2.6. It is possible to extend the robot to a four-wheel robot by adding passive caster wheels. The major advantages of the robot can be summarized as follows:

- A simple mechanical structure, a simple kinematic model, and low fabrication cost.
- A zero turning radius is available. For a cylindrical robot, the obstacle-free space can easily be computed by expanding obstacle boundaries by the robot radius r.
- Systematic errors are easy to calibrate.

On the other hand, its drawbacks are:

- Difficulty of moving irregular surfaces. When the robot goes over uneven surfaces, its orientation

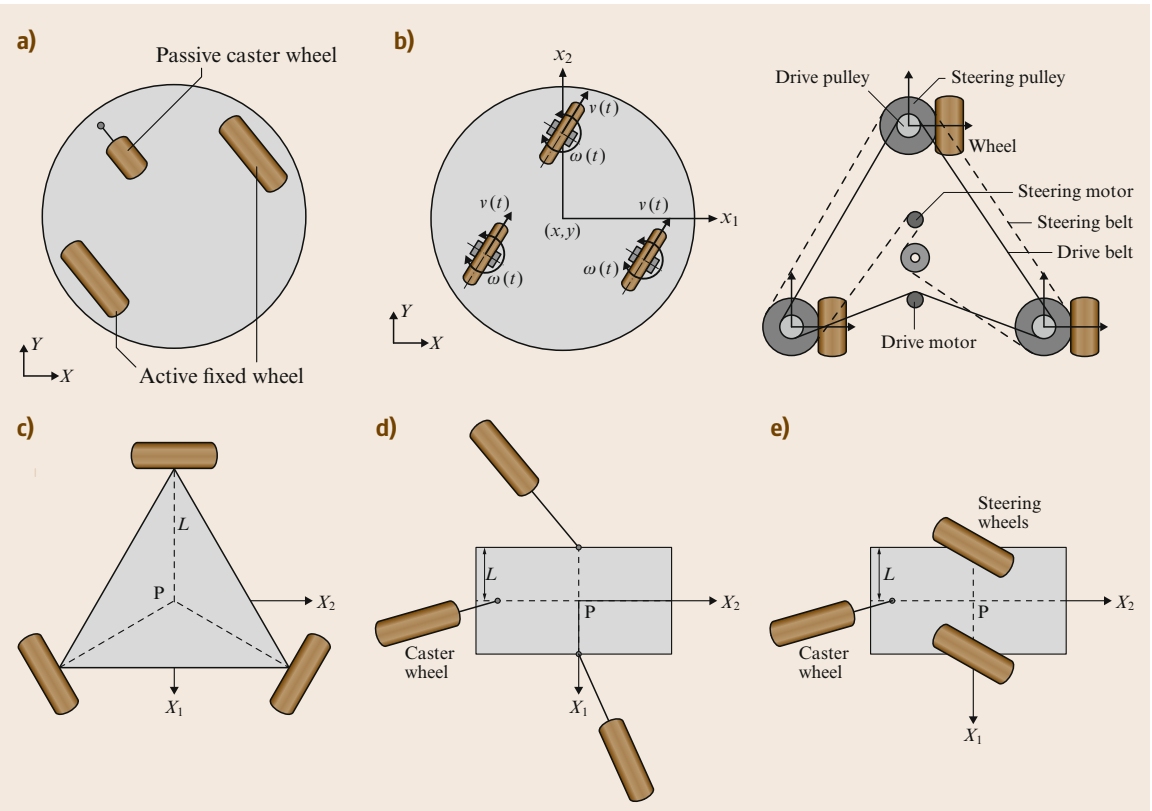


Fig. 24.7 (a) Two-wheel differential drive, (b) synchronous drive, (c) omnimobile robot with Swedish wheels, (d) omnimobile robot with active caster wheels, and (e) omnidirectional robot with active steerable wheels

might change abruptly if one of the active wheels loses contact with the ground.
- Only bidirectional movement is available.

Synchronous-Drive Robot

A synchronous-drive robot can be built by using centered or off-centered orientable wheels. The steering and driving motions of each wheel are mechanically coupled by chains or belts, and the motions are actuated synchronously, so the wheel orientations are always identical. The kinematic model of a synchronous drive robot is equivalent to that of the unicycle, a type (1,1) robot. Therefore, omnidirectional motion, i. e., motion in any direction can be achieved by steering the wheel orientations to the desired velocity direction. However, the orientation of the robot chassis cannot be changed. Sometimes a turret is employed to change the body orientation. The most significant advantage of the synchronous-drive robot is that omnidirectional movement can be achieved by using only two actuators. Since the mechanical structure guarantees synchronous steering and driving motions, less control effort is required for motion control. Other advantages include that odometry information is relatively accurate and driving forces are evenly distributed among all the wheels. The drawbacks of this approach can be summarized as:

- Complicated mechanical structure.
- If backlash or loose coupling is present in the chain transmission, velocity differences between wheels may occur.
- In order to achieve omnidirectional movement, the wheel orientations should be aligned to the desired velocity direction before movement, due to the nonholonomic velocity constraints.

Omnimobile Robot with Swedish Wheels

The omnimobile robot with Swedish wheels corresponds to type (3,0) in the nomenclature of Sect. 24.2.6. At least three Swedish wheels are required to build a holonomic omnidirectional robot. A major advantage of using the Swedish wheel is that omnidirectional mobile robots can be easily constructed. At least three Swedish wheels are required to build a holonomic omnidirectional robot. Since omnidirectional robots can be built without using active steering of wheel modules, the mechanical structures of actuating parts can have simple structures. However, the mechanical design of a wheel becomes slightly complicated. One drawback of the Swedish wheel is that there is a vertical vibration because of discontinuous contacts during motion. In order to solve this problem, a variety of mechanical designs have been proposed; examples can be found in [24.10] and [24.11]. Another drawback is its relatively low durability when compared to conventional tires. An example of a robot using Swedish wheels can be found in [24.12].

Omnimobile Robot with Active Caster Wheels

A holonomic omnidirectional robot can be constructed by using at least two active caster wheels, and the robot also belongs to type (3,0). The robot can be controlled to generate arbitrary linear and angular velocities regardless of the wheel orientations. Since the robot uses conventional tires, the disadvantages of Swedish wheels, for example, vertical vibrations or durability problems, can be solved. An example can be found in [24.13]. The disadvantages of this robot can be summarized as follows:

- Since the location of the ground contact point (i. e., footprint) changes with respect to the robot chassis, instability can take place when the distance between the wheels is too short.
- If the robot switches its movement to the reverse direction, an abrupt change of wheel orientations may take place. This is called the shopping-cart effect, which may result in instantaneous high steering velocities.
- If a driving motor is directly attached to the wheel, wires to the motor will be wound due to steering motions. In order to avoid this, a gear train should be employed to transmit the input angular velocity from the driving motor, which is attached to the robot chassis. In this case, the mechanical structure becomes quite complicated.
- If a robot is equipped with more than two active caster wheel modules, more than four actuators are used. Since the minimum number of actuator to achieve holonomic omnidirectional motion is three, this is an overactuated system. Therefore, actuators should be accurately controlled in a synchronous way.

Omnidirectional Robot with Active Steerable Wheels

Centered orientable wheels are also employed to build omnidirectional robots; at least two modules are required. A significant difference between the active caster wheel and the centered orientable wheel is that the wheel orientation should always be aligned with the desired direction of velocity direction, as computed by inverse kinematics. This fact implies that this robot is nonholonomic and omnidirectional: it is a type (1,2) robot. The control problem is addressed in [24.14]. The mechanical drawbacks are similar to those of using active caster wheels (i. e., many actuators and compli-

cated mechanical structures). Since the driving motor is directly attached to the driving axis in many cases, allowable steering angles are limited in order to prevent wiring problems.

There are a lot of design candidates for three-wheel robots, other than the five designs described earlier. They can be classified and analyzed according to the scheme presented in Sect. 24.1. The aforementioned designs can be extended to four-wheel robots to improve stability. Additional wheels can be passive wheels without adding additional kinematic constraints. Active wheels can also be added and should be controlled by solving the inverse kinematics problem. Four-wheeled robots require suspension to maintain contact with the ground to prevent wheels from floating on irregular surfaces.

24.3.4 Robots with Four Wheels

Among the various four-wheel robots, we focus on the car-like structure. The car-like structure has been called as the Ackermann steering geometry that was shown in Fig. 24.3c. The front two wheels should be synchronously steered to keep the same instantaneous center of rotation. It is clear that the orientations of two front wheels are slightly different because the curvatures of rotation are different. As a result, this solution is kinematically equivalent to a single orientable wheel and the robot can be classified as a type (1,1) robot. A major advantage of a car-like robot is that it is stable during high-speed motion. However, it requires a slightly complicated steering mechanism. If the rear wheels are actuated, a differential gear is required to obtain pure rolling of the rear wheels during the turning motion. If the steering angle of the front wheel cannot reach 90°, the turning radius becomes nonzero. Therefore, parking motion control in a cluttered environment becomes difficult.

24.3.5 Special Applications of Wheeled Robots

Articulated Robots

A robot can be extended to an articulated robot, which is composed of a robot and trailers. A typical example is the luggage-transporting trailer system at airports. By exploiting trailers, a mobile robot obtains various practical advantages. For example, modular and reconfigurable robots can change their configuration according to service tasks. A common design is a car with multiple passive trailers, which is the simplest design of an articulated robot. VIDEO 326 shows another example of a trailer robot. From the viewpoint of control, some significant issues have been made clear, including

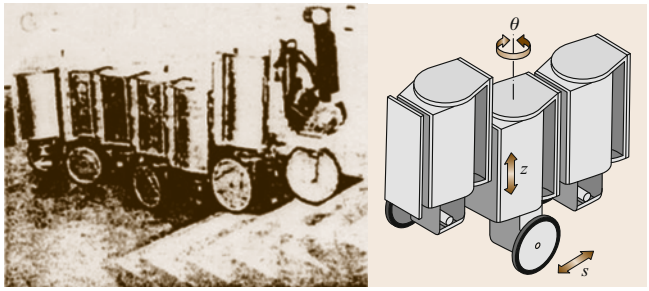


Fig. 24.8 An active trailer system (after [24.16])

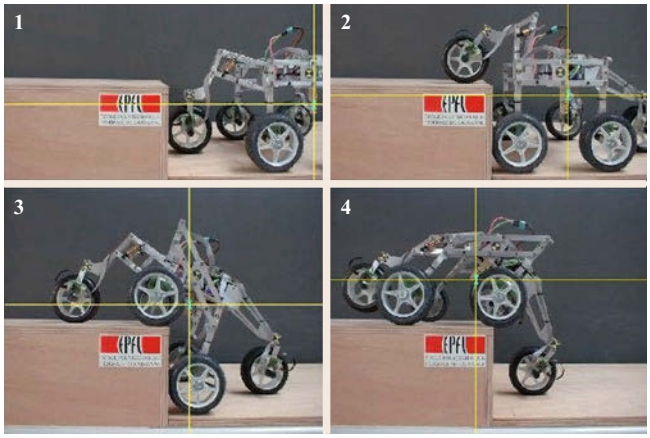


Fig. 24.9 A mobile robot for rough terrain (after [24.17])

a proof of controllability and the development of open- and closed-loop controllers using canonical forms such as the chained form. The design issues for trailer systems are the selection of wheel types and decisions regarding the link parameters. In practical applications, it is advantageous if trailers can move along the path of the towing robot. Passive trailers can follow the path of a towing robot within a small error by using a special design of passive steering mechanism for trailers; see, for example, [24.15].

On the other hand, active trailers can be used. There are two types of active trailers. A first approach is to actuate wheels of trailers. The connecting joints are passive, and two-wheel differential-drive robots can be used as active trailers. By using this type of active trailers, accurate path-following control can be achieved. The second approach is to actuate connecting joints. The wheels of the trailer are passively driven. By appropriate actuation of the connecting joints, the robot can move without wheel actuation, by snake-like motions. As an alternative design, we can use an active prismatic joint to connect trailers, in order to lift the neighboring trailer. By allowing vertical motion, a trailer system can climb stairs and traverse rough terrain. Examples of active trailers can be found in [24.16].

Hybrid Robots

A fundamental difficulty of using wheels is that they can only be used on flat surfaces. To overcome this problem, wheels are often attached to a special link mechanism. Each wheel is equipped with independent actuators and a linkage mechanism enables the robot to adapt its configuration to irregular ground conditions (VIDEO 329). A typical design can be found in [24.17] and can be understood as a hybrid robot that is a combination of a legged robot and a wheeled robot. Another hybrid example is a robot equipped with both tracks and wheels. Wheels and tracks have complementary advantages and disadvantages. Wheeled robots are energy efficient; however, tracked robots can traverse rough terrain. Therefore, a hybrid robot can selectively choose its driving mechanism according to environmental conditions, although fabrication cost increases.

24.4 Wheel–Terrain Interaction Models

A wheeled robot's mobility properties are governed by forces generated at the wheel–terrain contact interface. The ability to accurately model wheel–terrain interaction forces is therefore an important aspect of robot design, simulation, and control. These forces are strongly influenced by the relative stiffness of the wheel and terrain. Generally, there are four possible wheel–terrain interaction cases. The first case is that of a rigid wheel traveling on rigid terrain (Fig. 24.10a). The second case is that of a rigid wheel traveling on deformable terrain (Fig. 24.10b). The third case is that of a deformable wheel traveling on deformable terrain (Fig. 24.10c). The fourth case is that of a deformable wheel traveling on rigid terrain (Fig. 24.10d). Models for these four cases are presented as follows. It should be noted that while many different types of models (i. e., finite element, discrete element, empirical) have been developed for each of these four cases, the focus here is on analytical models that are broadly suitable for analysis, simulation, design, and control purposes.

24.4.1 Rigid Wheels on Rigid Terrain

Many robots employ wheels made of metal, stiff rubber, or other materials that deform very little when subject to loading experienced during operation. When operated on stiff surfaces such as indoor flooring, pavement, or stone, wheel–terrain interaction can be reasonably approximated as a point contact. An interaction model based on classical Coulomb friction can then be employed to describe bounds on available tractive forces, F_x, and lateral forces, F_y, as a function of the load on the wheel, W, for a robot with n wheels traveling on a surface with coefficient of friction μ

$$F_{xi} \le \mu_x W_i \,, \quad i = 1 \ldots n \,, \tag{24.11}$$

$$F_{yi} \le \mu_y W_i \,, \quad i = 1 \ldots n \,. \tag{24.12}$$

Since the frictional force can be generated in any direction, and its magnitude is limited, a bound on the norm of the frictional and lateral forces can be expressed as

$$\left(\frac{F_{xi}}{\mu_x W_i}\right)^2 + \left(\frac{F_{yi}}{\mu_y W_i}\right)^2 = 1 \,, \quad i = 1 \ldots n \,. \tag{24.13}$$

Equation (24.13) represents a concept known as the friction ellipse (Fig. 24.11). When the effective friction is equal in all directions the ellipse becomes a circle.

24.4.2 Rigid Wheels on Deformable Terrain

Robotic locomotion in outdoor, off-road terrain frequently results in deformation of the terrain, especially when the terrain is composed of a deformable material such as sand, silt, loam, or clay. When the wheel is constructed of a rigid material such as metal or stiff rubber, or the tire inflation pressure is high enough such that tire deformation is small, wheel–terrain interaction occurs along an arc of the wheel, rather than at a single point (Fig. 24.10b). In such scenarios, the Coulomb friction model described in Sect. 24.4.1 does not accurately represent the relationship between the wheel load

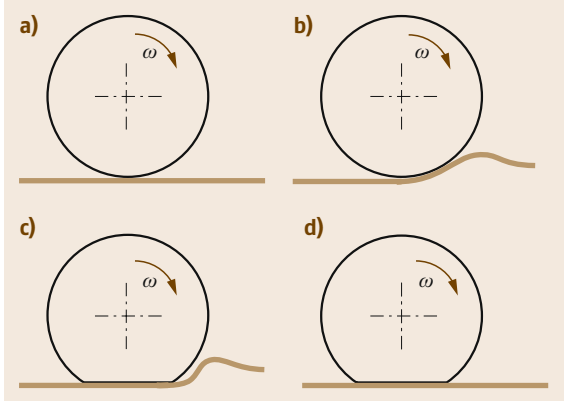


Fig.24.10a–d Four cases of wheel–terrain interaction mechanics: **(a)** rigid wheel traveling over rigid terrain, **(b)** rigid wheel traveling over deformable terrain, **(c)** deformable wheel traveling over deformable terrain, and **(d)** deformable wheel traveling over rigid terrain

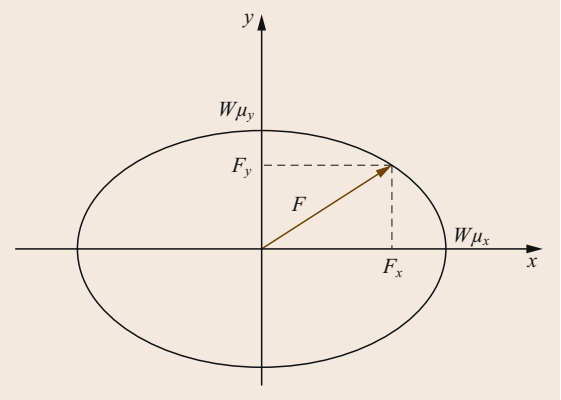


Fig. 24.11 Illustration of friction ellipse concept to represent constraint on longitudinal and lateral friction

and tractive forces. This is because the mechanism for force generation derives primarily from shearing along failure planes in the terrain, rather than from frictional contact at a point on the wheel–terrain interface.

Models of the interaction of rigid wheels on deformable surfaces were developed by *Bekker* in the 1950s and 1960s in the context of large military vehicles [24.18]. Bekker's research gave rise to the discipline of *terramechanics*, i. e., the study of vehicle-terrain interaction phenomena. Bekker's terramechanics models have subsequently been applied to robotic systems at a range of sizes [24.19] and [24.20]. However, Bekker performed his original analyses in the context of large military vehicles, which were typically several meters in length and with gross vehicle weights exceeding 500 kg. The accuracy of such models when used to analyze the performance of small robotic systems remains an open research question [24.21] and [24.22].

A description of terramechanics models for the case of rigid wheels on deformable surfaces is presented in Chap. 55, in the context of Space Robotics. However, Chap. 55 assumes that the wheel is constructed of a rigid material such as metal, which is typical for planetary exploration rovers but rare for robots operating on Earth. As noted earlier, a pneumatic tire will behave as a rigid wheel if the average tire ground pressure, which is a function of the tire carcass stiffness and inflation pressure, exceeds some critical ground pressure. The average tire ground pressure can be obtained experimentally by measuring the tire contact patch area while the tire is subjected to a series of known vertical loads and inflation pressures. The critical ground pressure is defined as

$$p_{\mathrm{gcr}} = \left(\frac{k_{\mathrm{c}}}{b} + k_{\phi}\right)^{\frac{1}{(2n+1)}} \left(\frac{3W}{(3-n)b_{\mathrm{ti}}\sqrt{D}}\right)^{\frac{2n}{2n+1}}, \tag{24.14}$$

where k_{c}, k_{ϕ}, n are the soil-dependent parameters, b is the smallest dimension of the (typically rectangular) wheel–terrain contact patch, b_{ti} is the tire width, W is the vertical load on the tire, and D is the tire diameter.

If the average tire ground pressure exceeds the critical ground pressure, a pneumatic tire can be modeled as a rigid wheel interacting with a deformable surface. This scenario is governed by the equations presented in Chap. 55. If the average tire ground pressure does not exceed the critical ground pressure, a pneumatic tire can be modeled either as a deformable wheel interacting with deformable terrain, or as a deformable wheel interacting with rigid terrain. Models of these scenarios are presented in Sects. 24.4.3 and 24.4.4.

24.4.3 Deformable Tires on Deformable Terrain

When the average tire ground pressure does not exceed the critical ground pressure, both the tire and the terrain surface may undergo deformation. Accurate modeling of this interaction thus requires consideration of phenomena related to both soil deformation (most notably, soil compaction, and shear deformation) and tire deformation. Several semiempirical models for this scenario, based on Bekker theory, have been developed in recent years.

A simple approach to modeling in this interaction scenario was initially proposed by Bekker, and further developed by *Harnisch* [24.23]. This approach relies on the observation that tire deformation serves to increase the size of the wheel–terrain contact patch, and thus reduce the average ground pressure and minimize sinkage. Bekker proposed the *substitute circle* concept, which embodies the notion that the deformable tire can be equivalently represented by a rigid tire of increased diameter, to yield identical sinkage (Fig. 24.12). The diameter of the larger rigid tire can be computed via the following method:

1. Assuming zero slip, sinkage for the original tire diameter D, assumed to be rigid, is calculated for the vertical load W and soil parameters.
2. Knowing the tire load-deflection characteristics, for the given inflation pressure the tire deflection δ can be calculated.
3. The diameter of the larger substitute circle D' will correspond to the diameter of the circle that passes through P_1, P_2, and P_3.

Once the outer circle diameter is known, all other quantities will be calculated following the standard methodology presented in Chap. 55, substituting D' for the original wheel diameter D.

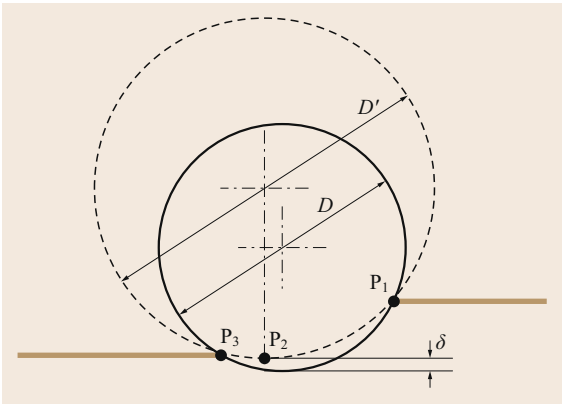


Fig. 24.12 Illustration of substitute circle concept

An alternative approach, based on a more physically realistic model of the deformed tire geometry, was proposed by *Wong* [24.24]. Here, the tire geometry is represented as a circular arc with a perfectly flat (i. e., horizontal) base (Fig. 24.13). The pressure distribution under the flat section BC is assumed to be uniform, and equal to the average ground pressure of the tire p_g. Disregarding shear effects and soil elastic rebound (section CD), it is possible to calculate the tire deflection δ by balancing vertical forces

$$W = bp_g l_t + W_{AB} \,, \tag{24.15}$$

where b is the tire width, l_t is the length of the flat section BC, and W_{AB} is the vertical reaction exerted by the soil along the undeformed tire section AB

$$W_{AB} = \left[b \left(\frac{k}{l} + k_\phi \right) \sqrt{D} (z_0 + \delta)^{n-1} \right] \frac{(3-n)(z_0+\delta)^{\frac{3}{2}} - (3-n)\delta^{\frac{3}{2}} - 3z_0\sqrt{\delta}}{3} \,, \tag{24.16}$$

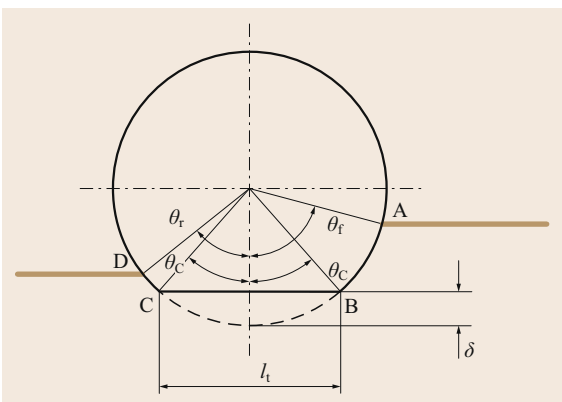


Fig. 24.13 Diagram of deformable wheel traveling over deformable terrain

where D is the undeformed tire diameter and z_0 is the static sinkage that can be calculated through standard pressure–sinkage relations

$$z_0 = \begin{cases} \left(\frac{p_g}{(k_c/l_t)+k_\phi} \right)^{\frac{1}{n}} & (l_t < b) \\ \left(\frac{p_g}{(k_c/b)+k_\phi} \right)^{\frac{1}{n}} & (l_t \geq b) \end{cases} \,, \tag{24.17}$$

where $l_t = 2\sqrt{D\delta - \delta^2}$ is the contact patch length. Although not strictly correct, the tire ground pressure p_g can be approximated by the tire inflation pressure, a known variable. Following this assumption, it is possible to solve (24.16) for the tire deflection δ.

Once the deformed tire shape has been calculated, all other relevant quantities (i. e., sinkage, drawbar force, and torque) can be calculated. Since the tire profile is nonsmooth, the normal and tangential stresses are described by piecewise expressions. The vertical load balance can be written as

$$W = br \int_{\theta_r}^{\theta_c} (\sigma \cos\theta - \tau \sin\theta) \mathrm{d}\theta + bDp_g \sin\theta_c + br \int_{\theta_c}^{\theta_f} (\sigma \cos\theta + \tau \sin\theta) \mathrm{d}\theta \,. \tag{24.18}$$

The aforementioned equation can be solved for the entry angle θ_f. The angle θ_c can be derived from the tire deflected shape. Once the tire-terrain geometry is determined, the sinkage can be found. Note that this new sinkage will differ from the static sinkage z_0. It is then possible to calculate drawbar pull F_x and driving torque

$$F_x = br \int_{\theta_r}^{\theta_c} (\sigma \cos\theta + \tau \sin\theta) \mathrm{d}\theta + b \int_0^{l_t} \tau \mathrm{d}x + br \int_{\theta_c}^{\theta_f} (-\sigma \cos\theta + \tau \sin\theta) \mathrm{d}\theta \,, \tag{24.19}$$

$$T = br^2 \int_{\theta_r}^{\theta_c} \tau \mathrm{d}\theta + br \cos\theta_c \int_0^{l_t} \tau \mathrm{d}x + br^2 \int_{\theta_c}^{\theta_f} \tau \mathrm{d}\theta \,. \tag{24.20}$$

Another flexible tire model developed by *Senatore* and *Sandu* is similar to the approach proposed by Wong, however it calculates the tire deflection as a function of inflation pressure, and also models multi-pass effects [24.25]. Generally speaking, all of the flexible tire models proposed to date have various advantages and drawbacks, and none have gained universal approval in the research community.

24.4.4 Deformable Tires on Rigid Terrain

In scenarios where the tire stiffness is significantly lower than the terrain stiffness, only the tire will experience significant deformation. This scenario has been studied extensively in the automotive community, since it is the typical interaction scenario for passenger and commercial vehicles operating on paved roads. Here, two common models of steady-state tire–terrain interaction are briefly described: the brush model and the magic formula model [24.26].

The brush model models the tire contact patch as a row of elastic bristles in contact with the ground. Carcass, belt, and tread element compliance is captured by a lumped bristle compliance. For the case of pure longitudinal slip, the tractive force can be calculated as

$$F_x = 2c_{px}a^2\frac{V_{sx}}{V_x}\,, \tag{24.21}$$

where c_{px} is the longitudinal tread element stiffness per unit length and other quantities are illustrated in Fig. 24.14.

In a similar manner, the lateral force can be calculated as follows

$$F_y = 2c_{py}a^2\alpha\,, \tag{24.22}$$

where α is the slip angle. As previously discussed for the case of rigid wheels on rigid terrain, combined slip operation will limit the availability of longitudinal and lateral thrust [24.26].

The magic formula model is an empirical model that has found widespread application due to its flexibility and reduced computational burden. The model is based on a combination of trigonometric functions that generate a curve that (usually) passes through the origin, reaches a peak value, and then tends to a horizontal asymptote: this behavior is typical of force/moment versus slip characteristics of modern tires, regardless of tire size, construction, inflation pressure, and other characteristics. The general form of the magic formula model is

$$y = D\sin\{C\arctan[Bx - E(Bx - \arctan Bx)]\}\,, \tag{24.23}$$

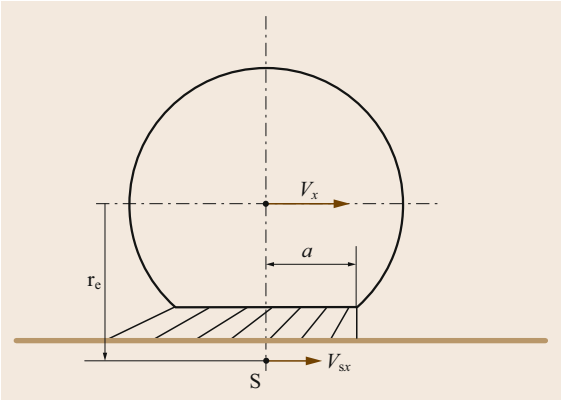


Fig. 24.14 Brush tire model for analyzing interaction of deformable tire and rigid terrain

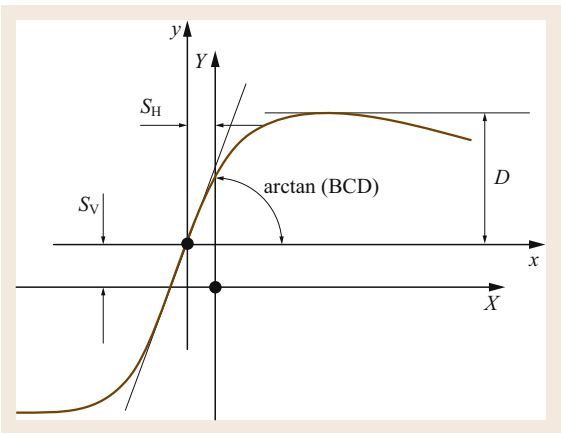


Fig. 24.15 Diagram illustrating influence of parameters of magic tire model

with $Y(X) = y(x) + S_V$ and $x = X + S_H$. Here, Y represents the output variable, which may be F_x, F_y or M_z, and X represents the input variable, which may be $\tan\alpha$ or k (i. e., the lateral or longitudinal slip). The relation also contains several empirically determined values, as follows: B represents a stiffness factor; C a shape factor; D a peak value; E a curvature factor; S_H a horizontal shift; and S_V a vertical shift. These factors are illustrated in Fig. 24.15.

24.5 Wheeled Robot Suspensions

A suspension is a system of linkages, springs, dampers, and actuators that govern the relative motion between a robot's wheels and body. These mechanisms include a degree of freedom allowing the wheel to spin, and optional degrees of freedom for steering and vertical translation. Suspension mechanisms are useful when driving over uneven surfaces for several reasons. First, a suspension allows the wheels to maintain contact with the ground despite disturbances from the uneven ground surface, which allows generation of traction, braking, and cornering forces. Additionally, suspensions provide load-bearing and dissipative elements (i. e., springs,

dampers, and active or semiactive actuators) to mitigate the effect of disturbances on robot body motion, and thus allow for improved obstacle negotiation and sensor or payload stabilization. Key issues in suspension design include the design of suspension linkage kinematics and selection of load-bearing and dissipative elements.

Suspension linkage kinematics are designed to allow vertical displacement of the wheels. Typical suspension mechanisms include simple prismatic joints, rotational joints with a wheel attached to a trailing arm, and 4-bar linkages. For car-like wheeled robots, substantial inspiration can be drawn from the design of passenger vehicle suspensions, which has received vast research attention [24.27]. A comprehensive treatment of the design of suspension linkage kinematics is beyond the scope of this chapter. Interested readers are referred to [24.28].

The selection of load-bearing and dissipative elements is typically based on analysis of low-order models of the suspension subject to expected operational conditions. Common methods for modeling passive and semiactive suspensions are presented in the following. Analysis of these models allows for principled selection of springs, dampers, and active or semiactive actuators that form the basis of nearly all wheeled robot suspensions.

24.5.1 Passive Suspension Dynamics

Arguably the most common passive suspension design consists of a parallel spring and damper mounted between the robot body and wheels. A simple model for analyzing the vertical dynamics of this suspension is commonly referred to as the quarter car model [24.29] and is illustrated in Fig. 24.16. The fraction of body mass supported by the suspension element, m_b, is connected to a wheel with mass m_w by a spring element with stiffness k_s and a damping element with damping coefficient b_s. The wheel stiffness is denoted by k_w, and transmits vertical excitation from the uneven terrain surface. The heights of the body mass, wheel mass, and terrain surface are given by z_b, z_w, and z_u, respectively.

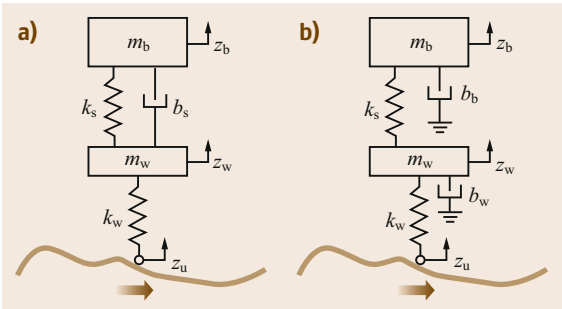


Fig.24.16a,b Quarter car model for passive suspension modelling **(a)** and quarter car model with ideal sky-hook semi-active suspension control **(b)**

A state vector x for this model is composed of the suspension displacement $d_s = z_b - z_w$, wheel spring displacement $d_w = z_w - z_u$, and velocities $\dot{z}_b$, $\dot{z}_w$, as shown in (24.24), along with a linear state space model (24.25)–(24.27). Note that the input to the model is the vertical terrain velocity $\dot{z}_u$. This is a convenient formulation for linear systems analysis since $\dot{z}_u$ can be modeled as white noise with intensity proportional to the vehicle's forward speed and a road roughness parameter [24.29].

$$x = \begin{pmatrix} d_s \\ d_w \\ \dot{z}_b \\ \dot{z}_w \end{pmatrix} \tag{24.24}$$

$$\dot{\boldsymbol{x}} = \mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{u} \tag{24.25}$$

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \\ -k_s/m_b & 0 & -b_s/m_b & b_s/m_b \\ k_s/m_w & -k_w/m_w & b_s/m_w & -b_s/m_w \end{pmatrix} \tag{24.26}$$

$$\mathbf{B} = \begin{pmatrix} 0 \\ -1 \\ 0 \\ 0 \end{pmatrix} . \tag{24.27}$$

To investigate the properties of this model, transfer functions $G(s) = \mathbf{Y}(s)/\mathbf{U}(s)$ are computed for outputs of interest $\boldsymbol{y} = \mathbf{C}\boldsymbol{x}$ as

$$G(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} . \tag{24.28}$$

With a white noise input, all frequencies are equally represented. Thus, the magnitude of the transfer function $|G(j\omega)|$ at a given frequency ω represents the frequency response of each output. Two outputs are of particular interest for suspension design purposes: the wheel displacement d_w and body acceleration $\ddot{z}_b$. This is because the wheel displacement corresponds to variations in the wheel contact force, and thus informs the road holding and maneuvering capability of the robot. The body acceleration represents the amount of vibration experienced by the sensors and payloads mounted on the robot body.

The output matrix for the wheel displacement $\mathrm{d}w$ is

$$\mathbf{C} = (0 \quad 1 \quad 0 \quad 0) , \tag{24.29}$$

and the output matrix for the body acceleration $\ddot{z}_b$ is

$$\mathbf{C} = \left(-\frac{k_s}{m_b} \quad 0 \quad -\frac{b_s}{m_b} \quad \frac{b_s}{m_b} \right) . \tag{24.30}$$

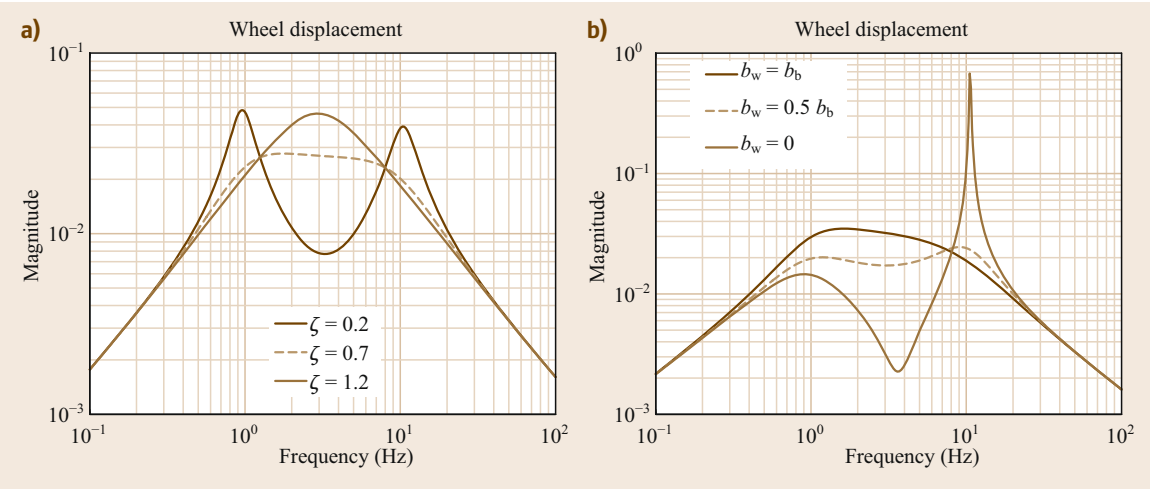


Fig.24.17a,b Wheel displacement frequency response for passive suspension (a) and semiactive suspension. Wheel displacement frequency corresponds to wheel contact force variation and road holding capability

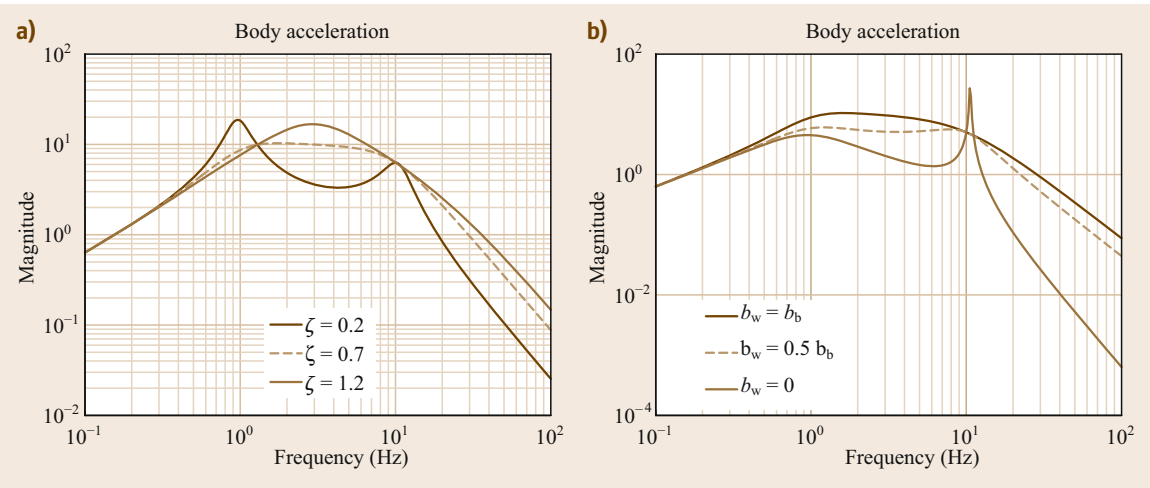


Fig.24.18a,b Body acceleration frequency response for passive suspension (a) and semiactive suspension (b). Body acceleration frequency corresponds to the vibration of sensors and payloads mounted on the robot body

The frequency response of wheel displacement and body acceleration are shown in Figs 24.17 and 24.18, respectively. These plots are shown with the following representative parameter values for a passenger vehicle: the ride frequency $\omega_b = \sqrt{k_s/m_b} = 2\,\pi$ rad/s, the wheel hop frequency $\omega_w = \sqrt{k_w/m_w} = 20\,\pi$ rad/s, the mass ratio $\rho = m_w/m_b = 0.1$, and the damping ratio $\zeta = b_s/(2\sqrt{k_s m_b}) = (0.2, 0.7, 1.2)$. Various damping ratios are shown to illustrate the behavior of underdamped, critically damped, and over-damped suspensions.

For the underdamped case, where $\zeta = 0.2$, resonant peaks (which are generally undesirable) exist in the frequency response near ω_b and ω_w for both the wheel displacement and body acceleration. For the over-damped case, where $\zeta = 1.2$, the peak response occurs between the resonant peaks. For the (approximately) critically damped case, where $\zeta = 0.7$, the peak responses are reduced, and for this reason many suspension are designed to exhibit critical damping.

24.5.2 Semiactive Suspension Dynamics

The introduction of controllable actuators to the suspension allows for shaping of the system's dynamic response, and can improve road holding performance

and reduce body vibration. Suspension systems integrated with controllable actuators are referred to as active or semiactive suspensions. Active suspensions refer to systems with powered rotational or linear actuators capable of both supplying and absorbing significant amounts of energy, such that the actuator effectively replaces the passive components in a traditional suspension. Such systems are rare due to their complexity and cost. A simpler and more inexpensive (and therefore more common) approach to controllable suspensions is to employ semiactive elements. This refers to elements that can absorb energy in a controlled manner, such as active (e.g., magnetorheological) fluid dampers. Semiactive suspensions can be modeled in a manner similar to passive suspensions with a time-varying, positive damping coefficient $b_s(t) \geq 0$.

Semiactive suspension control schemes attempt to control the damping coefficient $b_s(t)$ to emulate dampers with coefficients b, b acting on the absolute velocities of the body and wheels, in order to reduce the effect of wheel vibration on the robot body. A common technique known as a sky-hook controller is illustrated in Fig. 24.16. The technique can be approximated by applying the damper force F given in the following. Note that a semiactive system can only approximate this control, as F is constrained as $F(\dot{z} - \dot{z}) \leq 0$.

$$F = -b\dot{z} + b\dot{z} \tag{24.31}$$

To illustrate the potential improvement achievable by the use of a semiactive suspension with a sky-hook controller, frequency responses of the semiactive suspension quarter car model with F given earlier are plotted in Figs 24.17 and 24.18. The model parameters are identical to the values from the previous section for critical damping $\zeta = 0.7$, with damping coefficients $b = b$ and $b = (b, 0.5b, 0)$. Note that the case $b = b$ is identical to the passive, critically damped case.

With no wheel damping, $b = 0$, both frequency responses are improved for $\omega \ll \omega$, and the body acceleration response is improved for $\omega \gg \omega$. Near ω, however, there exist substantial resonant peaks for both outputs. With moderate wheel damping $b = 0.5b$, the amplitude of the resonant peaks near ω are reduced and both peak responses are reduced, though performance improvements of $b = 0$ are not matched for frequencies far from ω.

While the sky-hook and other suspension control algorithms can improve the dynamic response of the vehicle, it is important to note some inherent trade-offs in semiactive and active suspension control. It has been proven that road holding and attenuation of body vibration can be concurrently improved through active control for frequencies below the wheel hop frequency, ω [24.29]. At frequencies above ω, however, any attempts to improve one of these quantities necessarily causes the other to degrade.

24.6 Conclusions

Wheels are the most commonly employed running gear for mobile robots due to their relative simplicity, robustness, and low cost. The number of possible wheeled mobile robot realizations is almost infinite, depending on the number, type, implementation, geometric characteristics, and motorization of the wheels. This chapter has described several such realizations. Notwithstanding this variety, it is possible to classify WMRs into only five generic categories. This categorization aids understanding of wheel structures through simplification. Practical robot structures have been classified according to the number and type of wheels. Wheel-terrain interaction models have been presented to allow the analysis of tractive force generation capability, for design and simulation purposes. Finally, a brief description of common suspension systems has been presented, including a presentation of structures and dynamic models.

Video-References

VIDEO 325 An omnidirectional mobile robot with active caster wheels
available from http://handbookofrobotics.org/view-chapter/24/videodetails/325

VIDEO 326 Articulated robot – a robot pushing 3 passive trailers
available from http://handbookofrobotics.org/view-chapter/24/videodetails/326

VIDEO 327 An omnidirectional robot with 4 Mecanum Wheels
available from http://handbookofrobotics.org/view-chapter/24/videodetails/327

VIDEO 328 An omnidirectional robot with 4 Swedish wheels
available from http://handbookofrobotics.org/view-chapter/24/videodetails/328

VIDEO 329 An innovative space rover with extended climbing abilities
available from http://handbookofrobotics.org/view-chapter/24/videodetails/239

References

24.1 H. Asama, M. Sato, L. Bogoni: Development of an omnidirectional mobile robot with 3 DOF decoupling drive mechanism, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 1925–1930

24.2 L. Ferriere, G. Campion, B. Raucent: ROLLMOBS, a new drive system for omnimobile robots, Robotica **19**, 1–9 (2001)

24.3 W. Chung: *Nonholonomic Manipulators, Springer Tracts Adv. Robotics*, Vol. 13 (Springer, Berlin, Heidelberg 2004)

24.4 J.E. Colgate, M. Peshkin, W. Wannasuphoprasit: Nonholonomic haptic display, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 539–544

24.5 G. Campion, G. Bastin, B. dAndrea-Novel: Structural properties and classification of kinematic and dynamic models of wheeled mobile robots, IEEE Trans. Robotics Autom. **12**, 47–62 (1996)

24.6 R. Nakajima, T. Tsubouchi, S. Yuta, E. Koyanagi: A development of a new mechanism of an autonomous unicycle, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 906–912

24.7 G.C. Nandy, X. Yangsheng: Dynamic model of a gyroscopic wheel, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 2683–2688

24.8 Y. Ha, S. Yuta: Trajectory tracking control for navigation of self-contained mobile inverse pendulum, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1875–1882

24.9 Y. Takahashi, T. Takagaki, J. Kishi, Y. Ishii: Back and forward moving scheme of front wheel raising for inverse pendulum control wheel chair robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 3189–3194

24.10 K.-S. Byun, S.-J. Kim, J.-B. Song: Design of continuous alternate wheels for omnidirectional mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 767–772

24.11 M. West, H. Asada: Design and control of ball wheel omnidirectional vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 1931–1938

24.12 B. Carlisle: An omnidirectional mobile robot. In: *Development in Robotics*, ed. by B. Rooks (IFS, Bedford 1983) pp. 79–87

24.13 M. Wada, S. Mori: Holonomic and omnidirectional vehicle with conventional tires, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 3671–3676

24.14 D.B. Reister, M.A. Unseren: Position and constraint force control of a vehicle with two or more steerable drive wheels, IEEE Trans. Robotics Autom. **9**(6), 723–731 (1993)

24.15 Y. Nakamura, H. Ezaki, Y. Tan, W. Chung: Design of steering mechanism and control of nonholonomic trailer systems, IEEE Trans. Robotics Autom. **17**(3), 367–374 (2001)

24.16 S. Hirose: *Biologically Inspired Robots: Snake-Like Locomotion and Manipulation* (Oxford Univ. Press, Oxford 1993)

24.17 R. Siegwart, P. Lamon, T. Estier, M. Lauria, R. Piguet: Innovative design for wheeled locomotion in rough terrain, J. Robotics Auton. Syst. **40**, 151–162 (2003)

24.18 M.G. Bekker: *Introduction to Terrain-Vehicle Systems* (Univ. Michigan Press, Ann Arbor 1969)

24.19 H. Shibly, K. Iagnemma, S. Dubowsky: An equivalent soil mechanics formulation for rigid wheels in deformable terrain, with application to planetary exploration rovers, J. Terramech. **42**, 1–13 (2005)

24.20 G. Ishigami, A. Miwa, K. Nagatani, K. Yoshida: Terramechanics-based for steering maneuver of planetary exploration rovers on loose soil, J. Field Robotics **24**(3), 233–250 (2007)

24.21 G. Meirion-Griffith, M. Spenko: A modified pressure-sinkage model for small, rigid wheels on deformable terrains, J. Terramech. **48**(2), 149–155 (2011)

24.22 C. Senatore, M. Wulfmeier, P. Jayakumar, J. Maclennan, K. Iagnemma: Investigation of stress and failure in granular soils for lightweight robotic vehicle applications, Proc. Ground Vehicle Syst. Eng. Technol. Symp. (2012)

24.23 C. Harnisch, B. Lach, R. Jakobs, M. Troulis, O. Nehls: A new tyre–soil interaction model for vehicle simulation on deformable ground, Vehicle Syst. Dyn. **43**(1), 384–394 (2005)

24.24 J.Y. Wong: *Theory of Ground Vehicles*, 3rd edn. (Wiley, Hoboken 2001)

24.25 C. Senatore, C. Sandu: Off-road tire modeling and the multi-pass effect for vehicle dynamics simulation, J. Terramech. **48**(4), 265–276 (2011)

24.26 H.B. Pacejka: *Tire and Vehicle Dynamics*, 2nd edn. (Elsevier, Oxford 2005)

24.27 P. Barak: *Magic Numbers in Design of Suspensions for Passenger Cars*, SAE Tech. Pap. No. 911921 (SAE, Warrendale 1991)

24.28 J.C. Dixon: *Suspension Geometry and Computation* (Wiley, Chichester 2009)

24.29 J.K. Hedrick, T. Butsuen: Invariant properties of automotive suspensions, Proc. Inst. Mech. Eng. Part D J. Automob, Eng. **204**(1), 21–27 (1990)

Multimedia Contents

25. Underwater Robots

Hyun-Taek Choi, Junku Yuh

Covering about two-thirds of the earth, the ocean is an enormous system that dominates processes on the Earth and has abundant living and nonliving resources, such as fish and subsea gas and oil. Therefore, it has a great effect on our lives on land, and the importance of the ocean for the future existence of all human beings cannot be overemphasized. However, we have not been able to explore the full depths of the ocean and do not fully understand the complex processes of the ocean. Having said that, underwater robots including remotely operated vehicles (ROVs) and autonomous underwater vehicles (AUVs) have received much attention since they can be an effective tool to explore the ocean and efficiently utilize the ocean resources. This chapter focuses on design issues of underwater robots including major subsystems such as mechanical systems, power sources, actuators and sensors, computers and communications, software architecture, and manipulators while Chap. 51 covers modeling and control of underwater robots.

25.1 **Background** 595
25.2 **Mechanical Systems** 596
25.2.1 Frames and Pressure Hulls 596
25.2.2 Materials and Buoyancy 597
25.3 **Power Systems** 599
25.3.1 On-Board Power Sources 599
25.3.2 Surface Power Systems 601
25.4 **Underwater Actuators and Sensors** 601
25.4.1 Actuators 602
25.4.2 Nonacoustic Sensors 602
25.4.3 Acoustics Sensors 604
25.4.4 Other Sensor Issues 606
25.5 **Computers, Communications, and Architecture** 606
25.5.1 On-Board Computers 606
25.5.2 Communications 608
25.5.3 Software Architecture 610
25.6 **Underwater Manipulators** 614
25.6.1 Dynamics 614
25.6.2 Teleoperation 615
25.6.3 Autonomous Manipulation 616
25.7 **Conclusions and Further Reading** 617
Video-References 617
References 618

25.1 Background

The ocean covers about two-thirds of the earth and has a great effect on the future existence of all human beings. Underwater robots or unmanned underwater vehicles (UUVs) can help us better understand marine and other environmental issues, protect the ocean resources of the earth from pollution, and efficiently utilize them for human welfare.

Most UUVs commercially available in the market are tethered and remotely operated, referred to as remotely operated vehicles (ROVs). The operators on the mother vessel control the ROV by sending control signals and power to the vehicle and receiving video, status, and other sensory data via an umbilical cable. Most work-class ROVs are equipped with one or multiple video cameras and lights, sonars, a still camera, one manipulator, one grabber, and a wide range of sampling devices. The large deep-sea ROVs use a tether management system (TMS) that allows the vehicle to be deployed using a strong but heavy umbilical cable and then flown out from the TMS using a lighter, more flexible cable. ROVs must have high bandwidth for high-resolution video and data transmission.

The US Navy funded the early ROV technology development in the 1960s. They were used to recover underwater ordnance. ROVs have been used for various applications including military operations, environmental systems, scientific missions, ocean mining, and gas and oil industry [25.1]. Visiongains analysis indicates that total spending on ROVs in the energy industry will reach about $2.7 billion in 2022 as ROVs are required for more inspection, maintenance, drill support, and decommissioning activities than ever before.

However, since most commercial ROV systems currently have some operational issues such as very high operational costs with mother vessels and operator fatigue, the demand for advanced underwater robot technologies that would lead to next generation workclass ROV systems or autonomous underwater vehicles (AUVs) has been growing. A survey paper on AUVs was published by *Yuh* [25.2] in 2000. The paper describes key areas, such as dynamics, control systems, navigation, sensors, communications, power systems, pressure hulls, fairing, and mechanical manipulators. It also presents existing AUV systems and components commercially available in the market then.

This chapter focuses on design issues with each major subsystem of underwater robots, such as mechanical systems, power sources, actuators and sensors, computers and communications, software architecture, and manipulators.

25.2 Mechanical Systems

Every 10 m adds approximately 1 atm of water pressure on a robot body and the chemical environment of the seawater is highly corrosive. Therefore, mechanical systems that have excellent structural strength, and excellent chemical and environmental resistance are needed.

An underwater robot's mechanical structure can be divided into three functional subsystems: frame, pressure vessel, and fairing. Frame is the primary skeleton of the robot. It defines the robot's shape, supports its weight both in and out of the water, provides mechanical support, resists hydrodynamic forces of motion and additional force due to operation of tools, and protects against any damages [25.3]. Its design must be based on structural analysis. A pressure vessel should be a waterproof structure and it must be strong enough to withstand water pressure at a target design depth. Typically, a cylindrical or spherical shape is the most effective way to maximize strength at the expense of low space efficiency. A pressure vessel with a larger diameter would result in increased cost of materials and machining since it requires increasing the thickness of the pressure vessel for the same strength. Consequently, the pressure vessel would become heavier. Therefore, selecting appropriate materials for a pressure vessel is very important. Fairing is like a skin of the robot. A fairing that covers all or parts of an underwater robot offers some advantages, such as an aesthetically pleasing exterior, a decreased drag force, and protection.

25.2.1 Frames and Pressure Hulls

The frame must be designed considering the following operational conditions:

1. On the ground without motion
2. Underwater in motion
3. Hanging from a crane for launch and recovery.

The major points of acting force for each condition are totally different. The static gravitational force is enough to be considered for the first case. For the second case, we should consider buoyancy force, gravitational force as well as all thruster forces in a dynamic state. During the launch and recovery stages, the enormous snap force, due to the mother ship's vertical motion can be concentrated on a lifting point of the robot.

The structural design and analysis must be conducted to take advantages of shapes offering high strength and stiffness for their weight. At the same time, materials should be chosen to give the maximum structural performance with the minimum weight [25.4]. An important point is that weights in both air and water must be carefully calculated. One should note that buoyancy materials will also increase the overall weight of the robot in air. A factor of safety is one of crucial parameters in designing mechanical systems. Incorporating a factor of safety may result in 50% overdesign for a manned system and 10% overdesign for an unmanned system [25.5]. Figure 25.1 shows a deep-sea ROV, HEMIRE's open frame that is made of aluminum. HEMIRE was developed by Korea Research Institute of Ships and Ocean Engineering (KRISO) (

). On the other hand, a closed frame or a fairing is desirable for AUVs, due to limited on-board power sources for required hydrodynamic performance.

The first and ultimate goal in designing a pressure vessel is to provide a watertight environment, where on-board computers, sensors, or normal batteries are installed. Therefore, two important design factors are wa-

terproof sealing design and water pressure-resistant design. When given specifications, such as an inside diameter (ID) or an outer diameter (OD), length, thickness, and materials, we must analyze buckling stress and yield stress. They are the dominant factors for the deep-sea (high pressure) environment [25.6]. A spherical- or cylindrical-shaped pressure vessel would be good to withstand the hydrostatic pressure; however, a pressure vessel of cylindrical shape is considered a better alternative as it provides a better placement for the components. Hydrodynamic design would not be needed for pressure vessels placed inside a frame with fairing. However, if the body of an underwater robot is designed as a pressure vessel, hydrodynamics in designing a pressure vessel must be considered to reduce the drag, cavitation, and acoustic noise [25.7].

Frames and pressure vessels must have the structural analysis using the finite element method (FEM) software, and nondestructive tests and/or pressure testing to avoid any mechanical defects. Pressure testing is the best way to reveal hidden defects in materials and in machining process even though it requires additional time, cost and nontrivial facilities. It is a critical environmental test because a hydrostatic pressure test, cyclic pressure test for fatigue test due to repeated deployments, and failure mode test to compare a designed crush depth with its real depth are all involved [25.5]. Pressure testing facilities provide not only a chamber for external pressure testing but also various types of measurement sensors and systems to record any changes during the testing.

25.2.2 Materials and Buoyancy

Besides high strength and high stiffness, one should consider good corrosion resistance and machinability in choosing materials for frames and pressure vessels [25.7].

Fig. 25.1 Aluminum frame of an ROV, Hamire, developed by KRISO, Korea

A typical underwater robot for shallow water tends to use aluminum, polycarbonate (PC), and polyoxymethylene (POM), while high strength steel, titanium, and composites are considered for a deep-sea underwater robot. The hybrid ROV, *Nereus*, is for a depth of 11 000 m and employs two pressure vessels composed of 96% alumina ceramic because its high compressive strength-to-weight ratio allows for near neutrally buoyant pressure vessels capable of going to these extreme depths [25.8]. Basic properties of materials are described later.

Steel is commonly used because of its high tensile strength and low cost. For marine applications, surface treatment of steel must be done because it is susceptible to corrosion or rusting. It tends to distort magnetic fields. Type 316 stainless steel is usually regarded as the standard marine grade and it is typically used for hydraulic, pneumatic and fasteners, but it is difficult to machine and weld. The aluminum alloy is lightweight and has high strength. It is of reasonable value and not related to magnetic effects. Since it is vulnerable to corrosion, surface treatment is also required. High heat conductivity is another advantage for a pressure vessel of aluminum alloy. The titanium alloy has a very high strength-to-weight ratio, excellent corrosion resistance without any surface treatment, low electric conductivity, and no magnetic field distortion. Drawbacks include low heat conductance, difficulties in machining and welding, and high cost.

The most commonly used composite for marine applications is glass-fiber reinforced plastic (GFRP), which has a very high strength-to-weight ratio, excellent corrosion resistance, and a competitive price tag. Carbon fiber reinforced plastic (CFRP) performs better, but it is more expensive. For deep-sea applications, CFRP would bring lots of advantages. The pressure vessel of the Seaglider for 6000 m was made of CFRP because of its ability to achieve weight to displacement ratios of less than 0.5 [25.9]. However, it has not been popular in the market mainly because of difficulties in machining. Metal matrix composites (MMC) and ceramic alloy have many advantages over other materials. In particular, they have high elastic modulus and hardness, high melting points, low thermal expansion, and good chemical resistance.

Instead of metals and composite materials, acrylic (plexiglas), POM (polyoxymethylene, acetal, delrin), PC (polycarbonate lexan), ABS (acrylonitrile–butadiene–styrene), PVC (polyvinyl chloride), and plastic are alternatives for underwater robots operating in shallow-water environments. The main advantages of the abovementioned materials are excellent

Table 25.1 Properties of materials for an underwater robot

Specification	Density (kg/dm^3) (typical)	Tensile strength (MPa), Ultimate (typical)/Yield (typical)	Modulus of elasticity (GPa) (typical)
Steel (HY80)	7.87	482–689/276–517	190–210 (200)
Stainless steel (Type316)	7.75–8.1	620–560/415–290	193
Aluminum 6061-T6	2.7	310/276	68.9
Aluminum 7075-T6	2.81	572/503	71.7
Titanium Ti-6Al-4V	4.43	950/880	113.8
GFRP, E-glass	2.44–2.48	4800–4900/NA	85–90
GFRP, S-glass	2.54–2.60	3400–3500/NA	65–75
CFRP	1.75–1.95	2500–6000/NA	200–800
Acrylic	0.7–1.25 (1.16)	19.3–85.0 (64.3)/25.0–85.0 (60.2)	0.950–3.79 (2.69)
Polycarbonate	0.95–1.54 (1.2)	46.1–124 (64.6)/37.0–191 (64.3)	1.80–7.58 (2.37)
Unreinforced polyoxymethylene	1.14–2.23 (1.42)	5.00–115 (55.6)/22.0–120 (26.8)	0.586–12.1 (2.73)
Acrylonitrile–butadiene–styrene	0.35–3.5 (1.06)	24.1–73.1 (38)/20.0–73.1 (43.3)	0.778–6.10 (2.3)
Polyvinyl chloride	1.4	4.00–59.0/17.0–52.0	0.00159–3.24
Ceramic (96% alumina)	3.7–3.97	2070–2620/260–300	393
Metal matrix composite, Al/SiC	2.78–3.01	230–317	125–270

corrosion resistance and ease of machining and fabrication with relatively low cost. Additionally, POM has high stiffness, low friction, and excellent dimensional stability. PC and acrylic have high-impact strength. They are lightweight and flexible. Even though glass is brittle and, hence, subject to damage from impact, glass spheres are also appropriate for pressure vessels because of the immense strength-to-weight ratio, corrosion resistance, nonmagnetic and electrically nonconductive characteristics, and low cost [25.9]. Some properties are summarized in Table 25.1.

The metal corrosion in seawater is the gradual deterioration of a material based on the electrochemical reaction. There are galvanic corrosion and electrolytic corrosion which are known for fundamentally the same process and known for accelerating rates of the metal corrosion in underwater robot [25.3]. To protect against electrolytic corrosion, insulators, such as resin or rubber are attached to metal contact surfaces [25.6]. However, surface treatment is an effective way to avoid any direct contact of seawater. Anodizing is a surface conversion process that changes the exterior physical properties of a metal to form a hard coating on the surface as results of a controlled electrochemical process. One thing we should remember is that anodizing works well only when the entire surface remains intact. In other words, a small chip in the surface can become the point of accelerated corrosion if there is contact with a less active material [25.10]. A topcoat will provide additional protection against corrosion. Sacrificial anodes, made of highly active metal like zinc and magnesium, are alternative for corrosion protection. As sacrificial anodes corrode and deteriorate, they must be periodically inspected and replaced if needed.

When designing an underwater robot, it is recommended to have small positive buoyancy, except the reserve buoyancy for payload. It will help an underwater robot returning to the surface in the case of any electronics and power failure. The amount of positive buoyancy would depend on the size of robots, typically from a few hundred grams to several kilograms. There are several factors for consideration in choosing buoyancy materials, such as low density, high strength, nontoxic harmless, and weather resistance. The most important factors are low water absorption and low compressibility, which are all dependent on the target operating depth [25.4]. For an underwater robot, expandable polystyrene, polyisocyanurate foam, and syntactic foam are the most common buoyancy materials. The expandable polystyrene (EPS) is a super light and cheap material. However, it is only used for very shallow water because it is compressed by water pressure. Polyisocyanurate foams are generally low-density, insulation-grade foams, which have excellent insulating value and good compressive-strength properties. Since it is usually made in large blocks via a continuous extrusion process, these blocks are reasonably inexpensive and effective for shallow water [25.4]. The syntactic foam consists of glass microballoons and epoxy resin. The amount of trapped air within the resin structure will determine the density as well as the durability of the foam at deeper depths. It is also noted here that ballasting is a procedure of adding, removing, or relocating weights or floatation on an underwater robot to correct its buoyancy, pitch, and roll. There are two types of ballast systems: fixed (static) ballast and variable (active) ballast. In a fixed ballast system, an underwater robot is preset to the desired ballast and operates without any dynamic change of the ballast system. Fixed payload is

usually in the form of several lead blocks, which can be exchanged for equipment without adjusting the robots buoyancy. Variable ballast is not common in most ROVs but the hybrid ROV, *Nereus*, has one [25.8].

25.3 Power Systems

Underwater robots obtain power from any of three categories: on-board power, surface power by a tether, and the combination of the two. For an onboard power system, we can think of many types of power sources, such as primary/secondary batteries, internal/external combustion engines, and fuel cells. For surface powered underwater robots, the mother ship has to deliver power to the robot by a tether. Surface powered underwater robots have advantages of high power capacity and almost unlimited power but also entail additional cost for a winch, a drum, a TMS, and a mother ship. This section reviews typical power sources including batteries, fuel cells, and surface power systems.

25.3.1 On-Board Power Sources

The on-board power source of an underwater robot is an air-independent power (AIP) system, and the most common AIP system is batteries. Performance specifications of untethered vehicles like AUVs, such as working range, service time, and payloads are limited by the battery capacity.

Battery technology has improved over the last decade. One would like to have better battery sources in terms of high energy density, high charge and discharge rates, desirable current characteristics, wide operating/storage temperature range, reliability, long lifecycle, low cost for purchasing and maintenance, and, most importantly, safety against fire and explosion. We should also consider details on safety such as nongassing, spill-proof electrolyte, status monitoring as well as special requirements depending on the shape and volume of an AUV and particular working environment conditions such as depth, under sea ice near the North Pole [25.6, 11, 13].

Primary batteries usually have better endurance than secondary batteries. The lithium primary battery has a very high energy density but its operating cost is more expensive. The alkaline battery can be considered the most common primary battery as it is relatively cheap, but it has safety issues because it outgases hydrogen and it is sensitive to temperature [25.7].

There are a number of different kinds of secondary batteries including lead-acid, silver zinc, nickel–cadmium, nickel-metal hydride, lithium-ion, lithium-polymer batteries. Lead-acid batteries are the oldest type of battery having a low energy density. However, because of its relatively large power rate and low cost, they are the popular choice for many applications. Nickel–cadmium (Ni–Cd) batteries were widely used in commercial products because of their robust characteristics but had problems with the toxicity of the heavy metal cadmium. Nickel metal hydride batteries have been improved to eliminate drawbacks of Ni–Cd and replaced Ni–Cd in the market. During the last couple of decades, the silver–zinc battery was the most used power source in professional AUVs because of its better performance compared to that of old type secondary batteries such as lead-acid and Ni–Cd, even though it is very expensive. Today, lithium-ion and lithium-polymer batteries are popular in our daily life. We can find them in many IT devices. Li-ion batteries using lithium ions as an electrolyte have the advantages of higher energy density, longer cycle life, relatively low self-discharge, low maintenance, and light weight. However, the drawbacks include protection circuits and aging effects. In contrast, lithium-polymer batteries use a solid polymer composite. This reduces the chance of overcharging and electrolyte leakage and is more robust against physical damage [25.12], even though they have a lower energy density and a lower life cycle compared to Li-ion batteries. Some properties are summarized in Table 25.2.

In the literature, as one of the alternatives, seawater batteries with very long endurance capabilities are considered and have been studied to improve the performance. However, due to low power capacity, its applications are limited [25.14].

A fuel cell is an electrochemical device converting the chemical energy of fuels with oxygen directly into electricity to produce electric power, heat, and water. Since there is continuously replenished fuel, a fuel cell has a much larger capacity and longer endurance than batteries at the cost of system complexity. There are various types of fuel cells. The most common types are solid oxide (SOFC), direct methanol (DMFC), proton exchange membrane (PEMFC, or polymer electrolyte (PEFC)), phosphoric acid (PAFC), molten carbonate (MCFC), and alkaline (AFC), where each type operates at different temperature conditions using different materials [25.15, 16].

A closed-cycle PEFC system is known to be the most suitable for underwater and it consists of a fuel cell generator, a high pressure oxygen tank, and a metal hydride tank. Its operating temperature is around 60 °C

Table 25.2 Summary of characteristics of commonly used batteries (after [25.7, 11, 12])

Type	Advantages	Disadvantages	Energy/weight (W h/kg)	Energy/volume (W h/l)	Rating	Life cycle
Alkaline (primary)	Cheap	Leak gas at high temperature	140	–	–	–
Lithium (primary)	High energy density, expensive		375	–	–	–
Lead acid (secondary)	Low cost, long life cycle (1000)	Low energy density, leak hydrogen	20–30	60–80	−40–50	700
Silver–zinc (Ag–Zn) (secondary)	High energy density	Limited life cycle (40–100), need careful maintenance, high cost	100–120	180–200	−48–71	100
Nickel–cadmium (Ni–Cd) (secondary)	Long life cycle, flat discharge curve, good low temperature performance	Memory effects, environmentally unfriendly, exothermic charging process	40–60	50–150	−40–60	2000
Nickel–metal hydride (Ni–MH) (secondary)	High energy density, reduced memory effect	Self-discharging, limited discharge current	60–120	140–300	Ambient	1000
Lithium ion (secondary)	High-energy density, long life cycle, no memory effect	Aging effect	100–265	250–730	−20–45	400–1200
Lithium polymer (secondary)	Flexible form factor, light weight, Improved safety	Lower energy density compared to Li-ion	130–200	170–300	Ambient	300–3000

and its reactive product is only pure water. Urashima, the first prototype of a fuel cell-driven underwater vehicle developed by JAMSTEC, Japan, uses a solid polymer electrolyte fuel cell (PEFC) which allows 60 h endurance and 300 km cruising with max. 4 kW power, and they achieved 60 h of nonstop electric power generation and finished a sea trial [25.17]. In 2007, JAMSTEC started the research to develop a fuel cell for the second generation long-range cruising AUV (LCAUV) to cruise over 3000 km with over 600 h of continuous running [25.12].

The energy density of a fuel cell generator itself is high, but when we consider a whole fuel cell system including subsystems, such as hydrogen, oxygen, and reactant water tanks, auxiliary components, and control electronics, its energy density becomes low. Downsizing of these subsystems is needed for underwater applications. So, a hybrid fuel/battery system would be an alternative to share full loads because it allows both components to be of smaller dimensions and operates with higher efficiency using direct hydrogen and pure oxygen [25.18].

There are two types of loads: propulsion load for propulsion and hotel load for on-board computers, lights, navigation sensors, and payloads. If an AUV is operated for professional long-term missions, we need very accurate simulations for adjusting trade-offs among various constraints. Conventional lead-acid and Ni-based batteries would not be the best choice for AUVs. Instead, lithium-based batteries are the attractive solution that does not entail time-consuming estimation techniques. Nonetheless, lead-acid batteries remain a practical and low-cost option for education, experimentation, and testing.

There are two options for battery installation inside an AUV. One is that they are housed in a pressure vessel, the other one is using pressure-tolerant and waterproof batteries without a pressure vessel. The latter eliminates burdens related to leakage in assembling the system, thus increases system reliability. Lithium-polymer batteries using the solid electrolyte were made without air in the packaging. Hence, they would be a candidate in designing pressure-tolerant on-board power source for AUVs [25.19].

A battery management system (BMS) is a very important subsystem for performance and safety. It monitors the internal status of voltages, current, temperatures and state of charge. Based on the above information, a BMS controls circuits for balancing and protects batteries from over-current, over/under-voltage, over/under-temperature, and any predescribed conditions. As a part of health monitoring, an on-board computer of an AUV should collect status from the BMS to reflect a power budget and health to given missions.

Just like adding volume of auxiliary systems and fuel storage units to calculate an energy density of a fuel cell, we need a measure for fair comparison of the energy density among different power sources. Since a power source is installed inside of an AUV, which should be neutrally buoyant to save the energy in operation, energy density per neutrally buoyant

Table 25.3 Summary of pros and cons of DC and AC (alternating current) (data from [25.4, 20])

Form	Pro	Con	Remarks
DC	Two conductors with less shielding in tether, low voltage drop	Current protection, difficulties in voltage step-up and down	For observation class ROV
AC	Longer transmission distances, flexibility in voltage up and down	Inductance noise, reactive losses, need subsystem for freq. converting	For deep-sea ROV

kilogram should be a reasonable unit. This means that weight and volume for buoyancy materials and pressure vessels must be added to the power source itself, and this also means that the characteristics of buoyancy materials and pressure vessels (aluminum, titanium, composite material) under considering working (or maximum) depth should be designed before or at the same time [25.21]. Between batteries and fuel cells, a semi-fuel cell can be another alternative. It is a generator, but its usability is like a battery because one reactant is stored externally and brought to the reaction cell [25.11]. Aluminum/hydrogen peroxide energy semi-fuel cells can practically generate an energy density of about 400 Wh/kg (theoretically 3418 Wh/kg) which is three times that of a lithium-ion system [25.12]. HUGIN 3000 powered by a semi-fuel cell developed by FFI (Norwegian Defense Research Establishment) [25.7] operates at 4 km down to a sea depth of 3000 m for up to 60 h. This pressure-tolerant system is very attractive in deep-water systems. But it requires trained personnel for operation and maintenance as well as special facilities and tools [25.14].

25.3.2 Surface Power Systems

In the case of ROVs, the whole power is delivered from a mother ship to the ROV through an umbilical/tether cable, offering very high power and theoretically unlimited endurance. These electrical power transmission techniques may look simple, but actually involve a highly complicated procedure for optimization. A power system determines work capability of ROV as well as operating cost, subsystem size, and safety considerations. Typically, it consists of a surface system, a tether, and a subsea system. A surface system includes a shipboard power source (or power outlet), frequency converter, step-up transformers, and a ground fault monitoring device. A frequency converter is used only for high-frequency alternative current, but it is advantageous in handling the power of a subsea system. In a subsea system, there are step-down transformers, rectifiers, power busses, power switching devices for distribution, and a ground fault monitoring device. Also, system configuration and setup reflects the power requirement of the devices installed on a ROV including thrusters, lights, sensors and any circuits. Once again, these design parameters of a power system are not independent of each other [25.4, 20, 22, 23].

A detail configuration depends on the form of current and needed capacity. First of all, we have to make a decision on the form of current. There are three types of current for ROV systems:

1. 50/60 Hz
2. High frequency alternating current (HFAC) at 400 Hz
3. Direct current (DC).

The decision should be based on the design objectives of maximizing power transfer, efficiency and lifetime, as well as minimizing the overall system cost using steady-state analysis and transient analysis [25.20]. Table 25.3 shows pros and cons of the form of current.

Besides maximizing the power transfer to an ROV, we have to minimize the diameter of a cable, system cost, support systems on deck, and the weight of the ROV without losing a high level of electrical power quality. Therefore, most observation class ROVs use the DC power system mainly due to portability and cost, and work-class ROVs have a tendency to use a three phase AC power system.

25.4 Underwater Actuators and Sensors

It is inevitable that the overall performance of an underwater robot depends on the quality of its actuators and sensors. While one can think of different types of actuators for underwater robots, such as thrusters, water jets, and biomimetic actuators (VIDEO 793), most underwater robots use thrusters that use propellers involving hydrodynamics of water flow when generating force to move through water. It is noted that the mapping between thruster command and generated force is not linear, especially in the transient state of motion. The sensors for an underwater robot can be divided into three groups: navigation sensors, system sensors,

and mission sensors. They can also be categorized further by acoustic- or nonacoustic-based sensing. Sensing in underwater environments is very challenging, especially position sensing. There is no electromagnetic wave propagation under water except for very short ranges and, therefore, the GPS (global positioning system) is not available. Due to the limited performance of a single sensor, the sensor fusion approach with multiple sensors is used. For example, the inertial navigation system (INS) includes an IMU (inertial measurement unit) and a DVL (Doppler velocity logger) with highly complicated algorithms. The performance of optical sensing depends on the turbidity level, while acoustic sensing encounters significant challenges due to the dynamic properties and extreme nature of the ocean, such as multipath problems, variation of signal attenuation, variation of acoustic channel, nonlinear effects, and many more [25.24, 25]. This section describes underwater actuators and sensors, and their design-related issues.

25.4.1 Actuators

There are two major types of thrusters – electrical thrusters and electro-hydraulic thrusters – along with other types such as bio-inspired smart actuators and jet propulsion. The efficiency of an electro-hydraulic thruster is about 15%, while the efficiency of a fully electrical thruster is about 43% [25.13], where the definition of efficiency is the ratio of shaft mechanical power output versus electrical power input. In spite of the low energy efficiency, electro-hydraulic thrusters are preferred for large size work-class ROVs because of their extensive thrust and practicality in the rugged conditions of the offshore work environment. Various factors contributing to the power conversion efficiency of the hydraulic propulsion system for ROVs are described in [25.26].

Many small ROVs and AUVs have fully electrical thrusters because of their compact size and high efficiency. Torpedo-type AUVs for cruising commonly use the screw-type propellers, which generate much more power, though only in the forward direction. Small ROVs and AUVs requiring slow but fine motion typically use the propellers generating the same power in both forward/reverse directions. When two thrusters are operating on the same plane and the same direction of motion, an underwater robot must have counter-rotating thrusters to eliminate the counter torque effect of the thrusters rolling the robot opposite to the direction of propeller rotation.

There are many propeller design parameters including propeller diameter, the number of blades, blade pitch, blade thickness, hub size, rake, etc. Propeller diameter and blade pitch are critical design parameters that impact a robot's torque and power, and are related to the size and cost of an underwater robot. Therefore, we have to find an optimum combination of propeller diameter and blade pitch, along with careful matching to a motor to have an efficient electrical propulsion system [25.27]. A typical underwater electrical thruster includes an electric motor, a motor amplifier and controller, a thruster housing typically made of aluminum or titanium, and oil-filled, drive shafts designed for watertight using magnetic coupling or dynamic seal, propellers, and ducts. The Kort nozzle has a relatively simple geometry and it performs efficiently at low speed. The Rice nozzle has a lower drag coefficient and a better geometry to promote increased thrust [25.24].

25.4.2 Nonacoustic Sensors

Various nonacoustic sensors can be considered for measuring an underwater robot's position and pose, and detecting its linear and rotational motion. Magnetic sensors are the most common to get the north direction, but are susceptible to local magnetic disturbances generated by the active elements of a robot system or any magnetized materials in the environment.

Inertial sensors are sensors based on inertia, where linear acceleration is measured by accelerometers and the rate of rotation is measured by gyroscopes without an external reference. There are several types of accelerometers, such as pendulous accelerometers, pendulous integrating gyroscope accelerometers, force rebalance accelerometers, piezoelectric accelerometers, differential capacitor accelerometers, and vibrating beam accelerometers. They have each advantages and disadvantages. Some of them can be built using either conventional mechanical type or microelectromechanical system (MEMS) type. MEMS types like capacitive-based MEMS accelerometers and MEMS vibrating beam accelerometers, show generally higher sensitivity and better resolution [25.28, 29]. A gyroscope (gyro) is used to measure angular velocity of an object. There are three main types of gyro: spinning mass, vibratory, and optical. Spinning-mass type gyros operate on the principle of conservation of angular momentum, and this mechanical type gyro shows wide performance range (0.0001 to $> 100^\circ/\mathrm{h}$). However, it is very expensive. Vibratory (vibrating structure) type gyros use Coriolis forces that are small and of low cost because they use MEMS technology, but they have a limited performance range. Optical type gyros operate under the principle of the Sagnac effect, and there are two main types of optical gyros, RLG (ring-laser gyro) and interferometric fiber-optic gyro (IFOG). A RLG has high sensitivity

and stability, a wide range of measurement (from 0.001 to 1000°/h), excellent scale-factor stability and linearity, insensitivity to accelerations and fast turn-on (<1 s). The RLG technology is a mature technology, and is well established in the medium- and high-performance markets. However, RLGs are still complex and expensive. A fiber-optic gyro (FOG) can provide several important advantages of a pure solid-state device, such as reliable and low maintenance due to no moving part, no sealed cavity and no acoustic noise, and high performance. The FOG technologies are expected to replace many of the current RLG-based systems. However, one particular area where the RLG is expected to retain its superiority over the FOG is in applications requiring extremely high-scale-factor stability.

An attitude and heading reference system (AHRS) is a sensor providing three-dimensional (3-D) orientation information. Typically, it consists of (MEMS) gyroscopes, accelerometers, and magnetometers on all three axes, and an onboard microprocessor running a nonlinear estimation algorithm, such as an extended Kalman filter to compute the orientation. An integration drift is compensated by reference vectors, such as gravity and the earth magnetic field.

The fusion of several navigation sensors can minimize navigation errors. This sensor combination of accelerometers, gyros, navigation computer, and some electronics, called inertia navigation system (INS), provides estimations of velocity, position, and orientation by integrations of the linear accelerations and angular rates with an appropriate transformation between the body frame and the earth frame if given an initial position and orientation [25.12, 23]. Therefore, an INS is one of the most important devices for an underwater robot. Typically, the performance of an INS is usually rated in terms of its position error growth rate. USAF SNU84-1 categorizes INS units as shown in Table 25.4.

To increase the accuracy of a navigation system, an INS has been combined with DVL (Doppler velocity log). As an example, an integrated Doppler-inertial system, Marpos, calculates the relative position from a start point by dead reckoning, which shows a positioning accuracy of around 0.03% of the total distance traveled, the equivalent of 1.7 m per hour at a vehicle speed of three knots by AUV Maridan M600 [25.30].

Table 25.4 Performance classification of INS (USAF SNU84-1)

INS classification	Position error growth rate	Heading errors
Low	> 2 nm/h	> 0.2°
Medium	0.5–2 nm/h	0.05 to 0.2°
Precision	< 0.5 nm/h	< 0.05°

For the vertical motion of an underwater robot, its depth is easily measured by pressure sensors without any cumulative error. Actually, a pressure sensor measures the external pressure experienced by an underwater robot, where the two most common pressure sensor technologies are (i) strain gauges and (ii) quartz crystals. Strain gauge pressure sensors can typically attain overall accuracies of up to about 0.1% of full scale and resolutions of up to about 0.01% of full scale. Quartz crystal pressure sensors can typically attain overall accuracies of about 0.01% of full scale and overall resolution of up to about 0.0001% of full scale, that is, a resolution of one part per million [25.31]. Common units for pressure are atmosphere (atm), bar, kilopascals (kPa), pound of force per square inch (psi), Torr (= mmHg), and one atmosphere (1 atm) of pressure is 101.325 kPa and one bar (1 bar) of pressure is 100 kPa. In sea water, since pressure increases by 1 atm for every 10.0 m of depth, depth is an approximate reading of a pressure sensor multiplied by 10 m/atm. In fresh water, this simple formula underestimates the actual pressure by 2–3%, so you can get a more accurate depth by using 10.3 instead of 10 m/atm [25.3]. Measured pressure by a sensor is relative pressure. For absolute pressure, we have to add 1 atm to the sensor measurement.

Another nonacoustic sensor is an underwater computer vision system that can be considered as part of navigation sensors for underwater robots in addition to simply recording images of the underwater environment. When the underwater environment has high turbidity and insufficient and/or nonuniform light, it would not be easy to obtain enough visual information for the navigation through identifying objects, detecting obstacles, measuring distance between objects, and so on. However, when approaching its target very closely, for example, at the final stage of approaching for docking, the robot could gather useful information using a vision system.

On the other hand, underwater images are major products of underwater missions, especially for the inspection of submerged structures or observation of marine organisms, seafloor geology, etc. In this case, associating time stamp and localization data with the images increases the overall quality of the output. Because of the limited light range, it is very difficult to cover a wide area with one single shot. Thus, an image mosaicking is a popular strategy to overcome this in which several images are merged together in order to create a larger image, just as is done on Google Earth. Real-time image mosaicking is used for navigation stitches images with simple image manipulations, and high-quality image mosaicking is done by post-processing using various optimization techniques.

More recently, advanced, high-quality camera and video management systems have been developed such as HD (high definition) cameras, HD-SDI (serial digital interface) transmission through a fiber-optic cable, high-quality digital video management systems, holographic cameras, laser scanning systems, 3-D camera systems with 3-D monitors, a hybrid optical-acoustic imaging systems incorporating stereographic optical cameras, and high-resolution multibeam sonars [25.12, 32, 33]. Such developments would greatly help increasing the effectiveness of the underwater robot operation.

25.4.3 Acoustics Sensors

There are many ways to calculate the speed of an underwater robot. We can simply estimate the speed by the integral of linear acceleration from an accelerometer even though there are large accumulative errors. We can also measure the speed directly using a pitot tube velocity log, a rotor type velocity log, an electromagnetic velocity log, and an acoustic Doppler current profilers (ADCPs) [25.6], which are all water speed sensors, and are measuring speed with respect to the water, not the ground speed of a robot. ADCP provides measurements of the water's current direction and magnitude by measuring the velocity of particles in water using principles of Doppler shift. However, Doppler velocity logs (DVLs) can be used to measure the velocity of robots moving in the water with respect to the ocean floor. Since ADCP and DVL use the same principles, typically DVL is operated in ADCP mode for a hydrographic instrument and bottom-lock mode for a navigation instrument. Since a DVL measures the Doppler shift of sonar signals reflected by the ground to obtain the velocity, there is a maximum range from seabed to the sensor, which is inversely proportional to the frequency of a signal, typically from 300 kHz to 2 MHz. Table 25.5 shows the performances of DVLs currently available in the market. Recently phased array technology in DVL has been announced and has brought significant advantages over piston array type DVL in terms of size and weight, which have extended the areas of applications for DVLs [25.34].

Table 25.5 Summary of DVL specifications available in market

Specification	Model A	Model B	Model C
Frequency	300 kHz	600 kHz	1200 kHz
Max. range	200–300 m	90–110 m	20–30 m
Velocity range	±10 m/s		
Velocity resolution	0.1 cm/s		
Accuracy	0.4% ± 2 mm/s	0.2% ± 1 mm/s	0.2% ± 1 mm/s
Average power	3–10 W	2–6 W	2–3 W

There are several types of sonar systems such as single-beam directional sonar, scanning sonar, side scan sonar, multibeam sonar, interferometric sonar and synthetic aperture sonar. Single-beam sonars send a pulse of sound at a particular frequency that is reflected from the seabed, and its echo is received by a transducer. It can be implemented in the form of a single channel, for example, an altimeter measuring the distance between a robot and seabed, or multiple channels, for example, obstacle avoidance sonar measuring distances between a robot and any objects around it. A scanning sonar, that is, forward looking sonar, has a small transducer rotating on its axis and emits pulses at the particular direction, and then obtains a set of bins, where a bin represents the echo intensity returning from a specific place along the transducer axis. An image is synthesized by placing the set of bins in a Cartesian space with a set of colors assigning their intensities. On the other hand, a side scan sonar has a fixed transducer emitting fan-shaped acoustic pulses in the cross-track direction, and then the measured acoustic reflections are compiled to create and accurate image of the sea bottom. It is one of the most effective tools for quickly searching for any objects such as wreckage debris or mines on a large area of the seabed. However, it does not provide reliable information on their position [25.35].

An imaging sonar uses a fan beam typically with a 30° 3 dB beam width while retaining about 1.7° angular resolution, whereas a profiling sonar emits a spot or pencil beam with about a 1.7° 3 dB beam width from both horizontal and angular rotation, which means that it provides higher resolution measurements. A multibeam sonar has an array of transducer emitting acoustic pulses simultaneously to cover a large area. This beam pattern is wide in the across-track field and narrow in the along-track field, which is appropriate for collecting highly accurate seafloor information at the cost of the high purchasing expense and high power consumption. An interferometric sonar is one possible tool providing wide-swath coverage and high-resolution bathymetry in shallow water using the technique of superimposing two (or more) waves, which significantly improves the efficiency and safety of hydrographic survey operations. A synthetic aperture sonar (SAS) is a high-resolution acoustic imaging technique that sums multiple returns from multiple pings to produce output imagery with constant along track resolution which is independent of both range and frequency. This is a kind of artificial enlargement of a sonar array by accurate navigation data. Dual frequency technology has been announced for an imaging sonar and side scan

sonar, which offers more flexibility. A typical dual frequency imaging sonar uses frequency between 0.9 and 2.5 MHz to obtain medium and ultra-high-quality resolution from mid- to close-range. A side scan sonar's typical range is from 100 to 500 kHz. Based on a high-quality imaging sonar, researchers have been trying to identify underwater objects. Figure 25.2 shows a result of real-time tracking for artificial landmarks using DIDSON [25.36] (VIDEO 794), where the multiple landmarks are being tracked after being detected and recognized by a probabilistic based algorithm developed by the Korea Research Institute of Ships and Ocean Engineering (KRISO) [25.37].

Acoustic positioning systems are the popular approach to locate the underwater robot by solving a triangulation problem based on distances between an underwater robot and the preinstalled transponders. The distances are calculated by the time of flight of the acoustic signal between them, that is, half the round-trip time multiplied by the speed of sound in water. It is theoretically simple with no cumulative error. However, to obtain accurate position information, we have to have an exact understanding of the system, consideration of the inherent physical error sources, and lots of practical experience. The accuracy inherently depends on the configuration of the baseline, and the speed of sound underwater is affected by water temperature, pressure, and salinity. Additionally, in shallow water or in the highly reverberant underwater environment, multiple reflected copies of any transmitted signal reach the receiver at delayed intervals. It is known as the multipath problem. Therefore, we have to know the characteristics of the underwater acoustic environment at the work site. Before explaining the configuration of types of acoustic positioning system, we need to review functions of its basic component. When an interrogate sends a signal, a transponder installed on the seabed or an underwater robot receives the interrogation signal, and then sends a reply signal to the interrogator. A responder is a transmitter installed on the seabed or an underwater robot just like a transponder, but it can be triggered by a hard-wired external control signal to transmit a signal. If a transmitter sends out a pulse on a particular frequency, it is called a beacon or pinger [25.4].

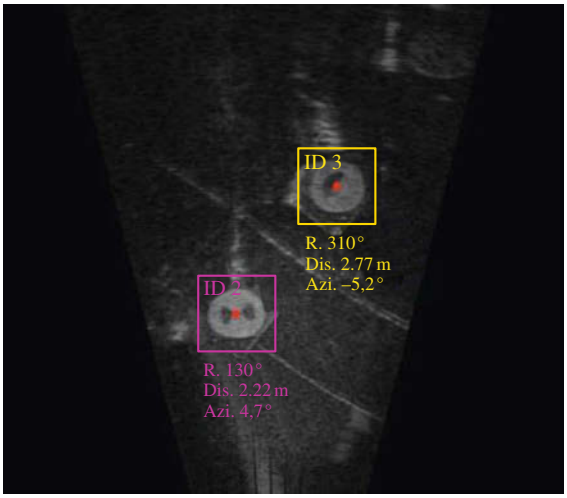


Fig. 25.2 Result of real-time tracking for artificial landmarks using DIDSON

There are three main types of acoustic positioning systems: long baseline (LBL), short baseline (SBL), and ultrashort baseline (USBL). An LBL system determines the position by measuring the slant ranges from three or more transponders mounted widely spaced on the seabed and typically at the corners of the operations site. This results in very high positioning accuracy and robustness regardless of water depth over wide operating area. However, this system requires multiple transponders which inevitably need a long deployment/recovery time and a calibration time required at each location. An update rate is relatively long because of two-way ranging. In an SBL system, there are three or more transducers connected by wires to a central control unit; these transducers construct a baseline installed on from a large barge to a small boat. It is just like an upside down version of an LBL system without requiring any seafloor premounted transponders. However, accuracy depends on the transducer spacing and mounting method. When a wider spacing is available, the performance can be similar to LBL systems. Otherwise, the accuracy goes down. A USBL system consists of a small integrated transducer array and a transponder/responder installed on what we want to trace, where the position of the transponder/responder is calculated by the combination of a target distance by the time of flight of a received signal and a target direction by the phase shift of the received signal on each of the acoustic elements in the transceiver array, called phase-differencing. Since a small error of the target direction calculated by a USBL will become a larger position error over a great distance, it should be mounted on the bottom end of a strong and rigid transducer pole, which is installed either on the side or in some cases on the bottom of a surface vessel. Additional sensors, including high-quality IMU and GPS, have to compensate for errors due to the any changing position and orientation of the integrated transducer array.

USBL systems offer many advantages, such as low complexity, ease of use and not requiring sea floor transponders. But they have also some disadvantages. An installation of a rigid pole for mounting of a transducer array is needed and a calibration process is required to determine offsets for the precise alignment

of a USBL system. Most importantly, their positioning accuracy and robustness are not as good as the LBL systems. In the case of range, all types are quite accurate assuming that the line of sight is maintained. However, the bearing accuracy of LBL systems is less than 1°, whereas SBL and USBL systems typically have 1–3° bearing accuracy [25.4].

The GPS intelligent buoys (GIB) system is available as a promising acoustic positioning system [25.38]. The configuration of GIB is an inverted LBL, where several floating buoys equipped with submerged hydrophones and GPS receivers measure the time of arrivals (TOAs) of the acoustic signals sent by the emitter of an underwater target and from the time measurements tracked position is calculated in real time. This system allows fast and easy installation without a calibration process and a dedicated mother ship and provides real-time multiobjects tracking with a fast update rate because of one-way acoustic signals from the emitter of an underwater target to the floating buoys. The number of floating buoys is determined by the size of the area and the accuracy requirement of given operations.

25.4.4 Other Sensor Issues

Many sensors and communication devices of an underwater robot, such as altimeter, DVL, imaging sonar, ranging sonar, acoustic modem, and acoustic positioning system are sonar-based devices. The integration and concurrent operation of multiple sonar sensors could cause cross-sensor acoustic interference problems. The contamination caused by such sensor crosstalk severely degrades the performance and sometimes, more seriously, mangles the whole sensor system. Basically, we have to mediate a sensor configuration including working frequencies of all sensors and their mounting positions to avoid sensor crosstalk. For mission sensors, we can selectively turn on the sensors as needed, considering other sensors installed on the mother ship. Additionally, potential problems of bio fouling and corrosion of underwater sensors should be taken into consideration and the quality of measurements should be regularly maintained.

Health-monitoring sensors are also very important. Even though they are small and relatively cheap compared to navigation sensors, it could save a robot. Health-monitoring sensors cover multipoint temperature and humidity values inside pressure vessels, status of the power system, for example, status of a battery using a battery management system (BMS) or status of tether cable using a ground fault-monitoring system, and status of the control system. At the same time, an emergency plan should be prepared for cases in which the system is headed toward complete failure. Examples include a situational warning for high temperature or clear thruster commands when communication is lost. An emergency beacon and a flash have to be operated independently from any system of the robot.

25.5 Computers, Communications, and Architecture

Design of computer systems, software architecture, and communication systems is a major task in designing an underwater robot. Designing the overall hardware and software structures and selecting appropriate components are very challenging and crucial. Nowadays, tethered robots or ROVs use a fiber-optic cable for real-time data transfer between a simple on-board computer system and a high-performance computing system on a mother ship. Untethered robots or AUVs require high computing power resources on-board with an underwater wireless communication system such as an acoustic modem.

It is worthwhile to note the Woods Hole Oceanographic Institute (WHOI)'s hybrid ROV, also known as *Nereus* [25.8]. It operates in two different modes. For broad area survey, the vehicle can operate untethered as an AUV capable of exploring and mapping the sea floor with sonars and cameras. For close-up imaging and sampling, *Nereus* can be converted at sea to operate as an ROV, using a lightweight, microthin, fiber-optic tether. *Nereus* uses its manipulator by teleoperation via the fiber-optic tether while the vehicle can be considered an AUV. In May 2009, *Nereus* explored the Mariana Trench, being the first vehicle to explore the Mariana Trench since 1998.

25.5.1 On-Board Computers

ROVs rely on computing systems on a mother ship and typical candidates are industrial PCs installed on a rack or on a shelf inside a control van as a surface control system. The overall system consisting of several PCs with a high-speed network handles data from the ROV and commands from its operator in order to control the ROV. Each computer is designated for a certain mission such as control, navigation, etc. However, the recent trend is to simplify the interface by an integration of multiple PCs and by specific software with expectation of cost saving benefits [25.39].

Unlike ROVs, AUVs must have compact, robust, reliable, and high-performance on-board computer sys-

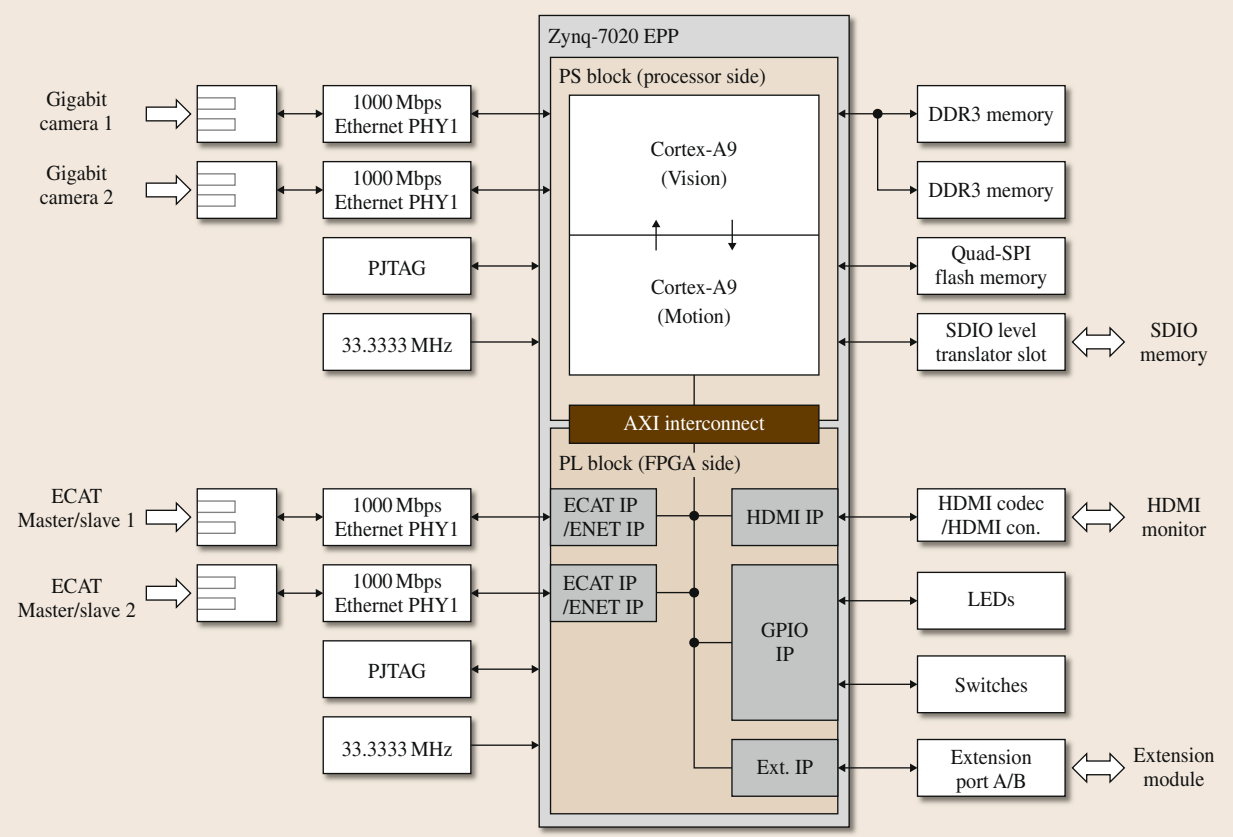


Fig. 25.3 Hardware configuration of Zynq-7020-based system, mController

tems. One popular choice is a VMEbus (Versa Module Europa) system [25.40]. Many AUVs before the year 2000 such as ODIN, Oceans Voyager II, and Aqua Explorer 1000 used a VME system based on Motorola CPU with a real-time operating system, such as VxWorks and OS9 [25.2]. Some AUVs like Kambara and Oberon use Compact peripheral component interconnect (PCI) systems [25.32, 41]. The controller area network (CAN) bus was originally designed as a vehicle bus standard to allow many subsystems such as actuators and sensors to communicate with each other using only two wires by a maximum 1 MHz data transmission rate [25.12]. The CAN bus could bring great advantages in designing the layout of wire harnesses for internal sensors of an AUV, but unfortunately, it has not been popular because there are few internal sensors that support it.

PC-104+ is an embedded computer standard which has no backplane but the boards are stacked on top of each other via stackable ISA, PCI, and PCIe bus connectors [25.33]. PC-104+ based systems are widely used in various types of embedded applications because of their advantages such as small size, stackable, rugged, compact, interoperable, PC compatible high-quality, low cost, and low power consumption [25.33, 40]. Basically, it consists of a CPU board with an optional power supply module. Other required functions can be added by stacking multiple peripheral modules such as frame-grabber, analog digital conveter/digital analog converter (ADC/DAC) converter, digital IO, relay, serial communication, CAN network, and GPS receiver from different manufacturers. Each module offers high-performance specifications and environmental specifications, such as extended operating temperature from -40 to $+85\,^{\circ}\mathrm{C}$. As an example, a decade ago, ODIN, originally developed with a VME-based onboard computer system, was refurbished to a PC-104+ based AUV, ODIN III [25.42]. Today, many recently developed AUVs employ PC-104+ based systems. It is also noted that as a possibility of a heterogeneous bus system, there is a PC-104+ bus-based system (CPU module) along with standard VME-based I/O modules using a protocol interface card [25.40].

PC-104+ is a good choice for a single CPU system. However, when we need more computing power for implementing complicated algorithms, additional PC-104s must be installed because multiple CPUs

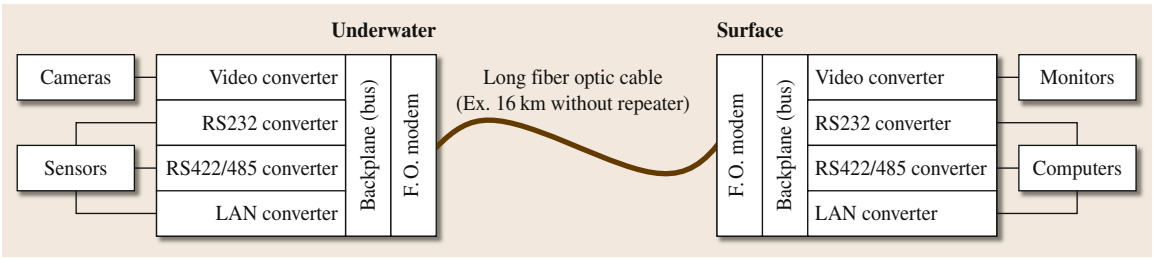


Fig. 25.4 Typical diagram of fiber-optic communication system for ROV

are not supported in a PC-104+ stack. For example, yShark [25.43] has three PC104+s with a high-speed local area network (LAN) for robot control, image processing, and acoustic signal processing. This causes many difficulties, such as a complicated system structure, wiring, space, and cooling in a pressure vessel. Alternative design of a distributed on-board system for an AUV is a combination of a PC-104+ system and an ARM-based system. Besides consumer mobile devices, many mobile robots have used ARM processors 32-bit RISC (reduced instruction set computer) microprocessors because of its high-performance, low power consumption, and low cost. As an example, Fig. 25.3 shows a hardware configuration of mController (announced by Redone technology, Korea) which is based on the Xilinx Zynq-7020 platform [25.44]. Its size is only 125 mm × 125 mm, and it consists of Dual core Cortex-A9 for typical motion control and vision processing, and an FPGA part for data parsing of various sensors. In this hardware hierarchy, PC-104 could be designated to operate high-level tasks.

An on-board-independent microcomputer-based emergency response system is very important to carry out critical functions, such as power disconnection of thrusters and weight drop when the on-board main system is dead. The emergency response system consists of a microprocessor that monitors the status of the on-board main system, an actuator, and a backup battery. Under normal conditions, the on-board main system would regularly check the status of the emergency response system.

25.5.2 Communications

For ROVs, an umbilical cable is used for communication between an ROV and its mother ship. An umbilical cable commonly has a coaxial core or a fiber-optic core. A fiber-optic cable is much better than a coaxial cable in terms of performance and physical dimensions. Light and thin fiber-optic cables offer high-speed, secured, long-distance data transmission without electromagnetic interference, and less drag force on an ROV, even though it is an expensive solution mainly because of optic fiber distribution devices, fiber-optic tools, and the special knowledge required for a handling fiber core. An umbilical cable of a typical ROV has been designed to have 2–3 fiber-optic cores including a spare core. A fiber-optic modem connected to each end of a fiber-optic cable sends and receives data at the same time, working together with several sub-boards that convert data in various formats, for examples, NTSC Video, RS232, and LAN as shown in Fig. 25.4.

Since the demand of high-quality video, such as multiple HD video, 3-D HD video, and UHD (ultrahigh definition) video is rapidly increasing, 100 Gbps optical communication systems are released to handle mass data using a technology known as dense wave division multiplex (DWDM) [25.12]. Cable designers are also making an effort in developing new techniques to add more fiber-optic cores inside an umbilical cable, where each core could be designated to high-quality video devices. The *Nereus* team used a light fiber tether system to send a lightweight battery-powered vehicle to the full depth of the ocean of about 11 km in the ROV mode. The selected Sanmina/SCI buffered fiber is 0.25 mm in diameter, weighs only 0.173 kg for 11 km cable in water, has a working strength of 8 N, and a breaking strength of 108 N [25.5].

For untethered vehicles like AUVs, wireless communication is considered. Typical radio communication would be fine when an AUV is on the surface. In an underwater environment, acoustic communication is the practical solution at the moment even though a commercial acoustic modem has a limited range and a very limited data rate. Table 25.6 shows the specifications of commercial acoustic modems. Acoustic communication is relatively slow compared to radio communication, and it is fundamentally limited by the speed of sound in water, roughly 1500 m/s. Because acoustic attenuation increases with sound frequency, lower frequencies are desirable for long-range communications. But, to generate signals at very low frequencies, larger size transducers are needed. This results in lower data rates that are undesirable [25.2, 13]. The speed and direction of sound propagation vary depending on various factors such as surface waves, currents, sea state, tides,

Table 25.6 Specifications of commercial acoustic modems

Specification	A modem	B modem	C modem	D modem
Size (length × dia)	585 × 183	200 × 165	235 × 126(87)	580 × 150
Material	Aluminum	Aluminum		
Weight (air/water)	11/6	5/1.5	4.2/2.3	21/(n/a)
Voltage	18–50 V_{DC}	10–20 V_{DC}	12–24 V_{DC}	18–28V_{DC}
Power (transmit/receive/sleep)	< 50 W/1/0.03	20/0.6/0.005	2/0.75/0.008	40/0.9/0.009
Working depth	3000	1000	200	7000
Working temperature (°C)	−5–40	−5–40	−5–45	−5–45
Beam width	±25	Hemispherical	210 (Omni)/120 (wide)/70 (narrow)	210 (Omni)/70 (Directional)
Working range	7000 m	Up to 20 km	350 mm	7000 m (Omni)/10 000 m(D)
Frequency	14–22 kHz	7.5–12 kHz	26.77–44.62 kHz	7.5–12.5 kHz
Modulation	QPSK	MFSK and DPSK	Hybrid	Hybrid
Raw data rate	1.5–15 kbps	2 kbps	9.6 kbps	2.5 kbps
User data rate	0.6–10 kbps	0.3 kbps	7 kbps	2.0 kbps
Bit error rate	10^{-9}	10^{-6}	10^{-9}	10^{-7}
Environment			Near vertical, horizontal	Near vertical, horizontal

DPSK: differential phase shift keying

temperature, conductivity, depth, and sea-bottom type. Hence, acoustic communication is inherently delayed, due to propagation delay. Several techniques in signal processing, data packaging, and coding schemes have been developed to overcome these difficulties due to signal absorption, geometric spreading losses, boundary effects, and multipath through.

Acoustic modems consist of three main parts: an underwater transducer, an analog transceiver including a matching preamp and an amplifier, and a hardware platform for control and signal processing, such as microprocessor, digital signal processor (DSP), or field programmable gate array (FPGA) [25.45]. When acoustic communication technology was in its infancy, an analog-based hardware platform was developed. Then digital techniques brought important improvements for increasing the reliability of transmissions using explicit error-correction techniques and increasing some level of compensation for the channel reverberation both in time (multipath) and frequency (Doppler spreading) [25.46]. There are several methods in digital modulation, where the most fundamental techniques are PSK (phase shift keying), FSK (frequency shift keying), ASK (amplitude shift keying), and quadrature amplitude modulation (QAM), and their improved versions are MFSK (multiple FSK), QPSK (quadrature phase shift keying), MPSK (M-ary phase shift keying), MSK (minimum shift keying), GMSK (Gaussian minimum shift keying), MQAM (M-ary quadrature amplitude modulation), and so on.

Just like a typical network, a data packet is packed with a header and a tail for addressing, error detection and correction not to be corrupted by transmission errors. In particular, the data link layer protocol implemented in some modems detects errors in a received packet and then requests retransmissions, which degrades the effective transfer rate. To reduce the retransmission rate, some modems have a function to adjust the speed of transmission depending on the channel condition. Another important technique is about minimizing the effects of multipath, which is most prominent over long ranges and shallow water because the original signal can be bounced between the surface and bottom before arriving at the receiver. There are a few algorithms for robust data transmission in multipath environments, such as convolutional coding, multipath guard period, and data redundancy. Also, a time-reversal technique is an alternative to reducing the effects of multipath [25.12]. From a practical point of view, acoustic communication has some other limitations, such as horizontal communication in shallow water, the loss of line of sight between the modems, the in-band noise (typically caused by another sonar system), echo sounder from ships, a shallow zone, and clouds of air bubbles near a receiver. These limitations might degrade its performance or make it unreliable.

Electromagnetic waves are strongly absorbed by sea water, but, for close-range communication, electromagnetic communication would be useful because it would be less affected by multipass interferences. JAMSTEC (Japan Agency for Marine-Earth Science and Technology) has developed a new communication using electromagnetic waves. This was experimentally evaluated up to 50 m distance [25.12].

Another approach is an optical underwater communication that achieves much higher data transfer rates

than an acoustic communication system at significantly lower power consumption and smaller packaging. However, the operation requires a point-to-point communication setting, requiring the alignment of the receiver and the transmitter for the communication to work effectively. Their range and scope are also affected by the water clarity, water light absorption, and power loss due to propagation spherical spreading [25.47]. Recently, the Massachusetts Institute of Technology (MIT) proposed a real-time video delivery solution based on underwater optical communication, AquaOptical II that consists of a high-bandwidth wireless optical communication device and a two-layer digital encoding scheme designed for error-resistant communication of high-resolution images. Its maximum transmission bandwidth of 4 Mbps and various video streaming tests showed similar performances at close ranges of up to 25 m in a pool trial [25.48], and up to 3–4 m on the Charles River near Boston. It requires accurate motion stabilization because of the narrow beams involved [25.49].

When an AUV is on the surface or an AUV exposes an antenna above the surface, radio communication is the best option. For a short range in which an AUV is operating near shore or its support craft, 900 MHz wireless modems or 2.4 GHz 802.11 wireless networks are common considering antenna sizes [25.49]. For longer ranges, radio links make use of an earth orbiter satellite like Argos satellites and Iridium satellites. In some cases, we have to make our own pressure-tight antenna, including a motion stabilizer to track the satellite regardless of fluctuation of motion. A tracking system for the *Urashima* vehicle was developed using data on network design and the oscillating characteristics of the vehicle [25.12]. Oscillation of the vehicle was measured with the high-accuracy inertial navigation system installed in *Urashima* in a sea trial. An oscillation angle of 7° and period of 0.15 Hz were estimated.

25.5.3 Software Architecture

includes various software components as shown in Fig. 25.5. The lower part of the software architecture is middleware as a class of software technologies designed to help managing the complexity and heterogeneity inherent in distributed systems [25.50]. It is defined as a layer of software above an operating system but below application programs that provide a common programming abstraction across a distributed system.

In the literature, AUV architectures were classified into four categories – the hierarchical architecture, the hierarchical architecture, the subsumption architecture, and the hybrid architecture [25.51] or into three categories – the deliberative architecture, the reactive architecture, and the hybrid architecture [25.52]. Since most of the recently developed architectures are hybrid, instead of describing each architecture, key features of recently proposed software architectures for AUVs are described below.

Teleo-Reactive EXecutive (T-REX) was developed for a specific underwater robot by the Monterey Bay Aquarium Research Institute [25.53–55]. It is a goal-oriented hybrid executive with an embedded automated planning adaptive execution using agents. In order to make the embedded automated planning scalable, the T-REX enables the scope of deliberation to be partitioned functionally and temporally inside units called Teleo-reactors. A Teleo-reactor agent is considered the coordinator of a set of concurrent control loops, which is characterized by functional scope, temporal scope, and timing requirements.

Huxley was developed by Bluefin Robotics originally for its fleet of AUVs. Since it is for their wide range of commercial products, the most fundamental requirement was the flexibility in terms of robustness and reliability, extensibility for new hardware, maintainability, and testability [25.56]. It consists of two layers – a reactive layer and an executive layer – and it allows expanding these core layers through a standard interface. Another key element is the Huxley messaging protocol, called the stream-oriented messaging architecture (SOMA), where SOMA is a publish-subscribe messaging protocol. Huxley's layered architecture with SOMA and SOMA interfaces can be extended along with new functionalities and/or modified configurations of future AUVs while localizing changes and increasing the reusability of components.

The lightweight communications and marshalling (LCM) system was designed as an alternative for implementing inter-process communication (IPC) for real-time robotics in the marine environment. It was

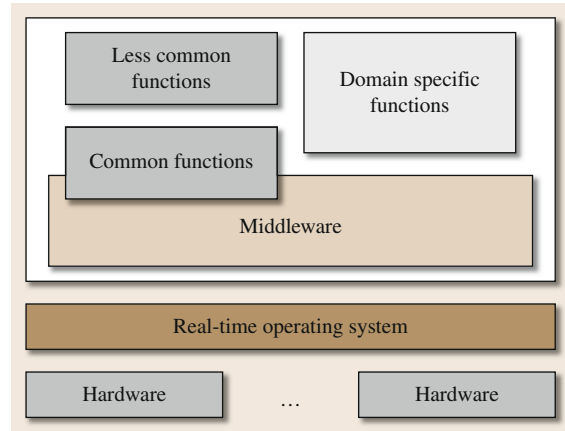


Fig. 25.5 Typical configuration of software for robot

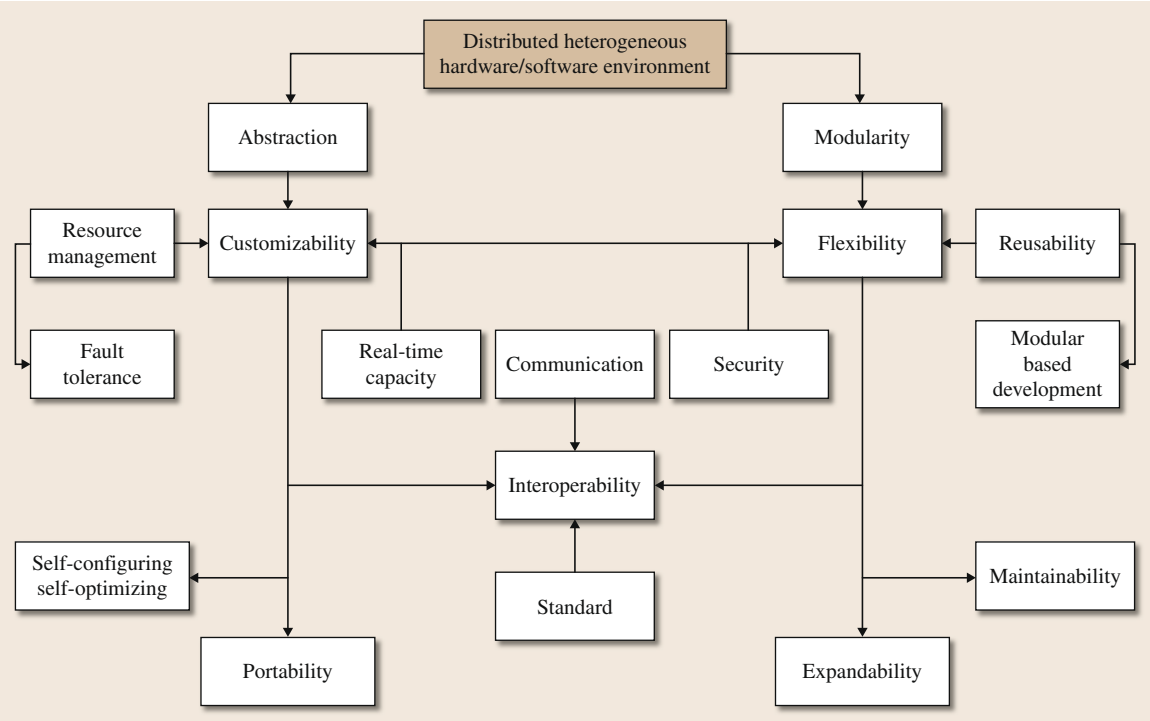


Fig. 25.6 Key characteristics of middleware

developed specifically for low-latency, high-throughput communication accomplished by a set of software tools for message passing between modules using user datagram protocol (UDP) multicast, marshalling (encoding and decoding) of LCM messages, language-independent data structures for messages of any digital data, allowing the transparent distribution of the system. This architecture was verified by two marine robotics applications [25.57].

MOOS-IvP consists of two distinct open source software projects. The Mission Oriented Operating Suite (MOOS) is, as a core middleware, a suite of libraries and executables developed by the Mobile Robotics Group at the University of Oxford. It provides inter-process communication using a publish-subscribe model, as a star topology where all messages go through a central MOOS server. IvP (interval programming) refers to a multiobjective optimization method used by the IvP helm for arbitrating between competing behaviors in its behavior-based architecture. MOOS provides modules for navigation, control and data logging, tools for mission replay from log files and communication debugging, and also provides driver modules for some sensors widely used with AUVs [25.57–59]. MOOS-IvP has been used for several AUVs including the Bluefin 21 in. UUV, the Hydroid REMUS-100 and REMUS-600 UUVs, the Ocean Server Iver2 UUV, the Ocean Explorer 21 in. UUV, autonomous kayaks from Robotic Marine Systems and SARA Inc., and two larger USVs from the NATO Underwater Research Center in La Spezia Italy [25.54, 58].

Let us look at middleware that should be easy to use, robust against change, reliable for any faults, easy to maintain, efficiently flexible, and supportive for heterogeneous and distributed hardware components. Well-structured proven middleware with some often-needed and value-added functions would minimize time and cost for development. In the literature [25.50–53, 56–58, 60–64], there are many demanded characteristics to realize above expectations, as shown in Fig. 25.6. Fundamental characteristics of middleware are abstraction and modularity. Abstraction is expected to simplify and to lessen burdens of application development process, and modularity is a necessary condition of flexibility and customizability. Abstraction allows developers to concentrate on building advanced algorithms themselves by hiding details of low-level hardware resources of heterogeneous and distributed hardware environments [25.51, 62]. This concept includes more specified subissues: interoperability by using simplified interfaces with communication and standard; customizability with real-time capacity; and flexibility and resource management. Additionally, resource management handles faults in hardware. From these characteristics,

a middleware is free from the specific hardware environments, and could obtain portability, self-configuring and self-optimizing. To achieve flexibility, we have to follow a modular-based development for increasing reusability of existing components based on modularity and interoperability. This development approach allows us to focus on building modules without considering specific relationships between existing modules. Based on this, expandability and maintainability can be realized, where expandability is the ability to incorporate new resources, for example, newly developed sensors without interfering with the existing parts, and maintainability is about how easy to embrace modification of components without affecting the rest. To implement a middleware, standards such as the common object request broker architecture (CORBA), the Internet communications engine (ICE), and the open dynamics engine (ODE) are applied [25.64], which bring many advantages. Basically, it requires less effort in implementing and evaluating a core engine from scratch, and some of these have additional functionalities such as standardized interface and security. Security aspects such as authentication, authorization, and secure communication become essential for collaborations of multiple robots or classified applications [25.64]. So far, many research groups have proposed their own middleware for their robots, such as MIRO (middleware for robot), ORCA (open robot control architecture), UPnP (universal plug and play), RT (robot-technology), the player and stage, OPRoS (the open platform for robotic services), ROS (robot operating system), OROCOS (open robot control software), ERSP, MRDS (Microsoft robotics developers studio), MOOS (the mission oriented operating suite), and Webots [25.50].

A development of middleware itself is not a trivial work. Understanding concepts is one thing and implementing them is another. The reasons for possible errors are mainly:

1. Many inherently distributed and heterogeneous components which will vary to improve performance.
2. The limited and restricted resources available on a robot platform.
3. The high rate of system failures from incorporating complicated algorithms and interacting with the environment, causing exceptional situations [25.50, 62].

Basically, an underwater robot's middleware has almost the same requirements as a robot, but the weighting of requirements and the point of view of requirements are little different. For an underwater robot's middleware, the following points must be considered:

1. *Limitation of resources*: Underwater robots use a low bandwidth of an underwater communication and thus, has to manage a mission using internal computing resources.
2. *Unstable sensor information*: Underwater sensors based on sonar are not stable and accurate enough to directly use data for robot control systems. Therefore, assisting algorithms such as well-tuned filtering algorithms and sensor fusion algorithms are needed.
3. *Lack of accessibility*: Underwater robots do not allow any access of an electrical contact used for debugging because of its water-proof pressure vessel.

This is one of the most challenging aspects in building an underwater robot. Thus, for the underwater robot, middleware should provide an independent channel for a supporting a debugging process.

No middleware is perfect, but we believe that we can achieve significant improvements through trial and error coupled with extensive feedback from the community [25.50]. When you develop system software and its architecture on your own, the implementation of abstraction and modularity can be understood with the white-box framework and the black-box framework concepts. The black-box framework supports extensibility by defining interfaces for components without detail information inside the components. White-box, as a base class in a class library can be designed for expanding with subclasses, based on object-oriented language features [25.63]. In the early stage of development, software architecture is mainly conceived as a white box framework. Hence, information inside the boxes, that is, classes, should be understood. As the architecture becomes more advanced and the components become more concrete, the black-box framework is the solution for most common expansions. At the moment, MOOS-IvP would be the best choice to start developing your vehicle software because it is open source software, and there is a relative wealth of information and case examples for AUVs online.

As an alternative to formal software architectures discussed above, we propose a simple but effective solution in testing a robot as shown in Fig. 25.7. This architecture has one real-time thread with several non-real-time threads for controlling actuators and devices installed on the robot.

In the Fig. 25.7:

1. The packet conveys three types of commands from an operator – commands for teleoperation, commands for RT-thread (real-time thread) itself and

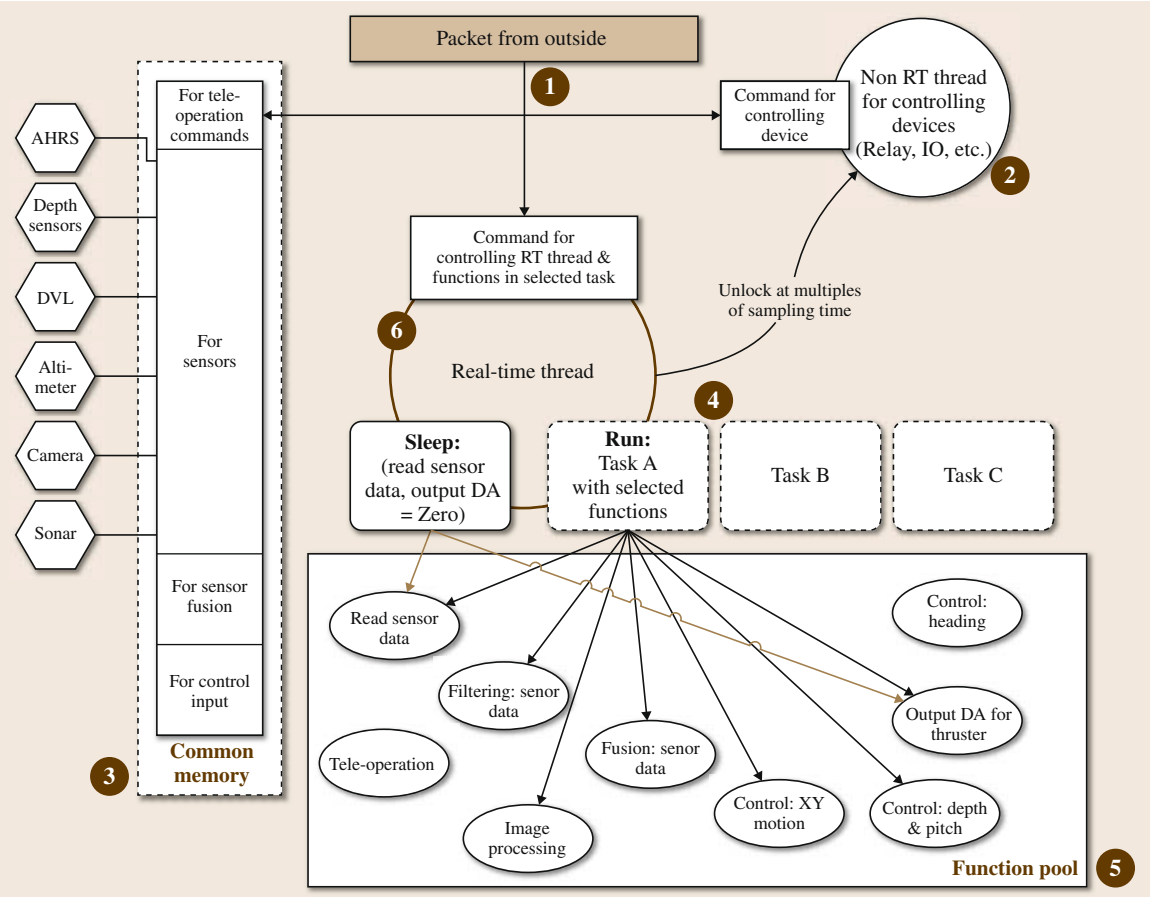


Fig. 25.7 Example diagram of simple software architecture

functions in the selected tasks, and commands for devices of the robot.

2. Device control commands run in a non-RT thread triggered by the RT-thread, typically every 1 s.
3. All sensor data including teleoperation commands and results of any sensor fusion algorithms are independently saved in common memory, and thruster commands, that is, control input are also saved here.
4. This architecture has a manual task coordinator and each task can be easily synthesized using functions from a function pool.
5. A robot is basically in one of two modes, sleep and run, where the sleep mode is doing only health-monitoring with basic sensors, the run mode runs a selected task which consists of subtasks in function pool. Tasks are designed for specific purposes, and one of them is selected by an operator before running. Of course, the default task should be the teleoperation task.
6. The RT-thread carries three main functions: doing any critical device control commands first, DA converting for control input of previous sampling time saved in the common memory, and processing algorithms using a sensor input saved in the common memory.

Since the function of this RT-thread is based on a basic sampling time, we could easily implement any multisampling time structures for various latency times of sensors and load-distributing structures to enhance the usage of computing resources. This would be done by calling functions at multiples of the basic sampling time. This architecture has been applied to several test-bed AUVs and smart ROVs [25.42, 43, 65] (VIDEO 797, VIDEO 799).

25.6 Underwater Manipulators

Most commercial underwater robots with manipulators are ROVs. A typical work-class ROV for intervention missions has a 7 degrees-of-freedom (DOF) manipulator and a 5 DOF grabber. The manipulator is used for interactive manipulation by a human operator on the mother ship, while the grabber is used to maintain a relative position of ROV to the target object by grabbing a target structure in the presence of an underwater current or wave. This section describes underwater manipulator systems based on materials presented in [25.66]. Table 25.7 lists samples of heavy duty underwater manipulators currently available in the market. They are quite expensive but utilize proven technologies in the field. However, most commercial underwater manipulators are hydraulic-driven and developed for ROVs and they are not suitable for AUVs that have limited on-board power and whose on-board sensors could be sensitive to noise. Therefore, electric-driven manipulators would be preferred for AUVs like SAUVIM.

A typical work-class ROV for intervention missions requires two individual operators, one steering the vehicle and the other operating the manipulator. One of the main tasks of the vehicle operator is to keep the ROV as steady as possible (station-keeping) since the vehicle motion can disturb the manipulator and greatly affects the accuracy of its end-effector interacting with the environment. The accuracy of the end-effector would be greatly improved if the manipulator controller could compensate for any disturbance due to the motion of the vehicle. However, most commercial ROV systems do not use vehicle motion compensation control for the manipulator. In the past, a very few AUV systems for intervention missions have been reported and they were used as a test-bed for developing autonomous manipulation or as a working model for research.

25.6.1 Dynamics

The analysis of the dynamics of underwater vehicle manipulators is much more complex than that of regular manipulators on the factory floor. For example, it is difficult to accurately introduce the effects of the added mass and the added moment of inertia in the modeling process for multiple links of the manipulator. The frictional force and the drag force due to the velocity of the manipulator itself, waves, and currents are also complex in modeling. The modeling and control of underwater manipulators were studied by several researchers, including [25.67–78]. In this section, it is not our intention to derive details of the full dynamic equations of the underwater vehicle manipulators. Instead, we would like to point out several specific elements that differentiate underwater vehicle manipulators from regular manipulators. The hydrodynamic forces acting on a rigid body moving in fluid can be found in [25.79, 80]. Similar to the vehicle main body, each link of the manipulator will be influenced by hydrodynamic forces including added mass, buoyancy acting at the center of buoyancy of the link, the fluid acceleration force resulting from the acceleration of the fluid itself, and the fluid drag forces exerted on the link.

As the profile drag forces are dominant for slowly moving objects, the drag force exerting on an infinitesimal element of the i-th link and the total drag force on the link can be expressed as

$$\mathrm{d}D_{\mathrm{f}i} = \frac{1}{2}\rho C_{\mathrm{D}} b_i v_{\mathrm{r}i} |v_{\mathrm{r}i}| \,\mathrm{d}x \,, \tag{25.1}$$

$$D_{\mathrm{f}i} = \frac{1}{2}\rho \int_0^l C_{\mathrm{D}} b_i v_{\mathrm{r}i} |v_{\mathrm{r}i}| \,\mathrm{d}x \,, \tag{25.2}$$

where C_{D} is the drag coefficient, ρ is the fluid density, $v_{\mathrm{r}i}$ is the relative velocity of the element to the fluid flow velocity, $b_i \,\mathrm{d}x$ is a projected area of the element, and l is the link length.

Considering the effects of the hydrodynamic variables above and assuming that the vehicle is stationary during the manipulation, the dynamic model of an underwater manipulator having a series of links and joints can be represented by the following equations,

$$\begin{aligned}\mathbf{M}_{\mathrm{m}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}_{\mathrm{m}}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{D}_{\mathrm{m}}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{F}_{\mathrm{m}}(\dot{\boldsymbol{q}}) \\ + \boldsymbol{G}_{\mathrm{m}}(\boldsymbol{q}) + \boldsymbol{\tau}_{\mathrm{d}} = \boldsymbol{\tau}_{\mathrm{m}} \,,\end{aligned} \tag{25.3}$$

where $\boldsymbol{q} \in \boldsymbol{R}^{\mathrm{n}\times 1}$ is a joint angle vector, $\boldsymbol{\tau}_{\mathrm{m}} \in \boldsymbol{R}^{n\times 1}$ a joint torque vector, $\mathbf{M}_{\mathrm{m}}(\boldsymbol{q}) \in \boldsymbol{R}^{n\times n}$ an inertia matrix including added mass terms, $\boldsymbol{C}_{\mathrm{m}}(\boldsymbol{q},\dot{\boldsymbol{q}}) \in \boldsymbol{R}^{\mathrm{n}\times 1}$ a nonlinear vector arising from centrifugal and Coriolis effects including added mass terms, $\boldsymbol{D}_{\mathrm{m}}(\boldsymbol{q},\dot{\boldsymbol{q}}) \in \boldsymbol{R}^{n\times 1}$ a nonlinear vector due to hydrodynamic forces, such as drag forces, $\boldsymbol{F}_{\mathrm{m}}(\dot{\boldsymbol{q}}) \in \boldsymbol{R}^{n\times 1}$ a nonlinear vector due to friction at the manipulator joints, $\boldsymbol{G}_{\mathrm{m}}(\boldsymbol{q}) \in \boldsymbol{R}^{n\times 1}$ a nonlinear vector due to gravity and buoyancy forces, $\boldsymbol{\tau}_{\mathrm{d}} \in \boldsymbol{R}^{n\times 1}$ a vector of unknown signals due to unmodeled dynamics or external disturbances such as current.

During the manipulation, the end-effector or tool of the manipulator can be in contact with the environment, and forces and moments would occur at the end-effector. In this situation, (25.3) must be modified by adding the reaction torque to the end-effector force on the right-hand side of (25.3)

$$\boldsymbol{\tau}_{\mathrm{r}} = \mathbf{J}_{\mathrm{m}}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{f}_{\mathrm{e}} \,, \tag{25.4}$$

Table 25.7 Samples of commercially available underwater manipulators

Specification	Eca robotics	International submarine engineering (ISE)	Kraft telerobotics	FMC/Schilling robotics	Western space and marine
Model	Arm 7H	Magnum-7	Predator	Titan 4	The ARM
DOF	6 plus gripper	6 plus gripper	6 plus gripper	6 plus gripper	6 plus gripper
Power source	Hydraulic	Hydraulic: max 1000 psi/19 lpm (5 gpm)	Hydraulic: 103 bar–207 bar (1500 psi–3000 psi), 19 lpm (5 gpm)	Hydraulic: 103 bar–207 bar (1500 psi–3000 psi), 19 lpm (5 gpm)	Hydraulic: max 204 bar (3000 psi), 7.6 lpm (2 gpm)
Material	Titanium or aluminum	Aluminum with stainless steel fittings	Anodized aluminum and stainless steel	Titanium	
Max grip force	490 N (110 lbf)	2009 N (450 lbf)	1334 N (300 lbf)	4092 N (920 lbf)	1467 N (330 lbf)
Wrist torque	108 Nm (80 ft – lbs)	108 Nm (80 ft – lbs)	135 Nm (100 ft – lbs)	170 Nm (125 ft – lbs)	
Actuator	Hydraulic cylinder	Hydraulic cylinder	Hydraulic cylinder	Hydraulic cylinder	Hydraulic cylinder
Max reach	1700 mm (66″)	1524 mm (60″)	2019 mm (79.50″)	1920 mm (75.7″)	1700 mm (66″) from azimuth axis to fingers
Max lift capacity		454 kg (1000 lbs)	227 kg (500 lbs)	454 kg (1000 lbs)	45.4 Kg (100 lbs)
Max lift at full extent	90 kg (198 lbs)	295 kg (650 lbs)	91 kg (200 lbs)	122 kg (270 lbs)	29.5 kg (65 lbs)
Working depth (standard)	7500 msw (24 600 fsw)	6000 msw (19 700 fsw)	3000 msw (9800 fsw)	4000 msw (13 100 fsw)	Unlimited
Working depth (Extended)	Unlimited	11 000 msw (36 000 fsw)	6500 msw (21 000 fsw)	7000 msw (22 967 fsw)	Unlimited
Homepage	www.ecarobotics.com	www.ise.bc.ca	www.krafttelerobotics.com/	www.fmctechnologies.com/Schillingrobotics.aspx	www.wsminc.com

where $\mathbf{J}_m \in \boldsymbol{R}^{6\times n}$ is the manipulator Jacobian matrix, and $\boldsymbol{f}_e \in \boldsymbol{R}^{6\times 1}$ a vector of forces, and torques at the end-effector.

The dynamics of the underwater vehicle manipulator is quite complicated, highly nonlinear, and involves coupled equations with unknown parameters and disturbances as shown in (25.3). It is almost impossible to accurately model the dynamic equations and to operate the manipulator at a nominal speed using a conventional controller. However, when the manipulator moves very slowly in a friendly environment with no or minimal current or wave, the dynamics of the manipulator can be represented by the simplified linear version of (25.3) since the effect of velocity-dependent terms become negligible. In fact, during the actual operation in the field, most commercial underwater manipulators move very slowly (much less than 1 rad/s) for safety and other reasons, and they use conventional joint controllers for the master-slave teleoperation.

25.6.2 Teleoperation

A typical setup for teleoperation involves the operator sitting in the control room located on the mother ship and holding a master arm that is a miniature of the actual underwater vehicle manipulator (slave arm) attached to the ROV. The operator controls the slave arm by moving the master arm and its motion is followed by the slave arm. During the operation, the operator relies on visual information of the work site in the form of a series of two-dimensional (2-D) video images that are captured by cameras on the vehicle main body and the slave arm. The images are then transmitted to the control room monitors by an underwater cable. The view of the work site is not only limited, but the visual information especially for the deep sea operation is also often delayed. If the vehicle is not holding or sitting on an underwater structure, the ROV operator must try to control the vehicle for station-keeping during the manipulation. However, even if there is no current or wave during the station-keeping, the vehicle moves like free floating within the accuracy of the vehicle position sensors in the order of a meter or so. Therefore, it is difficult to achieve the accuracy at the end-effector since underwater vehicle manipulators are attached to the vehicles that are constantly moving, unlike industrial manipulators, whose bases are fixed on the factory floor. As mentioned in [25.81], many simple tasks on the ground such as plugging become very difficult to perform by the underwater vehicle manipulator. Underwater plugging tasks, done on a trial-error basis as the required precision cannot be achieved by the ROV manipulator

system, take hours to complete. Therefore, operator fatigue often becomes a critical issue in tele-operating the underwater vehicle manipulator.

25.6.3 Autonomous Manipulation

Autonomous manipulation on a moving base, such as terrestrial mobile robots, humanoids, and underwater robotic vehicles is a very challenging task in the area of robotics in general, especially in unstructured environments, such as underwater. It is defined as the capability of a robot system that performs intervention tasks requiring physical contact with unstructured environments without continuous human supervision. Unlike industrial manipulators which have fixed bases on the floor, autonomous manipulation requires a system capable of assessing a situation, including self-calibration based on sensory information, and executing or revising a course of manipulating action without continuous human intervention. Therefore, developing a system capable of fully autonomous manipulation would be a great achievement and make a substantial impact on a variety of application areas with significant economical, societal, and scientific importance [25.82].

Let us look at one scenario of cutting an underwater cable. With given information about the cable's location and shape, the vehicle has to navigate to the location, identify the cable, position itself, and cut the cable. It may sound like a very simple task. However, it would not be so simple if it has to be done in autonomous mode. Even in teleoperation mode, it would not be an easy task without the coordinated motion control if the vehicle is floating in water. In fact, the low bandwidth and significant time delay inherent in acoustic subsea communications represent a considerable obstacle to remotely operate a manipulation system, making it impossible for remote controllers to react to problems in a timely manner. Nevertheless, robots for autonomous underwater intervention would pave the way for a different range of new operations, such as deep-ocean and under-ice exploration, tasks in hazardous areas, tasks in natural or man-made disastrous regions, automated searches, and surveillance missions, to name a few.

In the past, many researchers have studied advanced control of AUV itself as reported in [25.83–91], but only a few AUVs with manipulators were introduced. OTTER is an AUV equipped with a single degree-of-freedom arm, which was designed to be used as a test-bed for developing autonomous technologies at

Fig. 25.8 The SAUVIM vehicle (Sand Island, Hawaii, 2008)

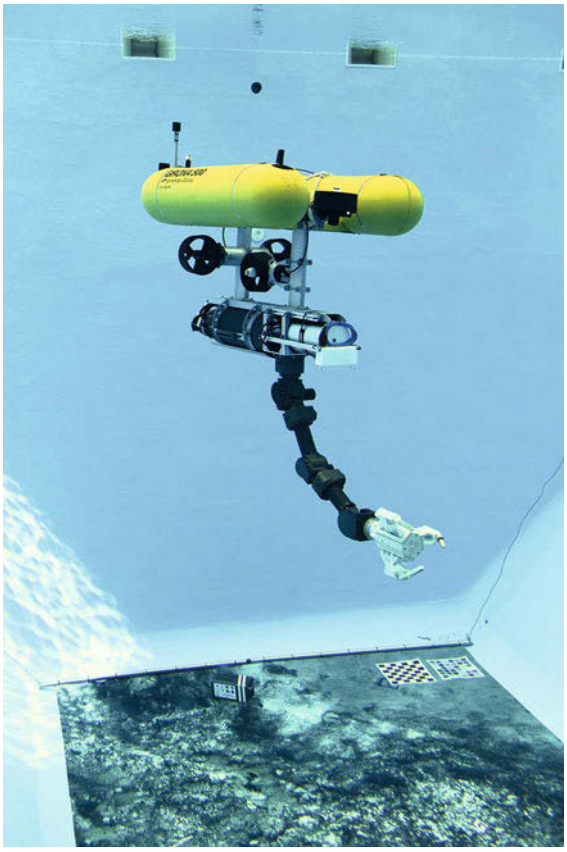


Fig. 25.9 Girona 500 I-AUV

the Stanford Aerospace Robotics Lab in 1996. It is a hovering-capable underwater vehicle that operates in a test tank at the Monterey Bay Aquarium Research Institute (MBARI) [25.92]. Another intervention AUV, namely ALIVE, was developed by Cybernetix in 2003. The aim of the EU-funded ALIVE project was to develop an intervention-AUV capable of docking to a subsea structure, which has not been specifically modified for AUV use. A description of the ALIVE vehicle was given in [25.93].

The key technology in underwater intervention performed with autonomous vehicles is autonomous manipulation. In the literature, no such system fully functional for autonomous manipulation was reported until recent developments with a semi-autonomous underwater vehicle for intervention missions, SAUVIM [25.94–97]. SAUVIM (Fig. 25.8) funded by the US Office of Naval Research was jointly developed by the Autonomous Systems Laboratory (ASL) of the University of Hawaii, Marine Autonomous Systems Engineering (MASE), Inc. in Hawaii, and Naval Undersea Warfare Center Division Newport (NUWC) in Rhode Island. The first fully autonomous manipulation in an unstructured ocean environment was demonstrated with SAUVIM at Snug Harbor, Honolulu, Hawaii on January 20, 2010. Following its success, more underwater vehicles having the capability of autonomous manipulation are expected to develop in the near future. Among some exemplar, underwater intervention tasks are object recovery/rescue, and maintenance/repairing of underwater facilities. Therefore, one may consider a task like black-box recovery, grasping an object of interest and placing it at a desired location for benchmarks in future development. One recent development after SAUVIM is the TRIDENT project [25.98–101] in Europe, which developed Girona 500 I-AUV (Fig. 25.9). They performed an autonomous underwater intervention task, with a black-box recovery in Port de Sóllers harbor (Mallorca) in October 2012 [25.14].

As subsea business in oil and gas industry is about to boom in the very near future, AUVs for intervention missions will receive more attention. While autonomous underwater manipulation is still an active research topic, recent development with SAUVIM and Girona 500 I-AUV would accelerate advancement in the field.

25.7 Conclusions and Further Reading

In this chapter, we described major subsystems of underwater robots – mechanical systems, power systems, actuators, sensors, computing systems, software architecture, communications, and manipulator systems, and also discussed critical design issues from the authors' practical experiences.

There are currently more than 1000 work-class ROVs. They are used for various applications, such as scientific research, military operations, and underwater constructions. About 57% of ROVs are used for offshore oil and gas industry. Most commercial ROVs have used old but proven technologies in the past 20 years. However, the ROV operation is very expensive as it requires a mother ship and, therefore, developing advanced ROVs or smart ROVs for efficient operations in terms of operating hours and costs has received much attention. There are more than 550 AUVs in use worldwide and some of them are commercially available. While the use of AUVs is still limited, advantages of AUVs have been recognized by the recent AUV operations in the field: post-incident monitoring of BP Macondo oil leakage in 2010 and the successful recovery of black boxes of Air France flight 447 that crashed into the Atlantic Ocean in 2009. Therefore, it is expected to see more underwater robots for various applications in the future.

Some web-based resources related to the topics of this chapter include the following sites: the Remotely Operated Vehicles Committee of the Marine Technology Society (http://www.rov.org/), the Autonomous Undersea Vehicles Application Center (AUVAC, http://www.auvac.org) and the technical committee on Marine Robotics of the IEEE Society of Robotics and Automation (http://webuser.unicas.it/MarineRoboticsTC). A list of AUVs and developers can be found at http://www.transitport.net/Lists/AUVs.Org.html. AUVSI (Association for Unmanned Vehicle Systems International) and ONR (US Office of Naval Research) sponsor International Autonomous Underwater Vehicle Competition as well as Autonomous Surface Vehicle competition (http://www.auvsifoundation.org/AUVSI/FOUNDATION/Competitions/).

It is worthwhile to note that several important initiatives [25.1, 101] in underwater robotics in Europe have been made including AMADEUS for developing dexterous arms for underwater manipulation; GREX – coordination and control of cooperating heterogeneous unmanned systems in uncertain environments; CO-3AUVs – cognitive cooperative control for AUVs; FREESUBNET – a Marie Curie research training network on marine robotics; and TRIDENT that developed Girona 500 I-AUV for autonomous underwater intervention tasks.

Video-References

VIDEO 793 Six-legged waking underwater robot, Crabster
available from http://handbookofrobotics.org/view-chapter/25/videodetails/793

VIDEO 794 Preliminary results of sonar-based SLAM using landmarks
available from http://handbookofrobotics.org/view-chapter/25/videodetails/794

VIDEO 796 First record of deep-sea diving of Hamire, depth was 5882m
available from http://handbookofrobotics.org/view-chapter/25/videodetails/796

VIDEO 797 Preliminary experimental result of an ROV, iTurtle
available from http://handbookofrobotics.org/view-chapter/25/videodetails/797

VIDEO 799 Preliminary experimental result of an AUV, yShark2
available from http://handbookofrobotics.org/view-chapter/25/videodetails/799

References

25.1 J. Yuh, G. Marani, D.R. Blidberg: Applications of marine robotic vehicles, Intell. Serv. Robotics **4**(4), 221–231 (2011)

25.2 J. Yuh: Design and control of autonomous underwater robots: A survey, Auton. Robots **8**(1), 7–24 (2000)

25.3 S.W. Moore, H. Bohm, V. Jensen: *Underwater Robotics: Science, Design and Fabrication* (Marine Advanced Technology Education MATE Center, Monterey 2010)

25.4 R.D. Christ, R.L. Wernli: *The ROV Manual: A User Guide for Observation Class Remotely Operated Vehicles* (Elsevier, Amsterdam 2007)

25.5 K. Hardy, S. Weston, J. Sanderson: Under pressure: Testing before deployment is integral to success at sea, Sea Technol. **50**(2), 19–25 (2009)

25.6 T. Hyakudome: Design of autonomous underwater vehicle, Int. J. Adv. Robotic Syst. **8**(1), 131–139 (2011)

25.7 W.H. Wang, R.C. Engelaar, X.Q. Chen, J.G. Chase: The state-of-art of underwater vehicles – Theories and applications. In: *Mobile Robots – State of the Art in Land, Sea, Air, and Collaborative Missions*, ed. by X.Q. Chen, Y.Q. Chen, J.G. Chase (InTech, Rijeka 2009)

25.8 A.D. Bowen, D.R. Yoerger, C. Taylor, R. McCabe, J. Howland, D. Gomez-Ibanez, J.C. Kinsey, M. Heintz, G. McDonald, D.B. Peter, S.B. Fletcher, C. Young, J. Buescher, L.L. Whitcomb, S.C. Martin, S.E. Webster, M.V. Jakuba: The Nereus hybrid underwater robotic vehicle for global ocean science operations to 11,000 m depth, Proc. MTS/IEEE Ocean (2007)

25.9 T.J. Osse, T.J. Lee: Composite pressure hulls for autonomous underwater vehicles, Proc. MTS/IEEE Ocean (2007)

25.10 K. Hardy: Anodizing aluminum for underwater applications, Ocean News Technol. **15**(3), 54–56 (2009)

25.11 S.M.A. Sharkh, G. Griffiths, A.T. Webb: Power sources for unmanned underwater vehicles. In: *Technology and Applications of Autonomous Underwater Vehicles*, ed. by G. Griffiths (Taylor Francis, New York 2002) pp. 19–35

25.12 H. Yoshida: Fundamentals of underwater vehicle hardware and their applications. In: *Underwater Vehicles*, ed. by A.V. Inzartsev (InTech, Rijeka 2009) pp. 557–582

25.13 L.L. Whitcomb: Underwater robotics: Out of the research laboratory and into the field, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 709–716

25.14 Ø. Hasvold, N.J. Størkersen, S. Forseth, T. Lian: Power sources for autonomous underwater vehicles, J. Power Sourc. **162**(2), 935–942 (2006)

25.15 A. Mendez, T.J. Leo, M.A. Herreros: Fuel cell power systems for autonomous underwater vehicles: State of the art, Proc. Int. Conf. Energy (2014)

25.16 K.L. Davies, R.M. Moore: Unmanned underwater vehicle fuel cell energy/power system technology assessment, IEEE J. Ocean Eng. **32**(2), 365–372 (2007)

25.17 H. Yoshida, T. Sawa, T. Hyakudome, S. Ishibashi, T. Tani, M. Iwata, T. Moriga: The high efficiency multi-less (HEML) fuel cell – A high energy source for underwater vehicles, buoys, and stations, Proc. MTS/IEEE Ocean (2011)

25.18 Q. Cai, D.J. Browning, D.J. Brett, N.P. Brandon: Hybrid fuel cell/battery power systems for underwater vehicles, Proc. 3rd SEAS DTC (2007)

25.19 K. E. Robinson: Li-poly pressure-tolerant batteries dive deep, Batter. Power Prod. Technol. **11**(2), 999999 (2007)

25.20 M.C. Wrinch, M.A. Tomim, J. Marti: An analysis of sub sea electric power transmission techniques from DC to AC 50/60 Hz and beyond, Proc. MTS/IEEE Ocean. (2007)

25.21 N. Størkersen, Ø. Hasvold: Power sources for AUVs, Proc. Sci. Def. Conf. (2004)

25.22 S. Cohan: Trends in ROV development, Mar. Technol. Soc. J. **42**(1), 38–43 (2008)

25.23 E. Mellinger: Power system for new MBARI ROV, Proc. IEEE Oceans (1993) pp. 152–157

25.24 M.R. Arshad: Recent advancement in sensor technology for underwater applications, Indian J. Mar. Sci. **38**(3), 267–273 (2009)

25.25 L. Lionel: Underwater robots part I: Current systems and problem pose. In: *Mobile Robotics*,

ed. by A. Lazinica (InTech, Rijeka 2006) pp. 335–360
25.26 S.M.A. Sharkh: Propulsion systems for AUVs. In: *Technology and Applications of Autonomous Underwater Vehicles*, ed. by G. Griffiths (Taylor Francis, New York 2002) pp. 109–1255
25.27 T. Schilling, W. Klassen, C. Barrett, J. Stanley: Power at depth: Efficient ROV power delivery and thrust generation for improved construction, repair, and maintenance support, Proc. Offshore Technol. Conf. (2005)
25.28 P.D. Groves: *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems* (Artech House, Boston 2013)
25.29 F. Viksten: *On the Use of an Accelerometer for Identification of a Flexible Manipulator*, Master Thesis (Linköping Univ., Linköping 2001)
25.30 J. Romeo, G. Lester: Navigation is key to AUV missions, Sea Technol. **42**(12), 24–30 (2001)
25.31 J.C. Kinsey, R.M. Eustice, L.L. Whitcomb: Aŝurvey of underwater vehicle navigation: Recent advances and new challenges, Proc. Int. Conf. Manoeuvering Control Mar. Craft, Lisbon (2006)
25.32 C. Silpa-Anan, T. Brinsmead, S. Abdallah, A. Zelinsky: Preliminary experiments in visual servo control for autonomous underwater vehicle, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1824–1829
25.33 RTD Embedded Technologies, Inc., What is PC/104? http://www.rtd.com/PC104/Default.htm (2014)
25.34 A. Kenny, G. Lopez: Advances in and extended application areas for Doppler sonar, Proc. MTS/IEEE Ocean. (2012)
25.35 E. Thurman, J. Riordan, D. Toal: Multi-sonar integration and the advent of senor intelligence. In: *Advances in Sonar Technology*, ed. by S.R. Silva (InTech, Rijeka 2009) pp. 151–164
25.36 Sound Metrics Corporation (Bellevue, WA): Didson Sonar, L http://www.soundmetrics.com/products/didson-sonars (2015)
25.37 Y. Lee, T.G. Kim, H.-T. Choi: Añew approach of detection and recognition for artificial landmarks from noisy acoustic images, Adv. Intell. Syst. Comput. **274**, 851–858 (2014)
25.38 A. Alcocer, P. Oliveira, A. Pascoal: Underwater acoustic positioning systems based on buoys with GPS, Proc. 8th Europ. Conf. Underw. Acoust., Vol. 8 (2006) pp. 1–8
25.39 L. Brun: ROV/AUV trends market and technology, Mar. Technol. Rep. **5**(7), 48–51 (2012)
25.40 G. Verma, M. Kalra, S.K. Jain, D.A. Roy, B.B. Biswas: Embedded PC based controller for use in VME bus based data acquisition system, Proc. 9th Int. Workshop Pers. Comput. Part. Accel. Controls (2012) pp. 65–76
25.41 S.B. Williams, P. Newman, G. Dissanayake, J. Rosenblatt, H. Durrant-Whyte: Aďecoupled, distributed auv control architecture, Int. Symp. Robotics **31**, 246–251 (2000)
25.42 H.-T. Choi, A. Hanai, S.K. Choi, J. Yuh: Development of an underwater robot, ODIN-III. Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (2003) pp. 836–841
25.43 D. Lee, G. Kim, D. Kim, H. Myung, H. Choi: Vision-based object detection and tracking for autonomous navigation of underwater robots, Ocean Eng. **48**, 59–68 (2012)
25.44 Xilinx, Inc.: Zynq-7000 all programmable SoC, http://www.xilinx.com/products/silicon-devices/soc/zynq-7000/index.htm (2014)
25.45 B. Benson, Y. Li, R. Kastner, B. Faunce, K. Domond, D. Kimball, C. Schurgers: Design of a low-cost, underwater acoustic modem for short-range sensor networks, Proc. MTS/IEEE Ocean. (2010)
25.46 D.B. Kilfoyle, A.B. Baggeroer: The state of the art in underwater acoustic telemetry, IEEE J. Ocean. Eng. **25**(1), 4–27 (2000)
25.47 Z. Jiang: Underwater acoustic networks–issues and solutions, Int. J. Intell. Control Syst. **13**(3), 152–161 (2008)
25.48 M.W. Doniec, A. Xu, D. Rus: Robust real-time high definition underwater video streaming with AquaOptical II, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Karlsruhe (2013)
25.49 J.W. Nicholson, A.J. Healey: The present state of autonomous underwater vehicle (AUV) applications and technologies, Mar. Technol. Soc. J. **42**(1), 44–51 (2008)
25.50 W.D. Smart: Is ačommon middleware for robotics possible?, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007)
25.51 K.P. Valavanis, D. Gracanin, M. Matijasevic, R. Kolluru, G.A. Demetriou: Control architectures for autonomous underwater vehicles, IEEE Control Syst. **17**(6), 48–64 (1997)
25.52 P. Ridao, J. Yuh, J. Batlle, K. Sugihara: On AUV control architecture, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 855–860
25.53 C. McGann, F. Py, K. Rajan, H. Thomas, R. Henthorn, R. McEwen: T-rex: Aḿodel-based architecture for AUV control, Proc. 3rd Workshop Plan. Plan Exec. Real-World Syst. (2007)
25.54 E.F. Perdomo, J.C. Gámez, A.C.D. Brito, D.H. Sosa: Mission specification in underwater robotics, J. Phys. Agents **4**(1), 25–34 (2010)
25.55 M.L. Seto (Ed.): *Marine Robot Autonomy* (Springer, New York 2013)
25.56 D. Goldberg: Huxley: Af̃lexible robot control architecture for autonomous underwater vehicles, Proc. MTS/IEEE Ocean. (2011)
25.57 B.S. Bingham, J.M. Walls, R.M. Eustice: Development of af̃lexible command and control software architecture for marine robotic applications, Mar. Technol. Soc. J. **45**(3), 25–36 (2011)
25.58 MOOS-IvP: http://oceanai.mit.edu/moos-ivp/pmwiki/pmwiki.php
25.59 M.R. Benjamin, H. Schmidt, P.M. Newman, J.J. Leonard: Nested autonomy for unmanned marine vehicles with MOOS-IvP, J. Field Robotics **27**(6), 834–875 (2010)
25.60 C. Lin, X. Feng, Y. Li, K. Liu: Toward ag̃eneralized architecture for unmanned underwater vehicles,

Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 2368–2373
25.61 N. Mohamed, J. Al-Jaroodi, I. Jawhar: Middleware for robotics: A survey, Proc. IEEE Conf. Robotics, Autom. Mechatron. (2008) pp. 736–742
25.62 M. Namoshe, N.S. Tlale, C.M. Kumile, G. Bright: Open middleware for robotics, Proc. 5th Int. Conf. Mechatron. Mach. Vis. Pract. (2008) pp. 189–194
25.63 D. Brugali, G.S. Broten, A. Cisternino, D. Colombo, J. Fritsch, B. Gerkey, G. Kraetzschmar, R. Vaughan, H. Utz: Trends in robotic software frameworks. In: *Software Engineering for Experimental Robotics*, ed. by D. Brugali (Springer, Berlin, Heidelberg 2007) pp. 259–266
25.64 A. Elkady, T. Sobh: Robotics middleware: A comprehensive literature survey and attribute-based bibliography, J. Robotics **2012**, 959013 (2012)
25.65 S. Kim, H.-T. Choi, J.-W. Lee, Y.J. Lee: Design, implementation, and experiment of an underwater robot for effective inspection of underwater structures, Proc. 2nd Int. Conf. Robot Intell. Technol. Appl. (2013)
25.66 T.W. Kim, J. Yuh, G. Marani: Underwater vehicle manipulators. In: *Springer Handbook of Ocean Engineering*, ed. by M. Dhanak, N. Xiros (Springer, Berlin, Heidelberg, 2016), in press.
25.67 N. Kato, D.M. Lane: Coordinated control of multiple manipulators in underwater robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1996) pp. 2505–2510
25.68 M.W. Dunnigan, D.M. Lane, A.C. Clegg, I. Edwards: Hybrid position/force control of a hydraulic underwater manipulator, Proc. IEEE Control Theory Appl. **143**, 145–151 (1996)
25.69 B. Lévesque, M.J. Richard: Dynamic analysis of a manipulator in a fluid environment, Int. J. Robotics Res. **13**(3), 221–231 (1994)
25.70 H. Mahesh, J. Yuh, R. Lakshmi: A coordinated control of an underwater vehicle and robotic manipulator, J. Robotics Syst. **8**(3), 339–370 (1991)
25.71 S. McMillan, D.E. Orin, R.B. McGhee: Efficient dynamic simulation of an underwater vehicle with a robotic manipulator, IEEE Trans. Syst. Man Cybern. **25**(8), 1194–1206 (1995)
25.72 T.W. McLain, S.M. Rock, M.J. Lee: Experiments in the coordinated control of an underwater arm/vehicle system, Autom. Robotics **3**, 213–232 (1996)
25.73 T.J. Tarn, G.A. Shoults, S.P. Yang: A dynamic model of an underwater vehicle with a robotic manipulator using kanes method, Autom. Robot. **3**, 269–283 (1996)
25.74 K. Ioi, K. Itoh: Modelling and simulation of an underwater manipulator, Adv. Robot. **4**(4), 303–317 (1989)
25.75 I. Schjølberg, T.I. Fossen: Modelling and control of underwater vehicle-manipulator systems, Proc. 3rd Conf. Mar. Craft Manoeuvering Control (1994)
25.76 K.N. Leabourne, S.M. Rock: Model development of an underwater manipulator for coordinated arm-vehicle control, Proc. MTS/IEEE Ocean., Vol. 2 (1998) pp. 941–946
25.77 M. Lee, H.-S. Choi: A robust neural controller for underwater robot manipulators, IEEE Trans. Neural Netw. **11**(6), 1465–1470 (2000)
25.78 J.-H. Ryu, D.-S. Kwon, P.-M. Lee: Control of underwater manipulators mounted on an ROV using base force information, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (2001) pp. 3238–3243
25.79 M.H. Patel: *Dynamics of Offshore Structures* (Butterworths, London 1989)
25.80 A.W. Troesch, S.K. Kim: Hydrodynamic forces acting on cylinders oscillating at small amplitudes, J. Fluids Struct. **5**(1), 113–126 (1991)
25.81 M. Hildebrandt, L. Christensen, J. Kerdels, J. Albiez, F. Kirchner: Realtime motion compensation for ROV-based teleoperated underwater manipulators, Proc. MTS/IEEE Ocean. Eur. (2009) pp. 1–6
25.82 O. Brock, R. Grupen: *Final Report of NSF/NASA Workshop on Autonomous Mobile Manipulation (AMM)* (Univ. of Massachusetts, Amherst 2005)
25.83 G. Antonelli: *Underwater Robots*, Springer Tracts in Advanced Robotics, Vol. 96, 3rd edn. (Springer, Berlin, Heidelberg 2014)
25.84 M. Carreras, J. Yuh, J. Batlle, P. Ridao: A behavior-based scheme using reinforcement learning for autonomous underwater vehicles, IEEE J. Ocean. Eng. **30**(2), 416–427 (2005)
25.85 M. Carreras, J. Yuh, J. Batlle, P. Ridao: Application of SONQL for real-time learning of robot behaviors, Robots Auton. Syst. **55**(8), 628–642 (2007)
25.86 S. Zhao, J. Yuh: Experimental study on advanced underwater robot control, IEEE Trans. Robotics **21**(4), 695–703 (2005)
25.87 A. Hanai, H.-T. Choi, S.K. Choi, J. Yuh: Experimental study on fine motion control of underwater robots, Adv. Robotics **18**(10), 963–978 (2004)
25.88 T.W. Kim, J. Yuh: Application of on-line neuro-fuzzy controller to AUVs, Inf. Sci. **145**(1), 169–182 (2002)
25.89 C.S.G. Lee, J.-S. Wang, J. Yuh: Self-adaptive neuro-fuzzy systems for autonomous underwater vehicle control, Adv. Robotics **15**(5), 589–608 (2001)
25.90 J. Yuh, J. Nie: Application of non-regressor-based adaptive control to underwater robots: Experiment, Comput. Electr. Eng. **26**(2), 169–179 (2000)
25.91 K.C. Yang, J. Yuh, S.K. Choi: Fault-tolerant system design of an autonomous underwater vehicle ODIN: An experimental study, Int. J. Syst. Sci. **30**(9), 1011–1019 (1999)
25.92 H.H. Wang, S.M. Rock, M.J. Lees: Experiments in automatic retrieval of underwater objects with an AUV, Proc. MTS/IEEE Ocean., Vol. 1 (1995) pp. 366–373
25.93 J. Evans, P. Redmond, C. Plakas, K. Hamilton, D. Lane: Autonomous docking for intervention-AUVs using sonar and video-based real-time 3D pose estimation, Proc. MTS/IEEE Ocean., Vol. 4 (2003) pp. 2201–2210

25.94 G. Marani, S.K. Choi, J. Yuh: Underwater autonomous manipulation for intervention missions AUVs, Ocean Eng. **36**(1), 15–23 (2009)
25.95 G. Marani, J. Yuh, S.K. Choi: Autonomous manipulation for an intervention AUV. In: *Advances in Unmanned Marine Vehicles*, IEE Control Engineering Series, ed. by B. Sutton, G. Roberts (Institution of Engineering and Technology, London 2006) pp. 217–237
25.96 G. Marani, S.K. Choi, J. Yuh: Real-time center of buoyancy identification for optimal hovering in autonomous underwater intervention, Intell. Serv. Robotics **3**(3), 175–182 (2010)
25.97 D. Beciri: SAUVIM robot completed its first fully autonomous mission, http://www.robaid.com/search/SAUVIM (2010)
25.98 P. Sanz, R. Ridao, G. Oliver, P. Casalino, C. Insaurralde, C. Silvestre, C. Melchiorri, A. Turetta: TRIDENT: Recent improvements about autonomous underwater intervention missions, Proc. IFAC Workshop Navig. Guid. Control Underw. Veh. (NGCUV) (2012)
25.99 D. Ribas, N. Palomeras, P. Ridao, M. Carreras, A. Mallios: Girona 500 AUV: From survey to intervention, IEEE/ASME Trans. Mechatron. **17**(1), 46–53 (2012)
25.100 M. Prats, D. Ribas, N. Palomeras, J.C. Garcia, V. Nannen, S. Wirth, J.J. Fernañdez, J.P. Beltrañ, R. Campos, P. Ridao, P.J. Sanz, G. Oliver, M. Carreras, N. Gracias, R. Marín, A. Ortiz: Reconfigurable AUV for intervention missions: Ačase study on underwater object recovery, Intell. Serv. Robotics **5**(1), 19–31 (2012)
25.101 Trident: Marine Robots and Dexterous Manipulation for Enabling Autonomous Underwater Multipurpose Intervention Missions: Newsletter October 2012, http://www.irs.uji.es/trident/files/2nd-TRIDENT-SCHOOL-Newsletter-Oct2012.pdf (2012)

Multimedia Contents

26. Flying Robots

Stefan Leutenegger, Christoph Hürzeler, Amanda K. Stowers, Kostas Alexis, Markus W. Achtelik, David Lentink, Paul Y. Oh, Roland Siegwart

Unmanned aircraft systems (UASs) have drawn increasing attention recently, owing to advancements in related research, technology, and applications. While having been deployed successfully in military scenarios for decades, civil use cases have lately been tackled by the robotics research community.

This chapter overviews the core elements of this highly interdisciplinary field; the reader is guided through the design process of aerial robots for various applications starting with a qualitative characterization of different types of UAS. Design and modeling are closely related, forming a typically iterative process of drafting and analyzing the related properties. Therefore, we overview aerodynamics and dynamics, as well as their application to fixed-wing, rotary-wing, and flapping-wing UAS, including related analytical tools and practical guidelines. Respecting use-case-specific requirements and core autonomous robot demands, we finally provide guidelines to related system integration challenges.

26.1 **Background and History** 624
26.1.1 A Glimpse of History 624

26.2 **Characteristics of Aerial Robotics** 625
26.2.1 Aerial Robots Classification 625
26.2.2 The Effect of Scale 626

26.3 **Basics of Aerodynamics and Flight Mechanics** 629
26.3.1 Properties of the Atmosphere 630
26.3.2 General Fluid Dynamics and 2-D Flow around Airfoils 630
26.3.3 Wing Aerodynamics 633
26.3.4 Performance of Rotors and Propellers 635
26.3.5 Drag 637
26.3.6 Aircraft Dynamics and Flight Performance Analysis 638

26.4 **Airplane Modeling and Design** 641
26.4.1 Forces and Moments 641
26.4.2 Static Stability 642
26.4.3 Dynamic Model 643
26.4.4 Design Guidelines 644
26.4.5 A Simple Autopilot 646

26.5 **Rotorcraft Modeling and Design** 647
26.5.1 Mechanical Design of Rotors and Propellers 648
26.5.2 Rotorcraft Dynamics 648
26.5.3 Simplified Aerodynamics 649
26.5.4 Nonuniform Inflow 651
26.5.5 Flapping Dynamics 651
26.5.6 Flight Dynamics Assessment 652

26.6 **Flapping Wing Modeling and Design** 653
26.6.1 Aerodynamic Mechanisms 653
26.6.2 Sizing New Flappers 655

26.7 **System Integration and Realization** 659
26.7.1 Challenges for Autonomous UAS 659
26.7.2 Levels of Autonomy 659
26.7.3 UAS Components 660

26.8 **Applications of Aerial Robots** 662
26.8.1 Demonstrated Applications of UAS 662
26.8.2 Current Applications and Missions 664
26.8.3 Aerial Robots: Emerging Categories 664
26.8.4 Open Issues 665

26.9 **Conclusions and Further Reading** 666

Video-References 666

References 667

26.1 Background and History

The field of *aerial robotics* encompasses a very broad class of flying machines that nowadays often possess the perception capabilities and decisional autonomy to accomplish complex tasks without the need for any direct human interventioning. Historically and within the aerospace jargon, robotic flying machines are commonly referred to as *unmanned aerial vehicles* (UAVs), while the entire infrastructures, systems and human–machine interfaces required for autonomous operation are often called *unmanned aerial systems* (UAS). Aerial robotic technologies are currently on the cutting edge of aerospace and robotic research. Breakthrough contributions take place in various fields such as design, estimation [26.1], perception [26.2], control [26.3], and planning [26.4], paving the way for a historical change on how flying systems are operated and what application challenges they fulfill.

As a class of systems, aerial robots have their roots in the first guided missiles; however, nowadays they refer to a wide variety of advanced intelligent systems. According to the American Institute of Aeronautics and Astronautics (AIAA) [26.5], a UAV is defined as

> *an aircraft which is designed or modified, not to carry a human pilot and is operated through electronic input initiated by the flight controller or by an onboard autonomous flight management control system that does not require flight controller intervention.*

As is generally the case in robotics, aerial robots tend to become more and more complex systems as a result of the effort to achieve advanced decision making and planning capabilities based on its on-board perception of the environment and a set of relatively abstract mission goals.

Aerial robots posses the unique capability to gently fly over terrain that other robots struggle to roll or crawl over. The price to be paid is related with the advanced challenges in terms of system design, propulsion, perception, control, and navigation. Autonomous flight requires handling of all six degrees of freedom and advanced cognition capabilities within challenging environments. In that sense, perception and navigation complexity drastically increase, while payload and available power consumption for processing tends to be limited, especially as scale decreases. Essentially, the design of aerial robots requires increased attention and thorough selection, or even combination, of one or more existing or new flying concepts, electronic components and algorithms. The design engineer has to assess specific optimization challenges and trade-offs as important desired goals like decreased weight and modularity typically contradict each other.

26.1.1 A Glimpse of History

Aerial robotics is a field of active research and promising perspectives, yet it already accumulates more than a century of developments. Figure 26.1 depicts some historical as well as recent examples of UAVs in the military and civilian sector. Starting as conceptual designs in the context of the human efforts to develop flying machines, aerial robots soon proved their extensive potential and have already created their own legacy. As was also the case for manned aviation, aerial robotic technologies accelerated within the framework of the 20th century world conflicts. Within World War I, Hewitt–Sperry developed an automatic plane that acted as a flying torpedo, carrying onboard intelligence to autonomously sustain flight over long periods of time. This page-turning success was achieved through the integration of (Sperry's self-made) gyroscopes which were then mechanically connected to the control surfaces and therefore established the necessary feedback control loop. During World War II, the German armed forces deployed one of the first successful cruise missiles, the V–1. Despite the fact that V–1 had limited success rate it did incorporate most of the elementary components, estimation algorithms and control loops that can allow autonomous navigation and reference tracking. Military applications kept being, and still are, the main driving force of aerial robotics research and the newest developments in the area change and shape the modern warfare. With the introduction of global positioning systems (GPSs), aerial robots managed to achieve the first completely autonomous surveillance missions. As information and intelligence gathering became one of the most important aspects of the world's open or silent conflicts, military research around the 1970s led to systems equipped with cameras and other sensory systems, giving birth to the UAV prototype the way we know it today. However, civil applications are currently emerging at a very fast pace and the majority of market predictions converge to the conclusion that this area will take dominant characteristics, and most importantly, will become an equally important – if not more – innovation drive.

Within this framework, the advancements in the field of microprocessors, miniaturized sensing, as well as actuator efficiency and downscaling greatly accelerated the field of aerial robots and paved the way for the great achievements we observe today. Aerial robots have advanced to a state in which sophisticated sensor

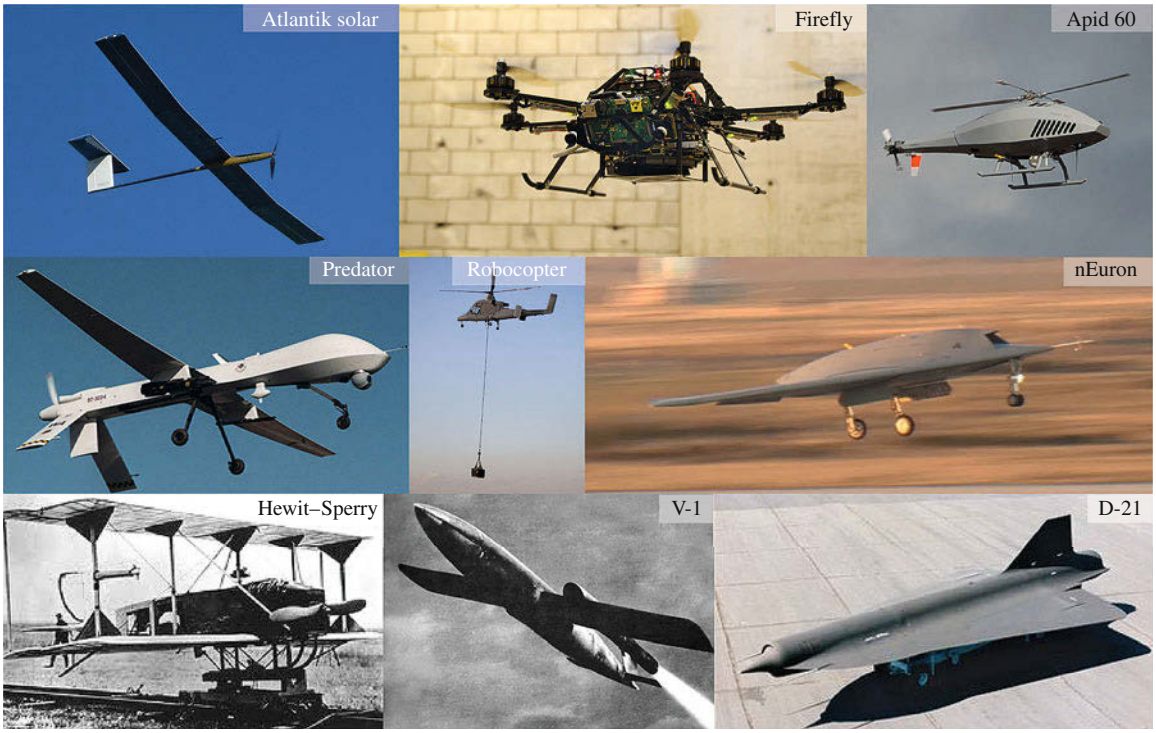


Fig. 26.1 A glimpse on the UAS history through some examples starting from the Hewitt–Sperry Automatic Airplane (1917), the V-1 flying bomb (1944), and the Lockheed D-21 (1962) until the recent examples of military (Predator, Robocopter, nEuron) and civilian (AtlantikSolar, Firefly, Apid 60) aerial robots

modules for onboard state estimation and environmental perception, powerful embedded processors running sophisticated navigation algorithms, potentially several communication interfaces, as well as high-end-mission-oriented payloads that enable the execution of challenging tasks, can be tightly integrated.

26.2 Characteristics of Aerial Robotics

This section aims to provide an overview of the key characteristic features and properties of different aerial robotic configurations as well as a classification based on the key advantages and limitations of some of the most common flying concepts found in unmanned aviation.

26.2.1 Aerial Robots Classification

Compared to the categorization of manned aviation, aerial robots classification is more complex, as the term currently refers to a very wide variety of systems of different scale, mechanical configuration, and actuation principles. In their vast majority, aerial robots correspond, in one way or another, to miniaturized versions of manned aircraft designs. Relatively classical fixed-wing unmanned aerial systems (FW-UAS) designs and rotary-wing unmanned aerial systems (RW-UAS) such as those shown in Fig. 26.2 are common vehicle configurations one may encounter in most applications, including those of surveillance, monitoring, inspection, mapping, or payload transportation. However, even within these relatively traditional concepts, several design aspects differ from those chosen for manned systems. This reflects the fact that for different scales, the variation of the physical properties behavior, along with the search for optimized designs, will naturally lead to modified and novel design considerations. This is further triggered by the fact that the absence of a pilot on-board unlocks a wide set of engineering choices, typically out of question or even forbidden in manned aviation. As expected for a multitude of engineering reasons, large UAS tend to follow design concepts closer to – while at smaller scale to classi-

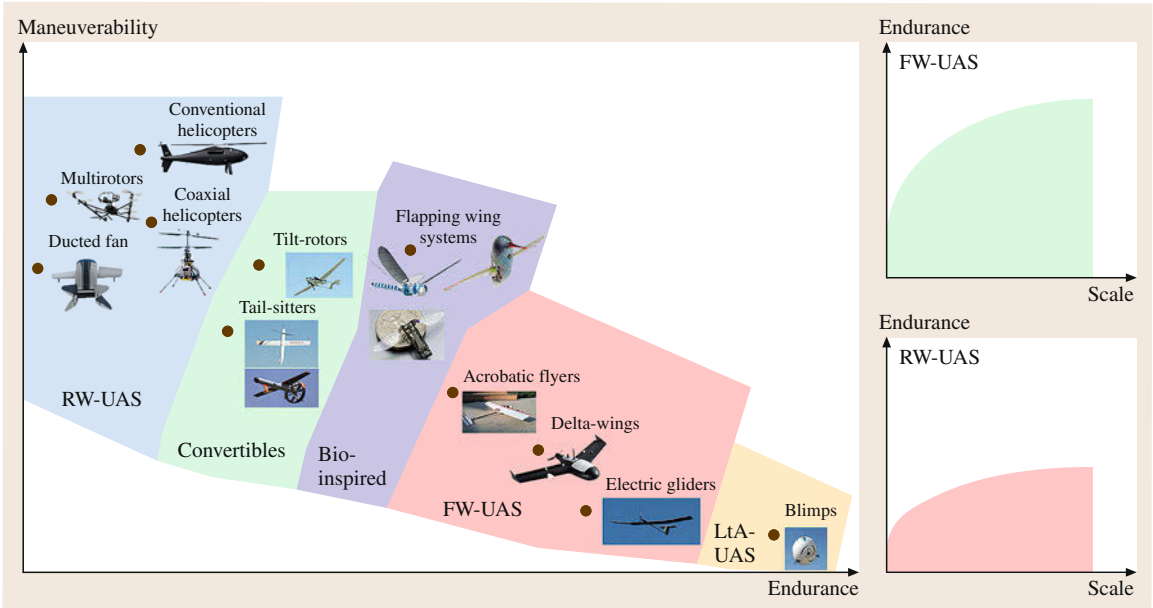


Fig. 26.2 Classification of aerial robotics based on their endurance and maneuverability properties. Also note the significant effect of scale which highlights that comparisons should be done on similar scales

cal designs while as scale decreases innovation – at the level of the flying principle – becomes more and more intense.

Apart from *lighter-than-air* systems (LtA-UAS), FW-UAS tend to be the most power efficient flying principle, while RW-UAS are tailored to increased maneuverability as well as the ability of stationary vertical flight (*hovering*). This general classification (also valid for manned aviation) is then further complicated with the relatively large class of convertible designs (such as tilt-rotors or cruise-flight-enabled ducted fans). This first attempt for aerial robots classification has then to be further augmented to account for the biologically inspired concepts, and especially the emerging field of flapping-wing UAS (FI-UAS). Figure 26.2 provides an abstract – yet incomplete – overview of the vehicle classes one may encounter in most of the application fields. As shown, a large diversity is observed as a result of the engineering efforts to propose designs with optimized endurance, agility, controllability, or even simplicity in a very wide scale range. In the following subsections, a brief overview on how the main aerodynamic forces and effects depend on the design scale of an aerial vehicle are provided.

26.2.2 The Effect of Scale

The understanding of how aerial vehicles manage to remain airborne, provides a useful insight into the effect of scale, and how different dimensioning has a huge impact on the efficiency of every flying machine. Table 26.1 provides an overview of the formulas expressing the lift force, as well as the drag forces that govern the flight of the most common UAS configurations. More detailed definitions on the aerodynamic forces can be found on the subsequent sections.

Within these equations, ρ is the density of the air while the remaining parameters are specific to the vehicle configuration. For FW-UAS, c_L and c_D represent the wing lift and drag coefficients, respectively, A is the wing area, and V_t denotes the airspeed. For the case of RW-UAS, c_T and c_Q denote the rotor thrust and drag coefficients, (πR^2) is the rotor disk area, Ω is the angular velocity of the rotor, and R is the rotor disk radius. Finally, for LtA-UAS, V^{LtA} is the volume of the blimp, c_D^{LtA} is the drag coefficient depending on the blimp shape, V_t is the blimp's airspeed, A^{LtA} is the blimp surface in the direction of motion and ρ_{gas} is the filling gas density. Figure 26.3 illustrates

Table 26.1 Formulas of the main aerodynamic forces and moments for common UAS configurations. FW stands for *fixed-wing* UAS, RW for *rotary-wing*, and LtA for *lighter-than-air*

UAS	Lift/Thrust	Drag force/Moment
FW	$L = \frac{1}{2} c_L \rho A V_t^2$	$D = \frac{1}{2} c_D \rho A V_t^2$
RW	$T = c_T \rho (\pi R^2)(R\Omega)^2$	$Q = c_Q \rho (\pi R^2)(R\Omega)^2 R$
LtA	$L_s = -g V^{LtA}(\rho_{gas} - \rho)$	$D = \frac{1}{2} c_D^{LtA} \rho A^{LtA} V_t^2$

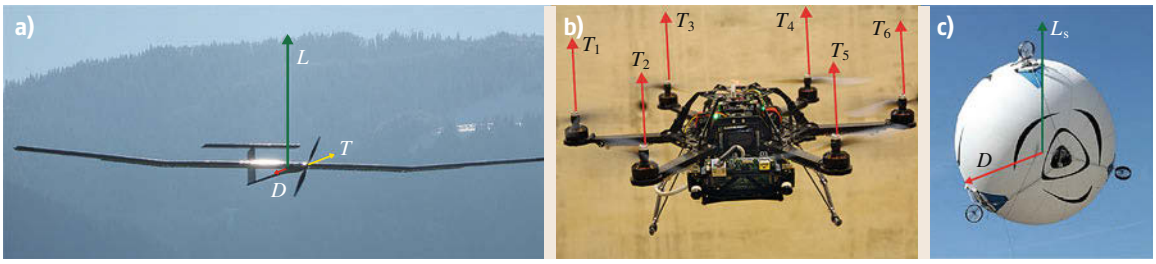


Fig.26.3a–c Main aerodynamic forces applied on **(a)** fixed-wing, **(b)** rotary-wing, and **(c)** lighter-than-air systems. **(a)** AtlantikSolar is a solar-powered FW-UAS developed by the Autonomous Systems Lab at ETH Zurich, **(b)** Firefly is developed by Ascending Technologies GMBH, while **(c)** Skye is developed by students of ETH Zurich

these forces on the body of the relevant aerial vehicle configurations.

Derivation of *scaling laws* starts with the observation of the lift and drag forces and how these are functions of scale-dependent parameters such as the area of the wing or the rotor radius. Proper dimensioning is essentially a very complex procedure where a multitude of factors has to be taken into account. Among others, one has to account for the issues of aerodynamics efficiency, availability of propulsion systems at a given scale, the technologies they employ (e.g., electric motors, jet engines) as well as the simplicity and robustness of the corresponding mechanical configuration. In the following, scaling laws and relevant design guidelines for fixed-wing, rotary-wing, and lighter-than-air systems are provided. Only a brief overview is provided for the case of flapping-wing systems, as the effect of scale on such UAS configurations is separately discussed within Sect. 26.6.

FW-UAS

Scaling laws express the dominant role of size and scale for a given vehicle configuration. In the case of fixed-wing systems, the wing loading, defined as the ratio of the weight (W) versus the wing area A, is the key parameter one has to focus to get some first insight on the role of scale. The Tennekes diagram shown in Fig. 26.4 provides a visual interpretation of this fact [26.6]. Working around the point that the lift force exactly counteracts the weight, the indicated trend line was derived using the following formulas [26.7]

$$\begin{aligned} \frac{W}{A} &= \sqrt[3]{W}47\,, \\ \frac{W}{A} &= \frac{1}{2}c_{\mathrm{L}}\rho V_{\mathrm{t}}^2\,, \\ A &= b_{\mathrm{w}}c_{\mathrm{w}}\,, \end{aligned} \tag{26.1}$$

where V_{t} is the airspeed, W is the weight, A is the wing area, b_{w} is the wing span, and c_{w} is the wing chord. These equations express the role of the lifting properties of the airfoil and airspeed against the ratio of the weight of the flying body and its wing area. For this analysis, a fixed aspect ratio ($\Lambda = b_{\mathrm{w}}/c_{\mathrm{w}}$) is assumed for all sizes of aircraft. Although such a simple analysis does not account for the details of the fluid dynamics environment between the different aircraft sizes, it is known that smaller aircrafts are typically built with lower aspect ratios, and that the difference in aspect ratio over existing aircraft within the size range of interest is significant.

RW-UAS

For the case of rotorcraft configurations, similar scaling laws regarding the vehicle efficiency may be derived. It is important to highlight however that especially for rotorcrafts, working with scaling laws demands that one has to simultaneously focus on both efficiency and dynamic response in order to avoid undesired effects in the vehicle flight dynamics such as unstable oscillations. Regarding the power efficiency, let power loading (PL) be defined as T/P, where P corresponds to the ideal power. As the induced ideal power to hover is given by $P = T\upsilon_{\mathrm{h}}$, the *ideal power loading* will be inversely proportional to the induced velocity at the rotor disk υ_{i}

$$\upsilon_{\mathrm{h}} = \upsilon_{\mathrm{i}} \Rightarrow \sqrt{\frac{T}{2\rho(\pi R^2)}} = \frac{P}{T} = (PL)^{-1}\,. \tag{26.2}$$

Observing Fig. 26.5, it is shown that the ratio T/P decreases quickly with increasing disk loading. Therefore, configurations with proportionally smaller rotors against their mass will tend to be less efficient in hovering flight; that is, the rotor will require proportionally more power to generate the required amount of thrust. It is also to be noted, however, that calculation of the actual power loading and rotor efficiency requires the consideration of viscous losses.

From the above brief analysis, we concluded that in general the tendency to increase the rotor dimension favors efficiency. However, this is not the only scaling law one has to consider. Rotorcrafts are particularly com-

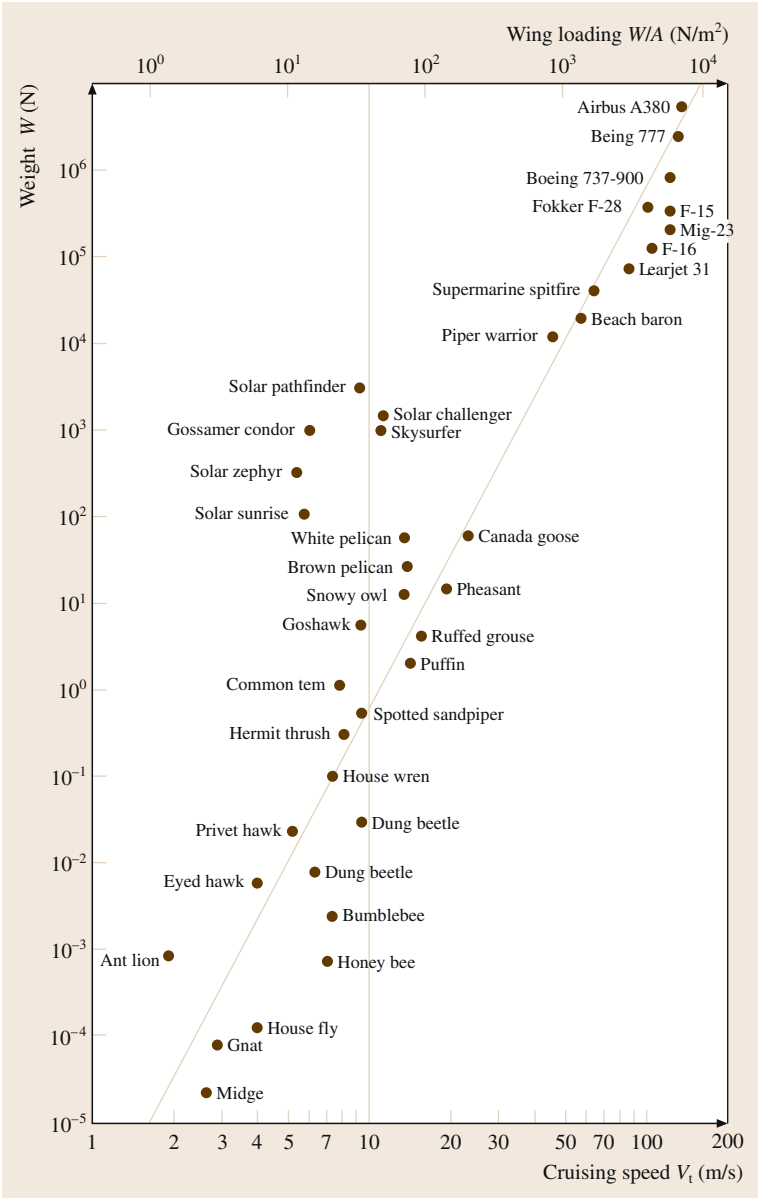


Fig. 26.4 Tennekes size trend relating wing loading and cruise speed to weight for insects, birds, and manned aircraft

plex dynamic systems and scaling considerations also have to focus on dynamic aspects of their flight. A more concrete analysis may take place using Froude or Mach scaling models. Let N denote the length scale between two vehicles, R_m the rotor radius of the model vehicle, and R_p the rotor radius of the prototype vehicle: thus, a scale factor N denotes a helicopter $1/N$ times the size of its prototype. Table 26.2 summarizes the Froude and Mach scaling laws that account for the role of scale in a set of significant parameters namely the length of the model and the prototype L_m, L_p, the dominant time constants t_m, t_p of the inner-loop characteristic response, the characteristic velocities V_m, V_p, the weight values W_m, W_p, the expected moments of inertia I_m, I_p, and the response-dominant frequencies ω_m, ω_p.

These, slightly more advanced scaling laws, further provide the opportunity to assess the aspects of main rotor performance, and more specifically the expected thrust margin. Traditional manned helicopters have small thrust margins in hover, typically 5–10% while miniaturized vehicles often present very high values. Mach models predict in general faster rotor speeds as compared to Froude scaled models, which consequently leads to a lower expected thrust coefficient. The

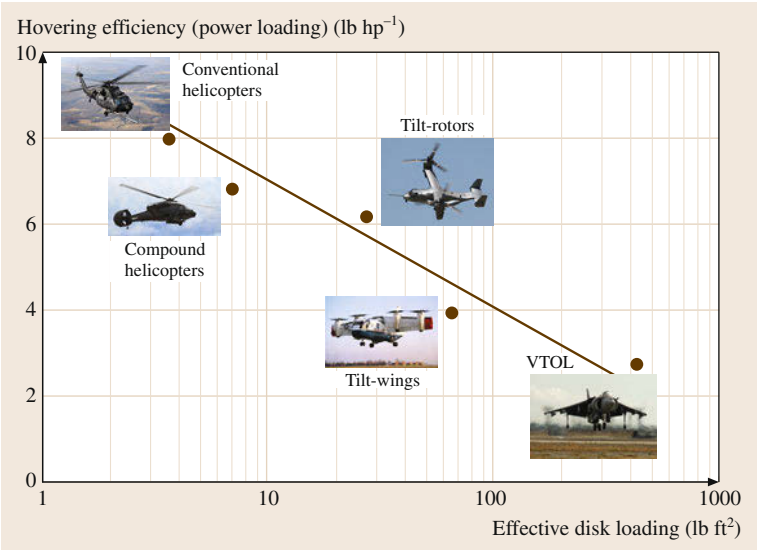


Fig. 26.5 Hovering efficiency versus disk loading for a range of vertical lift aircraft

Table 26.2 Scaling laws for conventional helicopters

Dimension	Froude	Mach
Length	$L_m = L_p/N$	$L_m = L_p/N$
Time constant	$t_m = t_p/\sqrt{N}$	$t_m = t_p/N$
Speed	$V_m = V_p/\sqrt{N}$	$V_m = V_p$
Weight	$W_m = W_p/N^3$	$W_m = W_p/N^3$
Inertia mat.	$I_m = I_p/N^5$	$I_m = I_p/N^5$
Frequency	$\omega_m = \omega_p\sqrt{N}$	$\omega_m = \omega_p N$

thrust coefficient reflects the lift loading of the rotor. For a given, single rotor configuration, the maximum thrust is provided by the following expression

$$T_{max} = (c_T/\sigma)_{max}\rho(\pi R^2)(\Omega R)^2 \,, \qquad (26.3)$$

where σ represents the blade solidity (Sect. 26.3.4). This relation gives a maximum thrust that scales as $T_{max} \propto 1/N^3$ for a Froude model and as $T_{max} \propto 1/N^2$ for a Mach model. Once divided by the vehicle weight which scales as $W \propto 1/N^3$, it is deduced that Froude models present a similar thrust-to-weight ratio. On the contrary, for a Mach model, there is an increasing expected maximum thrust-to-weight: $(T/W)_{max} \propto N$. Using these formulas, researchers in [26.8] calculated the scaling parameters for several conventional helicopters that provide intuitive insight on how scaling laws work.

LtA-UAS

For the case of lighter-than-air vehicles simple scaling laws regarding the efficiency of the system hold. Considering the example of a spherical blimp, it is directly deduced that the lift force scales with the cubic power of the radius. On the other hand, its mass, which depends on the surface, scales with the square power of the radius and also does the drag force. This essentially indicates that larger blimps will tend to have a higher maximum lift to weight and lift-against-drag force ratios.

FI-UAS

Analysis of the scaling laws for flapping wing systems requires a different treatment, as the flight modality changes while the robot operates in *hover* mode or navigates in *forward flight*. Furthermore, the lift and drag coefficients are dependent on the airfoil characteristics of the wing and also on the flapping frequency, a fact that further increases the complexity of the analysis on the effect of scale. Section 26.6 provides insight on how to deal with this challenging issue so that proper flapping wing systems design is achieved.

26.3 Basics of Aerodynamics and Flight Mechanics

Assembling an analytic representation of a UAS involves the derivation of approximative expressions for the aerodynamic forces, accounting for the actuator dynamics and appending the resulting effect to the vehicle body equations of motion. The goal of this section is to provide the necessary insight and understanding of

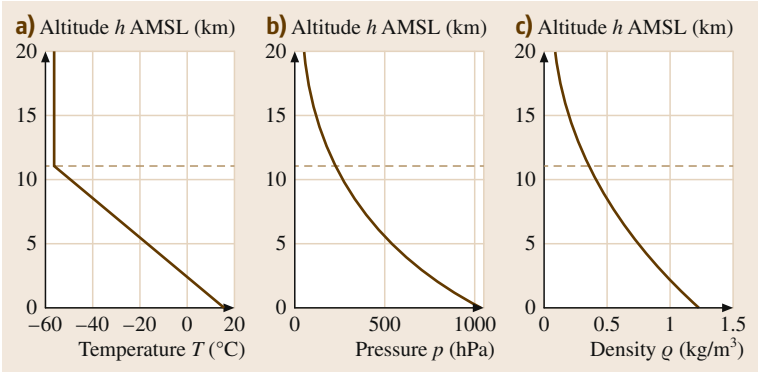


Fig. 26.6 Variation of **(a)** air temperature, **(b)** pressure, and **(c)** density with altitude in the lower part of the International Standard Atmosphere.The tropopause, above which the temperature is not further decreasing, corresponds to the red dashed line. It constitutes the upper limit of common weather phenomena

the underlying mechanisms and physical phenomena along with the derivation of the formulae for the most dominant effects one has to account for with any UAS configuration.

26.3.1 Properties of the Atmosphere

Assessing flow properties forms the basis for any further qualitative or quantitative aerodynamic analysis relevant for aircraft design, modeling and control. The international standard atmosphere (ISA) [26.9] provides a reference for the average main air characteristics as a function of altitude. Figure 26.6 shows the evolution of air temperature T_{air}, pressure p, and density ρ. These parameters largely affect the Reynolds number Re, which can be interpreted as the influence of inertial forces as compared to viscous forces of a flow, as well as the Mach number Ma representing the ratio of airspeed versus speed of sound.

It is noteworthy that the above parameters may be brought into relationship by the ideal gas law

$$p = \rho R T_{\text{air}} \,, \tag{26.4}$$

with $R = 286.97\,\text{m}^2/\text{s}^2/°\text{K}$ denoting the ideal gas constant of air.

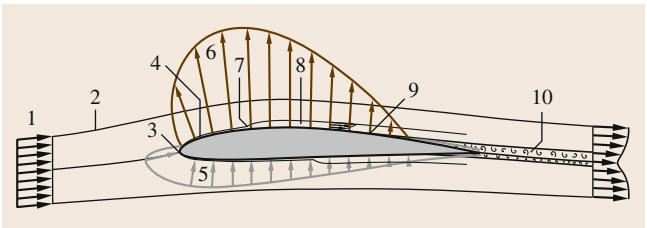


Fig. 26.7 Characteristics of a 2-D flow around an airfoil 1 – Free stream velocity field, 2 – Streamline, 3 – Stagnation point, 4 – yLaminar boundary layer, 5 – Overpressure, 6 – ySuction, 7 – Transition point, 8 – Turbulent boundary layer, 9 – Separation point, 10 – Separated flow

In contrast to *aerodynamics*, where forces, first and foremost lift, is generated by motion of an object through the air, the *aerostatic* lift force is formed solely by static properties of an object. It forms the basis of operation for a balloon or blimp.

According to Archimedes' principle, the aerostatic lift L_{stat} pointing upward, amounts to

$$L_{\text{stat}} = \rho V g - m g \,, \tag{26.5}$$

where g stands for the Earth gravitational acceleration, V for the volume of the object, and m for its mass. To state an example, consider a helium balloon of spherical shape with a diameter amounting to 1 m in the lower atmosphere. Neglecting its hull weight (thus with $m = \rho_{\text{helium}} V$), it will generate an aerostatic lift force of 5.4 N, representing an upper bound for any kind of total design mass including payload. Increasing the diameter of said sphere to 1.5 m has the huge effect of increasing lift to 18.1 N.

26.3.2 General Fluid Dynamics and 2-D Flow around Airfoils

A general airflow around an aircraft is three-dimensional, unsteady, and may be turbulent, even interacting with a nonrigid structure. In this setting, computations are almost intractable. We will thus rely on certain simplifications in order to allow simpler calculations and notably enhancing the understanding of a flow field together with resulting forces and moments. In both airplane and rotorcraft aerodynamics, the assumption of a locally two-dimensional (2-D) flow can be very helpful to serve as a starting point for more advanced computation.

Before diving into a formal treatment, the example in Fig. 26.7 depicts some characteristic elements of 2-D flow around an airfoil.

Notice that the pressure distribution on the airfoil contour is induced by the flow field, making the most

significant contribution to the aerodynamic force and moment. However, also viscous effects yield a typically unwanted share in the overall force (and moment) in the form of shear stress transmitted to the surface. The elements overviewed in Fig. 26.7 will now be explained in the following paragraphs.

Finite Control Volume Analysis: Mass and Momentum Conservation

Consider a finite control volume B bounded by the surface S with normal $\boldsymbol{n}$, which may contain a body or airfoil, depending on the context. For convenience, parts of the boundaries are often chosen to be streamlines (in 2-D) or stream surfaces (in 3-D). For this volume, the *conservation of mass* must be fulfilled

$$\iint_S \rho \boldsymbol{v} \cdot \boldsymbol{n} \, \mathrm{d}S \,, \tag{26.6}$$

where $\boldsymbol{n}$ denotes the fluid velocity vector.

From classical Newtonian mechanics, we can furthermore postulate the applicability of *conservation of linear and angular momentum*

$$\boldsymbol{F}_{\mathrm{tot}} = \iint_S \rho \boldsymbol{v} (\boldsymbol{v} \cdot \boldsymbol{n}) \, \mathrm{d}S + \iint_S p \boldsymbol{n} \, \mathrm{d}S + \frac{\partial}{\partial t} \iiint_B \rho \boldsymbol{v} \, \mathrm{d}B \,, \tag{26.7}$$

$$\boldsymbol{M}_{\mathrm{tot}} = \iint_S \rho (\boldsymbol{v} \times \boldsymbol{r}) (\boldsymbol{v} \cdot \boldsymbol{n}) \, \mathrm{d}S + \frac{\partial}{\partial t} \iiint_B \rho (\boldsymbol{v} \times \boldsymbol{r}) \, \mathrm{d}B \,, \tag{26.8}$$

where $\boldsymbol{r}$ denotes the position vector.

Differential Volume Analysis: Euler and Bernoulli Equations

When applied to a differential volume and assuming inviscid flow, (26.7) can be used to derive *Euler's equation*

$$\rho \left(\frac{\partial}{\partial t} + \boldsymbol{v} \cdot \nabla \right) \boldsymbol{v} + \nabla p = \boldsymbol{0} \,. \tag{26.9}$$

This equation forms the basis of many finite-element-based numerical tools neglecting viscous effects outside the boundary layer. Such methods employ potential flow theory along with some boundary layer analysis module; free example tools are JavaFoil [26.10] and xfoil [26.11] for 2-D flow computation as well as XFLR [26.12] allowing also 3-D flow extensions.

When applying (26.9) along a streamline and under the assumption that the flow is incompressible (a fair assumption for low-speed aerodynamics up to Ma = 0.3), *Bernoulli's equation* relating speed (V_{t}) and pressure can be formulated

$$\rho \frac{V_{\mathrm{t}}^2}{2} + \rho g h + p = \mathrm{const} \,, \tag{26.10}$$

where g denotes gravitational acceleration and h the elevation – the aerostatic pressure component $\rho g h$ can often be neglected due to small elevation changes along a streamline.

Viscous Effects and the Boundary Layer

While it is often a valid approximation to neglect viscous effects far enough from the body surfaces, they have to be considered within the boundary layer, where the fluid is slowed down to meet the speed of the surface. The friction shear stress τ_{w} transmitted to the surface is characterized by the gradient of the flow speed perpendicular to the surface

$$\tau_{\mathrm{w}} = \mu \frac{\mathrm{d}U}{\mathrm{d}n} \,, \tag{26.11}$$

where μ denotes the dynamic viscosity of the fluid, U stands for the airspeed parallel to the surface, and n for the coordinate along the surface normal $\boldsymbol{n}$. This tangential fluid velocity gradient in the boundary layer is visualized qualitatively in Fig. 26.7.

The boundary layer will be laminar around the nose, with the fluid moving parallel to the surface. At some point (at a critical local Reynolds number), however, influenced by disturbances such as surface roughness, a transition to a turbulent boundary layer will occur: it is characterized by stochastic fluctuations, significantly thicker and producing substantially more friction than before the transition.

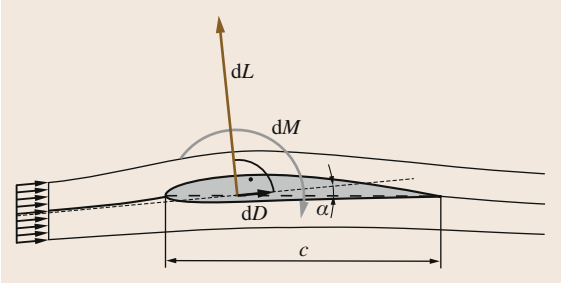


Fig. 26.8 Decomposition of the aerodynamic force by a 2-D flow around an airfoil: the section lift dL denotes the component perpendicular to the far-field inflow, and drag dD the one parallel to it

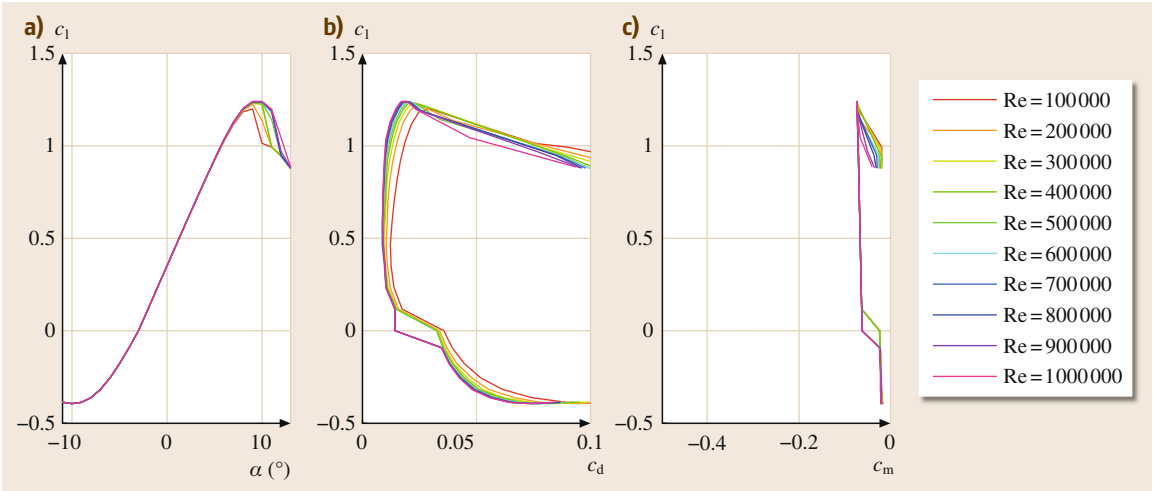


Fig. 26.9a–c SA7036 low-speed airfoil lift (**a**), drag (**b**), and moment (**c**) polars for various Reynolds numbers calculated by Javafoil

Section Lift, Drag, and Moment Representation with Dimensionless Coefficients

Historically and for practical reasons, the aerodynamic force is split into a component perpendicular to the inflow direction called *lift*, and a second one parallel to the inflow called *drag*. We write 2-D lift, drag, and moment as infinitesimal quantities $\mathrm{d}L$, $\mathrm{d}D$, and $\mathrm{d}M$, respectively, as opposed to L, D, and M designating physical forces of a whole airplane. Figure 26.8 visualizes these quantities. Furthermore, we define the angle of attack α as the angle between inflow direction and the chord line of length c connecting airfoil leading edge and trailing edge. Note that force and moment are reduced to the point at $0.25c$, i. e., one quarter of the chord behind the leading edge.

Dimension analysis suggests the formulation of aerodynamic forces and moments in terms of section lift, drag, and moment coefficients c_l, c_d, and c_m

$$\mathrm{d}L = \frac{1}{2}\rho V_\mathrm{t}^2 c_\mathrm{l} c\,\mathrm{d}y\,, \tag{26.12}$$

$$\mathrm{d}D = \frac{1}{2}\rho V_\mathrm{t}^2 c_\mathrm{d} c\,\mathrm{d}y\,, \tag{26.13}$$

$$\mathrm{d}M = \frac{1}{2}\rho V_\mathrm{t}^2 c_\mathrm{m} c^2\,\mathrm{d}y\,, \tag{26.14}$$

where V_t stands for the inflow speed and $\mathrm{d}y$ denotes an infinitesimal length element perpendicular to the 2-D flow (which can be interpreted as a length element into span-wise direction of an infinitely long wing).

These coefficients largely depend on the angle of attack α; but furthermore, the Reynolds and Mach numbers significantly influence them as well. The angle of attack dependences are typically given in the form of section lift, drag and moment polars, an example of which is provided in Fig. 26.9. Note that the drag component is originating both from viscous skin friction as well as form drag, caused by an asymmetric pressure distribution due to boundary layer development and separation. The lift curve shows its characteristic linear increase with increasing α for small angles of attack. The maximum and minimum lift values beyond which stall is entered are clearly visible in the lift polar. Note that the aerodynamic performance $c_\mathrm{l}/c_\mathrm{d}$ of the airfoil generally decreases with smaller Reynolds numbers as expected. The choice of reference point at $0.25c$ typically leads to a mostly constant moment coefficient when varying α, c_l, respectively, as can be seen in Fig. 26.9 as well.

Separation and Stall

At the upper side of the airfoil, the fluid is moving from the under-pressure region toward a higher pressure at the trailing edge; the slower moving fluid in the boundary layer will at some point not be able to follow this adverse pressure gradient, leading to flow separation. As the angle of attack is increased, the separation point suddenly moves far toward the leading edge: this condition is referred to as stall, with the catastrophic consequence of significant loss of lift and increase of drag. Figure 26.10 illustrates the changes in the flow and pressure distribution when varying the angle of attack. Note that the maximum lift and stall conditions are highly influenced by the choice of airfoil, Reynolds number, and Mach number.

26.3.3 Wing Aerodynamics

So far the 2-D flow characteristics around airfoils were treated. This will form the basis for the understanding and computation of lift and thrust forces generated on any type of aircraft; the following treatment of a finite wing serves as an important example of how to include three-dimensional (3-D) flow effects.

Recording lift and drag polars for a finite wing rather than just for its airfoil reveals less lift increase per angle of attack increase, less maximum lift, and higher drag at raised angles of attack. These observations are related to the concept of *induced flow* to be treated in the following.

Vortex System of a Wing

As a direct consequence of lift, we observe a downward flow deflection across an airfoil. This is intuitively explained with conservation of linear momentum as stated in (26.7). Assuming an inviscid and incompressible fluid, the flow may be modeled with potential field theory [26.13], where the velocity vector field is defined as the gradient of a scalar function. This concept allows for the insertion of singularities into a free stream, such as sources, sinks, and vortices. Figure 26.11 shows a first approximation using a single vortex – conceptually illustrating the flow characteristics around a simplified wing. The vortex system consists of the bound vortex and tip vortices; note that the vortex will in theory have to be closed to a ring by a starting vortex. In practice, i. e., in the presence of friction, the vortices will of course decay over time. Figure 26.12 illustrates the existence of tip vortices trailing an airplane wing.

The vortices induce a downwash area behind the wing; nevertheless, the trailing vortices will also induce some downward flow at the wing.

Induced Drag

With the simplified concept of the vortices around a wing in mind, we conclude that the wing lift induces downward flow, thus reducing the effective angle of attack when looking at the 2-D-flow of a wing cross-section. Figure 26.13 illustrates this reduction of the angle of attack from α_f (free stream) to α_e (effective) by the induced angle α_i that is caused by the induced flow component w_i. Note that this reduction of angle of attack is typically resulting in a smaller lift. Furthermore, when decomposing the lift into components parallel and perpendicular to the free stream velocity, it becomes apparent that a part dD_i of the lift results parallel to the effective inflow, thus will contribute to the overall drag of the wing. The integral of these components is referred to as induced drag. The actual amount of induced drag largely depends on the wing geometry; a variety of approaches have been employed in order to minimize induced drag, the most popular of which are winglets.

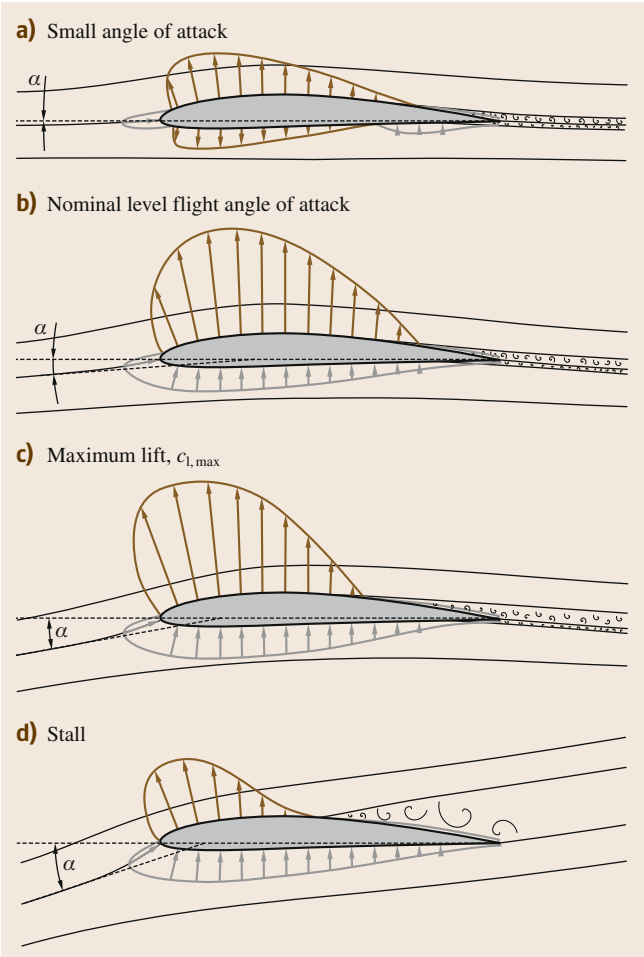


Fig.26.10a–d Changes in the flow characteristics with increasing angle of attack α. For a small α **(a)** the example nonsymmetric airfoil will generate some lift. **(b)** depicts nominal operation. At some α, the maximum lift $c_{l,max}$ is reached **(c)**. Beyond that angle of attack, stall occurs **(d)**

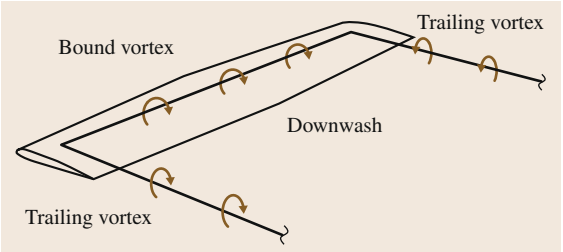


Fig. 26.11 Simplified representation of the wing vortex system: as a consequence of lift, a bound vortex is formed along with trailing wingtip vortices inducing downwash

Fig. 26.12 Wake vortex study by NASA at Wallops Island: the tip vortices are visualized using colored smoke rising from the ground

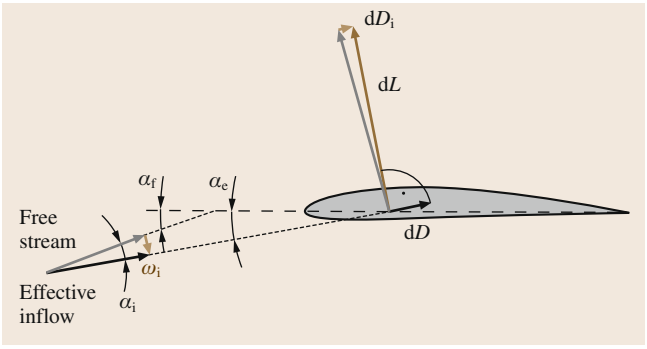


Fig. 26.13 Induced drag on a finite wing cross section: the induced downwash w_i causes a reduced effective angle of attack. As a consequence, the lift $\mathrm{d}L$ contains a component $\mathrm{d}D_\mathrm{i}$ parallel to the free stream velocity vector

For an approximately elliptical lift distribution, the induced drag coefficient can be roughly calculated as

$$c_{\mathrm{D,i}} = \frac{C_\mathrm{L}^2}{\pi e \Lambda} \,, \tag{26.15}$$

with the aspect ratio Λ and the Oswald efficiency e (deviation from the truly elliptic distribution) amounting from 0.7 to 0.85 for typical configuration.

Lifting Line Method

In the following, we present one example of how to numerically approximate the lift and drag distribution of a wing including induced drag. The lifting line method is a 2.5-D (two-and-a-half-dimensional) approach in which the induced flow is viewed as generated by several discrete horseshoe vortices rather than just one as introduced qualitatively earlier. Figure 26.14 depicts the geometry and variables involved. Note that the method only provides reliable results, if the assumption holds that spanwise flow is negligible; in particular, spanwise variation of parameters such as chord length and twist is supposed to be rather small. The Kutta–Joukowsky theorem relates circulation and lift; applied to a discrete wing segment, we obtain

$$\Gamma_k = \frac{1}{2} c_k c_{\mathrm{l},k}(\alpha_\mathrm{eff}) V_\mathrm{t} \,, \tag{26.16}$$

with the segment (index k) circulation Γ_k, the local chord length c_k, and the local airfoil lift coefficient $c_{\mathrm{l},k}$. The lift coefficient depends on the effective angle of attack: $\alpha_\mathrm{e} = \alpha_\mathrm{f} - \alpha_\mathrm{i}$. The induced downwash at position $\boldsymbol{m}_k$ is obtained by adding the induced speeds of all the individual vortices according to Biot–Savart

$$w_{\mathrm{i},k} = \sum_{j=1}^{n+1} \frac{\Gamma_j \boldsymbol{e}_\mathrm{V}}{4\pi \left\| (\boldsymbol{p}_j - \boldsymbol{m}_k) \times \boldsymbol{e}_\mathrm{V} \right\|} \times \left(1 + \frac{(\boldsymbol{p}_j - \boldsymbol{m}_k) \cdot \boldsymbol{e}_\mathrm{V}}{\left\| (\boldsymbol{p}_j - \boldsymbol{m}_k) \right\|} \right) \,, \tag{26.17}$$

where $\boldsymbol{e}_\mathrm{V}$ stands for the (unit) direction of flight. At $\boldsymbol{m}_k$, the induced angle of attack is calculated as

$$\alpha_{\mathrm{i},k} = \arctan \frac{w_{\mathrm{i},k}}{V_\mathrm{t}} \approx \frac{w_{\mathrm{i},k}}{V_\mathrm{t}} \,. \tag{26.18}$$

Together with the respective 2-D polar data, the above relations allow calculating the lift, drag, and moment distribution (with respect to the free inflow direction $\boldsymbol{e}_\mathrm{V}$) from a known circulation distribution, and can be summed and reduced to, e.g., the center of mass of a whole airplane. Note that the section lift coefficient is often approximated linearly as $c_\mathrm{l}(\alpha) \approx c_{\mathrm{l},0} + c_{\mathrm{l}_\alpha} \alpha$, which allows for a direct solution when applying (26.16) and (26.17). More accurate results, in particular in the domain near maximum lift, however, are obtained by using the nonlinear lift polar. In this case, a standard iterative numeric solver may be used. Furthermore, airfoil data with deflected control surfaces can be included, allowing to calculate control moments (and forces).

Note that the above method is just one example of numerically solving for the wing characteristics knowing the 2-D airfoil properties (polars) – well-suited for low-speed medium to high aspect ratio wings. For an overview on alternatives, the reader is referred to, e.g., [26.13].

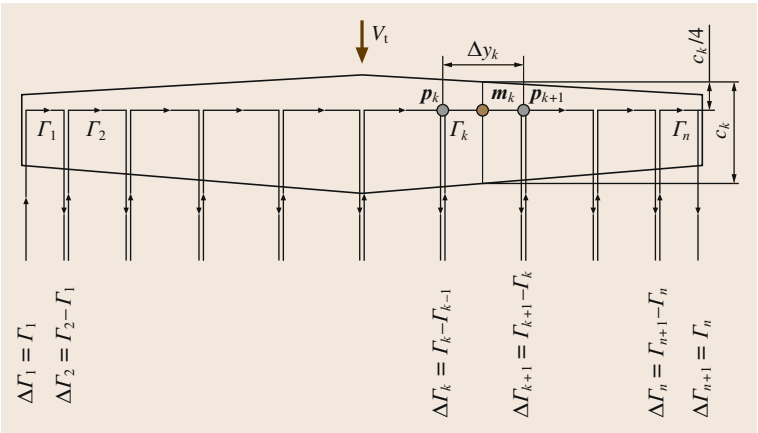


Fig. 26.14 k horse shoe vortices with circulation Γ_k placed on the wing to model the induced flow. The lifting line is imagined through the quarter-chord ($c_k/4$) locations. Vortex threads with strength $\Delta\Gamma_k$ are leaving the wing at the points $\boldsymbol{p}_k$ along the inflow and induce downwash at the locations $\boldsymbol{m}_k$

26.3.4 Performance of Rotors and Propellers

The propulsion mechanism found on many robotic aerial vehicles is commonly a specific configuration of propellers or rotors. In the case of a robotic airplane, forward facing propellers produce thrust forces compensating drag in forward flight. In case of a tail sitter or multicopter UAV, the propellers may be facing up (or down) and produce the main lift component compensating the vehicle's weight allowing it to hover in the air. Similarly, the more classic helicopter-type configurations (single rotor with tail rotor, coaxial rotor, tandem rotor, etc.), use rotors to generate the required thrust force to fly.

In order to decide on a suitable rotor or propeller geometry and to define requirements for the UAV motor drives, models must be available which allow for an assessment of the thrust and torque characteristics of a particular rotor or propeller. For this purpose, the blade element momentum theory (BEMT) has found widespread use, as it often provides a prediction accuracy which is acceptable for the UAV design process (despite its simplicity).

Blade Element Momentum Theory

One of the basic difficulties in aerodynamic rotor and propeller studies is the prediction of the induced inflow velocities discussed in Sect. 26.3.3. BEMT addresses this problem by combining two simple modeling approaches, namely momentum theory (MT) and blade element theory (BET) which individually cannot directly resolve this issue in an accurate manner [26.14].

The basic idea of momentum theory is to consider the revolving propeller or rotor as a propulsion disk which produces a thrust force by accelerating the surrounding (incompressible) air mass passing through it. A boundary volume is defined encapsulating the propulsion disc. Subsequently, the laws of mass, momentum and power conservation are formulated across the boundaries of the defined control volume. The concept of this propulsion disk as well as the corresponding control volume are visualized in Fig. 26.15.

From this simplistic model, two main conclusions may be drawn. First and foremost, it is possible to establish a relation between the induced velocity v_i at the propulsion disk and the produced thrust force T. In normalized form, it can be expressed as

$$c_T = 2\lambda_i(\lambda_i + \lambda_\infty) \,. \tag{26.19}$$

To simplify notation, the external airflow velocity v_∞ as well as the induced velocity v_i have been normalized with the rotor or propeller tip speed ΩR

$$\lambda_i = \frac{v_i}{\Omega R}, \qquad \lambda_\infty = \frac{v_\infty}{\Omega R} \,, \tag{26.20}$$

and the nondimensional thrust coefficient c_T is defined as

$$c_T = \frac{T}{\rho(\pi R^2)(\Omega R)^2} \,. \tag{26.21}$$

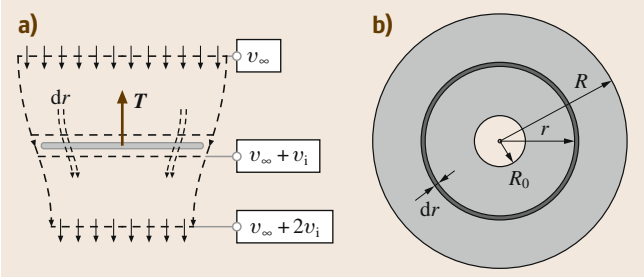


Fig. 26.15 **(a)** Side view on the slipstream control volume encompassing the MT propulsion disc. **(b)** Top view on MT propulsion disk with incremental annular section and root cutout

The parameter ρ corresponds to the density of air, R to the rotor or propeller radius, and Ω to the rotor or propeller angular speed.

In case of a robotic airplane, the velocity v_∞ at the start of the control volume corresponds to the body forward flight velocity V_t and in case of a rotorcraft configuration to the body climb respectively descent rate w.

Similarly, an incremental expression for the thrust coefficient may be found from MT by evaluating the mass, moment, and power conservation laws over an annular ring of the defined control volume only. The corresponding expression can be found as

$$\mathrm{d}C_\mathrm{T}^\mathrm{MT} = 4\lambda_\mathrm{i}(\lambda_\mathrm{i} + \lambda_\infty)\bar{r}\,\mathrm{d}\bar{r}\,, \tag{26.22}$$

where $\bar{r} = \frac{r}{R}$ and $\mathrm{d}\bar{r} = \frac{\mathrm{d}r}{R}$ are the normalized radial location and the radial increment of the propulsion disk annulus.

Another relevant conclusion that may be drawn from MT is that ideally, the induced component v_i of the slipstream velocity at the propulsion disk will accelerate to two times its initial value before leaving the control volume. In consequence to this acceleration of the flow field, the radial slipstream boundary will (in the ideal case) contract to half the propulsion disk area at the end of the control volume.

For the BET approach, the modeling process starts by investigating the aerodynamic lift and drag forces $\mathrm{d}L$ and $\mathrm{d}D$ on an individual rotor or propeller blade revolving around its shaft. These lift and drag forces produced by each airfoil segment depicted in Fig. 26.16, contribute to the total thrust and torque increments $\mathrm{d}T$ and $\mathrm{d}Q$ of the respective rotor or propeller annular section. The corresponding relation can be established as

$$\mathrm{d}T = N_\mathrm{b}(\mathrm{d}L - \Phi\,\mathrm{d}D) \approx N_\mathrm{b}\,\mathrm{d}L\,, \tag{26.23}$$

$$\mathrm{d}Q = N_\mathrm{b} r(\mathrm{d}D + \Phi\,\mathrm{d}L)\,. \tag{26.24}$$

In this context, N_b represents the number of rotor or propeller blades and Φ corresponds to the local inflow angle which is assumed to remain small. Under this assumption, the inflow angle Φ can be directly derived as the ratio between the local perpendicular inflow velocity $U_\mathrm{P} \approx v_\mathrm{i} + v_\infty$ and the tangential velocity $U_\mathrm{T} \approx \Omega r$ visualized in Fig. 26.16b. Additionally, the assumption introduced in (26.23) is justified by the fact that at low angles of attack α, the drag forces $\mathrm{d}D$ are at least one order of magnitude smaller than the corresponding lift forces $\mathrm{d}L$.

Based on (26.23) and the definition of the lift increment (26.12), the local thrust coefficient at each radial blade station r may be derived as

$$\mathrm{d}C_\mathrm{T}^\mathrm{BET} = \frac{1}{2}\sigma c_\mathrm{l}\bar{r}^2\mathrm{d}\bar{r}, \quad \text{with} \quad \sigma = \frac{N_\mathrm{b}c}{\pi R}\,. \tag{26.25}$$

The parameter c corresponds to the local blade chord and σ is the so-called rotor or propeller *solidity*. The solidity is a rough metric representing how much of a propulsion disk is covered by rotor or propeller blades. The aerodynamic parameter $c_\mathrm{l} = c_\mathrm{l}(\alpha, \mathrm{Re}, \mathrm{Ma})$ corresponds to the airfoil lift coefficient in function of the local angle of attack α and the local Reynolds and Mach numbers Re and Ma.

Accordingly, as established in (26.25), the thrust produced by a rotor or propeller strongly depends on the angle of attack α, which itself is a function of the local airfoil pitch angle θ and the inflow angle Φ

$$\alpha = \theta - \Phi \approx \theta - \frac{U_\mathrm{P}}{U_\mathrm{T}}\,. \tag{26.26}$$

In conclusion, MT as well as BET are capable of establishing a meaningful relation between the induced velocities v_i and the resulting thrust force T. However, none of the two theories are capable of accurately predicting rotor or propeller performance as either the radial distribution of the induced velocity or the radial distribution of the thrust coefficient must be known to compute the other.

The basic idea behind BEMT is to combine the thrust expression (26.22) resulting from MT with the thrust expression (26.25) from BET to compute the induced inflow velocity independently of the thrust force. Different BEMT implementations are possible depending on how willing one may be to introduce further assumptions for the section lift coefficient c_l.

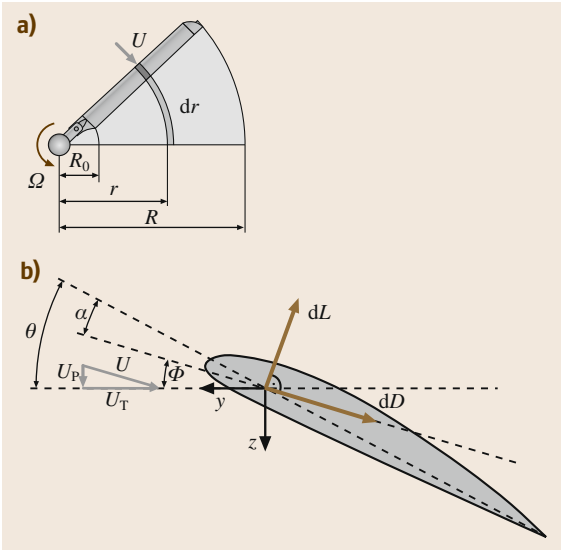


Fig. 26.16 **(a)** Rotor blade revolving around its shaft. **(b)** Blade element of a revolving rotor blade

Reference [26.14] presents a straightforward approach tailored toward helicopter rotors operating bellow stall by introducing a linear model for c_l in function of the angle of attack

$$c_l = c_{l1}\alpha + c_{l0} \,. \tag{26.27}$$

The parameters c_{l0} and c_{l1} can be computed from the lift polars of a particular airfoil geometry for a given range of angles of attack, Reynolds, and Mach numbers. This linear approximation may have limited validity for very low Reynolds numbers and strongly cambered airfoils but is in general acceptable for many typical airfoil geometries found on UAV rotors and propellers. Assembling (26.22) and (26.25) under the assumption (26.27), an algebraic expression of the radial induced inflow distribution can be derived as

$$\lambda_i(r, \lambda_\infty) = \sqrt{A^2 + B} - \overline{A} \,, \tag{26.28}$$

$$A = \frac{\sigma c_{l1}}{16} - \frac{\lambda_\infty}{2} \,,$$
$$\overline{A} = \frac{\sigma c_{l1}}{16} + \frac{\lambda_\infty}{2} \,,$$
$$B = \frac{\sigma c_{l1}}{8}\theta' \overline{r} \,.$$

Note that the lift-curve offset c_{l0} has been absorbed in the virtual pitch angle

$$\theta' = \theta + \frac{c_{l0}}{c_{l1}} \,, \tag{26.29}$$

to simplify the notation.

Once the approximate radial distribution of inflow velocities has been found, the local rotor or propeller thrust increments (26.25) can be computed. Similarly, the rotor or propeller torque increments derived from BET as

$$\mathrm{d}c_Q^{\mathrm{BET}} = \mathrm{d}c_{Qi} + \mathrm{d}c_{Q0} \,, \tag{26.30}$$

$$\mathrm{d}c_{Qi} = \frac{1}{2}\sigma c_l(\lambda_i + \lambda_\infty)\overline{r}^3 \,\mathrm{d}\overline{r} \,, \tag{26.31}$$

$$\mathrm{d}c_{Q0} = \frac{1}{2}\sigma c_d \overline{r}^3 \,\mathrm{d}\overline{r} \,, \tag{26.32}$$

can be evaluated based on the inflow distribution given in (26.28). For clarity, the total torque coefficient increment has been separated into its induced component $\mathrm{d}c_{Qi}$ originating from the lift forces and its profile component $\mathrm{d}c_{Q0}$ due the drag forces. The aerodynamic drag coefficient $c_d = c_d(\alpha, \mathrm{Re}, \mathrm{Ma})$ can be approximated using a quadratic function in dependency of the angle of attack [26.14]

$$c_d = c_{d2}\alpha^2 + c_{d1}\alpha + c_{d0} \,. \tag{26.33}$$

The parameters c_{d0}, c_{d1}, and c_{d2} can be computed from the drag polars of the modeled airfoil.

Consequently, the thrust and torque increments may be integrated along the radial direction of the rotor or propeller disk to compute the total thrust and torque coefficients

$$c_T = \int_0^R \mathrm{d}c_T^{\mathrm{BET}} \,, \tag{26.34}$$

$$c_Q = \int_0^R \mathrm{d}c_Q^{\mathrm{BET}} \,. \tag{26.35}$$

These thrust and torque integrals are usually evaluated numerically, as the blade pitch $\theta = \theta(r)$ as well as the blade chord $c = c(r)$ may be nonlinear functions of the radial direction r (blade *twist* and *taper*).

Finally, note that the presented theory may be extended to provide performance estimates under lateral inflow velocities such as in case of a rotorcraft in forward flight and may also be used to asses other types of rotor or propeller configurations such as, e.g., the coaxial rotor. Also note that the prediction accuracy of BEMT tools can be further improved by accounting for tip-loss effects and the nonlift producing rotor or propeller hub, e.g., using the Prandtl tip-loss function also presented in [26.14] and incorporating a root cutout radius R_0.

The resulting predictions are generally in good agreement with experimental data – nevertheless, an experimental verification is strongly recommended.

26.3.5 Drag

The sources of drag on aircraft are manifold: historically, the distinction between the lift-dependent *induced drag* and *parasite drag* is made. The latter is further subdivided into *skin friction drag* due to viscous shear stress at the surface, and *form drag*, generated by pressure loss along bodies (i. e., due to boundary layer development or even flow separation). Both these components contribute to the airfoil *profile drag*, i. e., the section drag coefficient c_d introduced before.

On a whole aircraft, many more drag sources are distinguished. For an exhaustive overview, the interested reader is referred to [26.13]. In the following, we will present an overview of drag generated by different typical shapes; note that when simply summing drag of different shapes associated with aircraft parts, the result may be a helpful initial estimate, but can be inaccurate, because of neglecting the interaction of flows resulting in *interference drag*. Depending on the stage of the design process or the desired modeling accuracy, 2.5-D

computations or even full 3-D computational fluid dynamics (CFD) simulations might be necessary to satisfy the needs of aerodynamics calculations.

Skin Friction

The simple but important example of a flat plate of length l in parallel flow is well studied. As introduced in the airfoil theory Sect. 26.3.2, the boundary that develops will be laminar near the leading edge and transitions into a turbulent one, generating more drag, at some point downstream. The friction coefficient is defined as

$$c_\mathrm{f} = \frac{2D_\mathrm{f}}{\rho V_\mathrm{t}^2 S_\mathrm{w}} , \tag{26.36}$$

with the wetted surface S_w and the friction drag force D_f. According to [26.13], the coefficients can be approximated by

$$\text{Laminar}: \quad c_\mathrm{f} = 1.328\mathrm{Re}_\mathrm{l}^{-0.5} , \tag{26.37}$$

$$\text{Turbulent}: \quad c_\mathrm{f} = 0.455(\log_{10}\mathrm{Re}_\mathrm{l})^{-2.58} . \tag{26.38}$$

Note that the point of transition is depending on the local Reynolds number $\mathrm{Re}_x = \rho V_\mathrm{t} x/\mu$, where x denotes the coordinate along the flow from the leading edge of the plate. Depending on the surface roughness and ambient turbulence, the critical (transition) Reynolds number varies; as an average guess for a flat plate, it will be in the order of $\mathrm{Re}_{x,\mathrm{crit}} = 3\times10^5$.

Drag Coefficients for Selected Bodies

In the following, drag coefficients for a selection of 2-D and 3-D bodies of rotation are given, obtained from [26.13]. Table 26.3 overviews a category of bodies, the drag coefficients of which are largely indifferent to the Reynolds number, owing to their geometrically defined (sharp edge) flow separation point.

Rounder bodies, most prominently the cylinder in cross-flow or the sphere, however, show a distinctively different behavior quantified in Table 26.4: below a critical Reynolds number $\mathrm{Re}_\mathrm{crit} \approx 4\times10^5$, the drag coefficient is significantly higher, where separation occurs before boundary layer transition. In contrast, above the critical Reynolds number, the turbulent, more energetic boundary layer separates only further downstream, reducing the wake and thus the amount of form drag.

A third important object category is formed by streamlined and fuselage-like bodies: due to their comparably high skin friction part, the fineness ratio largely influences the drag coefficient (along with the Reynolds number). The fineness is defined as body length divided by body diameter. We introduce a volumetric drag coefficient as $c_\mathrm{Dm} = 2D/(\rho V_\mathrm{t}^2 V_\mathrm{m}^{2/3})$ with the body volume V_m. Interestingly, this is minimal and approximately constant at fineness ratios between 4 and 10, which provides a range for optimal sizing of such a body when a certain volume needs to be fitted. The values are given in Table 26.5.

Table 26.3 Bodies the drag coefficients of which are largely Reynolds number independent

	c_D (2-D)		c_D (3-D)	
⟶		1.98		1.18
⟶		2.0		1.0–1.2
⟶		1.3		0.7
⟶		2.0		1.1
⟶		2.2		1.7
⟶		1.4		0.4

26.3.6 Aircraft Dynamics and Flight Performance Analysis

Sections 26.3.1–26.3.5 shortly presented the basic theory of wings, rotors, and propellers as well as a few tools to assess their respective aerodynamic performance. To develop fully functional UAV platforms, merely evaluating the individual flight mechanisms is a good starting point but generally not sufficient.

Designing high-performance aircraft systems requires a fundamental understanding of how the respective design parameters affect the full flight dynamic response and application-specific capabilities. In order to comprehend how design changes affect an aerial

Table 26.4 Sphere and cylinder drag coefficients

	c_D (2-D)	c_D (3-D)
⟶		
$\mathrm{Re} < \mathrm{Re}_\mathrm{crit}$	1.1	0.4
$\mathrm{Re} > \mathrm{Re}_\mathrm{crit}$	0.27	0.15

Table 26.5 Volumetric drag coefficients for fineness 4–10 (largely turbulent boundary layer)

Fuselages and nacelles	$c_\mathrm{Dm} \approx 0.027$
Streamlined bodies	$c_\mathrm{Dm} \approx 0.024$

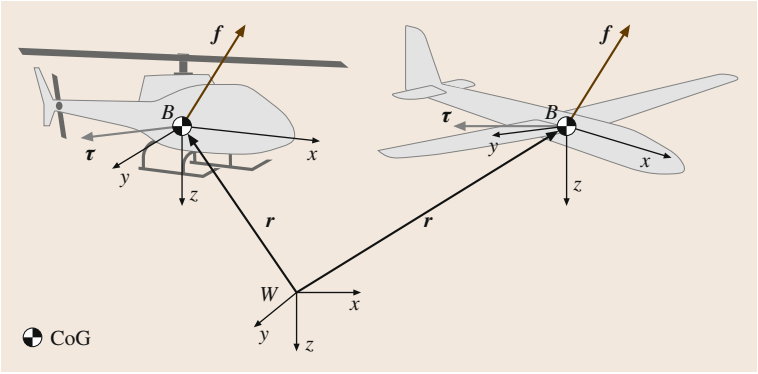


Fig. 26.17 Coordinate frames with external forces and moments of rotorcraft and fixed-wing UAVs

robotic system, representative models of its flight dynamics are required. Such models must be capable of capturing the dominant system dynamics within the relevant part of the flight envelope. Furthermore, especially in an interdisciplinary field such as robotics, these models must be accessible to the nonaerodynamic expert (the roboticist) and thus need to be simple enough to provide the required insight for the aircraft design process.

As many other types of robots, robotic flight platforms may be treated as a multibody system where a set of interlinked bodies exchanges kinetic and potential energy under the influence of external forces and moments. For aircraft systems it is common to treat the entire aircraft as a single rigid body first, with related body coordinate frame attached, as visualized in Fig. 26.17. Additional dynamics such as for example rotor flapping (as in case of a helicopter system) may be appended to these body dynamics in a subsequent step.

The modeling process thus starts by treating the aircraft system as a rigid body affected by external forces $\boldsymbol{F}$ and external moments $\boldsymbol{\tau}$. Using the Newton–Euler formalism to derive the aircraft body dynamics, one can directly write down the linear and angular momentum balance for a single rigid body

$$\begin{aligned} m({}^B\dot{\boldsymbol{v}} + {}^B\boldsymbol{\omega} \times {}^B\boldsymbol{v}) &= {}^B\boldsymbol{F} \,, \\ {}^B\mathbf{I}\,{}^B\dot{\boldsymbol{\omega}} + {}^B\boldsymbol{\omega} \times ({}^B\mathbf{I}\,{}^B\boldsymbol{\omega}) &= {}^B\boldsymbol{\tau} \,. \end{aligned} \tag{26.39}$$

For simplicity's sake, (26.39) is usually expressed with respect to a body fixed frame B located in the center of gravity of the aircraft. The velocity vectors ${}^B\boldsymbol{v} = (u, v, w)^{\mathrm{T}}$ and ${}^B\boldsymbol{\omega} = (p, q, r)^{\mathrm{T}}$ thus represent the aircraft linear and angular velocities with respect to B. The inertial properties of the above body dynamics are defined by the aircraft's total mass m and its second mass moment of inertia ${}^B\mathbf{I}$ also expressed with respect to B and its origin.

The most relevant contribution to the forces ${}^B\boldsymbol{F}$ and the moments ${}^B\boldsymbol{\tau}$ originates from the aerodynamic flight components such as wings, propellers and rotors. By integrating (26.39) over time one may compute a prediction of the aircraft's dynamic response to these external forces and moments and thus the evolution of its absolute pose. This pose is commonly represented by the position of the vehicle's center of gravity ${}^W\boldsymbol{r}_B = (x, y, z)^{\mathrm{T}}$ as well as the vehicle's orientation relative to an earth fixed world frame W which is considered inertial. The aircraft orientation is commonly represented using rotation matrices or quaternions. In the case of the rotation matrix representation, the aircraft orientation may be parameterized in three dimensional space by three consecutive rotations with the roll, pitch, and yaw angles $\varphi \in [-\pi, \pi]$, $\theta \in [-\pi/2, \pi/2]$ and $\psi \in [-\pi, \pi]$ as

$${}^W\boldsymbol{R}_B = \boldsymbol{R}_Z(\psi)\boldsymbol{R}_Y(\theta)\boldsymbol{R}_X(\varphi) \,. \tag{26.40}$$

In this case, relations between the body frame velocities ${}^B\boldsymbol{v}$ and ${}^B\boldsymbol{\omega}$ and the world frame pose can be expressed as

$$\begin{aligned} {}^W\dot{\boldsymbol{r}} &= {}^W\boldsymbol{R}_B\,{}^B\boldsymbol{v} \,, \\ {}^W\dot{\boldsymbol{R}}_B &= {}^W\boldsymbol{R}_B[{}^B\boldsymbol{\omega}]^{\times} \,, \end{aligned} \tag{26.41}$$

where $[{}^B\boldsymbol{\omega}]^{\times}$ corresponds to the skew-symmetric matrix of the vector ${}^B\boldsymbol{\omega}$.

In a minimal form, the orientation dynamics can be expressed in terms of roll, pitch, and yaw angles

$$\begin{pmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & \sin\varphi\tan\theta & \cos\varphi\tan\theta \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi/\cos\theta & \cos\varphi/\cos\theta \end{pmatrix}}_{\mathbf{J}_r} {}^B\boldsymbol{\omega} \,. \tag{26.42}$$

Note that the Jacobian $\mathbf{J}_r$ becomes singular at the boundaries of $\theta = \pm\pi/2$.

As a singularity-free, but still compact representation of orientation, quaternions may be used. Using the representation ${}^W\boldsymbol{q}_B = (q_w, q_x, q_y, q_z)^{\mathrm{T}}$ with real part q_w, the rotational kinematics become

$$ {}^W\dot{\boldsymbol{q}}_B = \frac{1}{2}\boldsymbol{\Omega}({}^B\boldsymbol{\omega})\,{}^W\boldsymbol{q}_B \,, \tag{26.43} $$

with the matrix $\boldsymbol{\Omega}$ defined as

$$ \boldsymbol{\Omega}({}^B\boldsymbol{\omega}) = \begin{pmatrix} 0 & {}^B\boldsymbol{\omega}^{\mathrm{T}} \\ -{}^B\boldsymbol{\omega} & [{}^B\boldsymbol{\omega}]^{\times} \end{pmatrix} . \tag{26.44} $$

In summary (using quaternions), we thus have the equations of motion

$$ \begin{aligned} {}^W\dot{\boldsymbol{r}} &= {}^W\boldsymbol{R}_B\,{}^B\boldsymbol{v} \,, \\ {}^W\dot{\boldsymbol{q}}_B &= \frac{1}{2}\boldsymbol{\Omega}({}^B\boldsymbol{\omega})\,{}^W\boldsymbol{q}_B \,, \\ {}^B\dot{\boldsymbol{v}} &= \frac{1}{m}\,{}^B\boldsymbol{F} - {}^B\boldsymbol{\omega}\times{}^B\boldsymbol{v} \,, \\ {}^B\dot{\boldsymbol{\omega}} &= {}^B\mathbf{I}^{-1}[{}^B\boldsymbol{\tau} - {}^B\boldsymbol{\omega}\times({}^B\mathbf{I}\,{}^B\boldsymbol{\omega})] \,. \end{aligned} \tag{26.45} $$

Note that the external forces and moments are related to the system's actuator inputs $\boldsymbol{u}$, the vehicle's orientation with respect to W and they typically are also functions of the linear and angular body motion as well as additional dynamics terms, represented here by the vector $\boldsymbol{\varepsilon}_r$

$$ {}^B\boldsymbol{F} = {}^B\boldsymbol{F}({}^W\boldsymbol{r}, {}^W\boldsymbol{q}_B, {}^B\boldsymbol{v}, {}^B\boldsymbol{\omega}, \boldsymbol{\varepsilon}_r, \boldsymbol{u}) \,, \tag{26.46} $$

$$ {}^B\boldsymbol{\tau} = {}^B\boldsymbol{\tau}({}^W\boldsymbol{r}, {}^W\boldsymbol{q}_B, {}^B\boldsymbol{v}, {}^B\boldsymbol{\omega}, \boldsymbol{\varepsilon}_r, \boldsymbol{u}) \,. \tag{26.47} $$

The additional dynamics $\boldsymbol{\varepsilon}_r$ may account for structural dynamics such as rotor flapping in case of a robotic helicopter or relevant actuator dynamics.

The resulting nonlinear system dynamics can usually be cast into state-space form and represented as

$$ \begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \,, \\ \boldsymbol{x} &= (\boldsymbol{x}_b, \boldsymbol{x}_r)^{\mathrm{T}} \,, \\ \boldsymbol{u} &= (u_1, \ldots, u_N)^{\mathrm{T}} \,, \end{aligned} \tag{26.48} $$

where the nonlinear functions $\boldsymbol{f}$ define the rate of change of the aircraft body states $\boldsymbol{x}_b$ as well as the additional states $\boldsymbol{x}_r$ affected by the set of N actuator inputs u_1 to u_N.

In order to gain a deeper understanding of how (26.48) is affected by changes of the flight system's geometric, structural, inertial, and aerodynamic parameters, three main problems are commonly of relevance [26.15]:

- The *trim problem* deals with the computation of the set of actuator inputs $\boldsymbol{u} = \boldsymbol{u}_0$ under which the nonlinear dynamic system presented in (26.48) remains in a desired trim point $\boldsymbol{x} = \boldsymbol{x}_0$ and thus $\boldsymbol{f}(\boldsymbol{x}_0, \boldsymbol{u}_0) = \boldsymbol{0}$. The most simple example of such a trim point $\boldsymbol{x}_0$ is the hover condition for a rotorcraft system where one may want to find the required rotor speed Ω_0 to hover or the steady forward flight condition for a fixed-wing UAV at a forward velocity V_t.
- The topic of *stability* deals with the question of how easily the system (26.48) will deteriorate from a specific trim condition $(\boldsymbol{x}_0, \boldsymbol{u}_0)$ under the influence of small disturbances $\Delta\boldsymbol{x}$ and $\Delta\boldsymbol{u}$. This investigation commonly involves the linearization of (26.48) according to

$$ \begin{aligned} \boldsymbol{A} &= \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)_{\boldsymbol{x}=\boldsymbol{x}_0, \boldsymbol{u}=\boldsymbol{u}_0} \,, \\ \boldsymbol{B} &= \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)_{\boldsymbol{x}=\boldsymbol{x}_0, \boldsymbol{u}=\boldsymbol{u}_0} \,, \\ \Delta\dot{\boldsymbol{x}} &= \boldsymbol{A}\Delta\boldsymbol{x} + \boldsymbol{B}\Delta\boldsymbol{u} \,, \end{aligned} \tag{26.49} $$

  where the eigenvalues and vectors of $\boldsymbol{A}$ will provide deeper insight into the motion characteristics and stability properties of an aircraft.
- Analyzing the *System Response*

$$ \boldsymbol{x}(t) = \boldsymbol{x}(0) + \int_0^t \dot{\boldsymbol{x}}\,\mathrm{d}t \tag{26.50} $$

  to characteristic inputs such as steps, pulses or specific input frequencies will provide additional information about the flight characteristics of a specific aircraft configuration.

These modeling and analysis concepts commonly find wide applicability for various types of robotic flight configurations and will be discussed in more detail for the specific UAV types presented hereafter.

Actuator Dynamics

Deriving the aerodynamic forces is combined with the dynamic equations of motion in order to assemble a complete model of the flying vehicle. However, as the employed actuators are of naturally limited bandwidth, accurate modeling furthermore requires the integration of the relevant motor or servo dynamics.

Nowadays, motors utilized in small size unmanned systems often belong to the category of brushless direct-current (DC) electric motors (BLDC). BLDCs are synchronous motors powered by a DC electric source via an integrated switching power supply. The equations of motion for a such a system are essentially nonlinear and

rather complex. However, working with small UAS, we may solely focus on the input–output dynamics which can be described with the following transfer function

$$\frac{\Delta\omega(s)}{\Delta Q_\mathrm{m}(s)} = \frac{-K_\mathrm{m}(1+\tau_\alpha s)}{(1+\tau_\mathrm{m}s)(1+\tau_\alpha s)+K_\mathrm{m}K_\alpha(Ki_{e0})^2}\,, \tag{26.51}$$

where $\Delta\omega(s)$, $\Delta Q_\mathrm{m}(s)$ correspond to the Laplace expressions of the linearized angular velocity and input torque, K_m is the mechanical gain, τ_m represents the mechanical time constant, K_α is the rotor gain, τ_α is the rotor time constant, K depends on electromagnetic properties of the motor and i_{e0} denotes the stator current linearization point [26.16]. Often, a satisfactory speed controller and BLDC dynamics description is obtained as the relation between a reference angular velocity and the actual output taking the even simpler first-order form

$$\frac{\Delta\omega(s)}{\Delta\omega^r(s)} = \frac{1}{1+\tau_\mathrm{mc}s}\,, \tag{26.52}$$

with the time constant τ_mc of the controlled motor.

Accounting for motor dynamics is essential for high-bandwidth control of agile vehicles that highly depend on such actuators (i. e., multirotors). However, in several other UAS configurations such as fixed-wing vehicles or conventional helicopters may, if needed, rather account for servo dynamics acting on control surfaces or a swashplate. Again, the relevant servo angle dynamics can be captured by an identified first-order transfer function of the form $1/(1+\tau_\mathrm{s}s)$, with the servo time constant τ_s.

26.4 Airplane Modeling and Design

Ever since the beginning of aviation, a broad spectrum of airplanes has been built and operated successfully: size, speed, and maneuverability vary widely and as a function of application. Since design and modeling are strongly related, we want to first give an overview of the physical principles common to all such configurations, and provide analysis tools for characterizing static and dynamic properties of an airplane. The design problem somewhat constitutes the inverse problem: for specified target characteristics, the engineer needs to find a suitable configuration; we therefore provide a summary of design guidelines aimed at fast convergence to a suitable design. Finally, a simple and classical autopilot scheme is presented underlining the need for models also at that stage.

26.4.1 Forces and Moments

Consider Fig. 26.18 for the introduction of airplane geometry definitions and main forces. Forces and moments are reduced to the airplane center of gravity (COG). Note that the angle of attack (AOA) α is defined as the angle between the x-axis and the true airspeed vector $\boldsymbol{v}_\mathrm{t}$ projected into the body x–z-plane, β denotes the sideslip angle, causing a typically unwanted sideslip force Y, L, and D denote lift and drag, W stands for the weight and T for thrust, which may act into a direction different from x (at a thrust angle ϵ_T). We furthermore write the aerodynamic moment vector as ${}^B\boldsymbol{\tau}_\mathrm{A} = [L_\mathrm{A}, M_\mathrm{A}, N_\mathrm{A}]^\mathrm{T}$. Also note the introduction of the main control surfaces that are designed to mainly influence the aerodynamic moment: with ailerons, elevator, and rudder, the roll (L_A), pitch (M_A), and yaw (N_A) moments are controlled. The indicated flaps, if available, are used for increasing lift for take-off and landing, in order to achieve a slower minimum speed.

Aerodynamic Forces and Moments

The aerodynamic forces and moments can be modeled to various accuracy using full 3-D CFD or with 2.5-D tools: Sect. 26.3.3 overviews such an approach which can be used to model the aerodynamic surfaces in incompressible flow. For enhanced accuracy, fuselages may be

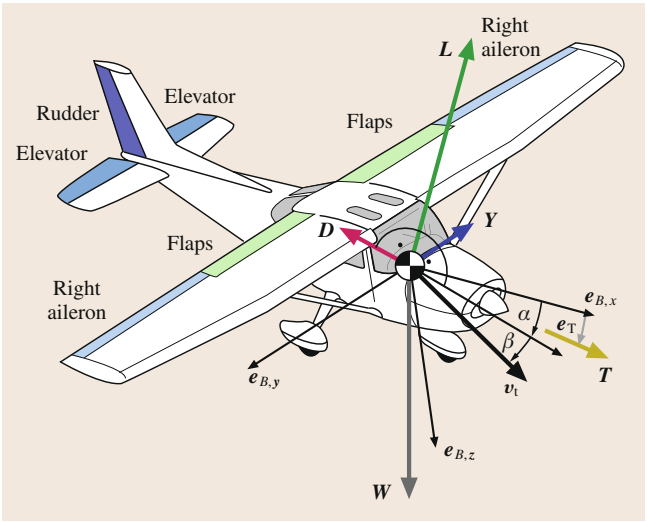


Fig. 26.18 Geometric definitions and main forces acting on the airplane (general case, not in equilibrium)

considered using again a combination of potential flow (placing singularities) and boundary layer theory. Respective ready-to-use software such as AVL [26.17] and XFLR [26.12] is available for free.

The forces and moments may again be written with dimensionless coefficients as

$$L = \frac{1}{2}\rho V_\mathrm{t}^2 c_\mathrm{L} A \,, \tag{26.53}$$

$$D = \frac{1}{2}\rho V_\mathrm{t}^2 c_\mathrm{D} A \,, \tag{26.54}$$

$$M_A = \frac{1}{2}\rho V_\mathrm{t}^2 c_\mathrm{M} \bar{c} A \,, \tag{26.55}$$

with the wing area A, the mean chord length $\bar{c}$, and the true airspeed $V_\mathrm{t} = \|\boldsymbol{v}_\mathrm{t}\|$. The moments L_A and N_A are made dimension-less with the wingspan b rather than the chord length.

Static Performance Considerations

Having characterized lift and drag of an airplane, three operating points are of particular interest.

First, stall is occurring at $c_{\mathrm{L,max}}$. This condition can be directly translated into constant-speed level-flight stall speed by applying the lift balance $L = mg$.

Second, the maximum $c_\mathrm{L}/c_\mathrm{D}$ ratio, or the *glide ratio* characterizes the maximum aerodynamic efficiency, i. e., the operating point for maximum range (assuming constant propulsive efficiency).

Finally, the maximum $c_\mathrm{L}^3/c_\mathrm{D}^2$ ratio, or the *climb factor* describes the condition at which power consumption is minimized, thus maximizing flight time (again assuming constant propulsion unit efficiency).

The latter two conditions have direct interpretation in gliding (or propulsion shut-off), in terms of maximum distance reached per altitude lost and minimum sink rate, respectively. Again, corresponding velocities can be found using the lift balance.

Thrust

For detailed insight into the variety of propulsion systems and respective models, the interested reader is referred to [26.13, 18]. As an approximation for the important case of a propeller, the BEMT method as described in Sect. 26.3.4 is suggested. For many applications, choosing the propeller speed as the system input and neglecting motor dynamics is sufficient.

26.4.2 Static Stability

Various forms of stability constitute central characteristics of an airplane related to whether or not it can be flown by a human pilot or flight controller. Simple stability criteria can be derived by requiring reaction forces and moments to be opposing a disturbance. We assume stationary conditions in the sense of constant linear and angular speeds: the respective force and moment balance is typically straightforward to apply in order to determine the starting point of the stability analysis.

Longitudinal Static Stability

We will take a close look at the example of longitudinal static stability playing a central role in airplane analysis and design. Leaving aside possible influence of the propulsion unit, the respective directional stability criterion is stated as

$$\frac{\partial c_\mathrm{M}}{\partial \alpha} < 0 \,, \tag{26.56}$$

at the equilibrium condition $c_\mathrm{M} = 0$. Figure 26.19 illustrates an exemplary moment coefficient as a function of AOA. Note that elevator actuation will *move* this curve up and down, and with it the equilibrium point (2) toward higher or lower angles of attack (i. e., lower or higher trimmed speeds). Figure 26.20 illustrates the forces and moments in the stable equilibrium

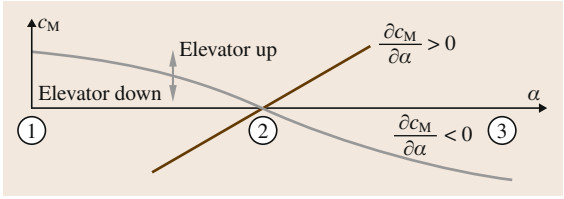


Fig. 26.19 Moment coefficient c_M as a function of AOA α the equilibrium point (2) is stable for $\partial c_\mathrm{M}/\partial\alpha < 0$ (*brown curve*)

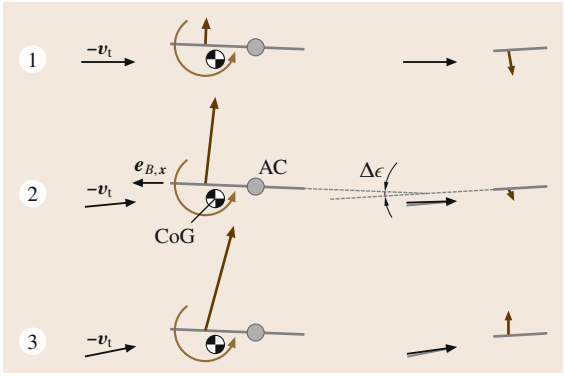


Fig. 26.20 Forces and moments at the main wing and tail for zero lift (1), the stable equilibrium AOA (2), and for high lift (3). The forces and moments are drawn into the individual surfaces' aerodynamic centers. Note the tilted inflow at the tail due to downwash. The green filled circle denotes the overall airplane aerodynamic center (AC)

with a simplified airplane side-view as compared to the points (1) and (3), i. e., zero-lift ($\alpha = 0$) and high-lift, respectively. The stability criterion can be equivalently stated as: the airplane COG needs to be *in front* of the airplane aerodynamic center. Note the main parameters that influence the stability are tail lever arm, tail area, the *longitudinal dihedral* $\Delta\epsilon$ (Fig. 26.20 for its geometric definition), and the COG location along the x-axis.

26.4.3 Dynamic Model

While some core characteristics such as static stability and performance measures may already be established using aerodynamics coefficients only, we now turn to analyze the dynamics, since they provides a much richer insight into airplane characteristics.

For application of the 6-D (six-dimensional) rigid body dynamics (26.45), the forces and moments from the various sources need to be assembled and represented in the body frame

$$ {}^B\boldsymbol{F} = \begin{pmatrix} L\sin\alpha - D\cos\alpha + T\cos\epsilon_\mathrm{T} \\ Y \\ -L\cos\alpha - D\sin\alpha + T\sin\epsilon_\mathrm{T} \end{pmatrix} + {}^B\boldsymbol{W} \,, \tag{26.57} $$

$$ {}^B\boldsymbol{\tau} = \begin{pmatrix} L_\mathrm{A} + L_\mathrm{T} \\ M_\mathrm{A} + M_\mathrm{T} \\ N_\mathrm{A} + N_\mathrm{T} \end{pmatrix} \,, \tag{26.58} $$

with the weight in body coordinates ${}^B\boldsymbol{W} = {}^B\boldsymbol{R}_W[0, 0, -mg]^\mathrm{T}$, and where the T-subscript indicates (possible) moment components from thrust. Note that the system inputs $\boldsymbol{u}$ are hidden inside these forces and moments. Also be aware of α and β containing parts of the state vector

$$ \alpha = \arctan2(w_\mathrm{t}, u_\mathrm{t}) \,, \tag{26.59} $$

$$ \beta = \arcsin(v_\mathrm{t}/V_\mathrm{t}) \,, \tag{26.60} $$

where the true airspeed components are used

$$ {}^B\boldsymbol{v}_\mathrm{t} := \begin{cases} u_\mathrm{t} \\ v_\mathrm{t} \\ w_\mathrm{t} \end{cases} = {}^B\boldsymbol{v} - {}^B\boldsymbol{R}_W {}^W\boldsymbol{w} \,, \tag{26.61} $$

with the wind vector ${}^W\boldsymbol{w}$.

Furthermore, due to the airplane symmetry plane, the inertia matrix becomes

$$ {}^B\mathbf{I} = \begin{pmatrix} I_{xx} & 0 & I_{xz} \\ 0 & I_{yy} & 0 \\ I_{xz} & 0 & I_{zz} \end{pmatrix} \,. \tag{26.62} $$

Parametric Force and Moment Models

Let us consider the example of a simple airplane configuration with ailerons, a rudder and an elevator plus a propeller, driven by an electric motor at rotation speed ω_p.

We define the system input vector $\boldsymbol{u} = [\delta_\mathrm{a}, \delta_\mathrm{e}, \delta_\mathrm{r}, \delta_\mathrm{T}]$ as normalized aileron, rudder, and elevator action, $\delta_\mathrm{a}, \delta_\mathrm{e}, \delta_\mathrm{r} \in [-1, 1]$ as well as (normalized) thrust $\delta_\mathrm{T} \in [0, 1]$.

The fully parametric nonlinear model provided below largely follows [26.19]. We approximate the lift, drag as well as sideslip coefficients with polynomials in α and β

$$ \begin{aligned} c_\mathrm{L} &\approx c_{\mathrm{L},0} + c_{\mathrm{L},\alpha}\alpha + c_{\mathrm{L},\alpha 2}\alpha^2 + c_{\mathrm{L},\alpha 3}\alpha^3 \,, \\ c_\mathrm{D} &\approx c_{\mathrm{D},0} + c_{\mathrm{D},\alpha}\alpha + c_{\mathrm{D},\alpha 2}\alpha^2 + c_{\mathrm{D},\beta 2}\beta^2 \,, \\ c_\mathrm{Y} &\approx c_{\mathrm{Y},\beta}\beta \,. \end{aligned} \tag{26.63} $$

For many applications except slow flying airplanes, the second-order and third-order term of c_L can be omitted.

As far as the torques are concerned, we introduce also dependencies on normalized angular rates

$$ {}^B\boldsymbol{\omega}_\mathrm{n} = (p_\mathrm{n}, q_\mathrm{n}, r_\mathrm{n})^\mathrm{T} = \left(\frac{pb}{2V_\mathrm{t}}, \frac{q\bar{c}}{2V_\mathrm{t}}, \frac{rb}{2V_\mathrm{t}} \right)^\mathrm{T} \,. \tag{26.64} $$

A suitable approximation of the moment coefficients is now made as

$$ \begin{aligned} c_\mathrm{l} &\approx c_{\mathrm{l},0} + c_{\mathrm{l},\delta_a}\delta_a + c_{\mathrm{l},\beta}\beta + c_{\mathrm{l},p_\mathrm{n}}p_\mathrm{n} + c_{\mathrm{l},r_\mathrm{n}}r_\mathrm{n} \,, \\ c_\mathrm{M} &\approx c_{\mathrm{M},0} + c_{\mathrm{M},\delta_e}\delta_e + c_{\mathrm{M},\alpha}\alpha + c_{\mathrm{M},q_\mathrm{n}}q_\mathrm{n} \,, \\ c_\mathrm{N} &\approx c_{\mathrm{N},0} + c_{\mathrm{N},\delta_r}\delta_r + c_{\mathrm{N},\beta}\beta + c_{\mathrm{N},r_\mathrm{n}}r_\mathrm{n} \,. \end{aligned} \tag{26.65} $$

Finally, the propeller thrust force needs to be modeled. Using the advance ratio $J = \frac{2\pi V_\mathrm{t}}{\omega_\mathrm{p} d}$, with propeller diameter d, we can approximate the thrust coefficient

$$ c_\mathrm{T} \approx c_{\mathrm{T},0} + c_{\mathrm{T},J}J + c_{\mathrm{T},J2}J^2 \,. \tag{26.66} $$

The thrust is then obtained as

$$ T = \rho \left(\frac{\omega_\mathrm{p}}{2\pi} \right)^2 d^4 c_\mathrm{T} \,. \tag{26.67} $$

Linearized Dynamics

As common throughout literature, the linearized airplane dynamics are written using Euler angles, which is why we will follow the same approach. But conceptually, they could be written in a singularity-free form using a minimal quaternion perturbation.

Typically, a separation into *longitudinal* and *lateral* dynamics is made, in order to assess related characteristics separately. Furthermore, the state is transformed to contain α, β, and V_t rather than ${}^B\boldsymbol{v}$.

The linear dynamics around a reference state $\boldsymbol{x}_0$ and input $\boldsymbol{u}_0$ vector takes the form

$$\Delta\dot{\boldsymbol{x}}_{\text{lon}} = \boldsymbol{A}_{\text{lon}}\Delta\boldsymbol{x}_{\text{lon}} + \boldsymbol{B}_{\text{lon}}\Delta\boldsymbol{u}_{\text{lon}}$$

and

$$\Delta\dot{\boldsymbol{x}}_{\text{lat}} = \boldsymbol{A}_{\text{lat}}\Delta\boldsymbol{x}_{\text{lat}} + \boldsymbol{B}_{\text{lat}}\Delta\boldsymbol{u}_{\text{lat}} .$$

The following formulation follows [26.19] to a large extent.

Figure 26.21 describes the separation in terms of inputs $\Delta\boldsymbol{u}$ and states $\Delta\boldsymbol{x}$ for the linearized system.

The longitudinal nonlinear equations are given as

$$\begin{aligned}
\dot{q} &= \frac{1}{I_{yy}}[M_A + M_T - (I_{xx} - I_{zz})pr + I_{xz}(p^2 - r^2)] ,\\
\dot{V}_t &= \frac{1}{m}[-D\cos\beta + Y\sin\beta\\
&\quad + T\cos(\alpha - \epsilon_T)\cos\beta + mg_1] ,\\
\dot{\alpha} &= \frac{1}{\cos\beta}\left[\frac{1}{mV_t}(-L - T\sin(\alpha - \epsilon_T) + mg_3) + q_A\right] ,\\
\dot{\theta} &= q\cos\varphi - r\sin\varphi ,
\end{aligned} \tag{26.68}$$

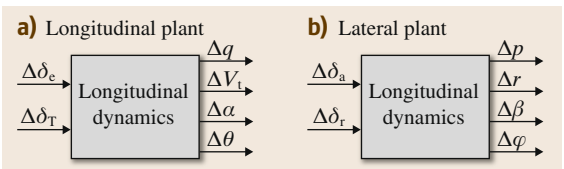


Fig.26.21a,b Linearized **(a)** longitudinal, and **(b)** lateral plants for inputs $\Delta\boldsymbol{u}$ (around $\boldsymbol{u}_0$) of the local states $\Delta\boldsymbol{x}$ (around $\boldsymbol{x}_0$)

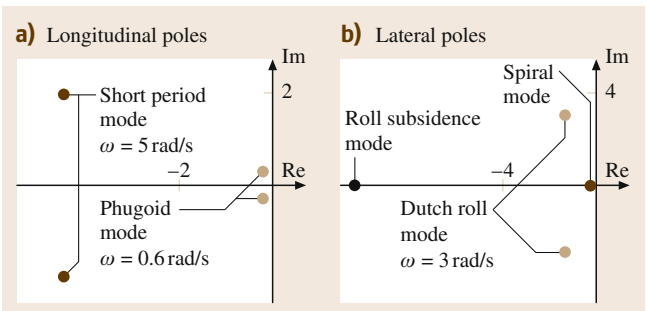


Fig. 26.22 **(a)** Longitudinal, and **(b)** lateral poles of an example aerobatic RC airplane

and the lateral nonlinear equations amount to

$$\begin{aligned}
\dot{p} &= \frac{I_{zz}(L_A + L_T - T_p)}{I_{xx}I_{zz} - I_{xz}^2} - \frac{I_{xz}(N_A + N_T - T_r)}{I_{xx}I_{zz} - I_{xz}^2} ,\\
\dot{r} &= \frac{I_{xz}(L_A + L_T - T_p)}{I_{xx}I_{zz} - I_{xz}^2} + \frac{I_{zz}(N_A + N_T - T_r)}{I_{xx}I_{zz} - I_{xz}^2} ,\\
\dot{\beta} &= -r_A + \frac{1}{mV_t}[Y\cos\beta + D\sin\beta\\
&\quad - T\cos(\alpha - \epsilon_t)\sin\beta + mg_2] ,\\
\dot{\varphi} &= p + q\sin\varphi\tan\theta + r\cos\varphi\tan\theta ,
\end{aligned} \tag{26.69}$$

where the following terms were used

$$\begin{aligned}
g_1 &= g(-\cos\alpha\cos\beta\sin\theta + \sin\beta\sin\varphi\cos\theta\\
&\quad + \sin\alpha\cos\beta\cos\varphi\cos\theta) ,\\
g_2 &= g(\cos\alpha\sin\beta\sin\theta + \cos\beta\sin\varphi\cos\theta\\
&\quad - \sin\alpha\sin\beta\cos\varphi\cos\theta) ,\\
g_3 &= g(\sin\alpha\sin\theta + \cos\alpha\cos\varphi\cos\theta) ,\\
q_A &= q\cos\beta - p\sin\beta\cos\alpha - r\sin\alpha\sin\beta ,\\
r_A &= r\cos\alpha - p\sin\alpha ,\\
T_p &= (I_{zz} - I_{yy})qr + I_{xz}pq ,\\
T_r &= (I_{yy} - I_{xx})qp - I_{xz}qr .
\end{aligned} \tag{26.70}$$

The linearizations of (26.68) and (26.69) are straightforward to obtain and not provided here due to space constraints. For a specific operating point, typically stationary ($\varphi_0 = 0$, $\theta_0 = \alpha_0$, and ${}^B\boldsymbol{\omega} = \boldsymbol{0}$), the standard tools of linear systems analysis can be employed. Most importantly, the *pole* locations in the imaginary plane will tell the characteristic modes and their dynamic stability. Figure 26.22 shows a pole location plot for an example RC airplane and introduces the related mode names.

In the case of a real pole π_i, it has the time constant $\tau_i = -1/\text{Re}(\pi_i)$. In the case of a complex conjugate pole pair, it is associated with a damping ratio $\zeta_i = -\text{Re}(\pi_i)/\sqrt{\text{Im}(\pi_i)^2 + \text{Re}(\pi_i)^2}$ and with an eigenfrequency $\omega_i = \sqrt{\text{Im}(\pi_i)^2 + \text{Re}(\pi_i)^2}$.

Figure 26.23 illustrates and characterizes the main modes.

26.4.4 Design Guidelines

Airplane design typically consists of the three stages *conceptual* design, *preliminary* design, and *detail* design. Here, we focus on the first two phases; due to space constraints, details of airplane structural design

and analysis are not covered here. The reader is referred to respective literature, e.g., [26.20], or [26.21], the latter covering RC-type aircraft. In the following, we provide a quick overview of practical guidelines for the typically iterative design process related to achieving characteristics as described above. The guidelines follow largely [26.22] and [26.18], with focus on slow-flying small-scale UAS.

Sizing and Geometry of Main Components

In the following, we provide some rules of thumb as initial guess for the design process in terms of sizing the wing, tail, control surfaces, and the propulsion unit. As a first and very general advice, the engineer is encouraged to minimize wetted area and cross section, as well as any kind of nacelles for increased aerodynamic efficiency.

Wing. First, an existing airfoil shall be chosen with characteristics meeting the requirements in the target flow regime (Re and Ma). When it comes to determining the overall wing size, a first estimate of design weight including structure, avionics, payload, propulsion unit, and energy storage is of crucial importance (26.2). With the target speed V_r and design lift coefficient c_l, a rough guess can be made for the wing area

$$A = 2mg/(\rho V_r^2 c_l) \, .$$

Concerning wing shape, clearly highest efficiency is reached with high aspect ratios (plus no multiple lifting surfaces), and, at low Ma, no sweep-back – as long as still implementable with a structural concept and staying at reasonably high Re numbers. For high efficiency, it is advisable to achieve an elliptic lift distribution to some extent by geometry. For benign stalling characteristics, it is furthermore highly advisable to twist the wing leading edge downward with increased spanwise distance (which also influences the lift distribution).

Finally, some dihedral should be considered for roll stability.

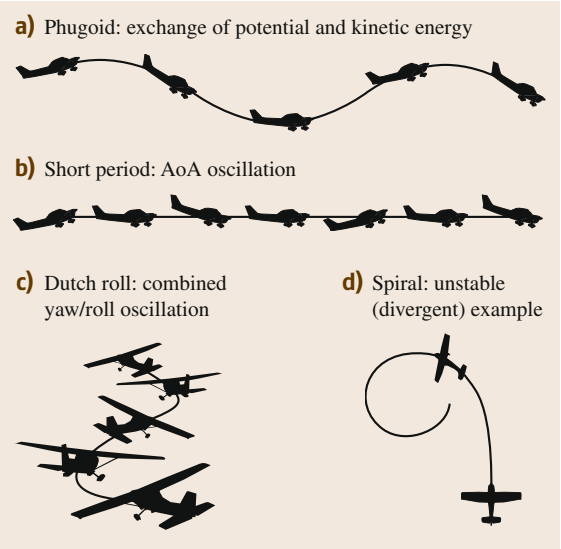


Fig.26.23a–d Characteristic trajectories of excited longitudinal modes (**a**,**b**) and lateral modes (**c**,**d**)

Tail. Various types of tails and even *exotic* configurations like the canard have been suggested; here, we want to simply point out the importance of the so-called *tail volume coefficient*, c_{VT} and c_{HT} concerning vertical and horizontal tail, respectively,

$$c_{VT} = l_{VT} A_{VT}/(bA) \, , \tag{26.71}$$

$$c_{HT} = l_{HT} A_{HT}/(\bar{c}A) \, , \tag{26.72}$$

with the wing to vertical tail lever arm l_{VT} and the wing to horizontal tail l_{HT} (these are referenced to the individual mean 1/4-chord points). Typical values for small-size slow airplanes are c_{VT}=0.02–0.04 and c_{HT} = 0.5–0.7. Furthermore, care should be taken that control surfaces are not completely blanketed in the case of stall (for stall/spin recovery).

Control Surfaces. Ailerons typically extend from around 50% in span direction to 90%; in this setting, 20–30% of wing chord is suggested as aileron depth. Tail control surface depth is typically chosen around 40% of the respective chord.

Propulsion. Finally, some advice is given concerning the propulsion unit. Some UAVs are required to be handlaunched: note that this imposes limits on the overall maximum take-off mass and minimum/stall airspeed. Experience shows that reasonable limits are <9 m/s minimum/stall speed and 7 kg airplane mass. For such small UAVs, a static thrust to weight ratio of at least 50% is highly recommended. In general, the propulsion unit must be sized to meet the specifications in terms of climb rates, maximum level flight speed and service ceiling. For the highest efficiency, the propulsion unit should be designed such as to provide highest efficiency at the design operating point (subscript r) $T_r/W = c_{D,r}/c_{L,r}$. For a hobbyist brushless DC outrunner type motor, the maximum power per motor mass ratio of 3.4 kW/kg can be used for estimation of the propulsion unit weight [26.23] (gearbox and propeller mass not included).

Handling Qualities

Manned aviation introduced the notion of *handling qualities*, assessing how well an aircraft can be flown by a human pilot as a basis for certification of both civil and military airplanes. Since UAS typically rely on autopilot systems enabling a certain degree of autonomy, these concepts may not be directly applied, but are still extremely relevant. Most importantly, the handling qualities concerning static and dynamic stability, as well as controllability determine the success of a UAS design.

While an autopilot can handle more and faster instabilities than a pilot, it can certainly not compensate for missing actuation authority. As detailed in Sect. 26.7, it is advisable to implement de facto manual operation mode as a testing, backup, or even standard operation mode, in which the airplane is either steered manually or through some stability augmentation system (SAS). Therefore, it is highly advisable that the resulting system complies with the following core requirements (simplified from [26.18]):

- Static longitudinal stability: most aft COG at least 5% of $\bar{c}$ in front of aerodynamic center (static margin, SM).
- Phugoid damping $\zeta_{\mathrm{ph}} > 0.2$.
- Short-period oscillation $\omega_{\mathrm{sh}} > 2$, $\zeta_{\mathrm{s}} > 0.5$.
- Spiral mode may be unstable, if $\tau_{\mathrm{sp}} > 20\,\mathrm{s}$.
- Roll acceleration at maximum aileron deflection $|\dot{p}(\delta_{a,\max})| > 5\,\mathrm{rad/s^2}$, roll subsidence time constant $\tau_{\mathrm{rs}} < 1\,\mathrm{s}$.
- Dutch roll damping $\zeta_{\mathrm{dr}} > 0.1$.
- Spins shall not be entered abruptly and must always be recoverable.

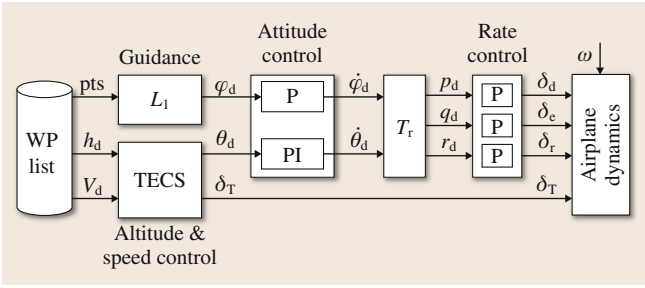


Fig. 26.24 Simple cascaded airplane guidance and control system

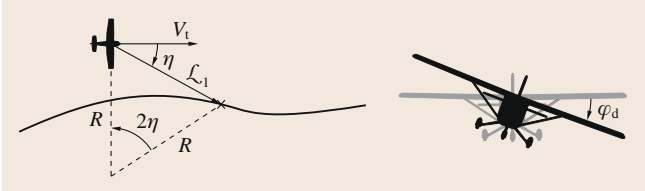


Fig. 26.25 Illustration of $\mathcal{L}_1$ lateral guidance

26.4.5 A Simple Autopilot

As shown earlier, airplane dynamics are nonlinear multiple-input-multiple-output (MIMO) systems with a significant amount of cross-coupling, thus they are inherently challenging to control. While a plethora of control strategies have been suggested as autopilots, we will provide a simple yet functional approach here that employs the popular concept of *cascaded* control loops as well as simple linear single-input–single-output (SISO) *PID* controllers acting on subparts of the dynamics. This approach is still widely deployed and well-understood, despite the fact that more advanced controllers, such as model-based linear quadratic regulators (LQR) with gain scheduling or nonlinear dynamic inversion (NDI) may achieve significantly better performance. Reference [26.24] constitutes an eccelent reference for in-depth treatment of small UAS guidance and control with cascaded control loops.

Cascaded Control Architecture

Figure 26.24 introduces the overall controller architecture. It presents the (typical) separation of per-axis *rate controllers* at the innermost loop, followed by an *attitude* controller and a combined altitude and speed controller TECS (total energy control system), as well as by a lateral $\mathcal{L}_1$ guidance. In general, care must be taken to separate successively closed loops by around a decade in terms of bandwidth.

Assuming little cross-axis sensitivity, the rate controllers may be implemented with simple individual P-controllers, optionally with $1/V_{\mathrm{t}}^2$ gain scaling. Also the attitude controllers can be as simple as P and PI-controllers for roll and pitch, respectively. Note that the desired roll and pitch angle derivatives need to be transformed in the static block T_{r} into angular reference rates: this can be achieved by applying the inverse Jacobian $\mathbf{J}_{\mathrm{r}}$ from (26.42) – where the missing yaw angle time derivative can be computed from the coordinated turn constraint $\beta = 0$, $\dot{\beta} = 0$ in (26.69)

$$\dot{\psi} = -\frac{g \sin\phi}{V_{\mathrm{t}} \cos\alpha \cos\phi} + \frac{\dot{\theta}_{\mathrm{d}} \sin\phi}{\cos\phi \cos\theta} \,. \tag{26.73}$$

The combined altitude and speed controller (TECS) inspired from [26.25] uses the difference to the reference altitude $\Delta h = h_{\mathrm{d}} - h$ to compute a desired climb rate of the form $\dot{h}_{\mathrm{d}} = \dot{h}_{\mathrm{traj}} + K_{P,\mathrm{alt}} \Delta h$, with the given trajectory rate of climb $\dot{h}_{\mathrm{traj}}$ and a P-gain $K_{P,\mathrm{alt}}$. Knowing the speed and corresponding angle of attack, $\dot{h}_{\mathrm{d}}$ can be simply converted into a desired pitch angle θ_{d} – to be saturated according to maximum thrust (climb) and drag (sink). Since climb rate must be provided via

additional thrust, the respective power component is computed as $\Delta T_{\text{climb}} = mg\dot{h}_{\text{d}}/V_{\text{t}}$. Concerning the speed control, the second thrust component is computed with the P-gain $K_{P,\text{vel}}$ as $\Delta T_{\text{acc}} = mK_{P,\text{vel}}\Delta V_{\text{t}}$.

As a last autopilot component, the $\mathcal{L}_1$ lateral guidance [26.26] proceeds as follows (illustrated in Fig. 26.25): a reference circular path of radius R is calculated that intersects the reference path given by the waypoint sequence at look-ahead distance L_1. In order to track this reference, a centripetal acceleration of V_{t}^2/R is needed, which can now be directly translated into a desired roll angle $\varphi = \arctan(V_{\text{t}}^2/(Rg))$ corresponding to a coordinated (level) turn – saturated with maximum bank angles.

Note that the suggested scheme should be enhanced by stall prevention and recovery (AOA monitoring and control), as well as by preventing sideslip for safer (and more efficient) operation.

26.5 Rotorcraft Modeling and Design

Various types of rotorcraft UAS configurations have been developed in the past (some examples are shown in Fig. 26.26), from helicopter-type UAVs such as [26.27, 28], over a vast selection of multicopter configurations such as [26.29, 30] and tail-sitter vehicles such as [26.31, 32] up to completely new types of flight mechanisms [26.33, 34]. The design, modeling, and system analysis process for all these RW-UAS types is essentially very similar and is largely based on the methodologies originally developed within the aerospace community for full-scale rotorcraft design and evaluation [26.15, 35]. In this context, it is important to realize that the rotorcraft design process goes beyond mere efficiency and payload considerations focused on the propulsion components (e.g., using BEMT). Designing an effective RW-UAS should in principle also include flight dynamics assessments of the entire robotic flight platform.

Flight performance assessment is commonly based on one of two types of modeling approaches referred to as *quasi-steady* and *hybrid* [26.36]. The quasi-steady modeling method employs a single rigid body representation of the aircraft affected by the steady-state forces and moments originating from the propulsion subsystem. Hybrid models treat the rotorcraft as a multibody system where the dynamics of the aircraft body are coupled with additional dynamics of the rotor or propeller blades (e.g., blade flapping dynamics). For propeller-based RW-UAS like multicopter and tail-sitter vehicles using the quasi-steady approach to, e.g., model attitude dynamics is most widespread. This may be related to the fact that for these vehicle configurations, properly accounting for motor dynamics may be more relevant than accounting for high-order effects related to structural deformations of the propeller blades. For helicopter-type UAVs the hybrid approach is more common as some dynamic modes of the rotor system are likely to couple with the attitude dynamics of the main rotorcraft body. Note that a proper application of the hybrid approach is considerably more involved than using quasi-steady models and should only be resorted to if justified.

A detailed treatment of the specific modeling and design procedures for every robotic rotorcraft configuration is beyond the scope of this chapter and thus the presented considerations focus on helicopter-type and multicopter UAVs. Based on the extensive theoretical aerospace-related background available in [26.14, 15, 35] amongst others, models for the most relevant rotor respectively propeller forces and moments are presented and subsequently appended to the rotorcraft body dynamics discussed in Sect. 26.3.6. A simplified

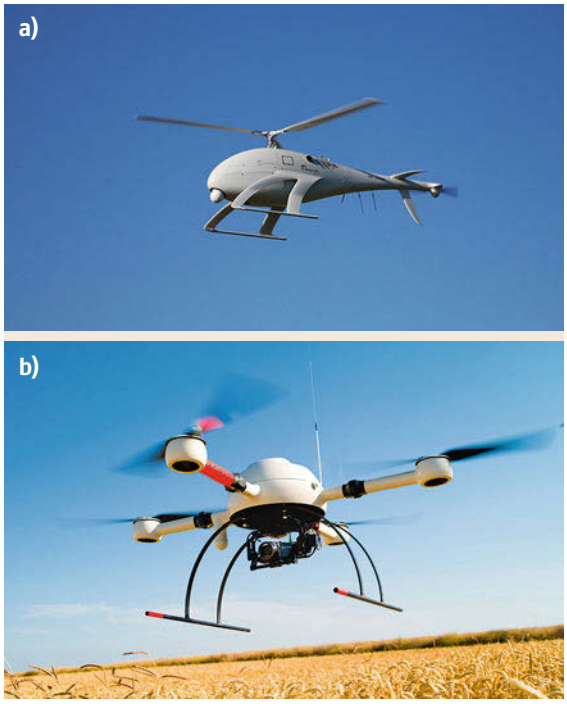


Fig. 26.26 **(a)** Conventional helicopter configuration. Swiss UAV Neo S-300. **(b)** Quadro-copter configuration. Microdrones MD4-1000

hybrid modeling approach is introduced and reductions to quasi-steady models are discussed where applicable. Finally, a few metrics meaningful for rotorcraft design and control purposes are discussed and summarized shortly.

26.5.1 Mechanical Design of Rotors and Propellers

The main control mechanism for any RW-UAS is its rotors or propellers. Accordingly, to understand the working principles and the dynamics of the rotorcraft type of aircraft, it is worth investigating a few of the main design characteristics found in these flying mechanisms.

The following discussion focuses on the operation principles of helicopter rotors and will be expanded to propellers subsequently. Figure 26.27 visualizes the typical rotor degrees of freedom realized via the *flap*, *lead-lag* and *feathering* (also referred to as *pitch*) hinges. The flap hinge allows the rotor blade to flap due to aerodynamic and inertial loads affecting the blade body during flight. The lead-lag hinge responds to lateral rotor blade moments due to Coriolis forces related to flapping. Where the flap and the lead-lag hinges are usually passive, possibly augmented with spring or damper elements, the pitch hinge is active in order to adjust the blade angle of attack and thus the generated aerodynamic forces.

Three types of rotor hubs are typically found in modern helicopters referred to as *teetering*, *articulated*, and *hingeless* depending on the mechanical realization of the flap hinge (Fig. 26.28). In the case of teetering rotor, a single hub flap hinge is located directly on the rotorshaft axis, rigidly connecting a set of two rotor blades. For the articulated rotor, the blades and the rotor hub are connected via mechanical hinges at a specific offset e from the rotor shaft axis, thus allowing each blade to flap individually. The hingeless rotor flaps through the deformation of elastic elements connecting the hub with each individual rotor blade or directly through structural deformation of the blades themselves. In this case, a virtual hinge offset can be defined at the intersection of the rotor hub plane and the tangent to the deflecting blade body at 75% of the rotor radius [26.14]. The specific characteristics of the flap hinge (offset from rotor shaft, stiffness, and damping) are fundamental for rotor blade flapping and in consequence for the rotorcraft pitch and roll dynamics as discussed later.

In the case of most propeller-based rotary-wing (RW) UAVs (e.g., multicopter systems) the feathering and lead-lag degrees of freedom do not exist. Nevertheless, propeller-based RW-UAVs may exert blade flapping by deformation of the propeller blades.

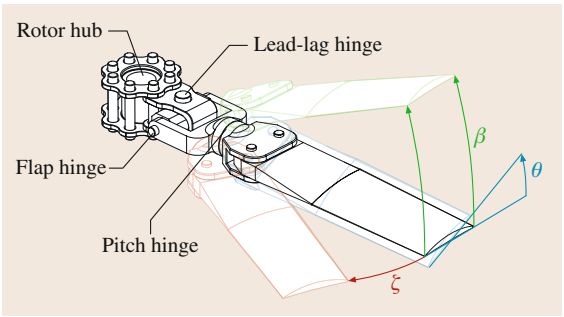


Fig. 26.27 Typical hinge configuration of an articulated rotor blade

26.5.2 Rotorcraft Dynamics

As discussed in Sect. 26.3.6, the main rotorcraft body dynamics can be directly described by the simplified differential (26.45) which for most rotorcraft systems maintain the presented mathematical structure. For helicopter-type and multicopter UAV configurations the dominant set of external forces and moments affecting these dynamics can be summarized as

$$\boldsymbol{F} = \boldsymbol{F}_{\mathrm{G}} + \sum_{i=1}^{N_{\mathrm{r}}} \boldsymbol{F}_{\mathrm{T}}^{i} + \sum_{i=1}^{N_{\mathrm{r}}} \boldsymbol{F}_{\mathrm{H}}^{i} + \boldsymbol{F}_{\mathrm{D}} \,, \tag{26.74}$$

$$\boldsymbol{\tau} = \sum_{i=1}^{N_{\mathrm{r}}} \boldsymbol{\tau}_{\mathrm{Q}}^{i} + \sum_{i=1}^{N_{\mathrm{r}}} \boldsymbol{\tau}_{\mathrm{T}}^{i} + \sum_{i=1}^{N_{\mathrm{r}}} \boldsymbol{\tau}_{\mathrm{H}}^{i} + \sum_{i=1}^{N_{\mathrm{r}}} \boldsymbol{\tau}_{\beta}^{i} \,. \tag{26.75}$$

The vector $\boldsymbol{F}_{\mathrm{G}}$ represents the weight force and $\boldsymbol{F}_{\mathrm{T}}^{i}$ is the thrust force of the i-th rotor or propeller out of the total set of N_r rotors or propellers (Fig. 26.29). These forces are related to the body heave dynamics but due

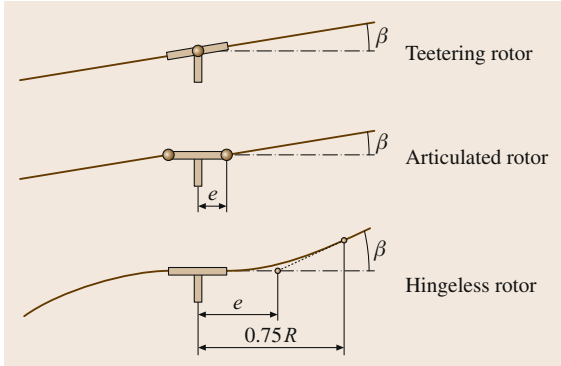


Fig. 26.28 Rotor hub design concepts of a teetering rotor, an articulated rotor with hinge offset e and a hingeless rotor with a virtual hinge offset e

to the underactuation of most rotorcraft are also responsible for the lateral rotorcraft acceleration.

The additional in-plane hub forces $\boldsymbol{F}_{\mathrm{H}}^{i}$ represent drag-related effects that may be neglected near hover but become more dominant for higher lateral flight velocities [26.15]. The vector $\boldsymbol{F}_{\mathrm{D}}$ represents the drag force associated with the rotorcraft main body. For simplicity's sake, it is assumed that the center of pressure of the rotorcraft body is collocated with its center of gravity which is not necessarily the case.

The relevant set of external moments is defined by the torques $\boldsymbol{\tau}_{\mathrm{Q}}^{i}$ affecting the vehicle yaw dynamics, the thrust-induced moments $\boldsymbol{\tau}_{\mathrm{T}}^{i}$, the moments introduced by the hub forces $\boldsymbol{\tau}_{\mathrm{H}}^{i}$ and the flapping moments related to the rotor hub stiffness $\boldsymbol{\tau}_{\beta}^{i}$.

More explicit expressions for the respective force and moment terms may be found in Tables 26.6 and 26.7, where T_i is the averaged thrust force magnitude, H_x^i and H_y^i are the hub force components along the rotorcraft body frame x- and y-axes, and Q_i is the torque generated by the i-th propeller or rotor. The aerodynamic drag of the main rotorcraft body, represented by the components D_x, D_y, and D_z, has been discussed in some detail in Sect. 26.3.5.

The vector ${}^{B}\boldsymbol{r}_i$ corresponds to the displacement of the i-th rotor or propeller hub from the rotorcraft body frame origin and ${}^{B}\boldsymbol{n}_i$ is the tip-path plane normal of the i-th propeller or rotor disk as explained in [26.37].

The coefficients β_{1c} and β_{1s} represent the longitudinal and lateral flapping coefficients [26.15] describing the tilting of the rotor or propeller disk as elaborated in more detail later in this section. Finally, the parameter k_b^i corresponds to the flapping spring stiffness of the i-th rotor or propeller hub. In the case of articulated or teetering hubs this torsional spring stiffness represents potential flap hinge springs and in case of hingeless hubs approximates the structural bending stiffness of a particular rotor or propeller.

Table 26.6 Typical rotorcraft forces

Type of force	Expression
Gravity	${}^{B}\boldsymbol{F}_{\mathrm{G}} = {}^{B}\boldsymbol{R}_{W}\begin{pmatrix}0\\0\\-mg\end{pmatrix}$
Thrust	${}^{B}\boldsymbol{F}_{\mathrm{T}}^{i} = {}^{B}\boldsymbol{n}_i T_i$
Hub forces	${}^{B}\boldsymbol{F}_{\mathrm{H}}^{i} = \begin{pmatrix}H_x^i\\H_y^i\\0\end{pmatrix}$
Rotorcraft body drag	${}^{B}\boldsymbol{F}_{\mathrm{D}} = \begin{pmatrix}D_x\\D_y\\D_z\end{pmatrix}$

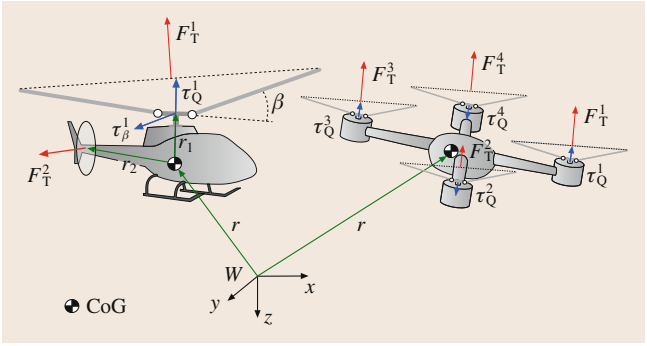


Fig. 26.29 A conventional helicopter and a quadro-copter platform. Flapping angles have been exaggerated and only the rotor and propeller forces relevant near hover have been visualized

In the case of a helicopter tail rotor, flapping is usually neglected and the thrust direction is modeled as, e.g.,

$$ {}^{B}\boldsymbol{n}_i = \begin{pmatrix}0\\1\\0\end{pmatrix} . \tag{26.76}$$

To finalize the models of these external forces and moments, the respective aerodynamic effects as well as the role and characteristics of blade flapping must be discussed next.

26.5.3 Simplified Aerodynamics

In order to accurately predict the forces and moments generated by a rotorcraft system, accounting for various ranges of operation conditions, detailed design specifications, aerodynamic interactions between the different rotors or propellers and possibly the rotorcraft body itself, highly sophisticated aerodynamics simula-

Table 26.7 Typical rotorcraft moments

Type of moment	Expression
Torque	${}^{B}\boldsymbol{\tau}_{\mathrm{Q}}^{i} = \begin{pmatrix}0\\0\\Q_i\end{pmatrix}$
Thrust moment	${}^{B}\boldsymbol{\tau}_{\mathrm{T}} = {}^{B}\boldsymbol{r}^{i} \times {}^{B}\boldsymbol{f}_{\mathrm{T}}^{i}$
Hub moment	${}^{B}\boldsymbol{\tau}_{\mathrm{H}}^{i} = {}^{B}\boldsymbol{r}^{i} \times {}^{B}\boldsymbol{f}_{\mathrm{H}}^{i}$
Flap moment	${}^{B}\boldsymbol{\tau}_{\beta}^{i} = k_b^i \begin{pmatrix}-\beta_{1s}^i\\\beta_{1c}^i\\0\end{pmatrix}$

tion tools are required. Such tools are usually not easily accessible to or operated by the roboticist and may hide some of the fundamental flight dynamics characteristics of a particular platform due to the involved complexity. For a repeated evaluation of the general flight properties of a rotorcraft it may thus be preferable to derive approximate, analytical models of the aerodynamic forces and moments. This may provide profound insight into the core working principles of rotorcraft operation.

The general approach to derive such models is the employment of BET as partially presented in Sect. 26.3.4. BET, as opposed to BEMT or MT, maintains the notion of individual rotor or propeller blades. Hence, it may account for changes in the angle of attack α, the aerodynamic inflow velocities U_T and U_P and in consequence of the local lift and drag increments $\mathrm{d}L$ and $\mathrm{d}D$, not only in function of the radial position r of the observed airfoil segment but also in dependency of the blade azimuth ξ. To further the understanding of this dependency in ξ, one may examine Fig. 26.16 as well as Fig. 26.30 where the hub coordinate frame H is introduced.

The perpendicular and tangential inflow velocities U_P and U_T (and thus also α) are strongly related to the rotor angular speed Ω and the induced inflow velocity v_i (as discussed for BEMT). In flight operation beyond hover, these inflow velocities are also affected by the rotorcraft linear and angular velocities ${}^B\boldsymbol{v}$ and ${}^B\boldsymbol{\omega}$, as well as the potential flapping motion of the rotor or propeller blades. From the perspective of a revolving rotor or propeller blade, these additional inflow velocities vary periodically in function of ξ.

Inspecting Fig. 26.30, one can establish the following relations for the airfoil inflow velocities [26.35]

$$U_\mathrm{T} = \Omega(e+r) - v_y + \beta\omega_x r\,, \tag{26.77}$$

$$U_\mathrm{P} = v_i - w + \omega_y(e+r) + \dot{\beta}r - \beta v_x\,. \tag{26.78}$$

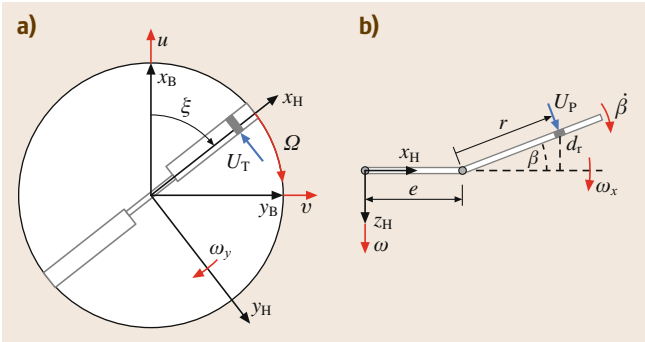


Fig. 26.30 **(a)** Top view of revolving rotor/propeller blade and the relevant velocities. **(b)** Side view of revolving rotor/propeller blade and the relevant velocities

The linear and angular velocities v_x, v_y, ω_x, and ω_y correspond to the projections of the vehicle body velocities u, v, p, and q into the hub frame H

$$v_x = u\cos\xi + v\sin\xi\,, \tag{26.79}$$

$$v_y = v\cos\xi - u\sin\xi\,, \tag{26.80}$$

$$\omega_x = p\cos\xi + q\sin\xi\,, \tag{26.81}$$

$$\omega_y = q\cos\xi - p\sin\xi\,. \tag{26.82}$$

For helicopter-type UAVs the azimuth dependency of α may also be related to periodic changes in the blade pitch angle θ. For helicopter configurations the feathering angle θ can usually be controlled collectively for the entire rotor disk or cyclically in dependency of the blade position ξ using a swashplate mechanism [26.38]

$$\theta = \theta_0 + \theta_{1\mathrm{c}}\cos\xi + \theta_{1\mathrm{s}}\sin\xi\,. \tag{26.83}$$

The swashplate mechanism essentially provides the means to adjust the collective pitch angle θ_0 as well as the cyclic pitch angles $\theta_{1\mathrm{c}}$ and $\theta_{1\mathrm{s}}$ individually. Note that for twisted propeller or rotor blades θ_0 merely represents the pitch angle at the blade root and an additional term accounting for the radial variation in pitch must be introduced [26.14].

By altering θ_0, the swashplate provides some control over the average angle of attack of the entire rotor disk and thus the generated average thrust and torque. Similarly, varying the cyclic components creates a lift imbalance between opposing sides of the rotor disc. This lift imbalance induces a periodic flapping motion of the individual rotor blades (also depending on ξ) which affects the pitch and roll moments $\boldsymbol{\tau}_\beta$ and $\boldsymbol{\tau}_\mathrm{T}$. For helicopter-type UAVs this is the main control mechanism for the pitch and roll attitude dynamics.

For the majority of multicopter vehicles θ_0 is usually fixed and the cyclic angles do not exist ($\theta_{1\mathrm{c}} = \theta_{1\mathrm{s}} = 0$). In this case, attitude control is realized through $\boldsymbol{\tau}_\mathrm{T}$ only by changing the propeller speed Ω_i of the individual propellers.

To describe the aforementioned periodic changes in blade flapping the following representation of the flapping angle β is customary when modeling rotorcraft dynamics [26.39]

$$\beta = \beta_0(t) + \beta_{1\mathrm{c}}(t)\cos\xi(t) + \beta_{1\mathrm{s}}(t)\sin\xi(t)\,. \tag{26.84}$$

The coefficient $\beta_0(t)$ corresponds to the coning angle of the entire rotor or propeller disc, whereas $\beta_{1\mathrm{c}}(t)$ and $\beta_{1\mathrm{s}}(t)$ are the longitudinal and lateral disk tilting angles.

Unlike the blade pitch coefficients in (26.83), the flap coefficients in (26.84) are assumed to be functions of time t which is later exploited when modeling flapping dynamics.

Although it is crucial to capture all these periodic dependencies in ξ, it can be obstructive for the purpose of creating an efficient rotorcraft simulation. Directly simulating the body dynamics (26.45) under the influence of the external forces and moments (26.74) and (26.75) while accounting for all dependencies in ξ, would require the blade motion to be simulated step by step while moving around the rotor or propeller shaft. This can lead to comparatively stiff differential equations which may be hard to simulate efficiently.

An alternative approach is to derive simplified force and torque models by integrating the lift and drag increments along the rotor or propeller radius, summing over the number of blades N_b and averaging around the rotor azimuth as

$$T = \frac{N_\mathrm{b}}{2\pi}\int_0^{2\pi}\int_0^R \mathrm{d}L\,, \tag{26.85}$$

$$Q = \frac{N_\mathrm{b}}{2\pi}\int_0^{2\pi}\int_0^R r\left(\mathrm{d}D + \frac{U_\mathrm{P}}{U_\mathrm{T}}\mathrm{d}L\right)\,, \tag{26.86}$$

$$H_x = \frac{N_\mathrm{b}}{2\pi}\int_0^{2\pi}\int_0^R \sin\xi\left(\mathrm{d}D + \frac{U_\mathrm{P}}{U_\mathrm{T}}\mathrm{d}L\right)\,, \tag{26.87}$$

$$H_y = \frac{N_\mathrm{b}}{2\pi}\int_0^{2\pi}\int_0^R \cos\xi\left(\mathrm{d}D + \frac{U_\mathrm{P}}{U_\mathrm{T}}\mathrm{d}L\right)\,. \tag{26.88}$$

Combining (26.77) to (26.88) with (26.12), (26.13), (26.27), and (26.33) and assuming simple rotor or propeller blade geometries (e.g., linear blade twist) the above expressions can be evaluated algebraically, e.g., using a symbolic computation program. As opposed to BEMT these averaged aerodynamic formulations are capable of accounting for the effects of the periodic changes of inflow velocities such as, e.g., the *dissymmetry of lift* phenomenon [26.15] as well as for the varying blade feathering and flapping angles.

To gain some appreciation of how these averaged models may be useful in providing insight into the core characteristics of rotorcraft operation, a strongly simplified thrust model directly resulting from (26.85) under the assumption of hover and negligible flapping angles is given as an example

$$T = \left(k_{\mathrm{T}1}\theta_0 + k_{\mathrm{T}2}\frac{v_0}{\Omega} + k_{\mathrm{T}3}\right)\Omega^2\,. \tag{26.89}$$

The coefficients k_{T1}, k_{T2}, and k_{T3} essentially depend on aerodynamic and geometric rotor or propeller properties. As one may deduce directly from (26.89), thrust is a linear function of the collective pitch angle θ_0 and the average induced rotor or propeller inflow velocity v_0 which is further defined in the next section.

26.5.4 Nonuniform Inflow

In order to account for the varying induced flow field over the rotor or propeller disk in hover and forward flight, the following approximate induced inflow distribution may be assumed as discussed in [26.14]

$$v_\mathrm{i} = v_0 + \frac{r}{R}(v_{1\mathrm{c}}\cos\xi + v_{1\mathrm{s}}\sin\xi)\,. \tag{26.90}$$

The velocity component v_0 represents the average induced inflow at the center of the modeled rotor respectively propeller disc. It is directly related to the thrust level the rotor or propeller is operating at and may be computed, e.g., based on the iterative method discussed in [26.15]. The inflow coefficients v_{1c} and v_{1s} account for changes in the induced flow field due to lateral rotorcraft flight velocities. Various steady-state models for $v_{1\mathrm{c}}$ and $v_{1\mathrm{s}}$ have been proposed in the past with moderate prediction quality only.

Reference [26.40] presents these models in detail and discusses an alternative method which also accounts for transient effects. Accounting for inflow dynamics, the transient response of v_0, $v_{1\mathrm{c}}$, and $v_{1\mathrm{s}}$ to, e.g., sudden blade pitch changes can be captured and the aforementioned iterative computations may be avoided.

26.5.5 Flapping Dynamics

The dynamics of rotor or propeller flapping corresponds to a second-order differential equation of the flapping angle β which essentially represents an aerodynamically damped oscillator [26.35]

$$\ddot{\beta} = \ddot{\beta}(\xi, \Omega, u, v, w, p, q, \dot{p}, \dot{q}, \beta, \dot{\beta}, \theta)\,. \tag{26.91}$$

Reference [26.41] and references therein elaborate the modeling process required to derive the above differential equation from first principles.

As for the simplified aerodynamic forces and moments, the dependency of (26.91) on rotor azimuth is important but problematic for efficient blade flap simulations. Here, the averaging operation discussed in [26.39] has been employed. This essentially leads to the following differential equation of the flapping coef-

ficients β_0, β_{1c} and β_{1s}

$$A_{\ddot{\beta}}\ddot{\boldsymbol{\beta}} + A_{\dot{\beta}}\dot{\boldsymbol{\beta}} + A_{\beta}\boldsymbol{\beta} = A_{\theta}\boldsymbol{\theta} + A_{\omega}\boldsymbol{\omega} + A_{\upsilon}\boldsymbol{\upsilon} \,,$$

$$\boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_{1c} \\ \beta_{1s} \end{pmatrix} , \qquad \boldsymbol{\upsilon} = \begin{pmatrix} \upsilon_0 - w \\ \upsilon_{1c} \\ \upsilon_{1s} \end{pmatrix} ,$$

$$\boldsymbol{\omega} = \begin{pmatrix} p \\ q \\ \dot{p} \\ \dot{q} \end{pmatrix} , \qquad \boldsymbol{\theta} = \begin{pmatrix} \theta_0 \\ \theta_{1c} \\ \theta_{1s} \end{pmatrix} . \tag{26.92}$$

The involved matrices from $A_{\ddot{\beta}}$ to A_{υ} depend on an abundance of aerodynamic, geometric, inertial, and structural parameters of the modeled rotor or propeller system as well as on the lateral flight velocity of the rotorcraft. The structure of these matrices is thus strongly related to the type of rotor or propeller hub that is being modeled. From the resulting mathematical relations it is for example possible to conclude that for a teetering rotor without hinge springs, the maximal flapping response β to a cyclic pitch input in θ will follow with a 90° phase shift in rotor azimuth [26.15].

Considering the body time constants of most rotorcraft configurations in comparison with the time constants of the above dynamics, one may in general assume that the fast poles in (26.92) are negligible and they mostly represent rotor or propeller vibrations. Accordingly, (26.92) is often reduced to a first-order differential equation by introducing the assumption $\ddot{\boldsymbol{\beta}} = \mathbf{0}$. For very rigid rotors or propellers one may even assume $\dot{\boldsymbol{\beta}} = \mathbf{0}$ and compute the steady-state flapping response only (quasi-steady model). However, this is not the case for all RW-UAS configurations. To what degree higher order effects such as flapping or inflow dynamics have to be accounted for, depends on the level of frequency separation between the body poles versus the poles of the rotor dynamics.

26.5.6 Flight Dynamics Assessment

Assembling the dynamics of the rotorcraft body with the forces and moments defined in the previous section, as well as possibly introducing higher order effects such as blade flapping and inflow dynamics, results in a set of nonlinear differential equations representing the flight characteristics of the modeled platform. In order to inspect these flight characteristics one may implement a flight simulator based on this set of nonlinear differential equations and analyze the simulated flight responses. The linearization process presented in Sect. 26.3.6 may provide additional conclusions and is often followed by a separation into subsystems analyzed individually.

One of the most crucial rotorcraft subsystems corresponds to the pitch and roll attitude dynamics as it is this subsystem which ultimately defines how agile, stable, or accurate a rotorcraft may be able to fly in the horizontal plane of motion. Assuming near hover operation and neglecting higher order dynamics, the linearized open-loop roll dynamics for example, can usually be reduced to a first-order transfer function

$$G(s) = \frac{P(s)}{U(s)} = \frac{K_{\mathrm{p}}}{s - D_{\mathrm{p}}} \,. \tag{26.93}$$

The frequency function $P(s)$ represents the Laplace transform of the rotorcraft roll rate p and $U(s)$ the Laplace transform of a control input affecting the roll subsystem. For a helicopter, this control input commonly corresponds to a cyclic swashplate command and for a quadro-copter to the differential speed between two propellers on opposing sides of its airframe. The abstracted system parameter K_{p} corresponds to the so-called *roll control derivative* and D_{p} to the so-called *roll damping* [26.37]. The control sensitivity K_{p} defines how strong the initial rotorcraft roll acceleration $\dot{p}$ will respond to a control input represented by $U(s)$. The parameter D_{p} defines how well the system's dynamic response will be damped in the following. For hybrid rotorcraft models the corresponding transfer functions are generally of higher order but are equally useful when analyzed with the well-known tools of linear system theory [26.15, 35].

In general, these transfer functions and the resulting metrics such as, e.g., K_{p} and D_{p} can be derived in dependency of a specific subset of physical rotorcraft parameters. Using these simple metrics and including the knowledge that may be gathered from the nonlinear flight simulator, flight performance trends can be assessed depending on the parameters of the UAV configuration under investigation. Such parameters may for example include the body pitch and roll inertia, the location of the body center of gravity or the location of the rotors or propellers. The fundamental understanding gained in this evaluation process is crucial for the development of effective robotic flight systems and the required control laws.

26.6 Flapping Wing Modeling and Design

A variety of animals, from insects to birds, are capable of flight maneuvers which are presently impossible in micro aerial vehicles, such as flying in turbulence or cluttered airspace. Additionally, animals are more maneuverable and can fly longer distances. People have made many attempts at building flapping robots or ornithopters. While several are successful, many either never take off or fly only for a short duration due to their higher complexity or poor design. Until recently, ornithopters represented a niche of flying vehicles. The development of lithium polymer batteries produced a light-weight high-power energy resource to power ornithopters. Among the first successful electric ornithopters were the Caltech and Aerovironment microbats in 1998 [26.42, 43]. Many designs still fail to fly despite the rapidly increasing population building electric ornithopters. A major problem in most designs is an inability to generate enough lift to take off in the first place. This precludes additional flight research, such as maneuverability, flight distance or time. Engineers have believed that flapping wings are essential to further development of micro aerial vehicles since the first electric ornithopters took off and biologists started to understand the aerodynamics of flapping insect wings. The main reason behind this focus is the idea that they are aerodynamically more efficient at the small Reynolds number of insects (10−10 000) when viscosity effects start to dominate airflow.

26.6.1 Aerodynamic Mechanisms

Our understanding of insect aerodynamics provides us with the most detailed model of the aerodynamic function of a flapping wing [26.45]. There is some evidence that wing flexibility can improve aerodynamic performance of a flapping wing by roughly 10% [26.46] if the angle of attack is not optimized for a stiff wing. However, a parametric study using a robot model of an insect wing suggests that wing flexibility does not improve performance if we can optimize angle of attack independently of wing stiffness [26.47]. Ignoring aeroelastic effects that change angle of attack distribution, the key known aerodynamic mechanisms of a flapping wing are [26.45]:

1. A stable leading edge vortex (LEV) that enables the wing to operate at high angles of attack without stall during the quasi-steady mid-stroke phase (Fig. 26.31). During stroke reversal the aerodynamics is not quasi steady. In this phase, five additional affects are thought to be important:
2. *Added mass* effects due to fluid acceleration in response to the reversal.
3. The Wagner effect explaining that changes in vortex strength need time to build-up over a few chord lengths of travel.
4. Rotational lift due to the timing of changes in angle of attack during stroke reversal and its effect on vortex lift through the *Kramer effect*.
5. Wake capture when the wing reverses direction and interacts with the momentum jet of its shed wake.
6. Clap and fling when the wings become close enough to (nearly) touch and air is forced out of the cavity formed by the two wings and sucked back in, which can increase lift [26.48].

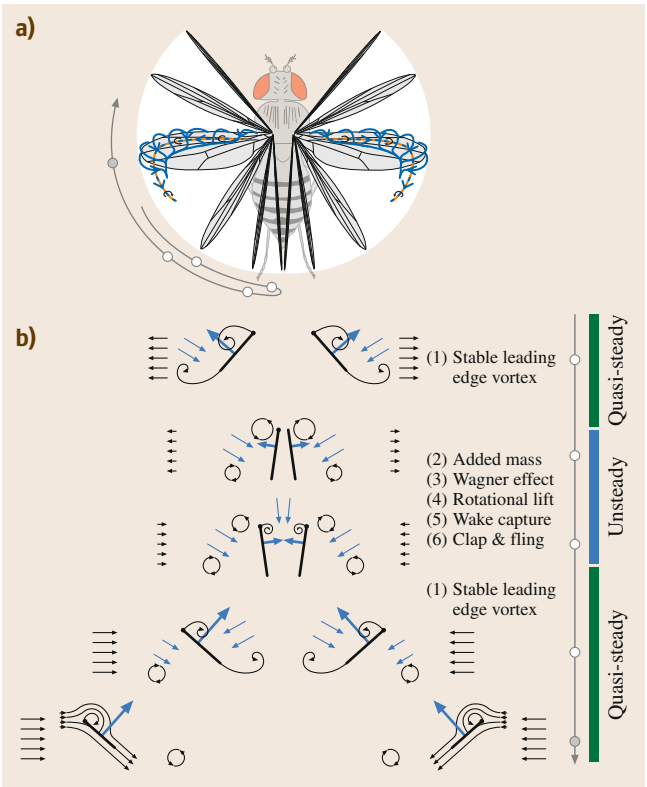


Fig.26.31a,b Flapping insect wing aerodynamics can be understood through the interaction of a myriad of complex aerodynamic mechanisms. **(a)** The key high-lift mechanism insects employ, is a stable leading edge vortex (LEV) generated during the up and downstroke. **(b)** A flapping cycle consists of a quasi-steady part during which the wing accelerates little. During this phase, the stable LEV is the key high-lift mechanism (1). During stroke reversal there is evidence that up to five effects (2)–(6) could be important (after [26.44])

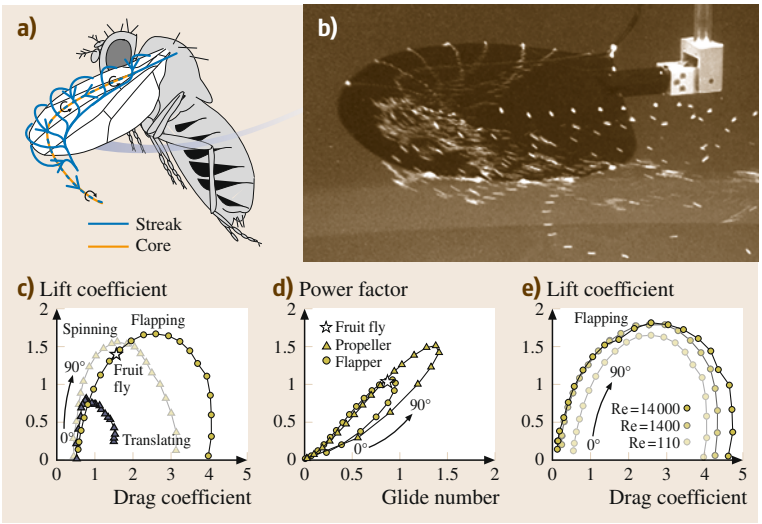
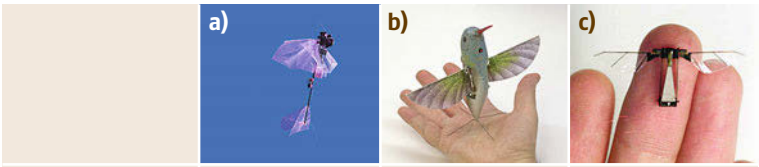


Fig.26.32a–e The aerodynamics of a flapping (insect) wing scale from insect to bird scale. **(a)** A stable LEV enables flapping wings to operate at high angles of attack without stall. **(b)** The key parameter explaining LEV stability is the wing's swing, its spinning motion, as demonstrated by this spinning model of a fly wing which generates a stable LEV and similarly elevated forces as in flapping wings. **(c)** At insect scale fixed (translating) wings underperform, whereas flapping and spinning wings generate similarly high lift. Spinning wings generate less drag which makes them more efficient. **(d)** The power factor of a spinning wing is higher than for a flapping wing, higher indicating that less power is needed to support body weight. **(e)** The dimensionless lift and drag averaged over a full flapping cycle is independent of scale to within good approximation (Reynolds number 110: fruit fly: 1400; house fly: 14 000; hummingbird). This makes flapping wing aerodynamics scalable enabling the use of dimensional analysis [26.49]

There exist, however no quantitative experimental studies or theories that fully dissect these effects and quantify their relative importance for aerodynamic lift and power. While flapping wing aerodynamics is complex and not fully understood, it is simple from a robot design perspective, because it is scalable from insect to bird size (Fig. 26.32). This enables prototyping at larger, more cost effective, scales and enables scaling the design down as technology advances, and smaller components and fabrication methods become available [26.49]. Flapping wings generate more lift than translating wings because they generate a stable LEV. To generate a stable vortex over the whole wing, the aspect ratio with respect to the center of rotation needs to be equal to or smaller than about 4 [26.53]. Flapping wings with an aspect ratio larger than 4 can stall outboard [26.53]; whereas more stubby flapping wings cannot. This can explain why the majority of insect, bird, and bat wings have an aspect ratio of around 2–4 with respect to the *shoulder* joint [26.53]. The



	Delfly II	Nano-Hummingbird	RoboBee
Wingspan (cm)	28	16	3
Mass (g)	16	19	0.06
m/m_0	1.26	1.37	N/A (tethered)
Flight time	15 min.	11 min.	N/A (tethered)
Frequency (Hz)	14	30	110
Mechanism	Gearbox and 4-bar	Gearbox and string rollers	Piezo-electric Elastic 4-bar like
Scale (mm)	10^2–10^0	10^2–10^0	10^2–10^{-1}
Power	1.4 W	3.27 W	N/A (tethered)
Current	380 mA	880 mA	N/A (tethered)

Fig.26.33a–c Examples of three different types of successful flappers. **(a)** Jaap Oldenkamp, Delfly II (after [26.49, 50]), **(b)** Nano Hummingbird (after [26.51]), **(c)** RoboBee (after [26.52])

main advantage of stubby wings is that they do not stall at high angles of attack enabling animals to take-off and land vertically by increasing angle of attack instead of flapping frequency [26.53] using LEVs [26.54]. Insects [26.55], bats, hummingbirds [26.56], and other birds [26.57], but also auto rotating seeds generate stable LEVs. This shows that stable LEVs are a convergent evolutionary solution for high lift at high angle of attack in nature [26.53].

Comparison of flapping versus spinning (propeller-like) insect wings shows spinning insect wings generate similar elevated lift forces by generating a LEV at lower drag. Helicopters with stubby rotors are, therefore, aerodynamically more efficient than stubby flapping wings, because they need less power to fly, as qualitatively presented in Fig. 26.32d [26.49]. This is confirmed experimentally for the most advanced hovering ornithopter at present, the Nano Hummingbird [26.51]. Comparing its flapping wing with a spinning wing showed for various forward speeds that flapping wings require more power for the same lift, in part due to aerodynamics [26.49, 53], and in part due to inertia losses [26.49, 51]. The key advantage of flapping wings seems to be the potential for extreme maneuverability and robustness. For instance, flapping wings may fare better in turbulence, close to the ground, near vertical surfaces and through clutter, when helicopters can become unstable due to stall and complex rotor-wake interactions [26.58].

26.6.2 Sizing New Flappers

An improved understanding of the detailed aerodynamics is scientifically invaluable, but perhaps not critical for designing successful ornithopters at a time when most struggle to take-off. Instead, sizing an ornithopter in terms of gross design parameters such as wing span, weight, and flapping frequency is more critical for take-off. The design methodology introduced here explains how one can transform successful designs to meet other mission perspectives. These designs can then enable flight studies that can advance our understanding of ornithopters versus Ro-UAS and FW-UAS to better appreciate their unique advantages.

Amongst successful flappers, there are three main archetypes as shown in Fig. 26.33. Historically, most flappers have relied on variants of a four-bar mechanism to generate the flapping motion which generates lift. One example of this is the Delfly family of ornithopters, which are capable of both fast forward flying and hover using this approach (VIDEO 493). A recent design which demonstrates both prolonged hovering flight and maneuverability, although lacks the ability to fly fast forward, is the Aerovironment Nano-Hummingbird [26.51]. The Nano-Hummingbird uses a flapping mechanism composed of rollers and strings, while still using a geared down motor to provide power at the right frequency. Additionally, the wings provide control, rather than traditional tail control surfaces. Another more modern development is centimeter scale ornithopters which use piezoelectric actuators to generate flapping motion and control such as the Harvard Robobee [26.52] and the Berkeley Micromechanical Flying Insect. These are capable of tethered flight only, because no batteries exist that can supply high enough power in an enough lightweight package.

Despite the differences in design, these flappers share common trends in parameters, as shown in Fig. 26.34. To design a functional ornithopter, we start with a desired mission such as surveillance, search and rescue, or military applications. The mission determines an appropriate wingspan, and also determines a minimum time for task completion. Figure 26.34 shows that empty weight (mass without battery) follows an exponential pattern with wingspan, especially over the mid-range of wingspans. The main observation is that the power defining scale is not 3, but approximately 1.5. This may be because significant portions of the mass of smaller ornithopters comes from electronics, gearboxes and actuators, whose masses are not dependent on wingspan. Additionally, required flapping frequency decreases with wingspan, enabling an approximation of required flapping frequency based on wingspan that works well for all sizes of ornithopters, as expected using scaling relations.

Using initial design parameters from a successful ornithopter, we can design another ornithopter that is also capable of flapping flight using scaling relationships of geometry, fluid mechanics, and battery physics [26.27]. We need to decide on design parameters for the new flapper, including the wingspan b, weight W, aspect ratio Λ, and battery weight W_{batt}. Here, the aspect ratio is wingspan divided by chord length, as these are both easily measured design parameters. Example of initial parameters for the Delfly II are: $b_1 = 28\,\text{cm}$, $m_1 = 16\,\text{g}$ ($W = \text{mg}$), $\Lambda_1 = 3.5$, $f_1 = 14\,\text{Hz}$, $P_1 = 1.4\,\text{W}$, $W_{\text{batt},1} = 2.7\,\text{g}$, $t_1 = 15\,\text{min}$. Initial design parameters are denoted with subscript 1; while new design parameters are denoted with subscript 2. Using the curve fitted through successful ornithopters as shown in Fig. 26.34, one can make an initial approximation of empty weight. First, we can calculate the wing area, A_{fl}, of the new flapper and the old flapper using the same equation for each

$$A \propto \frac{b^2}{\Lambda} \,. \tag{26.94}$$

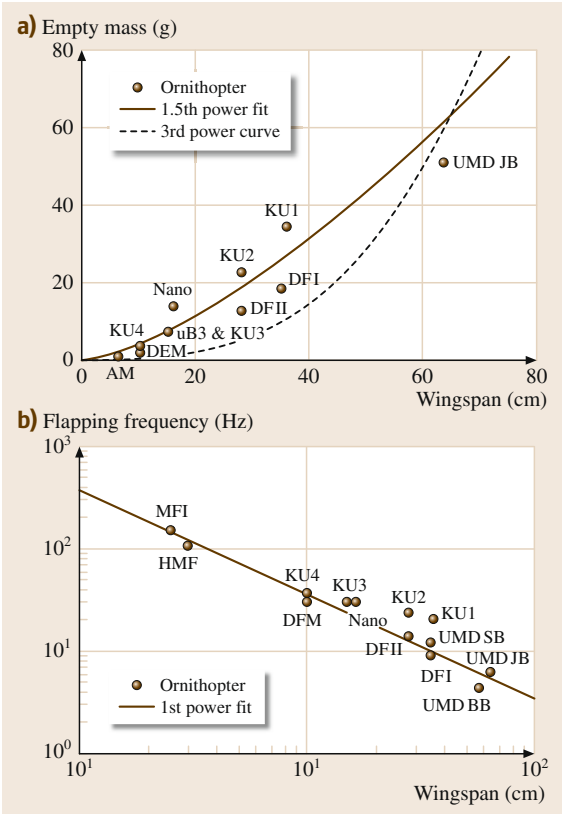


Fig.26.34a,b Current ornithopter trends of empty mass and flapping frequency with changes in wingspan. **(a)** The empty mass of successful ornithopters does not scale with wingspan cubed, but with wingspan to the power 1.5 ($R^2 = 0.79$). The power law predicts the approximate masses effectively in the 10–50 cm wingspan range, while it overestimates the mass for those with wingspans below 10 cm. The curve to the third power consistently underestimates the unloaded masses of current ornithopters. **(b)** To support the weight of the ornithopter, flapping frequency needs to increase inverse to wingspan for smaller wingspans. Ornithopters in **(a)** fly freely and have a flight time of at least one minute. The Micromechanical Flying Insect and Harvard Robobee follow the same trend line for flapping frequency as larger ornithopters; even though they fly tethered (they would need to flap faster with batteries onboard). The relationship here fits a power curve with the exponent equal to −1.01 with $R^2 = 0.96$. Abbreviations are as follows: MFI – Berkeley Micromechanical Flying Insect; HMF – Harvard Robobee; KU1,2,3,4 – Konkuk University ornithopters; DFI,II,M-Delfly I,II and Micro; Nano – Aerovironment Nano-Hummingbird; UMD SB, JB, BB – University of Maryland Small Bird, Big Bird, Jumbo Bird; AM – Brian's Ornithopter; uB3 – NiCad powered Caltech Microbat

In hovering or steady forward flight, it is reasonable to assume that weight is proportional to lift

$$W \propto \frac{1}{2} c_L \rho V_t^2 A_{fl} \,. \tag{26.95}$$

We assume that c_L (lift coefficient), ρ (density,) and g (gravitational acceleration) are constant [26.49], which is reasonable for flights on earth at low altitudes. Then, rearranging produces the following relationship between forward velocities, V_t

$$V_{t,2} = V_{t,1} \sqrt{\frac{W_2}{W_1} \frac{A_{fl,1}}{A_{fl,2}}} \,. \tag{26.96}$$

We can then assume that the advance ratio J is constant for both vehicles, which is a reasonable approximation for ornithopters with similar wing kinematics, shape, and deformation. The advance ratio J is the ratio of maximum forward speed to wingtip speed

$$J = \frac{V_t}{4 f \Phi R} \,. \tag{26.97}$$

Since wingspan is twice the radius, and we can use the assumption that J is constant to obtain the following relationship for flapping frequencies

$$f_2 = \frac{V_{t,2}}{V_{t,1}} \frac{b_1}{b_2} \frac{\Phi_1}{\Phi_2} f_1 \,. \tag{26.98}$$

Then, assuming that flapping amplitude, Φ is constant between the two designs (reasonable for designs that follow the same parameters and keep the same gearboxes) we can simplify the relationship for flapping frequencies

$$f_2 = \frac{V_{t,2}}{V_{t,1}} \frac{b_1}{b_2} f_1 \,. \tag{26.99}$$

The required power to fly is proportional to the weight and flight speed

$$P \propto mgV_t = WV_t \,. \tag{26.100}$$

Thus, we can calculate the power required of the new flapper relative to that of the old flapper

$$P_2 = P_1 \frac{V_{t,2}}{V_{t,1}} \frac{W_2}{W_1} \,. \tag{26.101}$$

Using the power calculated above, the flight time can be estimated as

$$t = \frac{C_{LiPo} U_{LiPo}}{P} m \,, \tag{26.102}$$

in which $U_{\text{LiPo}} = 3.7\,\text{V}$ for a LiPo battery, and where, as in Fig. 26.35, the capacity can be approximated as

$$C_{\text{LiPo}} = m_{\text{batt}} k_{\text{batt}} \,. \tag{26.103}$$

From the scaling equations (particularly (26.101) and (26.102)), we can produce a set of graphs as in Fig. 26.36, allowing us to use the wingspan and flight time to design a scaled ornithopter. Beginning with the approximate wingspan and flight time desired, we use Fig. 26.36a to choose the appropriate battery mass. An increase in wingspan creates the option for heavier batteries and an increase in flight time as does an increase in battery mass. The wingspan and battery mass specify the required flapping frequency. This allows us to choose a motor and gear ratio. If this turns out to be impractical with available components, we can adjust parameters and iterate between the equations shown in Fig. 26.36. In general, for an ornithopter with equal mass, increasing the wingspan decreases the necessary flapping frequency. Alternatively, increasing the battery mass to improve flight time also requires increasing flapping frequency, electric power, and current to carry the extra payload. This explains why increasing battery mass beyond empty weight causes little increase in flight time, because the airframe needs to become much stronger at the cost of weight. A penalty in the flight time scaling equation needs to be implemented to correct for the increase in structural weight. The required flapping frequency and battery mass ratio specify the required power. Power increases significantly with wingspan. Additionally, power increases with added battery mass due to the increase in flapping frequency required to lift the larger mass. Finally, we can determine the current the battery needs to supply, which is proportional to the power assuming we use the same kind of battery and efficiency of motor. Iterating between these steps enables finding solutions that best meet the mission specifications. We note that many ornithopters could fly significantly longer by doubling their current battery mass (Fig. 26.36a) at the expense of control response (inertia) and airframe loading.

If flight time needs to increase for a wingspan-constrained ornithopter design, and battery mass and chemistry is already optimized, we should reduce air-

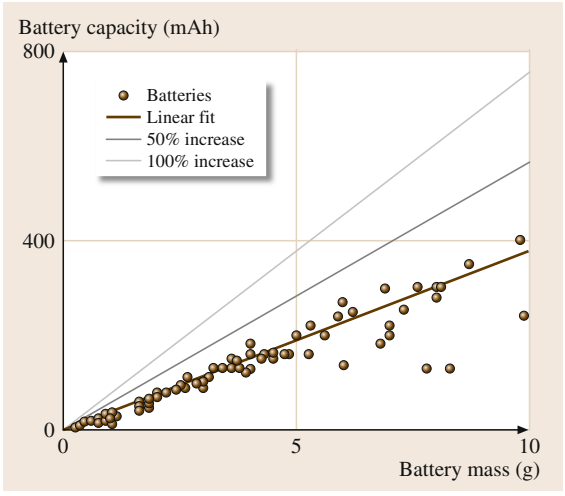


Fig. 26.35 Battery capacity as a function of mass for many small lithium polymer (LiPo) batteries in the size range (<10 g) which would be used for ornithopters with 10–50 cm wingspan. The graph shows the technology is linearly scalable. The approximate capacity density of small LiPo cells (3.7 V) is 37 mAh/g

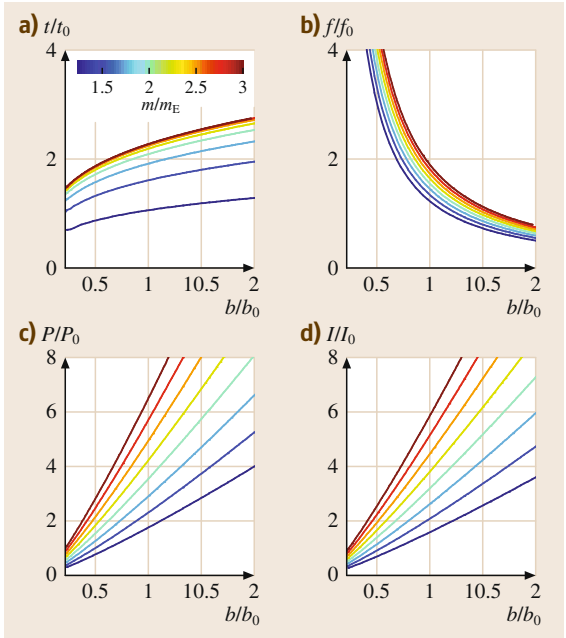


Fig.26.36a–d These four figures show the effects of changing wingspan and adding battery mass to an ornithopter on the flight time, power consumption, current requirement, and flapping frequency requirement. The value of the empty mass, m_e, is determined using the fitted curve in Fig. 26.34a for each wingspan. The figures are then scaled from the initial reference (Delfly II) whose position is at (1, 1) in each figure. **(a)** Increasing the battery mass ratio increases the flight time up until the ratio becomes equal to 3. This ignores additional airframe mass needed to carry these batteries. **(b)** However, increasing the battery mass also increases the required flapping frequency. **(c,d)** Increasing the frequency also increases the necessary power (P) and current (I). Using these parameters, we can iterate back and forth between the plots until a feasible design is found

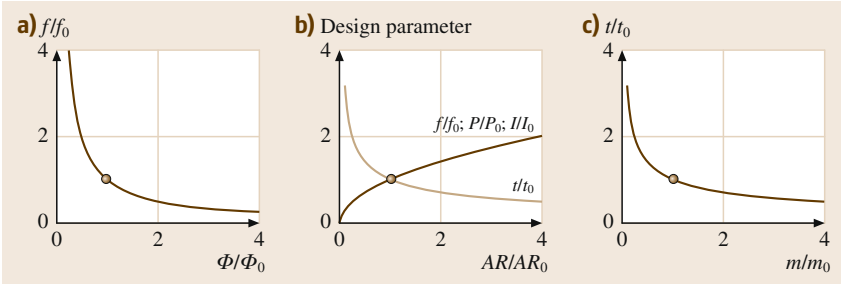


Fig.26.37a–c Changing additional parameters can modify performance of a scaled vehicle. **(a)** Adjusting the flapping amplitude allows the user to change the required flapping frequency to use available motor/gearbox combinations. Generally, larger flapping angles result in increased lift coefficient and decreased drag (after [26.59]). Thus, increasing the amplitude to match it with the motor and gear train can decrease the required power to fly. **(b)** As the aspect ratio increases at a constant wingspan, the wing area decreases, and therefore the flight time decreases while the required flapping frequency (and hence the power and current) increases. **(c)** Flight time decreases with additional payload (weight)

frame mass (Fig. 26.37) and increase wing area [26.49]. Mass can be further decreased by airframe optimization using underutilized aerospace optimization strategies, and by critically reevaluating the payload. Wing area can be increased by decreasing aspect ratio and selecting a biplane instead of a monoplane configuration. Whereas such wing design changes reduce aerodynamic efficiency of the wing, they increase the overall vehicle energy efficiency, and therefore increase flight time. Ornithopters that fly long enough to complete missions are often controlled by low-weight underpowered actuators that sacrifice maneuverability.

To control the ornithopter's flight and to utilize its maneuverability, we need to generate enough control torques with lightweight actuators. Designs optimized for flight time, such as the Delfly, use control surfaces added to the tail in the style of a traditional rudder or elevator. More maneuverable designs use the flapping wings as control surfaces, by changing their angle of attack (Nano-Hummingbird [26.51]) or left versus right wing relative flapping motions (Robobee [26.52]). The two dominant off the shelf actuators are standard servos and magnetic actuators. Standard servos have small electric motors and potentiometers and move to specified positions; while magnetic actuators have a small magnet inside a small coil of wire and apply specified amounts of torque. Magnetic actuators are available at lower masses than servos, which proves critical in optimizing performance of smaller ornithopters. This shows that selecting appropriate actuators involves a tradeoff between flight duration and maneuverability. Ornithopters that are more maneuverable require more powerful and precise servo actuators. The required servo torque of a scaled ornithopter can be estimated assuming isometric scaling: Torque should be proportional to total weight times wingspan, because aerodynamic force is proportional to weight, and arm length to wingspan. Knowing the required torque, we need to find a servo that can provide it. To reduce trial and error we have plotted current servo data to determine how torque correlates with mass to budget for its weight. The data in Fig. 26.38 shows that torque is proportional to mass squared for current servo technology,

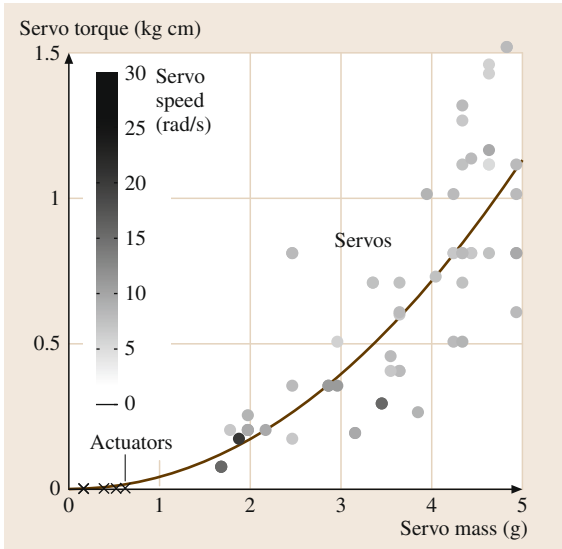


Fig. 26.38 Servo (*dots*) and actuator (*crosses*) torques increases with mass. The intensity of dots represents the servo speed, with darker dots representing faster servos (the magnetic actuators do not have speeds shown, as they apply a force rather than specify a position). The servo speed does not correlate strongly with mass, as it is dependent on the motors, gears, and other internal hardware of the servo, as well as the supply voltage. There are magnetic actuators available in the range of 0.8–1.8 g, they are not included here due to lack of data available from manufacturers

while empty ornithopter mass scales with wingspan to the power of 1.5 (Fig. 26.34), so as wingspan increases the actuator mass can become proportionally smaller.

We have demonstrated current design strategies based off scaling successful designs that ensure ornithopters fly. These upgraded *rules of thumb* are powerful because current aerospace design analysis and optimization techniques for ornithopters lack predictive power and are therefore less informative than estimates based on scaled flying designs. If current designers base their first iteration of new ornithopters on current state-of-the-art ornithopters, the field can progress at a faster pace through successful flight testing of new concepts that meet novel mission criteria.

26.7 System Integration and Realization

Enabling autonomous flights with UAS incorporates solving many challenges. This requires an interdisciplinary approach, bringing together expertise from many different fields. As shown in Fig. 26.39, knowledge in the field of aircraft design, as detailed in this chapter, is required, as well as in many fields of engineering and robotics.

26.7.1 Challenges for Autonomous UAS

Given the agility of UAS and their strict limitations on weight and power consumption, the choice of sensors, processors, and algorithms impose great technical and scientific challenges. Also, major differences exist between ground vehicles and UAS – sensors and algorithms that work well on ground vehicles cannot simply be applied on UAS due to inherent challenges.

Limited Payload and Power Supply

Weight and size restrictions require that lightweight sensors have to be used, usually at the cost of having noisier and less accurate data. These limitations also restrict the choice of onboard computers being used to process information from the sensors. Thus, algorithms need to cope with such data to achieve robust estimates, while having limited computational power. As a reference, the take-off weight of a commonly used small multirotor UAS (e.g., shown in Fig. 26.40) is 1.5 kg, including 0.5 kg of payload, while the flight autonomy is approximately 15 min. For this class of UAS, approximately 100 mW are required per 1 g additional take-off weight for hovering – a fact that has to be considered even for mounting small additional payload. Detailed studies of take-off weight, payload, and flight time of such systems can be found in [26.60].

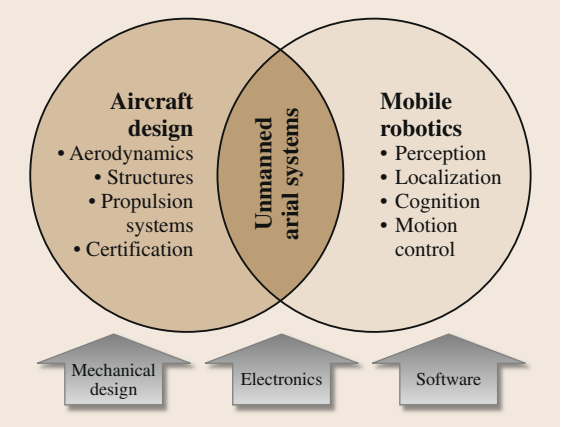


Fig. 26.39 UAS design and research is interdisciplinary as challenges from both aircraft design and mobile robotics are combined

Degrees of Freedom

Compared to typical ground vehicles, there are two additional degrees of freedom (DOF) for the vehicle's attitude (pitch and roll angle) as well as one additional degree of freedom for the altitude, that need to be estimated and controlled. This requires state estimation, control, and planning to be performed in full 6-D space, and without simplifying assumptions.

Under-Actuated Systems

The types of UAS studied here are usually underactuated, as there are less control inputs available than DOF. As a result, the attitude has to be changed for many maneuvers. This in return changes the field of view of onboard sensors interacting with the UAS' environment like cameras, or distance sensors.

Constant Motion and Inherent Instability

UAS cannot simply stop to acquire sensor readings, when state estimation is delayed or contains high uncertainty. While waiting for measurements or re-evaluation of uncertain state estimates, the vehicle continues moving and further falsifies these estimates.

26.7.2 Levels of Autonomy

We classify levels of autonomy and interaction with the pilot or operator into three categories: Manual flight, semiautonomous and autonomous. Industry and the re-

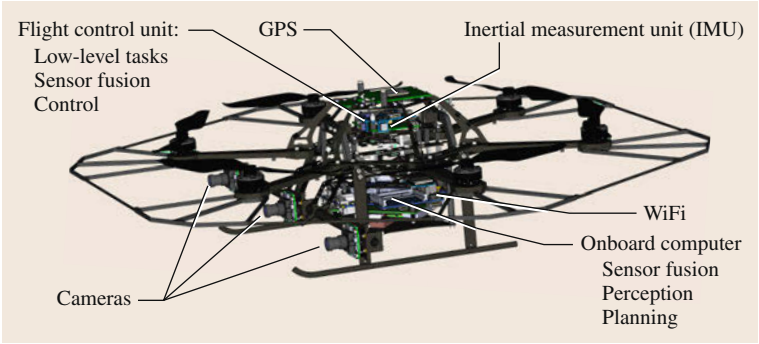


Fig. 26.40 Multi-rotor UAS and its main components (after [26.60])

search community have made great progress toward semiautonomous flights, while there are still many open questions and research topics for truly autonomous operation.

Manual Flight

We refer to this mode when operation of the UAS requires pilot skills. A remote-pilot has to handle attitude dynamics and throttle/thrust in a way such that the UAS remains in a stable state. That is, the pilot cannot leave the hands off the remote control. This also means that line-of-sight has to be maintained up to a distance where the pilot can observe the state of the UAS properly. Piloting may be aided by a stability augmentation system (SAS), like rate stabilization commonly used on fixed wing aircraft, or attitude control commonly used on multirotor UAS.

Semiautonomous (Automatic)

This flight mode does not require piloting skills anymore, which is why we refer to a pilot rather than an operator. Onboard sensors and algorithms are in charge of stabilizing the UAS, such that the operator can leave the hands off the remote control and the UAS remains in a stable state, waiting for input such as waypoints or desired velocities. However, it is still the full responsibility of the operator to find feasible paths, safely navigate around obstacles and to interact with other air traffic. This requires line of sight to be maintained, however, first-person view could be thought of, as long as legal requirements are fulfilled. Examples for such modes are off-the-shelf multirotor UAS [26.61] equipped with GPS sensors or small fixed-wing aircraft [26.62] used for surveying.

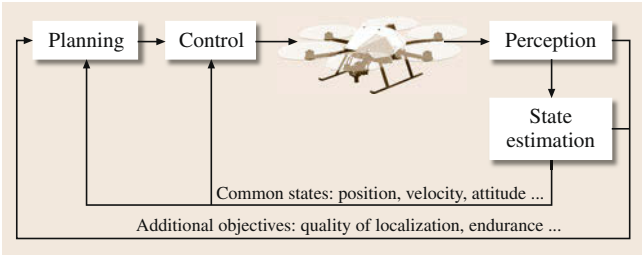


Fig. 26.41 Components for an autonomous UAS

Autonomous

Full autonomous mode relaxes the constraints from semiautonomous or automatic mode. Only mission goals are set, or high-level task allocation is handled by the operator, while the UAS navigates through the environment safely by itself. This includes global path planning, collision avoidance with both static and dynamic objects and replanning where necessary. The main idea is that the UAS can be *left alone* while performing the task at hand, and, in case, warns the operator ahead of time if intervention becomes necessary.

26.7.3 UAS Components

Autonomous UAS require numerous components from various fields, which need to be carefully designed in order to address the challenges detailed above. We highlight important design considerations of building blocks first, and show implementation aspects for real systems. Figure 26.40 shows the components for a commonly used multirotor UAS.

Components for Autonomous Flights

The levels of autonomy defined in Sect. 26.7.2 require several components to be designed and to work with each other. These components are shown in Fig. 26.41 and are described in the following:

Perception and State Estimation. Perception for autonomous UAS mainly involves both localization of the UAS and sensing of its environment. For localization, commonly used satellite-based localization systems such as GPS (GLONASS or GALILEO in the future) may not be accurate enough, especially during tasks that involve operation in close proximity to (man-made) structure, and certainly not indoors. Di-

rect view to satellites is obstructed in these cases, compromising localization accuracy. Also, it may be considered whether sole reliance on these services is acceptable, as authorities may deteriorate the accuracy on purpose (*selected availability*). Thus, additional sensors such as cameras or laser-rangefinders should be used onboard UAS in combination with simultaneous localization and mapping (SLAM), or visual or laser odometry algorithms, yielding additional localization information [26.63–67].

Another important requirement to enable autonomous flights on the levels defined above, is sensing or reconstruction of the environment in order to navigate around obstacles, and for sense-and-avoid maneuvers with other air traffic. Obstacle sensing is especially important when navigating close to terrain at low altitude. Satellite-based navigation in combination with terrain maps may not be accurate enough, outdated, and cannot handle dynamic obstacles. Again, onboard sensors such as cameras, laser-rangefinders, or even radar, in combination with appropriate algorithms need to be used to overcome the limitations mentioned before [26.68, 69].

Having information from a multitude of sensors, from IMU over cameras to satellite-based, intelligent sensor fusion methods are required, taking all information into account and yielding a best estimate of the state of the UAS and its environment. As such, perception and state estimation are building blocks, providing essential information to the components described in the following.

Control. As UAS typically present highly dynamic uncertain flight behavior, appropriate and robust control techniques are required across all levels of autonomy. This ranges from SAS used for supporting the pilot during manual operation, to higher level control such as waypoint following, and more advanced techniques such as trajectory tracking, and may include handling of failure situations. Cascaded control structures are commonly used for this task [26.70]: inner- or low-level control loops stabilize attitude dynamics (these may be cascaded already), while outer- or high-level control loops stabilize translational dynamics (velocity or position). Employed control approaches range from simple P(I)D structures to more advanced techniques like model predictive control (MPC) [26.3]. During waypoint following, specified waypoints have to be reached and via-points have to be passed, while the velocity and attitude profile is left to the controller to optimize. For trajectory tracking in contrast, a tracking controller [26.71] has to follow specific profiles for position and attitude dynamics. These profiles are usually planned with approaches presented below, and yield more smooth paths than simple waypoint following.

Planning. Planning in the context of UAS depicts the process of planning appropriate waypoints or dynamic trajectories for a UAS depending on the mission at hand. This involves the latter two levels of autonomy described in the previous section. Requirements range from (dynamic) obstacle avoidance, taking obstacles and the dynamics of the UAS into account, to more sophisticated complete mission plans [26.72]. While taking into account vehicle dynamics is a straightforward choice, additional constraints such as quality of state estimation along a path, or battery endurance may have to be considered as well. Furthermore, there may be additional objectives like area covered or, for instance, energy optimizations for solar airplanes in order to stay airborne 24 h or longer.

While analytic approaches [26.73] can be used for simple planning tasks, random sampling-based planners [26.74–76] dominate the literature in the area of path planning for UAS, due to their ability to cope with nonlinear vehicle dynamics and high-dimensional state spaces. Based on these planners, a number of successful approaches exist for UAS, even taking into account localization uncertainty [26.4, 77–79].

Communication. Communication can be thought of the *glue* between the components mentioned above. While components have to communicate onboard across multiple computation devices, there is also the need of communication with a ground station, or with the operator remote control device, via radio-links. Here, the requirements in terms of range, delay, and transmission rate (commonly referred to as *bandwidth*) can vary greatly and should be tailored to the application: computationally heavy tasks, being offloaded to a ground station require high-bandwidth connections, where either WiFi (IEEE 802.11n or IEEE draft 802.11ac) or ultra wide band (UWB) techniques can be applied. However, a reasonable maximum range is in the order of 100 m. More autonomy means also less dependence on radio-links and realtime constraints. Missions involving larger distances have to rely on longer range radio-links, even up to satellite links, but at the expense of bandwidth and delay or significant weight and power requirements. That radio-links should never be used inside any (real-time) critical control loops, goes without saying.

Integration Onboard UAS

The components identified above have different requirements in terms of real-time constraints and computational complexity. Instead of real time, it is more

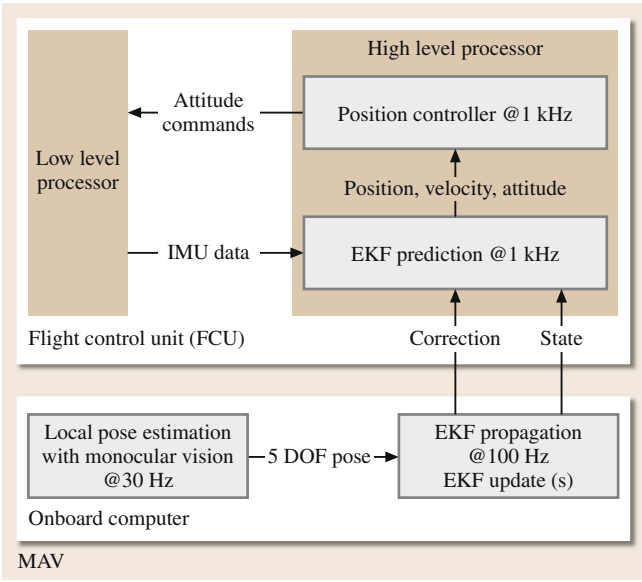


Fig. 26.42 Example implementation of flight-relevant components onboard a multirotor UAS. All time-critical tasks such as control and the prediction part of an extended Kalman filter (EKF) are performed on real-time microcontroller hardware, while computationally more demanding, but less time-critical, tasks (visual localization, EKF update) are processed on the onboard computer using a standard operating system

appropriate to consider that a task has to be completed *in time*. As an example, vehicle *control* tasks have to be computed within a few milliseconds, but these task are computationally less complex. *Path planning* in contrast is fairly complex, but it is usually perfectly fine if a path is computed within seconds, since planning horizons span much longer time. *Perception* and parts of *state estimation* lie in-between and need computation times in the order of tens of milliseconds. This defines how, and on which computation devices these components should be implemented. An example for a multirotor UAS is shown in Fig. 26.42. The microcontrollers on the flight-control-unit (FCU) are closest to inertial sensors and actuators, but least powerful. Control loops and the prediction parts of the state estimator are implemented here, running at guaranteed rates on a real-time operating system, or even without. This relaxes real-time constraints on the onboard computer, which can compute demanding tasks such as visual localization and update steps for state estimation. Local planning tasks are computed here as well, while less critical parts can be offloaded to a ground station.

Application Related Payload

While there is a set of sensors required to enable autonomous navigation, which are referred to as navigation sensors, additional sensors, or payload in general, need to be considered in the design phase. Not only in terms of payload, i. e., weight, but also such that the UAS is kept balanced. Furthermore, it has to be guaranteed that the additional payload does not interfere (electromagnetically) with components in the critical stabilization loop. Applications and their required sensors are detailed in the next section.

26.8 Applications of Aerial Robots

From an application perspective, one may distinguish between UAS that mostly operate as remote controlled or semiautonomous systems (typically referred to as drones) and intelligent systems, robots that present advanced levels of autonomy. Drones are essentially tele-operated aircraft or systems capable of tracking predefined trajectories while they further integrate onboard sensors to provide situational awareness. Most often such situational awareness is visual (using optics) but can also include meteorological and environmental tasks like hurricane-monitoring and chemical plume detection. As such, most drones fly high with a predetermined and structured flight plan and mission profile. Drones will continue to be valuable assets and as featured in daily media headlines world-wide, positively impact both civilian and military missions.

Nowadays, a constant trend is to develop aerial robots of advanced intelligence. Machine cognition, perception and vehicle control algorithms work in concert to perform applications that go beyond just situational awareness. Reaching new levels of autonomy, these robots are designed to handle unforeseen events, interact with their environment, and adapt to a broad range of scenarios. In essence, drones were in their vast majority *passive*, providing *eyes* on scene whereas modern aerial robots tend to become *active*, allowing their users to *engage* the scene, act *autonomously* and possibly *interact* with surroundings. Figure 26.43 presents indicative examples of current and emerging applications.

26.8.1 Demonstrated Applications of UAS

The Handbook's first edition listed eight categories of *possible* applications. Since then, all these applications have been realized, albeit with various levels of ma-

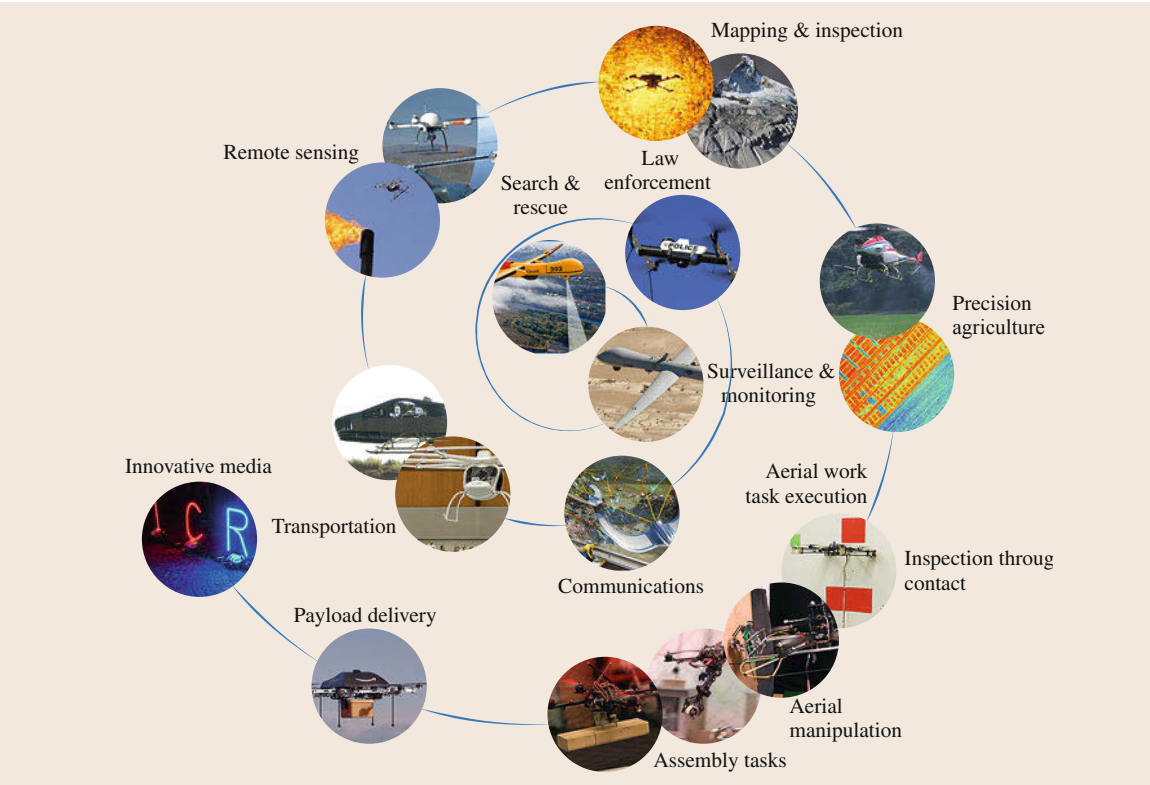


Fig. 26.43 Indicative demonstrated and emerging applications of aerial robotics

turity. UAS (both drones and aerial robots) have been deployed in:

1. Remote sensing
2. Disaster response
3. Surveillance
4. Search and rescue
5. Image acquisition
6. Communications
7. Transportation
8. And payload delivery.

The first five categories broadly fall into the area of reconnaissance, surveillance, and target acquisition (RSTA). Drones have continued to successfully perform RSTA-based tasks like volcanic sampling, damage assessment, border patrol, and cinematography. In contrast, the latter three categories are being realized with advanced aerial robots. Equipped with computational intelligence aerial robots can flock. Aerial robots can also airlift, maneuver and interact in near-Earth environments like in-and-around buildings, forests, caves and tunnels. As such current trends in aerial robotics push the application envelope beyond RSTA. The eight categories are now revisited with context on the expected breakthroughs to come while the topic of visual inspection for industrial purposes is separately discussed:

- *Remote sensing*: Drones are already used for pipeline spotting, power line monitoring, volcanic sampling, mapping, meteorology, geology, agriculture and unexploded mine detection. Advanced aerial robots will be able to conduct pipeline risk assessment and repair, power line maintenance, real-time mapping, crop care and mine defusing.
- *Disaster response*: Drones are also used for chemical sensing, flood monitoring, and wildfire management. Advanced aerial robots will be able to conduct infrastructure repair, flood mitigation, and wildfire fighting.
- *Surveillance*: Drones are employed for law enforcement, traffic monitoring, coastal and maritime patrol, and border patrols. Advanced aerial robots will be able to accomplish tasks like crowd control, traffic redirection, and inspection of maritime and trucking containers.
- *Search and rescue*: The vision of using drones for search and rescue operations within low-density or hard-to-reach areas has already become a reality.

Advanced aerial robots will be able to go beyond casualty extraction by assessing care and delivering first-aid support.

- *Visual inspection*: For quite some time, drones are utilized to provide direct visual feeds to ground operators inspection industrial and civil structures. Future aerial robots will be able to act as a new high fidelity autonomous inspection tools that could also be capable of conducting maintenance work tasks.
- *Transportation*: Drones are already used for small- and large-cargo transport, and possibly passenger transport. Advanced aerial robots will be able to conduct in-flight refueling of other aircraft, pick-up and drop-off of cargo as well as loading and extraction of casualties.
- *Communications*: The use of drones as permanent or ad hoc communication relays for voice and data transmission, as well as broadcast units for television or radio is currently a reality. Advanced aerial robots will be able to conduct perch-and-stare to serve as *bug on the wall* listening devices, perch-and-stare to harvest energy from power lines, flocking to establish networks especially in areas with degraded communications.
- *Payload delivery*: Drones are already employed for firefighting or crop dusting. Advanced aerial robots will be able to accomplish dexterous manipulation of payloads like tools for repairing structures, delivering, and insertion of logistics as well as handling crops.
- *Image acquisition*: Cinematography or real-time entertainment is another relatively established use-case of drones. Advanced aerial robots will be able to conduct acting as *pixels* and operate in formation flight to serve as physical displays in the air.

26.8.2 Current Applications and Missions

Since the Handbook's first edition, media headlines continue to document UAS and their impact in all the aforementioned application categories. No longer are such applications notional but are rather considered as routine. The following provides recent state-of-the-art examples in each category to underscore this message.

Drone usage for RSTA-based military operations is routine. Tracking targets and, in growing instances, destroying them, is commonly performed. But also non-military missions in remote sensing, disaster response, and surveillance are becoming routine. Beyond image acquisition, drones are routinely used to gather meteorological data. NASA and NOAA regularly deploy drones for real-time monitoring of hurricanes. Onboard sulfur dioxide sensors gather airborne samples from volcanic plumes. Drones are frequently deployed after disasters: in Fukushima (2011) and Hurricane Sandy (2012) aerial images were gathered to assess building damage. Monitoring of maritime piracy and cartel drug-trafficking also underscore the usefulness of drones to persistently survey using high-definition cameras and night vision systems.

Currently, drones are deployed to open and/or restricted areas where airborne collision risk is reduced. The need for situational awareness will however continue to be fulfilled by UAS. The demand for more data will push the development of newer drones. As such, well-known and cutting-edge aerodynamic design principles will be applied so that UAS fly longer and farther and incorporate different flight modalities like vertical take-off and landing (VTOL). Sensor suites will also grow in sophistication to collect higher-resolution and/or multispectral data. Propulsion technologies will also be engineered to enable UAS to carry more sensors. The net effect is that as missions become more routine and frequent, the variations of drones and their performance will grow to keep up with expanding RSTA-based needs.

26.8.3 Aerial Robots: Emerging Categories

The road ahead is full of challenging emerging applications that will benefit from the utilization of aerial robotic technologies. This is especially the case when one considers near-Earth environments like in-and-around buildings, through forests and down tunnels and caves. Beyond RSTA, such environments provide opportunities for aerial robots to dexterously interact with objects. Today's drones release retardants over wildfires but tomorrow's aerial robots would attach hoses and breach walls for firefighting. Interaction demands advances in areas like aircraft design (flight modalities, payload capability), algorithms (perception, control, motion planning, and grasping) and manipulators (arms and end-effectors). The latter two areas overlap with the greater domain of robotics research. As deeper understanding and realizations develop, the application space, demand and impact of aerial robots will grow even greater. In the following, a brief overview of exciting emerging applications will be listed.

Assembly Work Task Execution

One notional concept employs aerial robots to assembly physical structures. Such assembly pushes the boundaries of the categories of transportation and payload delivery. Labs like those of University of Pennsylvania [26.80], Switzerland (ETH Zurich [26.81]) have used gripper-mounted quadrotors to illustrate proof-of-concept. Demonstrations include the pickup and drop-off of workpieces to defined locations. These

workpieces self-connect using magnets or premachined joints but illustrate capabilities to coordinate multiple aircraft, mitigate disturbances like ground effect, sense and map, and execute agile maneuvers. Groups at Drexel, Twente, Seville, and Yale are equipping rotorcraft robots with manipulators to illustrate more dexterous tasks like inserting, screwing, and removing workpieces. Assembly and in general physical work-task execution will find applications in categories like disaster response (repairing structures) and remote sensing (inserting sensors).

Inspection Through Contact

Inspecting containers at docks, ships, and border crossings is a daunting task. Sensor technology is advancing to look inside such containers. However, dexterous aerial robots that can open hatches and trunks could also yield many applications. Inspection as an emerging category can employ aerial robots to physically interact with structures. Aerial robots can physically probe bridges, pipelines, and power lines to repair and replace parts. Interactive inspection could emerge to applications like crop handling and mine defusing. Labs like those of ETH Zurich [26.82] and the University of Bologna [26.83] have presented relevant results in the topic.

Innovative Media

Both the number of vendors and affordability of rotorcraft robots have increased since the Handbook's first edition. In mid-2000, it was not surprising to see a commercial quadrotor costing 40 000 USD. Today, there are dozens of companies providing robotic rotorcraft ranging in price (hundreds to under 10 000 USD), size (from pucks to bike wheels) and configurations (mono, co-axial, and multirotor). Such range will likely yield innovative and entrepreneurial applications.

The Firefly project at the MIT SENSEable Lab perhaps best illustrates the notion of coordinating multiple aerial robots to display images. Each aerial robot acts as flying *pixel* and maneuver into position to form 2 or 3-D images. Beyond visual and dynamic art, such capabilities could yield airborne displays for tasks like crowd control, traffic redirection and SOS signals. Airborne displays could also expand creative expression to enhance concerts, advertise at stadiums, and inform audiences.

Autonomous Structural Inspection and 3-D Reconstruction

From the perspective of the role of aerial robotics in accelerating growth, mapping of areas, structural inspection of buildings and infrastructure as well as recognition of objects or areas of interest is among the most important application fields. Aerial robots have achieved great milestones in such scenarios. Among others, recently researchers managed to extract a high-fidelity 3-D model of the Swiss mountain top Matterhorn utilizing a small FW-UAS and sophisticated robotic vision techniques [26.84], other efforts led to the inspection of a real power plant boiler using a multirotor vehicle and tightly integrated visual-inertial algorithms [26.85], while in terms of industrial adoption aerial robots have provenly minimized the inspection times of power infrastructure [26.86] and are often used in combination with geographic information system (GIS) data [26.84]. It is worth noting that some of the largest asset owners and service providers in the field of civil infrastructure and civil engineering works have shown interest and participate in large consortia that aim to make such aerial robotic technologies an integral and game-changing factor of how things are done (see e.g., Petrobot Project [26.87], ARCAS: Aerial Robotics Cooperative Assembly System [26.88], EuRoC: European Robotics Challenges [26.89], ARGOS Challenge [26.90].).

Precision Agriculture

Precision agriculture is considered to be among the fields where the use of aerial robots may become a key factor boosting growth and improving the production quality. The Japanese farming sector has historically been in a position to lead the relevant research efforts with successful designs like the Yamaha RMAX being extensively used for monitoring as well as spraying operations [26.5]. Already in December 2002, 1687 aerial robots were used in Japan to conduct precision agriculture operations. However, current research contributions and pioneering technology early adopters have opened an even more promising channel for the widest possible utilization of aerial robotic technologies for the benefit of agriculture production. Mainly through the use of miniaturized aerial robotics at farms of smaller scale and via the integration of advanced multispectral perception sensors (i. e., NDVI) [26.91], accurate mapping and analysis regarding the quality of the field, the level of plant growth and existence of diseases becomes possible. Such technologies are expected to become a critical tool to optimize and enhance agricultural services. Furthermore, manipulators-equipped aerial robots will be able to autonomously physically act based on their perception of the field and minimize the time and effort required for several agriculture services and tasks.

26.8.4 Open Issues

Emerging categories underscore both the distinction between more traditional drones and emerging aerial

robotic technologies as well as how much further the state-of-the-art has to be pushed. Clearly, today's UAS have limited payloads but adding manipulators goes beyond lift issues. When interacting with objects, aerial robots must handle reaction forces and torques. Understanding such reactions remains an open issue. However, both space and underwater roboticists have looked at dexterous manipulators for tasks like satellite repair and turning valves on ocean-based oil rigs. As research clarifies this issue and as lift capacities increase, dexterous aerial robots will be realized.

Airspace access is an often cited issue that may limit aerial robot development and vast utilization. A US congressional bill was passed in 2012 that sets requirements for UAS to fly in the national airspace by 2015. This has also sparked competition among a dozen or so states to establish UAS flight testing sites. In Europe, similar events have happened and there are UAS test sites in places like Finland (Kemijarvi), Sweden (NEAT), and Wales (Parc Aberporth).

Lastly, somewhat ironic is that today's unmanned drones require a crew of highly skilled operators. In the case of some Predator missions, crew sizes can be up to a dozen people. Also ironic is that human error is the most cited cause for drone accidents. As the number of UAS in the national airspace increases, the need for even more operators will also grow. This has the potential to raise the risk of UAS-related accidents. The issues of effective UAV pilot training, certifying operators, handling emergency landings, and sharing airports with manned aircraft will also emerge as pressing ones.

26.9 Conclusions and Further Reading

Design of aerial robots requires background knowledge in a multitude of subjects, from aerodynamics to dynamics, control, and system integration: we have overviewed the relevant basics along with analytical tools and guidelines to go through the stages of designing, modeling, and setting up operation of various types of unmanned aerial systems (UAS). An emphasis was given on custom tailoring a system to a specific application, in order to optimally meet related requirements in terms of endurance, range, agility, size, complexity, as well as from a system integration point of view. The compilation at hand shall serve as a starting point, further motivating the reader to study the various fields with their related literature, ranging from aircraft and system design to the classical autonomous robotics challenges involving perception, cognition and motion control.

Video-References

VIDEO 493 Delfly II in hover
available from http://handbookofrobotics.org/view-chapter/26/videodetails/493

VIDEO 602 AtlantikSolar field-trials
available from http://handbookofrobotics.org/view-chapter/26/videodetails/602

VIDEO 603 senseSoar UAV Avionics testing
available from http://handbookofrobotics.org/view-chapter/26/videodetails/603

VIDEO 604 Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics
available from http://handbookofrobotics.org/view-chapter/26/videodetails/604

VIDEO 688 sFly: Visual-inertial SLAM for a small helicopter in large outdoor environments
available from http://handbookofrobotics.org/view-chapter/26/videodetails/688

VIDEO 689 UAV stabilization, mapping & obstacle avoidance using VI-sensor
available from http://handbookofrobotics.org/view-chapter/26/videodetails/689

VIDEO 690 Project Skye – autonomous blimp
available from http://handbookofrobotics.org/view-chapter/26/videodetails/690

VIDEO 693 Flight stability in aerial redundant manipulators
available from http://handbookofrobotics.org/view-chapter/26/videodetails/693

VIDEO 694 The astounding athletic power of quadcopters
available from http://handbookofrobotics.org/view-chapter/26/videodetails/694

VIDEO 695 Robots that fly ... and cooperate
available from http://handbookofrobotics.org/view-chapter/26/videodetails/695

VIDEO 696 A robot that flies like a bird
available from http://handbookofrobotics.org/view-chapter/26/videodetails/696

VIDEO 697 Robotic insects make first controlled flight
available from http://handbookofrobotics.org/view-chapter/26/videodetails/697

VIDEO 719 Towards valve turning using a dual-arm aerial manipulator
available from http://handbookofrobotics.org/view-chapter/26/videodetails/719

References

26.1 S. Leutenegger, M. Chli, R.Y. Siegwart: Brisk: Binary robust invariant scalable keypoints, Proc. IEEE Int. Conf. Comput. Vis. (ICCV) (2011) pp. 2548–2555

26.2 D. Scaramuzza, M.C. Achtelik, L. Doitsidis, F. Fraundorfer, E.B. Kosmatopoulos, A. Martinelli, M.W. Achtelik, M. Chli, S.A. Chatzichristofis, L. Kneip, G.H. Lee, S. Lynen, L. Meier, M. Pollefeys, A. Renzaglia, R. Siegwart, J.C. Stumpf, P. Tanskanen, C. Troiani, S. Weiss: Vision-controlled micro flying robots: From system design to autonomous navigation and mapping in GPS-denied environments, IEEE Robotics Autom. Mag. **2014**(9), 1–10 (2014)

26.3 K. Alexis, G. Nikolakopoulos, A. Tzes: Model predictive quadrotor control: Attitude, altitude and position experimental studies, IET Control Theory Appl. **6**(12), 1812–1827 (2012)

26.4 M.W. Achtelik, S. Lynen, S. Weiss, M. Chli, R. Siegwart: Motion and uncertainty aware path planning for micro aerial vehicles, J. Field Robotics **31**(4), 676–698 (2014)

26.5 K. Nonami: *Autonomous Flying Robots: Unmanned Aerial Vehicles and Micro Aerial Vehicles* (Springer, Berlin, Heidelberg 2010)

26.6 H. Tennekes: *The Simple Science of Flight: From Insects to Jumbo Jets* (MIT, Cambridge 2009)

26.7 W.J. Pisano, D.A. Lawrence: Control limitations of small unmanned aerial vehicles in turbulent environments, Proc. AIAA Guid. Navig. Control Conf. (2009)

26.8 B. Mettler, C. Dever, E. Feron: Scaling effects and dynamic characteristics of miniature rotorcraft, J. Guid. Control Dyn. **27**(3), 466–478 (2004)

26.9 ICAO: *Manual of the ICAO Standard Atmosphere: Extended to 80 Kilometres (262 500 feet)* (Int. Civil Aviation Organization (ICAO), Montréal 1993)

26.10 M. Hepparle: JavaFoil – Analysis of java airfoils, http://www.mh-aerotools.de/airfoils/javafoil.htm (2007)

26.11 M. Drela: XFOIL – Subsonic airfoil development system, http://www.web.mit.edu/drela/Public/web/xfoil (2000)

26.12 Techwinder: xlfr5, http://www.xflr5.com (2000)

26.13 B.W. McCormick: *Aerodynamics, Aeronautics, and Flight Mechanics* (Wiley, New York 1979)

26.14 G.J. Leishman: *Principles of Helicopter Aerodynamics* (Cambridge Univ. Press, Cambridge 2006)

26.15 G.D. Padfield: *Helicopter Flight Dynamics* (Blackwell, New York 2007)

26.16 L. Zaccarian: *Dc Motors: Dynamic Model and Control Techniques* (Lecture Notes Univ. Rome, Rome 2012)

26.17 M. Drela: AVL (Software for aerodynamic and flight-dynamic analysis), http://web.mit.edu/drela/Public/web/avl (2004)

26.18 B. Etkin: *Dynamics of Atmospheric Flight* (Wiley, New York 1972)

26.19 G.J.J. Ducard: *Fault-Tolerant Flight Control and Guidance Systems: Practical Methods for Small Unmanned Aerial Vehicles*, Advanced in Industrial Control (Springer, Berlin, Heidelberg 2009)

26.20 M.C.Y. Niu: *Airframe Structural Design* (Conmilit, Hong Kong 1988)

26.21 R. Randolph: *R/C Airplane Building Techniques*, R/C Encyclopedia (Air Age, Wilton 1991)

26.22 D.P. Raymer: *Aircraft Design: A Conceptual Approach* (AIAA, Washington 1989)

26.23 A. Noth: Design of Solar Powered Airplanes for Continuous Flight, Ph.D. Thesis (Ecole Polytechnique Fedenale de Lausanne, Lausanne 2008)

26.24 R.W. Beard, T.W. McLain: *Small Unmanned Aircraft: Theory and Practice* (Princeton Univ. Press, Princeton 2012)

26.25 A.A. Lambregts: *Vertical flight path and speed control autopilot design using total energy principles*, AIAA Paper (AIAA, Washington 1983)

26.26 S. Park, J. Deyst, J.P. How: A new nonlinear guidance logic for trajectory tracking, Proc. AIAA Guid. Navig. Control Conf. (2004) pp. 16–19

26.27 S. Bouabdallah, C. Bermes, S. Grzonka, C. Gimkiewicz, A. Brenzikofer, R. Hahn, D. Schafroth, G. Grisett, W. Burgard, R. Siegwart: Towards palm-size autonomous helicopters, Proc. Int. Conf. Exhib. Unmanned Aer. Veh. (2010)

26.28 P.I.E. Pounds, D.R. Bersak, A.M. Dollar: Grasping from the air: Hovering capture and load stability, Proc. IEEE Conf. Robotics Autom. (ICRA) (2011)

26.29 D. Gurdan, J. Stumpf, M. Achtelik, K.-M. Doth, G. Hirzinger, D. Rus: Energy-efficient autonomous four-rotor flying robot controlled at 1 khz, Proc. IEEE Int. Conf. Robotics Auton. Syst. (2007)

26.30 G.M. Hoffmann, H. Huang, S.L. Waslander, C.J. Tomlin: Quadrotor helicopter flight dynamics and control: Theory and experiment, Proc. AIAA Guid. Navig. Control Conf. (2007)

26.31 A. Ko, O.J. Ohnaian, P. Gelhausen: Ducted fan uav modeling and simulation in preliminary design, Proc. AIAA Model. Simul. Technol. Conf. Exhibit. (2007)

26.32 R. Naldi, F. Forte, L. Marconi: A class of modular aerial robots, Proc. 50th IEEE Conf. Decis. Control Eur. Control Conf. (2011)

26.33 E.R. Ulrich, J.S. Humbert, D.J. Pines: Pitch and heave control of robotic samara micro air vehicles, J. Aircr. **47**, 1290–1299 (2010)

26.34 C.Y. Yun, I. Park, H.Y. Lee, J.S. Jung, I.S. Hwang, S.J. Kim: A new vtol uav cyclocopter with cycloidal blades system, Proc. 60th AHS Annu. Forum Amer. Helicopter Soc. (2004)

26.35 R.W. Prouty: *Helicopter Performance, Stability and Control* (Krieger, New York 2005)

26.36 M.B. Tischler, R.K. Remple: *Aircraft and Rotorcraft System Identification: Engineering Methods with Flight-Test Examples* (AIAA, Washington 2006)

26.37 B. Mettler: *Identification, Modeling and Characteristics of Miniature Rotorcraft* (Kluwer, Boston 2002)

26.38 A.R.S. Bramwell, G. Done, D. Balmford: *Bramwell's Helicopter Dynamics* (Butterworth-Heinemann, London 2001)

26.39 R.T.N. Chen: *Effects of primary rotor parameters on flapping dynamics*, Tech. Rep. (National Aeronautics and Space Administration, Washington 1980)

26.40 R.T.N. Chen: *A survey of nonuniform inflow models of rotorcraft flight dynamics and control applications*, Tech. Rep. (National Aeronautics and Space Administration, Washington 1989)

26.41 R. Cunha: Advanced Motion Control for Autonomous Air Vehicles, Ph.D. Thesis (Instituto Superior Tecnico, Universidade Tecnica de Lisbon, Lisbon 2007)

26.42 T.N. Pornsin-Sirirak, S.W. Lee, H. Nassef, J. Grasmeyer, Y.C. Tai, C.M. Ho, M. Keennon: MEMs wing technology for a battery powered ornithopter, Proc. 13th IEEE Annu. Int. Conf. MEMS (2000) pp. 799–804

26.43 T.N. Pornsin-Sirirak, Y.C. Tai, C.M. Ho, M. Keennon: Microbat: A palm-sized electrically powered ornithopter, Proc. NASA/SPL Workshop Biomorphic Robotics (2001) pp. 14–17

26.44 S.P. Sane: The aerodynamics of insect flight, J. Exp. Biol. **206**(23), 4191–4208 (2003)

26.45 M.H. Dickinson, F.O. Lehmann, S.P. Sane: Wing rotation and the aerodynamic basis of insect flight, Science **284**(5422), 1954–1960 (1999)

26.46 J. Young, S.M. Walker, R.J. Bomphrey, G.K. Taylor, A.L.R. Thomas: Details of insect wing design and deformation enhance aerodynamic function and flight efficiency, Science **325**(5947), 1549–1552 (2009)

26.47 L. Zhao, Q. Huang, X. Deng, S.P. Sane: Aerodynamic effects of flexibility in flapping wings, J. R. Soc. Interface **7**(44), 485–497 (2010)

26.48 F.O. Lehmann, S.P. Sane, M. Dickinson: The aerodynamic effects of wing-wing interaction in flapping insect wings, J. Exp. Biol. **208**(16), 3075–3092 (2005)

26.49 D. Lentink, S.R. Jongerius, N.L. Bradshaw: *Flying Insects and Robots* (Springer, Berlin, Heidelberg 2009)

26.50 G.C.H.E. de Croon, M.A. Groen, C. De Wagter, B. Remes, R. Ruijsink, B.W. van Oudheusden: Design, aerodynamics, and autonomy of the DelFly, Bioinspiration Biomim. **7(2)**, 025003 (2012)

26.51 M. Keennon, K. Klingebiel, H. Won, A. Andriukov: Tailless flapping wing propulsion and control development for the nano hummingbird micro air vehicle, Proc. Am. Helicopter Soc. Futur. Vert. Lift Aircr. Des. Conf. (2012)

26.52 K.Y. Ma, P. Chirarattananon, S.B. Fuller, R.J. Wood: Controlled flight of a biologically inspired, insect-scale robot, Science **340**(6132), 603–607 (2013)

26.53 D. Lentink, M.H. Dickinson: Rotational accelerations stabilize leading edge vortices on revolving fly wings, J. Exp. Biol. **212**(16), 2705–2719 (2009)

26.54 W. Shyy, H. Aono, C.-K. Kang, H. Liu: *An Introduction to Flapping Wing Aerodynamics* (Cambridge Univ. Press, Cambridge 2013)

26.55 C.P. Ellington, C. van den Berg, A.P. Willmott, A.L.R. Thomas: Leading-edge vortices in insect flight, Nature **384**, 626–630 (1996)

26.56 D.R. Warrick, B.W. Tobalske, D. Powers: Lift production in the hovering hummingbird, Proc. R. Soc. Biol. Sci. (2009) pp. 3747–3752

26.57 F.T. Muijres, L.C. Johansson, A. Hedenstrom: Leading edge vortex in a slow-flying passerine, Biol. Lett. **8**(4), 554–557 (2012)

26.58 J. Koo, T. Oka: *Experimental Study on the Ground Effect of a Model Helicopter Rotor in Hovering*, Tech. Rep. (NASA, Washington 1966)

26.59 S.P. Sane, M.H. Dickinson: The control of flight force by a applying wing: Lift and drag production, J. Exp. Biol. **204**(15), 2607–2626 (2001)

26.60 M.C. Achtelik, K.-M. Doth, D. Gurdan, J. Stumpf: Design of a multi rotor MAV with regard to efficiency, dynamics and redundancy, Proc. AIAA Guid. Navig. Control Conf. (2012)

26.61 Ascending Technologies Ltd.: http://www.asctec.de (2015)

26.62 Sensefly (Parrot Company): http://www.sensefly.com (2000)

26.63 M.W. Achtelik: Advanced Closed Loop Visual Navigation for Micro Aerial Vehicles, Ph.D. Thesis (ETH Zurich, Zurich 2014)

26.64 A. Bachrach, S. Prentice, R. He, N. Roy: RANGE – Robust autonomous navigation in GPS-denied environments, J. Field Robotics **28**, 644–666 (2011)

26.65 S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, R. Siegwart: Keyframe-based visual-inertial slam using nonlinear optimization, Proc. Robotics Sci. Syst. (RSS) (2013)

26.66 S. Weiss: Vision Based Navigation for Micro Helicopters, Ph.D. Thesis (ETH Zurich, Zurich 2012)

26.67 A.I. Mourikis, S.I. Roumeliotis, J.W. Burdick: SC-KF mobile robot localization: A stochastic cloning kalman filter for processing relative-state measurements, IEEE Trans. Robotics **23**(4), 717–730 (2007)

26.68 A. Bachrach, S. Prentice, R. He, P. Henry, A.S. Huang, M. Krainin, D. Maturana, D. Fox, N. Roy: Estimation, planning, and mapping for autonomous flight using an RGB-D camera in GPS-denied environments, Int. J. Robotics Res. **31**, 1320–1343 (2012)

26.69 T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. Grixa, F. Ruess, M. Suppa, D. Burschka: Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue, IEEE Robotics Autom. Mag. **19**(3), 46–56 (2012)

26.70 S. Bouabdallah: Design and Control of Quadrotors with Application to Autonomous Flying, Ph.D. Thesis (STI School of Engineering, EPFL, Lausann 2007)

26.71 T. Lee, M. Leoky, N.H. McClamroch: Geometric tracking control of a quadrotor UAV on SE(3), Proc. 49th IEEE Conf. Dec. Control (CDC) (2010) pp. 5420–5425

26.72 P. Doherty, J. Kvarnström, F. Heintz: A temporal logic-based planning and execution monitoring framework for unmanned aircraft systems, Auton. Agents Multi-Agent Syst. **19**(3), 332–377 (2009)

26.73 P.E. Hart, N.J. Nilsson, B. Raphael: A formal basis for the heuristic determination of minimum cost paths, Trans. Syst. Sci. Cybern. **4**(2), 100–107 (1968)

26.74 S. Karaman, E. Frazzoli: Incremental sampling-based algorithms for optimal motion planning, Proc. Robotics Sci. Syst. (RSS), Zaragoza (2010)

26.75 L.E. Kavraki, P. Švestka, J.-C. Latombe, M.H. Overmars: Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robotics Autom. **12**(4), 566–580 (1996)

26.76 S.M. LaValle, J.J. Kuffner: Randomized kinodynamic planning, Int. J. Robotics Res. **20**(5), 378–400 (2001)

26.77 A. Bry, N. Roy: Rapidly-exploring random belief trees for motion planning under uncertainty, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 723–730

26.78 H. Cover, S. Choudhury, S. Scherer, S. Singh: Sparse tangential network (SPARTAN): Motion planning for micro aerial vehicles, IEEE Proc. Int. Conf. Robotics Autom. (ICRA) (2013)

26.79 R. He, S. Prentice, N. Roy: Planning in information space for a quadrotor helicopter in a gps-denied environments, IEEE Proc. Int. Conf. Robotics Autom. (ICRA) (2008) pp. 1814–1820

26.80 Q. Lindsey, D. Mellinger, V. Kumar: Construction with quadrotor teams, Auton. Robots **33**(3), 323–336 (2012)

26.81 J. Willmann, F. Augugliaro, T. Cadalbert, R. D'Andrea, F. Gramazio, M. Kohler: Aerial robotic construction towards a new field of architectural research, Int. J. Archt. Comput. **10**(3), 439–460 (2012)

26.82 G. Darivianakis, K. Alexis, M. Burri, R. Siegwart: Hybrid predictive control for aerial robotic physical interaction towards inspection operations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

26.83 L. Marconi, R. Naldi, L. Gentili: Modelling and control of a flying robot interacting with the environment, Automatica **47**(12), 2571–2583 (2011)

26.84 Z. Lin: UAV for mapping – low altitude photogrammetric survey, Proc. 21st ISPRS Congr. Techn. Commis. I, Beijing (2008) pp. 1183–1186

26.85 J. Nikolic, M. Burri, J. Rehder, S. Leutenegger, C. Huerzeler, R. Siegwart: A UAV system for inspection of industrial facilities, Proc. IEEE Aerosp. Conf. (2013) pp. 1–8

26.86 Cyberhawk: Aerial Inspection and Suervying Specialists, http://www.thecyberhawk.com (2015)

26.87 Petrobot Project: http://www.petrobotproject.eu/

26.88 ARCAS: Aerial Robotics Cooperative Assembly System, http://www.arcas-project.eu/

26.89 EuRoC: European Robotics Challenges, http://www.euroc-project.eu/

26.90 ARGOS Challenge: http://www.argos-challenge.com/

26.91 E.R. Hunt Jr., W.D. Hively, S.J. Fujikawa, D.S. Linden, C.S.T. Daughtry, G.W. McCarty: Acquisition of nir-green-blue digital photographs from unmanned aircraft for crop monitoring, Remote Sens. **2**, 290–305 (2010)

Multimedia Contents

27. Micro-/Nanorobots

Bradley J. Nelson, Lixin Dong, Fumihito Arai

The field of microrobotics covers the robotic manipulation of objects with dimensions in the millimeter to micron range as well as the design and fabrication of autonomous robotic agents that fall within this size range. Nanorobotics is defined in the same way only for dimensions smaller than a micron. With the ability to position and orient objects with micron- and nanometer-scale dimensions, manipulation at each of these scales is a promising way to enable the assembly of micro- and nanosystems, including micro- and nanorobots.

This chapter overviews the state of the art of both micro- and nanorobotics, outlines scaling effects, actuation, and sensing and fabrication at these scales, and focuses on micro- and nanorobotic manipulation systems and their application in microassembly, biotechnology, and the construction and characterization of micro and nanoelectromechanical systems (MEMS/NEMS). Material science, biotechnology, and micro- and nanoelectronics will also benefit from advances in these areas of robotics.

27.1 **Overview of Micro- and Nanorobotics** ... 671
27.2 **Scaling** ... 674
27.2.1 The Size of Things ... 674
27.2.2 Predominate Physics at the Micro- and Nanoscales ... 674
27.3 **Actuation at the Micro- and Nanoscales** 675
27.3.1 Electrostatics ... 675
27.3.2 Electromagnetics ... 676
27.3.3 Piezoelectrics ... 676
27.3.4 Other Techniques ... 676
27.4 **Imaging at the Micro- and Nanoscales** ... 676
27.4.1 Optical Microscopy ... 677
27.4.2 Electron Microscopy ... 677
27.4.3 Scanning Probe Microscopy ... 678
27.5 **Fabrication** ... 678
27.5.1 Microfabrication ... 679
27.5.2 Nanofabrication ... 680
27.6 **Microassembly** ... 681
27.6.1 Automated Microassembly Systems ... 681
27.6.2 Microassembly System Design ... 683
27.6.3 Basic Microassembly Techniques ... 685
27.7 **Microrobotics** ... 687
27.7.1 Introduction ... 687
27.7.2 Bio-Microrobotics ... 689
27.7.3 Bio-Mimetic/Untethered Microrobots ... 690
27.8 **Nanorobotics** ... 692
27.8.1 Introduction ... 692
27.8.2 Nanorobotic Manipulation Systems ... 695
27.8.3 Nanorobotic Manipulation and Assembly ... 696
27.8.4 Nanorobotic Systems ... 702
27.9 **Conclusions** ... 704
Video-References ... 704
References ... 705

27.1 Overview of Micro- and Nanorobotics

Progress in robotics over recent years has dramatically extended our ability to explore, perceive, understand, and manipulate the world on a variety of scales extending from the edges of the solar system, to the bottom

of the sea, down to individual atoms (Fig. 27.1). At the lower end of this scale, technology has been moving toward greater control of the structure of matter, suggesting the feasibility of achieving thorough control of the molecular structure of matter atom by atom, as *Feynman* first proposed in 1959 in his prophetic article on miniaturization [27.1]:

> *What I want to talk about is the problem of manipulating and controlling things on a small scale... I am not afraid to consider the final question as to whether, ultimately – in the great future – we can arrange the atoms the way we want: the very atoms, all the way down!*

He asserted that

> *At the atomic level, we have new kinds of forces and new kinds of possibilities, new kinds of effects. The problems of manufacture and reproduction of materials will be quite different. The principles of physics, as far as I can see, do not speak against the possibility of maneuvering things atom by atom.*

This technology is now labeled *nanotechnology*.

The *great future* of Feynman began to be realized in the 1980s. Some of the capabilities he dreamed of have been demonstrated, while others are being actively pursued. Feynman foresaw the possibility of employing a microrobotic manipulator (*a master–slave system*) for bottom-up manufacturing (manipulation, assembly, etc.) of minute machines; one such device he described as *swallowing the surgeon*, which he attributed to his friend Albert R. Hibbs. He also imagined we could *build a billion tiny factories, models of each other, which are manufacturing simultaneously, drilling holes, stamping parts, and so on* [27.1]. Micro- and nanorobotics research has progressed from these seemingly *far-out* concepts of the 1960s and 1970s to reality when microelectromechanical systems (MEMSs) began to emerge in the late 1980s. These building blocks took the form of surface-micromachined micromotors and microgrippers made of polysilicon fabricated on a silicon chip [27.2]. In the late 1980s and early 1990s, more concrete suggestions on how one could realize MEMS-based microrobotic devices using such micromotors as well as potential applications were published [27.3, 4]. Today, a variety of microrobotic devices are enabling new applications in various fields.

In industry, interesting areas for microrobotics include assembly [27.5, 6], characterization, inspection and maintenance [27.7, 8], microoptics (positioning of microoptical chips, microlenses and prisms) [27.9], and microfactories [27.10]. Many of these applications require automated handling and assembly of small parts with accuracy in the submicron range.

Other important fields include biology (manipulation, capturing, sorting and combining cells [27.11]) and medical technology [27.12, 13]. In surgery, the use of steerable catheters and endoscopes is very attractive and the development of increasingly small microrobotic devices is rapidly progressing. Wireless untethered microrobots that will explore and repair our bodies (*swallowing the surgeon*) appear to be simply a matter of time. In fact, endoscopy using wireless capsules (camera pills) are already on the market and allow for endoscopic imaging of the entire gastrointestinal tract [27.14], something currently not possible using standard scopes. Magnetic steering or crawling-type motions serve as promising ways for such devices to locomote in a controlled fashion [27.15]. Doctors could steer pill-mounted cameras and other actuators to areas of interest for visual investigation and biopsies beyond the range of current endoscopes.

Nanorobotics represents the next stage in miniaturization for maneuvering nanoscale objects. Nanorobotics is the study of robotics at the nanometer scale, and includes robots that are nanoscale in size, i. e., nanorobots, and large robots capable of manipulating objects that have nanometer dimensions with nanometer resolution, i. e., nanorobotic manipulators. The field of nanorobotics brings together several disciplines, including nanofabrication processes used for producing nanoscale robots, nanoactuators, nanosensors, and physical modeling at nanoscales. Nanorobotic manipulation technologies, including the assembly of nanometer-sized parts, the manipulation of biological cells or molecules, and the types of robots used to perform these types of tasks also form a component of nanorobotics.

As the 21st century unfolds, the impact of nanotechnology on the health, wealth, and security of humankind is expected to be at least as significant as the combined influences in the 20th century of antibiotics, the integrated circuit, and human-made polymers. For example, N. Lane stated in 1998, [27.16]

> *If I were asked for an area of science and engineering that will most likely produce the breakthroughs of tomorrow, I would point to nanoscale science and engineering.*

The great scientific and technological opportunities nanotechnology presents have stimulated extensive exploration of the nanoworld and initiated an exciting worldwide competition, which has been accelerated by the publication of the *National Nanotechnology Initiative* by the US government in 2000 [27.17]. Nanorobotics will play a significant role as an enabling nanotechnology and could ultimately be a core part of

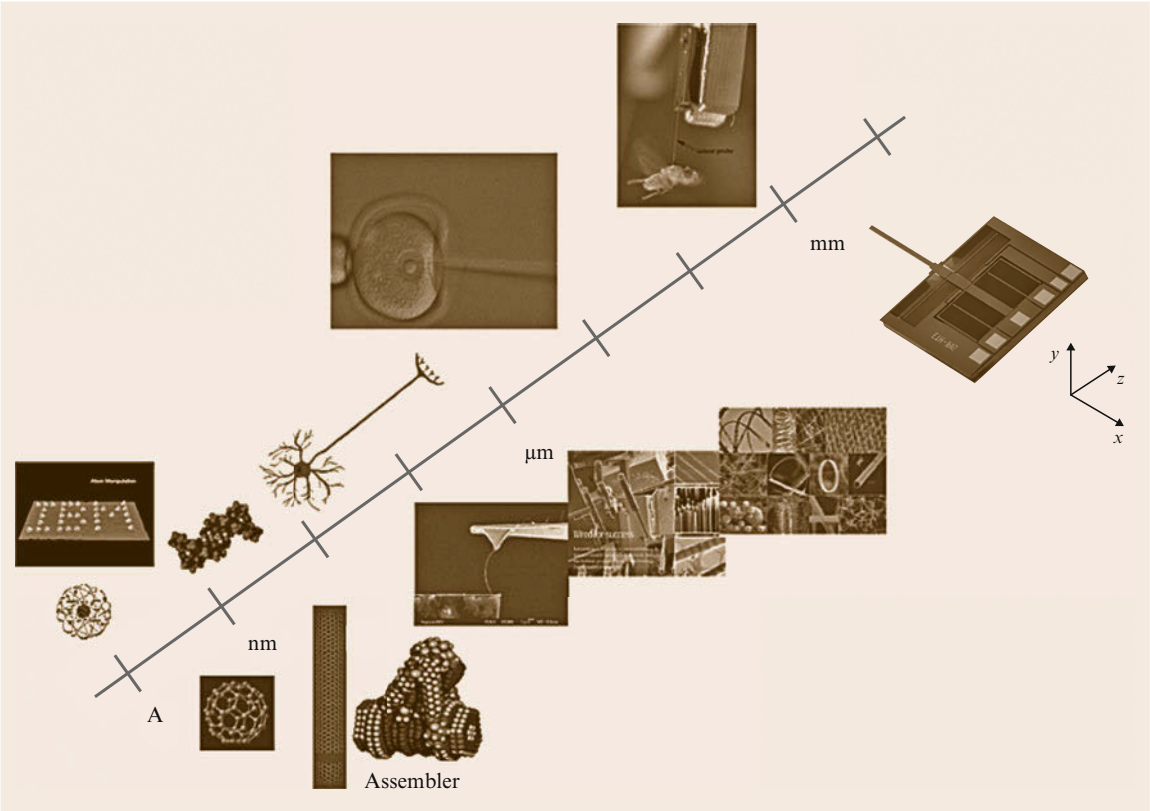


Fig. 27.1 Robotic exploration at micro- and nanometer scales

nanotechnology if *Drexler*'s machine-phase nanosystems based on self-replicative molecular assemblers via mechanosynthesis can be realized [27.18].

By the early 1980s, scanning tunneling microscopes (STMs) [27.19] radically changed the way in which we interacted with and even regarded single atoms and molecules. The very nature of proximal probe methods encourages the exploration of the nanoworld beyond conventional microscopic imaging. Scanning probes now allow us to perform *engineering* operations on single molecules, atoms, and bonds, thereby providing a tool that operates at the ultimate limits of fabrication. They have also enabled exploration of molecular properties on an individual nonstatistical basis.

STMs and other nanomanipulators are nonmolecular machines that use bottom-up strategies. Although performing only one molecular reaction at a time is obviously impractical for making large amounts of a product, it is a promising way to provide the next generation of nanomanipulators. Most important, it is possible to realize the directed assembly of molecules or supermolecules to build larger nanostructures through nanomanipulation. The products produced by nanomanipulation could be the first step of a bottom-up strategy in which these assembled products are used to self-assemble into nanomachines.

One of the most important applications of nanorobotic manipulation will be nanorobotic assembly. However, it appears that, until assemblers capable of replication can be built, the combination of chemical synthesis and self-assembly are necessary when starting from atoms; groups of molecules can self-assemble quickly due to their thermal motion, enabling them to *explore* their environments and find (and bind to) complementary molecules. Given their key role in natural molecular machines, proteins are obvious candidates for early work in self-assembling artificial molecular systems. *Degrado* [27.20] demonstrated the feasibility of designing protein chains that predictably fold into solid molecular objects. Progress is also being made in artificial enzymes and other relatively small molecules that perform functions like those of natural proteins. Several bottom-up strategies using self-assembly appear feasible [27.21]. Chemical synthesis, self-assembly, and supramolecular chemistry make it possible to provide building blocks at relatively large sizes beginning from the nanometer scale. Nanorobotic manipulation serves as the base for a hybrid approach to

construct nanodevices by structuring these materials to obtain building blocks and assembling them into more complex systems.

Despite the claims of many *futurists*, the form nanorobots of the future will take and what tasks they will actually perform remain unclear. However, it is clear that nanotechnology is progressing towards the construction of intelligent sensors, actuators, and systems that are smaller than 100 nm. These nanoelectromechanical systems (NEMSs) will serve as both the tools to be used for fabricating future nanorobots as well as the components from which these nanorobots may be developed. Shrinking device size to these dimensions presents many fascinating opportunities such as manipulating nano-objects with nanotools, measuring mass in femtogram ranges, sensing forces at piconewton scales, and inducing gigahertz motion, among other new possibilities waiting to be discovered. These capabilities will, of course, drive the tasks that future nanorobots constructed by and with NEMS will perform. NEMS and the components of nanorobots will be the products of nanorobotic manipulation. Large nanorobotic manipulators will be able to shrink in size due to this development, thus enabling nanosized robotic manipulators and other forms of nanorobots. All of these form the scope of the area of nanorobotics.

This chapter focuses on micro- and nanorobotics including actuation, manipulation, and assembly at the micro- and nanoscale. The main goal of these fields of robotics is to provide an effective technology for the experimental exploration of the micro- and nanoworld, and to push the boundaries of this exploration from a robotics research perspective.

27.2 Scaling

27.2.1 The Size of Things

Things we can potentially observable range from 10^{-35} m (the Planck length) to 10^{26} m (the radius of the observable universe). A nanometer, 10^{-9} m, is about ten times the size of the smallest atoms, such as hydrogen and carbon, while a micron is barely larger than the average wavelength of visible light, thus invisible to the human eye. A millimeter, the size of a pinhead, is roughly the smallest part typically fabricated using traditional machining techniques. The range of scales from millimeters to nanometers is one million (Fig. 27.1), which is also about the range of scales in present-day mechanical technology from the largest skyscrapers to the smallest conventional mechanical machine parts. The vast opportunity to make new machines spanning almost six orders of magnitude from 1 mm to 1 nm, is one take on *Feynman*'s famous statement, *there is plenty of room at the bottom* [27.1]. If L is taken as a typical length, 0.1 nm for an atom, perhaps 2 m for a human, this scale range in L would be 2×10^{10}. If the same scale range were to apply to an area, 0.1 nm × 0.1 nm versus 2 m × 2 m, the scale range for area L^2 is 4×10^{20}. Since a volume L^3 is enclosed by sides L, we can see that the number of atoms of size 0.1 nm in a (2 m^3) volume is about 8×10^{30}, recalling that Avogadro's number $N_A = 6.022 \times 10^{23}$ is the number of atoms in a gram-mole, supposing that the atoms were ^{12}C, molar responding to a density 1.99×10^4 kg/m^3. A primary working tool of the nanotechnologist is facility in scaling the magnitudes of various properties of interest, as the length scale L shrinks, e.g., from 1 mm to 1 nm.

Clearly, the number of atoms in a device scales as L^3. If a transistor on the micron scale contains 10^{12} atoms, then on the nanometer scale, $L'/L = 10^{-3}$ it will contain 1000 atoms, likely too few to preserve its function.

Normally, we will think of scaling as an isotropic scale reduction in three dimensions. However, scaling can be thought of usefully when applied only to one or two dimensions, scaling a cube to a two-dimensional (2-D) sheet of thickness a or to a one-dimensional (1-D) tube or nanowire of cross-sectional area a^2. The term *zero-dimensional* (0-D) is used to describe an object small in all three dimensions, having volume a^3. In electronics, a zero-dimensional object (a nanometer-sized cube a^3 of semiconductor) is called a quantum dot (QD) or *artificial atom* because its electronic states are few and sharply separated in energy, and thus resemble the electronic states of an atom.

27.2.2 Predominate Physics at the Micro- and Nanoscales

The predominate physics at the micro- and nanoscale can be dramatically different from at the macroscale. Surface and intermolecular forces, such as adhesion forces originating from surface tension forces, van der Waals forces, and electrostatic forces, become more significant than volumetric forces such as gravitational forces for objects with sizes well below 1000 μm [27.22]. Although the laws of classical Newtonian physics may well suffice to describe changes in behavior down to 10 nm (100 Å), the range of scaling

is tremendous. Therefore, the changes in magnitudes of many important physical properties, such as resonant frequencies, are so great, that completely different applications may appear.

The more challenging question for the nanotechnologist is to understand and hopefully to exploit those changes in physical behavior that occur at the end of the classical scaling range. The *end of the scaling* is the size scale of atoms and molecules, where nanophysics [27.23] is the proven conceptual replacement of the laws of classical physics. Modern physics, which includes quantum mechanics as a description of matter on a nanometer scale, is a very well-developed and proven subject whose application to real situations is limited only by modeling and computational competence.

In the modern era, simulations and approximate solutions increasingly facilitate the application of nanophysics to almost any problem of interest. Many central problems are already (adequately, or more than adequately) solved in the extensive literatures of theoretical chemistry, biophysics, condensed matter physics, and semiconductor device physics. The practical problem is to find the relevant work, and, frequently, to convert the notation and units systems to apply the results to the problem at hand.

27.3 Actuation at the Micro- and Nanoscales

The positioning of nanorobots and nanorobotic manipulators depends largely on nanoactuators. While nanosized actuators for nanorobots are still under exploration and relatively far from implementation, MEMS-based efforts are focused on shrinking their sizes [27.24]. Nanometer resolution motion has been extensively investigated and can be generated using various actuation principles. Electrostatics, electromagnetics, and piezoelectrics are the most common ways to realize actuation at nanoscales. For nanorobotic manipulation, besides nanoresolution and compact sizes, actuators generating large strokes and high forces are best suited for such applications. The speed criteria are of less importance as long as the actuation speed is in the range of a couple of hertz and above. Table 27.1 provides a small selection of early works on actuators [27.25–30] suitable for micro- and nanorobotic applications (partially adapted from [27.24]).

Several extensive reviews on various actuation principles have been published [27.4, 31–34]. During the design of an actuator, the trade-offs among range of motion, force, speed (actuation frequency), power consumption, control accuracy, system reliability, robustness, load capacity, etc. must be taken into consideration. This section reviews basic actuation technologies and potential applications at nanometer scales.

27.3.1 Electrostatics

Electrostatic charge arises from a build up or deficit of free electrons in a material, which can exert an attractive force on oppositely charged objects, or a repulsive force on similarly charged objects. Since electrostatic fields arise and disappear rapidly, such devices will likewise demonstrate very fast operation speeds and be little affected by ambient temperatures.

Previous investigations have produced many examples of miniature devices using electrostatic force for actuation including silicon micromotors [27.35, 36], microvalves [27.37], and microtweezers [27.38]. This type of actuation is important for achieving nanoscale actuation.

Electrostatic fields can exert great forces, but generally across very short distances. When the electric field must act over larger distances, a higher voltage will be required to maintain a given force. The extremely low current consumption associated with electrostatic devices makes for highly efficient actuation.

Table 27.1 Actuation with MEMS

Actuation principle	Type of motion	Volume (mm^3)	Speed (s^{-1})	Force (N)	Stroke (m)	Resolution (m)	Power density (W/m^3)	Reference
Electrostatic	Linear	400	5000	1×10^{-7}	6×10^{-6}	NA	200	[27.25]
Magnetic	Linear	$0.4 \times 0.4 \times 0.5$	1000	2.6×10^{-6}	1×10^{-4}	NA	3000	[27.26]
Piezoelectric	Linear	$25.4 \times 12.7 \times 1.6$	4000	350	1×10^{-3}	7×10^{-8}	NA	[27.27]
Actuation principle	**Type of motion**	**Volume (mm^3)**	**Speed (rad/s)**	**Torque (Nm)**	**Stroke (rad)**	**Resolution (rad)**	**Power density (W/m^3)**	**Reference**
Electrostatic	Rotational	$\pi/4 \times 0.5^2 \times 3$	40	2×10^{-7}	2π	NA	900	[27.28]
Magnetic	Rotational	$2 \times 3.7 \times 0.5$	150	1×10^{-6}	2π	$5/36\pi$	3000	[27.29]
Piezoelectric	Rotational	$\pi/4 \times 1.5^2 \times 0.5$	30	2×10^{-11}	0.7	NA	NA	[27.30]

Table 27.2 Comparison of nano actuators

Method	Efficiency	Speed	Power density
Electrostatic	Very high	Fast	Low
Electromagnetic	High	Fast	High
Piezoelectric	Very high	Fast	High
Thermomechanical	Very high	Medium	Medium
Phase change	Very high	Medium	High
Shape memory	Low	Medium	Very high
Magnetostrictive	Medium	Fast	Very high
Electrorheological	Medium	Medium	Medium
Electrohydrodynamic	Medium	Medium	Low
Diamagnetism	High	Fast	High

27.3.2 Electromagnetics

Electromagnetism arises from electric current moving through a conducting material. Attractive or repulsive forces are generated adjacent to the conductor and proportional to the current flow. Structures can be built which gather and focus electromagnetic forces, and harness these forces to create motion.

Electromagnetic fields arise and disappear rapidly, thus permitting devices with very fast operation speeds. Since electromagnetic fields can exist over a wide range of temperatures, performance is primarily limited by the properties of the materials used in constructing the actuator.

One example of a microfabricated electromagnetic actuator is a microvalve which uses a small electromagnetic coil wrapped around a silicon micromachined valve structure [27.39]. The downward scalability of electromagnetic actuators into the micro- and nanorealm may be limited by the difficulty of fabricating small electromagnetic coils. Furthermore, most electromagnetic devices require perpendicularity between the current conductor and the moving element, presenting a difficulty for planar fabrication techniques commonly used to make silicon devices.

An important advantage of electromagnetic devices is their high efficiency in converting electrical energy into mechanical work. This translates into less current consumption from the power source.

27.3.3 Piezoelectrics

Piezoelectric motion arises from the dimensional changes generated in certain crystalline materials when subjected to an electric field or to an electric charge. Structures can be built which gather and focus the force of the dimensional changes, and harness them to create motion. Typical piezoelectric materials include quartz (SiO_2), lead zirconate titanate (PZT), lithium niobate, and polymers such as polyvinylidene fluoride (PVDF).

Piezoelectric materials respond very quickly to changes in voltages and with great repeatability. They can be used to generate precise motions with repeatable oscillations, as in quartz timing crystals used in many electronic devices. Piezoelectric materials can also act as sensors, converting tension or compression strains to voltages.

On the microscale, piezoelectric materials have been used in linear inchworm drive devices and micropumps [27.40]. STMs and most nanomanipulators use piezoelectric actuators.

Piezoelectric materials operate with high force and speed, and return to a neutral position when unpowered. They exhibit very small strokes (under 1 %). Alternating electric currents produce oscillations in the piezoelectric material, and operation at the sample's fundamental resonant frequency produces the largest elongation and highest power efficiency [27.41]. Piezoelectric actuators working in the *stick–slip* mode can provide millimeter to centimeter strokes. Most commercially available nanomanipulators adopt this type of actuators, such as Picomotors from New Focus and Nanomotors from Klock.

27.3.4 Other Techniques

Other techniques include thermomechanical, phase change, shape memory, magnetostrictive, electrorheological, electrohydrodynamic, diamagnetism, magnetohydrodynamic, shape changing, polymers, and biological methods (living tissues, muscle cells, etc.). Table 27.2 compares these techniques.

27.4 Imaging at the Micro- and Nanoscales

A brief overview is given of common imaging tools used in the research area of micro- and nanorobotics including optical, electron, and scanning probe microscopy. The application and integration of these tools for micro- and nanorobotics are discussed in separate sections.

In selecting a proper tool for imaging/sensing at these scales, the following factors should be first considered:

1. Specimen: size, conductivity, and environment compatibility are among the most important aspects

Table 27.3 Comparison of optical microscopes (OM) and electron microscopes

Feature	OM	TEM	SEM
General use	Surface morphology and sections (1–40 μm)	Section (40–150 nm) or small particles on thin membranes	Surface morphology
Source of illumination	Visible light	High-speed electrons	High-speed electrons
Best resolution	ca. 200 nm	ca. 0.2 nm	ca. 3–6 nm
Magnification range	10–1000×	500–500.000×	20–150.000×
Depth of field	0.002–0.05 nm (N.A. 1.5)	0.004–0.006 mm (N.A. 10^{-3})	0.003–1 mm
Lens type	Glass	Electromagnetic	Electromagnetic
Image ray-formation spot	On eye by lenses	On phosphorescent plate by lenses	On cathode tube by scanning device

to consider. For example, in vivo bioapplications generally require air or liquid, so lower-resolution, light microscopy should be the first choice. If higher resolution is needed, atomic force microscopy or scanning near-field optical microscopy can be used.

2. Resolution: the ability to see fine details of a specimen. Once you can resolve fine details then you can magnify them. Every microscope has a finite resolution; if you magnify objects beyond the resolution the result will be empty magnification. Roughly speaking, a light microscope cannot provide a resolution better than 200 nm. The best commercially available scanning electron microscopes (SEM) have approximately 1 nm resolution, while transmission electron microscopes (TEM) can achieve approximately 0.2 nm, and a scanning tunneling microscope (STM) working in ultrahigh vacuum under very low temperature can resolve atomic-level structures. A comparison of optical microscopy and electron microscopy is given in Table 27.3.
3. Depth of field: the range of depth that a specimen is in acceptable focus. A microscope that has a small depth of field will have to be continuously focused up and down to view a thick specimen.
4. Contrast: the ratio between dark and light. Typically, most microscopes use absorption contrast, i. e., the specimen is subjected to stains in order to be seen. This is called bright-field microscopy. There are other types of microscope that use more exotic means to generate contrast, such as phase contrast, dark field, and differential interference contrast.
5. Brightness: the amount of light. The higher a microscope magnifies the more light will be required. The illumination source should also be at a wavelength (color) that will facilitate interaction with the specimen. All microscopes fall into either of two categories based on how the specimen is illuminated. In the typical compound microscope the light passes through the specimen and is collected by the image forming optics. This is called diascopic illumination. Dissecting (stereo) microscopes generally use episcopic illumination for use with opaque specimen. The light is reflected onto the specimen and then into the objective lens.

27.4.1 Optical Microscopy

Since their invention in the late 1500s, light microscopes have enhanced our knowledge in basic biology, biomedical research, medical diagnostics, and materials science. Light microscopes can magnify objects up to 1000 times, revealing microscopic details. Light-microscopy technology has evolved far beyond the first microscopes of Robert Hooke and Antoni van Leeuwenhoek. Special techniques and optics have been developed to reveal the structures and biochemistry of living cells. Most optical microscopes in current use are known as compound microscopes, where a magnified image of an object is produced by the objective lens, and this image is magnified by a second lens system (the ocular or eyepiece) for viewing. Microscopes have even entered the digital age, using charge-coupled devices (CCDs) and digital cameras to capture images.

The development of modern microscopy has enabled a large family of optical microscopes. For special purposes, other types of optical microscopes can be selected. These include phase-contrast microscopy, fluorescence microscopy, confocal scanning optical microscopy, and deconvolution microscopy image reconstruction.

27.4.2 Electron Microscopy

Scanning Electron Microscope (SEM)

Since the commercial availability of the SEM in 1966, it has been a valuable resource for viewing samples at a much higher resolution and depth of field than the typical optical microscope. Conventional SEMs can resolve down to the nanometer scale (≈ 1 nm) whereas an optical microscope can only resolve down to approximately 200 nm [27.42]. Unlike conventional optical microscopes, SEMs have a high depth of field which gives imaged samples a three-dimensional appearance.

Early SEMs were limited to viewing conductive samples. However, many of today's SEMs can image nonconductive samples in addition to conductive samples using variable-pressure chambers.

Transmission Electron Microscope (TEM)

The TEM can resolve to an atomic scale down to about 1 d.h. (i.e., 0.1 nm). The TEM mode of operation is similar to that of the SEM in that both microscopes contain an electron gun source of illumination. However, the TEM detects the electrons that pass through a given sample. As a result, the electron gun of the TEM operates at higher energy levels between 50–1000 kV, while the SEM's electron gun operates at around 1–30 kV. In order for proper imaging to take place, the sample must be very thin so that electrons from the beam can pass through the specimen. Electrons that do not pass through the sample cannot be detected. Unlike the SEM, the TEM produces images that are two dimensional in appearance.

27.4.3 Scanning Probe Microscopy

Scanning Tunneling Microscope (STM)

Similar to the TEM, the STM [27.19] can also resolve specimens down to the atomic scale. The scanning probe of the STM is comprised of a noble metal sharpened to an atomic-sized tip, which is mounted on a piezoelectrically driven (x, y, z) linear stage. The STM makes use of the quantum-mechanical effect known as tunneling. Electron tunneling occurs when electrons, driven by a small potential difference, flow across the gap between the probe tip and sample. This event takes place at Ångström-scale distances between the probe tip and the sample [27.43]. The tunneling current, which is typically on the order of a few nanoamperes, is directly related to the tip–sample separation distance. Thus, the tunneling current can be measured and is kept at a constant value by controlling the tip–sample gap distance (z) with a feedback control system. The probe tip is then scanned (x, y) along the entire surface of the sample. Since the control system maintains a constant tunneling current, and thus maintains a constant tip–sample distance (z), the result of a scan yields a $z(x, y)$ terrain map of the sample with enough resolution to detect atomic-scale features. The STM can achieve faster imaging by operating in so-called constant-height mode in which the probe tip is scanned in a plane parallel to the average surface portion. The tip–sample distance (z) can then be inferred directly from the measured tunneling current [27.43].

Atomic Force Microscope (AFM)

The AFM [27.44] is considered to be a spin-off of the STM. One shortcoming of the STM is that it requires conductive probe tips and samples to work properly. The AFM was developed in order to view nonconductive samples, giving it a wider applicability than the STM. In addition to imaging nonconductive samples, the AFM can also image samples immersed in liquid, which is useful for biological applications [27.45]. Although the STM and AFM are similar in that they both scan the surface of a sample with an atomically sharp probe, they operate under slightly different principles. The AFM is based on interatomic forces as opposed to the electron tunneling used in the STM. The AFM probe tip is mounted on the end of a microscale cantilever beam. At very short separations, the forces between atoms in the probe tip and atoms in the sample cause the cantilever to deflect. This deflection is usually measured by striking the back of the cantilever with a laser. The reflection of the laser beam hits a photodetector, which can be used to recover the deflection of the cantilever. The force can then be calculated by using Hooke's law, which simply relates the applied force to the stiffness and deflection of a material. The forces that are measured can be on the scale of piconewtons [27.43]. The AFM has three main modes of operation known as contact mode, noncontact mode, and tapping mode.

Unlike the SEM and TEM, both the STM and AFM do not require a vacuum environment in order to function. However, a high vacuum is advantageous in order to keep the samples from becoming contaminated from the surrounding environment as well as controlling humidity. In addition, atomic resolution in air is hardly possible with an AFM due to humidity. As a result of humidity, a water film is formed and creates problems because of capillary forces. This can be resolved by operating in vacuum environment or completely in a liquid solution [27.43].

27.5 Fabrication

The design of micro- and nanorobotic devices is inextricably linked to available fabrication techniques. However, though the development of microfabrication processes has become somewhat stable over the past decade, nanofabrication processes are still being actively pursued, and the design constraints generated by these processes are relatively unexplored. This section will briefly highlight the processes used in conventional

microfabrication including lithography, thin-film deposition, chemical etching, and electrodeposition, and describe some emerging techniques for nanofabrication.

Most micro- and nanofabrication techniques have their roots in the standard fabrication methods developed for the semiconductor industry [27.46–48]. Therefore, a clear understanding of these techniques is necessary for anyone embarking on a research and development path in the micro/nano area.

27.5.1 Microfabrication

In this section, we will discuss the major microfabrication techniques used most frequently in the manufacturing of microstructures.

Photolithography

Lithography is the technique used to transfer a computer-generated pattern onto a substrate (silicon, glass, GaAs, etc.). This pattern is subsequently used to etch an underlying thin film (oxide, nitride, etc.) for various purposes (doping, etching, etc.). Although photolithography, i. e., lithography using an ultraviolet (UV) light source, is by far the most widely used lithography technique in microelectronic fabrication, electron-beam (e-beam) and X-ray lithography are two alternatives that have attracted considerable attention in the MEMS and nanofabrication areas. We will discuss photolithography in this section and postpone the discussion of e-beam and X-ray techniques to the subsequent sections dealing with nanofabrication.

The starting point following the creation of the computer layout for a specific fabrication sequence is the generation of a photomask. This involves a sequence of photographic processes (using optical or e-beam pattern generators) that results in a glass plate having the desired pattern in the form of a thin (≈ 100 nm) chromium layer. Following the generation of photomask, the lithography and etching process can proceed as shown in Fig. 27.2. After depositing the desired material on the substrate, the photolithography process starts with spin-coating the substrate with a photoresist. This is a polymeric photosensitive material that can be spun onto the wafer in liquid form; usually an adhesion promoter such as hexamethyldisilazane (HMDS) is used prior to the application of the resist. The spinning speed and photoresist viscosity will determine the final resist thickness, which is typically 0.5–2.5 μm. Two different kinds of photoresist are available: positive and negative. With a positive resist, the UV-exposed areas will be dissolved in the subsequent development stage, whereas with a negative photoresist, the exposed areas will remain intact after the development. After spinning the photoresist on the wafer, the substrate is soft-baked (5–30 min at 60–100 °C in order to remove the solvents from the resist and improve the adhesion. Subsequently, the mask is aligned to the wafer and the photoresist is exposed to a UV source.

After exposure, the photoresist is developed in a process similar to the development of photographic film. The resist is subsequently hard-baked (20–30 min at 120–180 °C in order to further improve the adhesion. The hard-bake step concludes the photolithography sequence by creating the desired pattern on the wafer. Next, the underlying thin film is etched, and the photoresist is stripped in acetone or other organic solvent. Figure 27.3 shows a schematic of the photolithography steps with a positive photoresist.

Thin-Film Deposition and Doping

Thin-film deposition and doping are used extensively in micro- and nanofabrication technologies. Most of the fabricated structures contain materials other than that of the substrate, which are obtained by various deposition techniques, or by modification of the substrate. These techniques include oxidation, doping, chemical vapor deposition (CVD), physical vapor deposition (PVD), and electroplating.

Etching and Substrate Removal

For micro- and nanofabrication, in addition to thin film etching, often the substrate (silicon, glass, GaAs, etc.) needs to be removed in order to create various mechanical structures (beams, plates, etc.). Two important figures of merit for any etching process are selectivity and directionality.

Selectivity is the degree to which the etchant can differentiate between the masking layer and the layer to

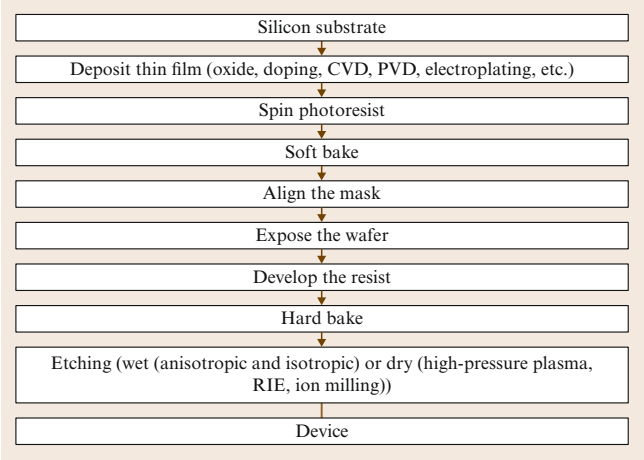


Fig. 27.2 Typical microfabrication process flow

be etched. Directionality has to do with the etch profile under the mask. In an isotropic etch, the etchant attacks the material in all directions at the same rate, creating a semicircular profile under the mask (Fig. 27.4a). In an anisotropic etch, the dissolution rate depends on specific directions, and one can obtain straight sidewalls or other noncircular profiles (Fig. 27.4b). One can also divide the various etching techniques into wet and dry categories. Due to the lateral undercut, the minimum feature size achievable with wet etchants is limited to $> 3\,\mu m$. Photoresist and silicon nitride are the two most common masking materials for the wet oxide etch. Anisotropic and isotropic wet etching of crystalline (silicon and gallium arsenide) and noncrystalline (glass) substrates are important topics in micro- and nanofabrication [27.49–53].

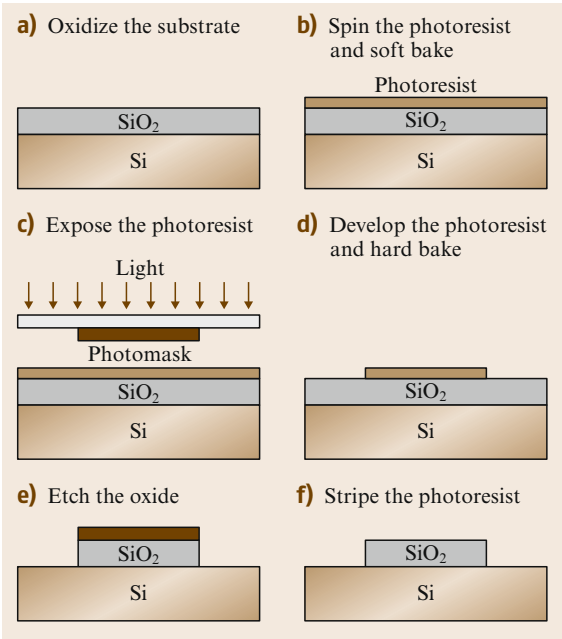


Fig.27.3a–f Schematic drawing of the photolithographic steps with a positive photoresist (PR) **(a)** Oxidize the substrat **(b)** spin the photoresist and soft bake **(c)** expose the photoresist **(d)** develop the photoresist and hard bake **(e)** etch the oxide **(f)** stripe the photoresist

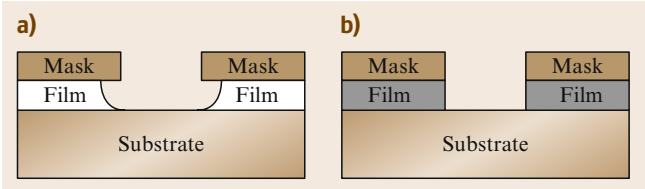


Fig.27.4a,b Profile for isotropic **(a)** and anisotropic **(b)** etch through a photoresist mask

The anisotropic behavior of these etchants with respect to the (111) planes have been used extensively to create beams, membranes, and other mechanical and structural components. Figure 27.5 shows the typical cross sections of (100) silicon wafers etched with an anisotropic wet etchant. As can be seen, the (111) slow planes are exposed and creating 54.7° sloped sidewalls. Depending on the dimensions of the mask opening, a V-groove or a trapezoidal trench is formed in the (100) wafer. A large enough opening will allow the silicon to be etched all the way through the wafer, thus creating a thin dielectric membrane of the other side. It should be mentioned that exposed convex corners have a higher etch rate than the concave ones, resulting in an undercut that can be used to create dielectric (e.g., nitride) cantilever beams.

Dry etching techniques are largely plasma based. They have several advantages compared with wet etching. These include smaller undercut (allowing smaller lines to be patterned) and higher anisotropy (allowing high-aspect-ratio vertical structures). However, the selectivity of dry etching techniques is lower than the wet etchants, and one must take into account the finite etch rate of the masking materials. The three basic dry etching techniques, namely high-pressure plasma etching, reactive-ion etching (RIE), and ion milling, utilize different mechanisms to obtain directionality.

27.5.2 Nanofabrication

The design and fabrication of NEMS is an emerging area being pursued by an increasing number of researchers. Two approaches to nanofabrication, top-down and bottom-up, have been identified by the nanotechnology research community and are being independently investigated by various researchers. Top-down approaches are based on microfabrication and include technologies such as nanolithography, nanoimprinting, and chemical etching. Presently, these are 2-D fabrication processes with relatively low resolution. Bottom-up strategies are assembly-based techniques. Currently these strategies include techniques such as self-assembly, dip-pen lithography, and directed self-assembly. These techniques can generate regular nanopatterns at large scales.

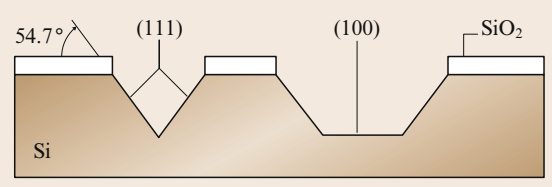


Fig. 27.5 Anisotropic etch profiles for (100) silicon wafers

In this section, we will discuss three major nanofabrication techniques. These include:

1. e-beam and *nanoimprint* fabrication
2. *Epitaxy* and *strain engineering*, and
3. Dip-pen nanolithography.

E-beam Lithography and Nanoimprint Fabrication

In previous sections, we discussed several important lithography techniques commonly used in MEMS and microfabrication. These include various forms of UV (regular, deep, and extreme) and X-ray lithography. However, due to the lack of resolution (in the case of the UV), or the difficulty in manufacturing mask and radiation sources (X-ray), these techniques are not suitable for nanometer-scale fabrication. E-beam lithography is an attractive alternative technique for fabricating nanostructures [27.53]. It uses an electron beam to expose an electron-sensitive resist such as polymethyl methacrylate (PMMA) dissolved in trichlorobenzene (positive) or polychloromethylstyrene (negative).

The e-beam gun is usually part of an SEM, although a TEM can also be used. Although electron wavelengths on the order of 1 Å can easily be achieved, electron scattering in the resist limits the attainable resolutions to > 10 nm. The beam control and pattern generation are achieved through a computer interface.

E-beam lithography is serial and hence has a low throughput. Although this is not a major concern in fabricating devices used in studying fundamental microphysics, it severely limits large-scale nanofabrication. E-beam lithography, in conjunction with such processes as lift-off, etching, and electrodeposition, can be used to fabricate various nanostructures.

An interesting new technique that circumvents the serial and low-throughput limitations of the e-beam lithography for fabricating nanostructures is nanoimprint technology [27.54]. This technique uses an e-beam-fabricated hard material master (or mold) to stamp and deform a polymeric resist. This is usually followed by a reactive-ion etching step to transfer the stamped pattern to the substrate. This technique is economically superior, since a single stamp can be used repeatedly to fabricate a large number of nanostructures.

Epitaxy and Strain Engineering

Atomic-precision deposition techniques such as molecular-beam epitaxy (MBE) and metallo-organic chemical vapor deposition (MOCVD) have proven to be effective tools in fabricating a variety of quantum confinement structures and devices (quantum well lasers, photodetectors, resonant tunneling diodes, etc.) [27.55–57].

Dip-Pen Nanolithography

In dip-pen nanolithography (DPN), the tip of an AFM operated in air is *inked* with a chemical of interest and brought into contact with a surface. The ink molecules flow from the tip onto the surface as with a fountain pen. Line widths down to 12 nm with spatial resolution of 5 nm have been demonstrated with this technique [27.58]. Species patterned with DPN include conducting polymers, gold, dendrimers, deoxyribonucleic acid (DNA), organic dyes, antibodies, and alkanethiols.

27.6 Microassembly

Assembly is often required in macroscale product manufacturing in order to reduce the complexity and cost of the manufacturing process. Assembly makes it possible to build complex products from relatively simple parts and to integrate incompatible manufacturing processes. It also makes maintenance and replacement possible. The extension of assembly techniques into the microscale is driven by the development of modern design and manufacturing technologies in the pursuit of miniaturization and function integration, especially by the development of integrated circuit (IC) [27.59] and MEMS fabrication techniques. With the extension of manufacturing technology into the microscale and even nanoscale domain, the term microassembly has been created to refer specifically to assembly operations performed at the micro/mesoscale [27.60]. To give a formal definition, microassembly is the assembly of objects with microscale and/or mesoscale features under microscale tolerances [27.61].

27.6.1 Automated Microassembly Systems

Microassembly plays the role of an enabling technology in various processes of MEMS fabrication, including device fabrication, packaging, and interconnection. MEMS device fabrication is fundamentally different from the highly modular IC fabrication in that it often requires the machining of complex-shaped three-dimensional (3-D) mechanical structures [27.51]. However, almost all current MEMS fabrication tech-

niques are subject to constraints in limited allowable materials, limited capabilities in true 3-D fabrication, and the requirement of fabrication process compatibility. Microassembly provides a possible solution to these constraints. For example, incompatible fabrication processes can be integrated through assembly. This makes it possible to use *nontraditional* fabrication techniques that are not necessarily based on semiconductor materials, such as laser cutting, microwire electrical discharge machining (EDM), and micromilling [27.62]. Complex 3-D structures can also be developed using parts with relatively simple geometry [27.63, 64]. Microassembly is also crucial to MEMS packaging and interconnection [27.65, 66].

From the perspective of robotic systems and automation, MEMS device fabrication and packaging share many common assembly requirements. A fundamental commonality of both processes is the requirement of the ability to manipulate micro/mesoscale objects so that precise (i. e., microscale tolerance) spatial relations can be established (e.g., die alignment, part insertion) and certain physical/chemical processes (e.g., die bonding, surface coating) can be performed. Another common requirement is to control the interaction force involved. MEMS devices often have fragile structures such as thin beams or membranes. This requires controlling the interactive force in manipulation operations. Typically, the force magnitude resides in the range from millinewtons to micronewtons.

MEMS devices are often three-dimensional. MEMS packaging requires both electrical interconnection for signal transmission and mechanical interconnection for the interaction of the packaged device with its external environment [27.51, 67]. Many such mechanical interconnections require three-dimensional manipulation and three-dimensional force control. The actual operations are highly application specific, which pose great challenges to the development of automated microassembly systems. Automated IC packaging systems can be used for the packaging of certain MEMS devices such as accelerometers and gyros. However, packaging of microfluidic devices, optical MEMS devices, and hybrid microsystems often require the development of new automated microassembly techniques and systems.

The selection of assembly mode is among the first decisions to be made in developing automated microassembly systems. Sequential microassembly requires the use of micromanipulators and sensory feedback. At each moment, only one or a few parts are being assembled. Depending on the physical effects used, parallel microassembly can be either deterministic or stochastic [27.68]. Die bonding is an example of deterministic parallel microassembly. In stochastic parallel microassembly, large numbers of parts are assembled simultaneously using distributed physical effects such as electrostatic force, capillary force, centrifugal force, or vibration [27.60, 69–71]. In fact, the basic philosophy of stochastic parallel microassembly is to minimize the use of sensory feedback.

Each of these assembly modes has both advantages and disadvantages. Each has its own suitable applications. Deterministic parallel assembly shares several commonalities with sequential microassembly. For example, sensory feedback is often used in deterministic parallel microassembly. However, deterministic parallel microassembly requires high relative positioning accuracy between parts. In addition, only simple planar structure features can be assembled in order to make parallel operation possible.

Due to the requirement of MEMS packaging for three-dimensional manipulation and microassembly, it can be expected that automated sequential microassembly will be the most widely adopted solution. In particular, those MEMS packaging applications that require the control of multiple-degree-of-freedom (DOF) interaction force should use sequential microassembly. The major possible disadvantage of sequential microassembly is its low throughput. This constraint can often be overcome by appropriate system design.

Here we introduce a three-dimensional microassembly example that originates from an industrial application [27.61]. It is significantly different from wire bonding and die bonding applications in that high-precision 3-D part insertion is required. Together with other examples, it will be used throughout this section to illustrate the major concepts and techniques and the logic connections between each functional unit of an automated microassembly system.

The assembly task is to pick up micromachined thin metal parts that are transferred to the assembly workcell on a vacuum-release tray (Fig. 27.6a–c), and insert them into vertically deep reactive ion etching (DRIE) etched holes in a silicon wafer (Fig. 27.6d). The wafers can have diameters up to 8 inches. The holes on each wafer form regular arrays consisting of approximately 50 holes each. However, these arrays may not be regularly distributed on the wafer. Typically, hundreds of parts are to be assembled on each wafer. In general, each assembly operation is a typical rectangular-peg-into-a-rectangular-hole problem. Each metal part is approximately half a millimeter in width and less than 100 μm in thickness at its rectangular tip. The total assembly tolerance is typically smaller than 10 μm in the vertical direction, and smaller than 20 μm in the horizontal direction. This task is a typical application of 3-D microassembly techniques.

27.6.2 Microassembly System Design

This section discusses the design of automated microassembly systems from the perspective of robotic systems and automation. Performance objects to be achieved include high reliability, high throughput, high flexibility, and low cost.

General Guidelines

Taking a System Perspective. An automated microassembly system consists of many functional units and must integrate techniques from a diverse range of areas such as robotics, computer vision, microscope optics, physics, and chemistry. It is therefore important to consider the interaction between these units.

Emphasizing the Coupling with Packaging Processes. Although in this section microassembly techniques are addressed primarily from the perspective of robotics for the clarity of presentation, robotic system designers must recognize the strong dependence of architecture design of the robotic systems on the packaging process being implemented. This connection should be emphasized from the beginning of system development.

Design for Reconfigurability. Automated packaging machines must be designed so that they can be customized for a wide variety of applications. Reconfigurability is therefore a basic design requirement. Typically, modular design based on function decomposition is desirable. Support for tool replacement is essential.

The design of automated microassembly systems is strongly dependent on the assembly tolerance required. First of all, the repeatability of the motion control system and the micromanipulator used is determined by the required assembly tolerance. In addition, assembly tolerance often determines the minimum resolution of the microscope optics. General microassembly tasks may only require microscopic vision feedback and manipulators with microscale repeatability, while complex microassembly tasks may also require the integration of microforce and vision feedback [27.72, 73].

Automated microassembly systems must be able to support a wide variety of material-handling tools, which include part transfer tools, bulk feeders, wafer-handling tools, magazine loaders and unloaders, etc.

The role of the micromanipulator is to provide multiple-DOF fine motion control. The role of the microgripper is to grasp objects in pick-and-place and other assembly operations. Their efficiency and robustness will to a great extent decide the performance of the entire system. Microgripper design is also closely related to the design of fixtures for microassembly operations.

Major environment factors include clean-room requirements, temperature, humidity, airflow, etc. Certain assembly operations must be performed in a clean room. This requires that the design of automated microassembly systems comply with relevant standards. Some packaging processes such as eutectic bonding must be performed under high temperature. Consequently, the potential influence of high temperature on motion control system and microscope optics must be considered. For the manipulation of microscale objects, environment conditions such as temperature and humidity can have a major influence on adhesion forces [27.74, 75]. Therefore, it is often important to consider environment control in system design.

Assembly Process Flow: An Example

The micromachined metal parts (Fig. 27.6c) are horizontally transferred to the workcell on the vacuum-release tray (Fig. 27.6b). The wafer is placed on a wafer mount perpendicular to the horizontal plane (Fig. 27.6b). This configuration does not require the flipping of the thin metal parts and is advantageous in terms of reliability and efficiency. There are two major operations in each assembly cycle: pickup and insertion. Under the configuration shown in Fig. 27.7a, all the operations are performed by the same workcell. Complex packaging operations must often be decomposed and performed by multiple workcells [27.76].

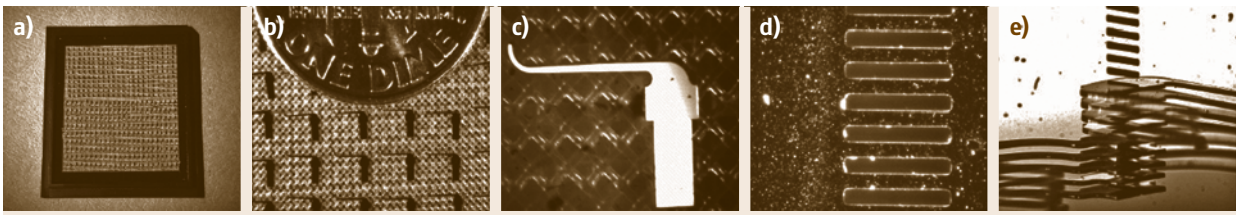


Fig.27.6a–e A 3-D microassembly example **(a)** Parts on release tray **(b)** parts versus a US dime **(c)** part shape **(d)** DRIE holes **(e)** assembly results

General System Architecture

An automated microassembly system typically consists of the following functional units.

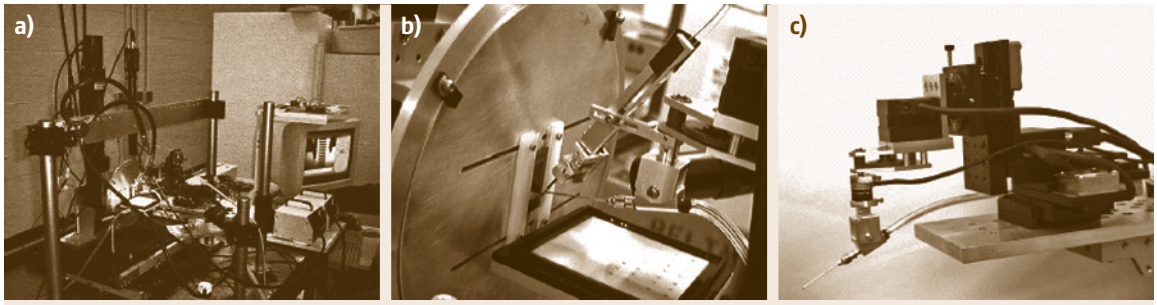
Large Workspace Positioning Unit. A large workspace and long-range positioning motion are required in most microassembly operations. The large workspace is necessary to accommodate different functional units, part feeders, and various tools.

In general, off-the-shelf motion control systems developed for automated IC packaging equipment can be adopted directly. For the task described in Sect. 27.6.1, the DRIE-etched holes are distributed on wafers up to 8 inches in diameter. This requires the assembly system to have a commensurate working space and high positioning speed. The coarse positioning unit has four DOFs (Fig. 27.7a). Planar motion in the horizontal direction is provided by an open-frame high-precision *XY* table with a travel of 32 cm (12 inch) and a repeatability of 1 μm in both directions. Position feedback with a resolution of 0.1 μm is provided by two linear encoders. A dual-loop PID (proportional–integral–derivative) plus feedforward control scheme is used for each axis. The internal speed loop is closed on the rotary encoder on the motor. The external position loop is closed on the linear encoder. Each wafer is placed on the vertical wafer mount that provides both linear and rotational control (Fig. 27.7b). Vertical motions of the wafer mount are provided by a linear slide with a travel of 20 cm (8 inch) and a repeatability of 5 μm. It is also controlled using a PID plus feedforward algorithm. Both the *XY* table and the vertical linear slides are actuated using alternating-current (AC) servo motors. The rotation of the wafer mount is actuated by an Oriental PK545AUA microstep motor with a maximum resolution of 0.0028° step. All low-level controllers are commanded and coordinated by a host computer [27.61].

For applications requiring repeatability of one micron or greater, it is convenient to use conventional positioning tables to implement coarse range motions. These tables typically use leadscrew or ballscrew drives and ball or roller bearings. For applications requiring submicron or nanometer repeatability, a few solutions are also commercially available. For example, piezoactuators are often used for nanometer repeatability motion. The disadvantage of piezoactuators is that their travel range is small, typically on the order of 100 microns. As another example, a series of positioning stages with submicron repeatability based on direct-drive linear actuators and air bearings is available from Aerotech. It is also possible to use parallel structure mechanisms such as a Stewart platform [27.77]. In general, the development of IC manufacturing towards the deep submicron level provides a major driving force behind the development of these motion control techniques.

Micromanipulator Unit. Fine pose (position and orientation) control is required in operations such as 3-D precision alignment and assembly. For example, the six DOFs required by the task introduced in Sect. 27.6.1 are implemented on separate structures. The three Cartesian DOF are provided by an adapted Sutter MP285 micromanipulator, which also provides yaw motions with its rotational DOF (Fig. 27.7c). Roll motions are implemented on the wafer mount (Fig. 27.7b). The pitch movement of the metal part after pickup is not motorized and is implemented through manual adjustment and calibration before assembly. More discussion of micromanipulator configuration can be found in [27.61].

Two principles need to be considered in implementing motion control for automated 3-D microassembly. The first is the partition of large-workspace coarse positioning unit and the fine positioning micromanipulator unit. The second is the decomposition and distributed implementation of multiple DOFs. In practice, the actual implementations of these principles are highly application specific. For certain applications, if a large-range positioning unit is sufficient to satisfy assembly



System overview | Assembly scene | Micromanipulator unit

Fig.27.7a–c An experimental microassembly workcell **(a)** System overview **(b)** assembly scene **(c)** micromanipulator unit

tolerance requirements, the separate implementation of a micromanipulator may even be unnecessary.

In general, the separation of a micromanipulator unit will facilitate the implementation of high-bandwidth motion control. However, this separation normally brings redundancy to the entire motion control system. The functioning of the high-precision micromanipulator must also rely on closed-loop feedback control, especially microscopic vision feedback.

Automated Microgripper Unit. The function of a microgripper is to provide geometrical and physical constraints (grasping) in pick-and-place and assembly operations. The reliability and efficiency of the microgripper is critical to the performance of the entire automated microassembly system. Several factors must be considered in micromanipulator design. First of all, as the micromanipulator end-effector, the microgripper must be constantly monitored under a microscope. Therefore, it must be small in size and suitable in shape to remain in the microscope's field of view and to minimize occlusion. Secondly, it is important to consider the different governing physics and to explore the use of various gripping forces [27.77]. Thirdly, microgrippers are used in both part pick-and-place and assembly. Since assembly operations typically require more constraints, microgrippers designed for pick-and-place operations may not necessarily be suitable for assembly operations. In fact, the microassembly task described in Sect. 27.6.2 is performed using a combined microgripper [27.61]. Microgripper development is often closely related to the development of fixtures that can also have microscale sizes.

Microscope Optics and Imaging Unit. The function of the microscope optics and imaging unit is to provide noncontact measurement of the geometry, motion, and spatial relations of assembly objects. Typical configurations of commercial device bonding systems use one or two vertical microscopes. An inverted microscope configuration is commonly used for backside alignment. On the other hand, 3-D microassembly may require two camera views in a stereo configuration. In the system shown in Fig. 27.7a, a total of four different views can be provided to its human operator: a global view of the entire assembly scene, a vertical microscopic view for part pickup, and two lateral microscopic views for the fine position and orientation adjustments during the final microassembly operations. Each view uses a CCD camera with a matching optical system. All images are captured using a Matrox Corona peripheral component interconnect (PCI) frame grabber.

Microscopic visual feedback is crucial for precise 3-D alignment. Provided that resolution requirements of the assembly task can be satisfied, microscope optics with larger working distances is desirable. For the assembly task addressed in Sect. 27.6.1, an Edmund Scientific VZM 450i zoom microscope with a 1× objective is used to provide the right view. Its working distance is approximately 90 mm, with a resolution of 7.5 μm. The vertical view is also provided by a VZM 450i microscope with a 0.5× objective to guide pickup operations. Its field of view can range from 2.8 × 2.8 mm to 17.6 × 17.6 mm. Its working distance is approximately 147 mm.

If visual servoing is required in automate assembly operations, a stereo configuration formed by adding another lateral view may be necessary. Higher resolution may be necessary in some microassembly tasks. In such cases, the same configuration can be used with higher-resolution microscope optics, for example, two Navitar TenX zoom microscopes with Mitutoyo ultralong-working-distance M Plan Apo 10× objectives have been used in this configuration by the authors. The resolution of each microscope is 1 μm, with a working distance of 33.5 mm. Consequently, the usable workspace of the micromanipulator is reduced [27.61].

The global view is implemented using a miniature Marshall V-1260 board camera to monitor the status of the entire assembly scene. It plays an important role in helping the operator to understand gross spatial relations and preventing operation errors.

From the perspective of robotic systems, the development of automation microassembly systems for complex 3-D microassembly operation will rely on the development of:

1. Compact, robust and high-speed micromanipulators with 5–6 DOFs.
2. Highly reliable and efficient microgrippers that are suitable for working under microscopes. Such grippers should have active force control or passive compliance to avoid damage to MEMS devices.
3. Three-dimensional microscopic computer vision techniques, three-dimensional microforce measurement and control techniques, and their integration.

27.6.3 Basic Microassembly Techniques

This section introduces a few supporting techniques important to automated microassembly systems, including machine vision techniques, microforce control techniques, and simulation verification of assembly strategy.

Machine Vision Techniques

Machine vision techniques are widely used in the semiconductor industry. The major difference between ma-

chine vision and general computer vision [27.78] is that, unlike natural objects and scenes, industrial objects and scenes can often be artificially designed and configured. This advantage often makes it possible to significantly reduce the complexity and enhance the robustness of vision techniques. The applications of machine vision techniques can generally be categorized into the following two classes based on the requirement of real-time processing.

Non-Time-Critical Vision Applications. These applications do not require visual feedback for high-bandwidth real-time control. Examples include object recognition and packaging quality inspection [27.79].

Time-Critical Vision Applications. These applications require real-time visual feedback. Examples include vision-guided pick-and-place, alignment, insertion, etc. [27.80]. An introduction to 3-D computer vision techniques can be found in [27.81]. A standard introduction to visual servoing techniques can be found in [27.82].

Several commercial software packages are available from suppliers such as Cognex, Coreco Imaging, and National Instruments.

Microforce Control Techniques

The theory of force control has been studied by the robotics community for more than 50 years [27.83, 84]. Several theoretical frameworks and many control algorithms have been proposed and experimentally verified. A variety of macroscale multiple-DOF force sensors have been developed.

Force control is also crucial to microassembly. For example, contact forces in device bonding must often be programmed and precisely controlled. In general, many of the macroscale force control techniques can be applied at micro/mesoscales. Force control for device bonding is essentially 1-D and involves force on the order of several newtons. On the other hand, in the manipulation of micro/mesoscale parts, the magnitude of interactive force typically ranges from millinewtons (10^{-3} N) to micronewtons (10^{-6} N). Forces of this magnitude are often referred to as *microforces*. A major technical challenge in implementing microforce control is the lack of multiple-DOF microforce sensors. A basic requirement for multiple-DOF microforce sensors is that they must be miniature in size. The manufacturing of these sensors normally requires the use of micromachining, including MEMS techniques. There are two major microforce sensing configurations.

Stand-Alone Force Sensor. The advantage of this configuration is that the sensor is general purpose and can be used with different microgrippers. Most macroscale multiple-DOF force sensors are of this type. However, this also requires the sensor to have sufficient structural stiffness to support the static load of microgripper, which is often significantly larger than the force resolution to be reached.

Embedded Force Sensor. Micro strain gages can be attached to microgrippers [27.85]. Force-sensitive materials can also be deposited on microgrippers. This configuration avoids the issue of static load. However, such force sensing capabilities are dependent on the microgripper design, which often is not necessarily optimal for the measurement of multiple-DOF microforce/torque.

Complex and high-precision microassembly tasks also require the integration of microscopic machine vision with microforce control. Simple integration techniques use a gating/switching scheme [27.72, 86]. For more integrated approaches, visual impedance can be used [27.73].

Simulation Verification of Assembly Strategies

In many microassembly tasks, the distances between adjacent features are often on the meso/microscale. Therefore, selecting the correct assembly sequence is important for collision avoidance. In addition, due to the limited working distance of microscopes, microassembly operations must often be performed in a limited space using micromanipulators. Collision avoidance is critical to avoiding equipment or device damage. Potential collisions can be found and avoided by using offline simulation software. Many commercially available offline robot programming tools can provide this function.

Microassembly Tool

The end-effector of a micromanipulator in an automated microassembly system is often in the form of a microgripper, whose reliability and efficiency greatly influence the reliability and efficiency of the entire system. The microgripper must often be as small as possible. Its design must also minimize potential damage to fragile MEMS parts. This often requires passive compliance in structural design.

Microgrippers with integrated MEMS actuators can be fabricated monolithically and thus can be more compact in size. Several physical effects are commonly used in MEMS actuators, including electrostatic force and piezoelectric force [27.87], shape memory alloy (SMA) [27.88, 89], and thermal deformation [27.63, 90]. Currently, the major limitation is that it is difficult for MEMS actuators to generate sufficient travel, force, and power output. Alternatively, another solution

is to provide actuation externally [27.61, 91] The advantage is that sufficient travel and force and power output can be more easily obtained. The major disadvantage is that the microgripper is less compact in size. This could become a major obstacle to its applications.

27.7 Microrobotics

Today, more and more microrobotic devices are enabling new applications in various fields. Besides microassembly, microrobotics can play important roles in other industry fields for manipulation, characterization, inspection, and maintenance, and in biotechnology for, e.g., manipulating cells, a field referred to as biomicrorobotics.

27.7.1 Introduction

Microrobotics is a field that combines the established theory and techniques of robotics with the exciting new tools provided by MEMS technology in order to create intelligent machines that operate at micron scales. As stated by authors reviewing the microrobotics field [27.31, 92], many *micro* terms such as *micromechatronics*, *micromechanism*, *micromachines*, and *microrobots* are used synonymously to indicate a wide range of devices whose function is related to a *small* scale; however, *small* scale is a relative term so a clearer definition is needed.

The obvious difference between a macrorobot and a microrobot is the size of the robot. Thus, one definition of a microrobot is *a device having dimensions smaller than classical watch-making parts (i. e., μm to mm) and having the ability to move, apply forces and manipulate objects in a workspace with dimensions in the micrometer or submicrometer range* [27.93]. However, in many cases it is important that the robot can move over much larger distances. This task-specific definition is quite wide and includes several types of very small robots as well as stationary micromanipulation systems, which are a few decimeters in size but can carry out very precise manipulation (in the micron or even nanometer range) [27.92].

Besides classification by task or size, microrobots can also be classified by their mobility and functionality [27.31, 94]. Many robots usually consist of sensors and actuators, a control unit, and an energy source.

Depending on the arrangement of these components, one can classify microrobots according to the following criteria: locomotive and positioning possibility (yes or no), manipulation possibility (yes or no), control type (wireless or tethered), and autonomy. Figure 27.8 illustrates 15 different possible microrobot configurations by combining the four criteria [27.31, 94].

As depicted in Fig. 27.8 (taken from [27.24]), the classification is dependent on the following microrobot components: the control unit (CU), the power source (PS), the actuators necessary for moving the robot platform (i. e., the robot drive for locomotion and positioning; AP), and the actuators necessary for operation (i. e., manipulation using robot arms and hands; AM). Besides the different actuation functions, sensory functions are also needed, for example, tactile sensors for microgrippers or charge-coupled device (CCD) cameras for endoscopic applications (compare Fig. 27.8d and a).

The ultimate goal is to create a fully autonomous, wireless mobile microrobot equipped with suitable microtools according to Fig. 27.8o. Because this is a very difficult task, a good start is to investigate the possibility of making silicon microrobot platforms that are steered and powered through wires, like the one in Fig. 27.8c, and to study their locomotion capability.

The majority of MEMS-based microrobotic devices developed so far could be categorized as moveable links – microcatheters [27.95, 96], according to Fig. 27.8a, or microgrippers [27.25], such as those in Fig. 27.8d, or the microgrippers [27.97, 98] shown in Fig. 27.8e. Among the research publications covering locomotive microrobots, most publications have addressed microconveyance systems (Fig. 27.8b) [27.99–102]. Robots using external sources for locomotion could be used (compare Fig. 27.8b,f,j,n). According to *Fatikow* and *Rembold* [27.92], several researchers are working on methods to navigate micromechanisms through human blood vessels; however, these microrobots are difficult to control. Examples of partially autonomous systems (compare Fig. 27.8j) are the concept for so-called *smart pills*. Centimeter-sized pills for sensing temperature and/or pH inside the body have been presented [27.103, 104] as well as pills equipped with video cameras [27.105]. The pill is swallowed and transported to the part of the body where one wants to measure or record a video sequence. The information of the measured parameter or the signals from the camera is then transmitted (telemetrically) out of the body. More sophisticated approaches involving actuators for drug delivery of various kinds have also been proposed [27.92, 104]. The position of the pill inside the body is located by an X-ray monitor or ultrasound.

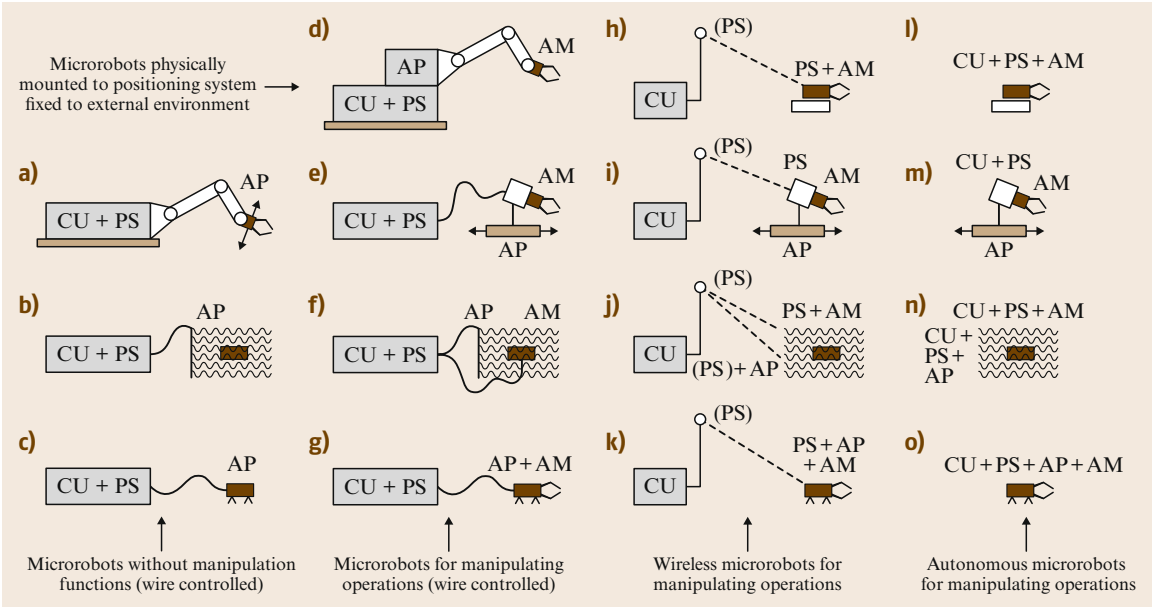


Fig.27.8a–o Classification of microrobots by functionality (modification of earlier presented classification schemes [27.12, 31, 92, 93]). (CU indicates the control unit; PS, the power source or power supply; AP, the actuators for positioning; AM, the actuators for manipulation): **(a–c)** Microrobots without manipulation functions (wire controlled), **(d–g)** microrobots for manipulation operations (wire controlled), **(h–k)** wireless microrobots for manipulation operations, **(l–o)** autonomous microrobots for manipulationg operations, **(d,h,l)** microrobots physically mounted to positioning system fixed to external environment

As soon as the pill reaches an infected area, a drug encapsulated in the pill can be released by the actuators onboard. External communication could be realized through radio signals.

Several important results have been presented regarding walking microrobots (Fig. 27.8c,g,k) fabricated by MEMS technologies and batch manufacturing. Different approaches for surface-micromachined robots [27.106, 107] and for a piezoelectric dry-reactive-ion-etched microrobot should be mentioned. A suitable low-power application-specific integrated circuit (ASIC) for robot control has been successfully tested and is planned to be integrated on a walking microrobot [27.108]. The large European Esprit project *MINIMAN* (1997) has the goal of developing moveable microrobotic platforms with integrated tools with six degrees of freedom for applications such as microassembly within an SEM and involves different MEMS research groups from several universities and companies across Europe. Further, miniature robot systems with MEMS/MST (microsystem technology) components have been developed [27.109].

Several research publications on *gnat* minirobots and actuator technologies for MEMS microrobots [27.3, 110] were reported by US researchers in the early 1990s, and several groups in Japan are also currently developing miniaturized robots based on MEMS devices [27.8]. In Japan, an extensive ten-year program on *micromachine technology*, supported by the Ministry of International Trade and Industry (MITI), started in 1991. One of the goals of this project is to create microsized and miniature robots for microfactory, medical technology, and maintenance applications. Several microrobotic devices, including locomotive robots and microconveyers, have been produced within this program. Miniature robot devices or vehicles [27.111] for locomotive tasks, containing several MEMS components, have been presented. Even though great efforts have been made on robot miniaturization using MEMS technologies, no experimental results on MEMS batch-fabricated microrobots suitable for autonomous walking (i. e., robust enough to be able to carry its own power source or to be powered by telemetric means) have been presented yet. The first batch-fabricated MEMS-based microrobot platform able to walk was presented in 1999 [27.112]. However, this robot was powered through wires and was not equipped with manipulation actuators. Besides walking microrobotic devices, several reports on flying [27.113–115] and swimming [27.116] robots have been published. Micromotors and gear boxes made using LIGA technology (a high-precision, lithograph-

ically defined plating technology) are used to build small flying microhelicopters, which are commercially available from the Institute of Microtechnology in Mainz, Germany, as rather expensive demonstration objects [27.114]. Besides the pure mechanical microrobots, hybrid systems consisting of electromechanical components and living organisms such as cockroaches have also been reported [27.117].

27.7.2 Bio-Microrobotics

Biomanipulation entails such operations as positioning, grasping, and injecting material into various locations in cells. Research topics in bio-microrobotics include the autonomous manipulation of single cells or molecules, the characterization of biomembrane mechanical properties using microrobotic systems with integrated vision and force sensing modules, and more. The objective is to obtain a fundamental understanding of single-cell biological systems and provide characterized mechanical models of biomembranes for deformable cell tracking during biomanipulation and cell injury studies.

Existing biomanipulation techniques can be classified into noncontact manipulation including laser trapping [27.118–121] and electrorotation [27.122–124], and contact manipulation, referred to as mechanical micromanipulation [27.125]. When laser trapping [27.118–121] is used for noncontact biomanipulation, a laser beam is focused through a large-numerical-aperture objective lens, converging to form an optical trap in which the lateral trapping force moves a cell in suspension toward the center of the beam. The longitudinal trapping force moves the cell in the direction of the focal point. The optical trap levitates the cell and holds it in position. Laser traps can work in a well-controlled manner. However, two features make laser trapping techniques undesirable for automated cell injection. The high dissipation of visible light in aqueous solutions requires the use of high-energy light close to the UV spectrum, raising the possibility of damage to the cell. Even though some researchers claim that such concerns could be overcome using wavelengths in the near-infrared (IR) spectrum [27.120], the question as to whether the incident laser beam might induce abnormalities in the cells' genetic material still exists. One alternative to using laser beams is the electrorotation technique. Electric-field-induced rotation of cells was demonstrated by *Mischel* et al. [27.126], *Arnold* and *Zimmermann* [27.127], and *Washizu* et al. [27.124]. This noncontact cell manipulation technique is based on controlling the phase shift and magnitude of electric fields. These fields, appropriately applied, produce a torque on the cell. Different system configurations have been established for cell manipulation based on this principle [27.122, 123], which can achieve high accuracy in cell positioning. However, it lacks a means to hold the cell in place for further manipulation, such as injection, since the magnitude of the electric fields has to be kept low to ensure the viability of cells. The limits of noncontact biomanipulation in the laser trapping and electrorotation techniques make mechanical micromanipulation desirable. The damage caused by laser beams in the laser trapping technique and the lack of a holding mechanism in the electrorotation technique can be overcome by mechanical micromanipulation.

To improve the low success rate of manual operation, and to eliminate contamination, an autonomous robotic system (shown in Fig. 27.9) has been developed to deposit DNA into one of the two nuclei of a mouse embryo without inducing cell lysis [27.11, 128]. The laboratory's experimental results show that the success rate for the autonomous embryo pronuclei DNA injection is dramatically improved over conventional manual

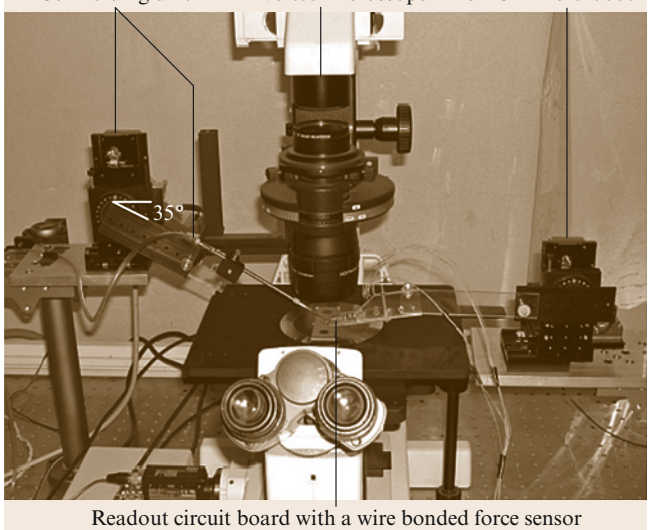


Fig. 27.9 Robotic biomanipulation system with vision and force feedback

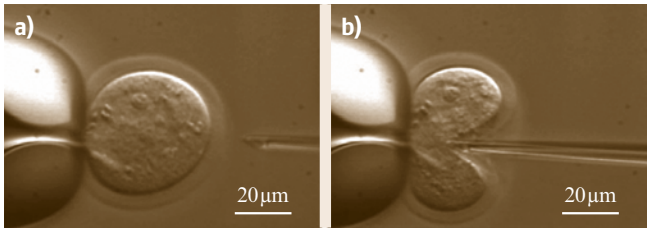


Fig. 27.10 Cell injection process. **(a)** Before and **(b)** during the injection of a mouse oocyte zona pellucida (ZP)

injection methods. The autonomous robotic system features a hybrid controller that combines visual servoing and precision position control, pattern recognition for detecting nuclei, and a precise autofocusing scheme. Figure 27.10 illustrates the injection process.

To realize large-scale injection operations, a MEMS cell holder was fabricated using anodic wafer-bonding techniques. Arrays of holes are aligned on the cell holder, which are used to contain and fix individual cells for injection. When well calibrated, the system with the cell holder makes it possible to inject large numbers of cells using position control. The cell injection operation can be conducted in a move–inject–move manner.

A successful injection is determined greatly by injection speed and trajectory, and the forces applied to cells. To further improve the robotic system's performance, a multi-axial MEMS-based capacitive cellular force sensor is being designed and fabricated to provide realtime force feedback to the robotic system. The MEMS cellular force sensor also aids research in biomembrane mechanical property characterization.

MEMS-Based Multi-Axis Capacitive Cellular Force Sensor

The MEMS-based two-axis cellular force sensor [27.129] shown in Fig. 27.11 is capable of resolving normal forces applied to a cell as well as tangential forces generated by improperly aligned cell probes. A high-yield microfabrication process was developed to form the 3-D high-aspect-ratio structure using deep reactive ion etching (DRIE) on silicon-on-insulator (SOI) wafers. The constrained outer frame and the inner movable structure are connected by four curved springs. A load applied to the probe causes the inner structure to move, changing the gap between each pair of interdigitated comb capacitors. Consequently, the total capacitance change resolves the applied force. The interdigitated capacitors are orthogonally configured to make the force sensor capable of resolving forces in both the x- and y-directions. The cellular force sensors used in the experiments are capable of resolving forces up to 25 μ N with a resolution of 0.01 μ N.

Tip geometry affects the quantitative force measurement results. A standard injection pipette (Cook K-MPIP-1000-5) tip section with a tip diameter of 5 μ m is attached to the probe of the cellular force sensors.

The robotic system and high-sensitivity cellular force sensor are also applied to biomembrane mechanical property studies [27.130]. The goal is to obtain a general parameterized model describing cell membrane deformation behavior when an external load is applied. This parameterized model serves two chief purposes. First, in robotic biomanipulation, it allows online parameter recognition so that cell membrane deformation behavior can be predicted. Second, for a thermodynamic model of membrane damage in cell injury and recovery studies, it is important to appreciate the mechanical behavior of the membranes. This allows the interpretation of such reported phenomena as mechanical resistance to cellular volume reduction during dehydration, and its relationship to injury. The establishment of such a biomembrane model will greatly facilitate cell injury studies.

Experiments demonstrate that robotics and MEMS technology can play important roles in biological studies such as automating biomanipulation tasks. Aided by robotics, the integration of vision and force sensing modules, and MEMS design and fabrication techniques, investigations are being conducted in biomembrane mechanical property modeling, deformable cell tracking, and single-cell and biomolecule manipulation.

27.7.3 Bio-Mimetic/Untethered Microrobots

Similar to the larger scale, bio-mimicking is a significant approach for the design of micro- and nanorobots. However, at the micro-/nanoscale, the objectives we are learning from are not animals, fishes, birds, or insects anymore, rather we learn from such categories as bacteria, molecular motors, DNA molecules, and so on.

For instance, it is well known that a variety of microorganisms swim in liquid using flagella, a swimming strategy that is particularly well suited to their low-Reynolds-number regime [27.131]. Eukaryotic flagella

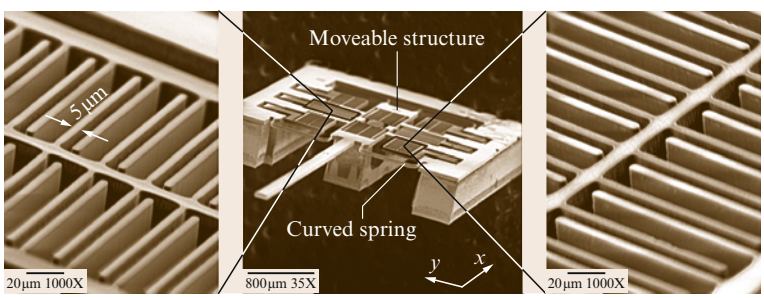


Fig. 27.11 A cellular force sensor with orthogonal comb drives detailed

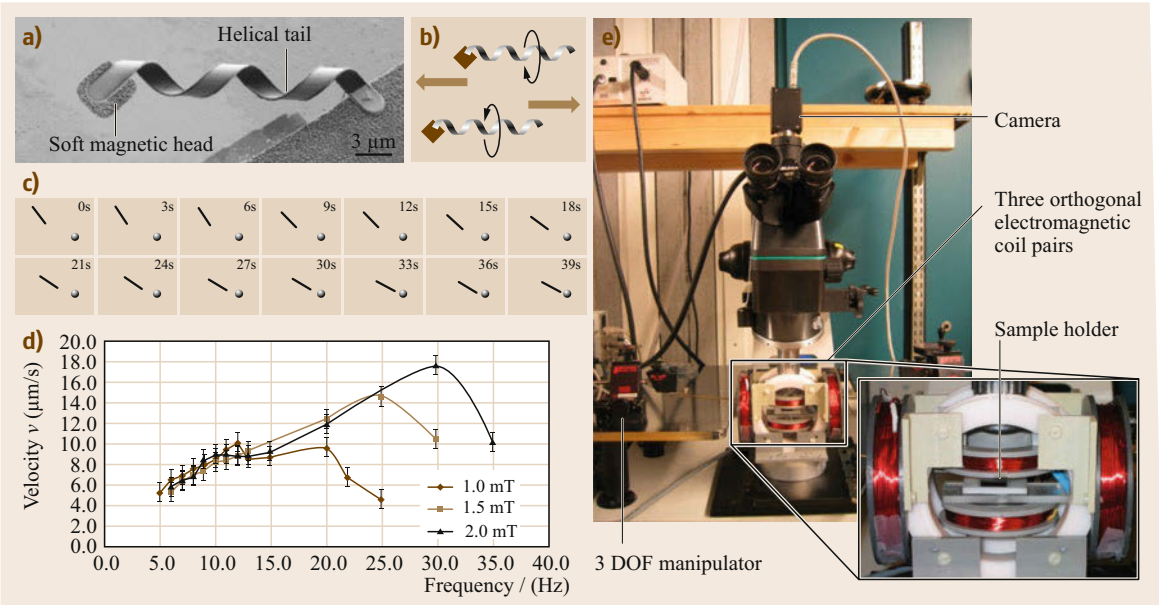


Fig.27.12a–e Artificial bacteria flagella. **(a)** SEM micrograph of an as-fabricated ABF with a diameter of 2.7 μm. **(b)** Schematic drawing of driving an ABF swims forward and backward as a rotary-to-linear motion converter. For forward and backward motion, the helical tail turns clockwise (CW) and counterclockwise (CCW), respectively (as seen by an observer in front of the ABF). **(c)** A series of frames taken from a video camera showing an ABF swim forward in water. The ABF has an 81 μm long InGaAs/GaAs/Cr helical tail, and a 4.5 $\mu m \times 4.5$ μm square head. B, f, θ, φ represent four parameters, i. e., magnetic field, field frequency, yaw and pitch, which can be controlled by the coil pairs. **(d)** The dependence of ABF velocity to frequency of external magnetic field generated by the coil pairs. **(e)** Experimental setup. All examples are from the authors' work

are active organelles that deform to create paddling motions, such as propagating waves or circular translating movements. Previous work experimentally demonstrated that this kind of motion can be approximated by a microscopic artificial swimmer consisting of a chain of DNA-bound magnetic beads connected to a red blood cell [27.132]. In contrast, bacterial (prokaryotic) flagella work differently by using a molecular motor to turn the base of a flagellum or bundle of flagella, which deform passively into a helical structure [27.133].

One example inspired by this is artificial bacterial flagella (ABFs) [27.134], which are proposed as microrobots capable of locomotion in liquid. An artificial bacterial flagellum consists of a helical *tail* resembling a natural flagellum, fabricated by the self-scrolling of helical nanobelts [27.135, 136], and a thin soft-magnetic *head*, consisting of a Cr/Ni/Au multilayer. Experimental investigation shows that an ABF can be propelled forward, backward and steered by a controlled external rotating magnetic field. ABFs have comparable swimming speed to bacteria propelled by flagella but under precise control.

An artificial bacterial flagellum (ABF) consists of two parts: a helical *tail* and a soft-magnetic metal *head* as shown in Fig. 27.12. Previous work has shown that the geometrical shape of the helical tail, i. e., chirality, helicity angle and diameter, can be precisely controlled. To control the locomotion of the ABF remotely, three orthogonal electromagnetic coil pairs were employed to generate a uniform rotating magnetic field [27.137].

In principle, for forward and backward motion, the ABF acts as a corkscrew to convert rotary motion to linear motion as shown in Fig. 27.12b. Figure 27.12c shows a left-handed ABF swim controlled by coil pairs. It can be seen in Fig. 27.12c that the ABF swim forward until the 10th second by rotating the magnet field clockwise (CW). In contrast to a left-handed ABF, when the same field is applied to a right-handed ABF, though it rotates in the same direction as left-handed one, the swimming direction will be opposite. The magnetic torque for rotation is generated by the thin head attempting to align with the applied field, and torque for steering is generated by the diagonal axis of the square head attempting to align with the applied field, since the diagonal axis is the easy magnetization axis of thin square plates in weak applied fields. Thus, to steer the ABFs in water, two more parameters were applied, i. e.,

yaw (θ) and pitch (φ), for the turning and tilting of the ABF.

The velocity of ABF is investigated as a function of magnetic field strength and rotation frequency in water (Fig. 27.12d. A 45 μm long ABF is used for the tests with three different field strengths: 1.0 mT, 1.5 mT and 2.0 mT. The field's rotational frequency is increased from 5 Hz to 35 Hz. The results show that at low frequency the swimmer's rotation is synchronized with the applied field, and ABF velocity increases linearly with frequency, as expected in the low-Reynolds-number regime [27.131]. After reaching a maximum value, the velocity reduces and becomes less deterministic with increasing field frequency, since the available magnetic torque is no longer sufficient to keep the swimmer synchronized with the applied field. The fluctuations in the curves are attributed to unmodeled boundary conditions, such as wall effects and intermolecular interactions of the ABF with the substrate. The maximum velocity we have achieved is 18 μm/s, which is comparable to bacteria, such as Escherichia coli [27.138], which swim by rotating their flagella with a frequency of about 100 Hz at room temperature. The results also indicate that by exerting a stronger magnetic torque on the ABF, higher driving frequencies can be achieved resulting in higher linear velocity. By extrapolating experimental results, if a frequency of 100 Hz can be achieved, the resulting swimming speed of ABFs may be beyond bacteria swimming speeds.

Self-propelled devices such as these are of interest in fundamental research and biomedical applications for mimicking and understanding natural organisms, for signaling inter- or intra-cellular information, for manipulating cellular or sub-cellular objects, and for targeted drug delivery. In contrast to bacteria that are propelled by nature flagella and move somewhat randomly in liquid, ABF motion can be precisely controlled by magnetic fields. ABFs can be used as a test bed for understanding the swimming behavior of bacteria and other swimming microorganisms, and can potentially be used for targeted drug delivery and as wireless manipulators for medical and biological applications.

27.8 Nanorobotics

Nanorobotics represents the next stage in miniaturization for maneuvering nanoscale objects. Nanorobotics is the study of robotics at the nanometer scale, and includes robots that are nanoscale in size, i. e., nanorobots, and large robots capable of manipulating objects that have dimensions in the nanoscale range with nanometer resolution, i. e., nanorobotic manipulators. Robotic manipulation at nanometer scales is a promising technology for structuring, characterizing, and assembling nanoscale building blocks into NEMS. Combined with recently developed nanofabrication processes, a hybrid approach is realized to build NEMS and other nanorobotic devices from individual carbon nanotubes and SiGe/Si nanocoils. Material science, biotechnology, electronics, and mechanical sensing and actuation will benefit from advances in nanorobotics.

27.8.1 Introduction

Nanomanipulation, or positional and/or force control at the nanometer scale, is a key enabling technology for nanotechnology by filling the gap between top-down and bottom-up strategies, and may lead to the appearance of replication-based molecular assemblers [27.18]. These types of assemblers have been proposed as general-purpose manufacturing devices for building a wide range of useful products as well as copies of themselves (self-replication).

Presently, nanomanipulation can be applied to the scientific exploration of mesoscopic physical phenomena, biology, and the construction of prototype nanodevices. It is a fundamental technology for property characterization of nanomaterials, nanostructures, and nanomechanisms, for the preparation of nanoscale building blocks, and for the assembly of nanodevices such as NEMS.

Nanomanipulation was enabled by the inventions of the STM [27.19], AFMs [27.44], and other types of scanning probe microscopes (SPMs). Besides these, optical tweezers (laser trapping) [27.139] and magnetic tweezers [27.140] are also potential nanomanipulators. Nanorobotic manipulators (NRMs) [27.141, 142] are characterized by the capability of 3-D positioning, orientation control, independently actuated multiple end-effectors, and independent real-time observation systems, and can be integrated with scanning probe microscopes. NRMs largely extend the complexity of nanomanipulation.

A concise comparison of STM, AFM, and NRM technology is shown in Fig. 27.13. With its incomparable imaging resolution, an STM can be applied to particles as small as atoms with atomic resolution. However, limited by its 2-D positioning and available strategies for manipulations, standard STMs are ill-suited for complex manipulation and cannot be used in 3-D space. An AFM is another important type of nanomanipulator. Manipulation with an AFM can

be done in either contact or dynamic mode. Generally, manipulation with an AFM involves moving an object by touching it with a tip. A typical manipulation starts by imaging a particle in noncontact mode, then removing the tip oscillation voltage and sweeping the tip across the particle in contact with the surface and with the feedback disabled. Mechanical pushing can exert larger forces on objects and, hence, can be applied for the manipulation of relatively larger objects. 1- to 3-D objects can be manipulated on a 2-D substrate. However, the manipulation of individual atoms with an AFM remains a challenge. By separating the imaging and manipulation functions, nanorobotic manipulators can have many more degrees of freedom including rotation for orientation control, and hence can be used for the manipulation of 0-D (symmetric spheres) to 3-D objects in 3-D free space. Limited by the lower resolution of electron microscopes, NRMs are difficult to use for the manipulation of atoms. However, their general robotic capabilities, including 3-D positioning, orientation control, independently actuated multiple endeffectors, separate real-time observation system, and integration with SPMs inside, make NRMs quite promising for complex nanomanipulation.

The first nanomanipulation experiment was performed by *Eigler* and *Schweizer* in 1990 [27.143]. They used an STM and materials at low temperatures (4 K) to position individual xenon atoms on a single-crystal nickel surface with atomic precision. The manipulation enabled them to fabricate rudimentary structures of their own design, atom by atom. The result is the famous set of images showing how 35 atoms were moved to form the three-letter logo *IBM*, demonstrating that matter could indeed be maneuvered atom by atom as *Feynman* suggested [27.1].

A nanomanipulation system generally includes nanomanipulators as the positioning device, microscopes as *eyes*, various end-effectors including probes and tweezers among others as its *fingers*, and various types of sensors (force, displacement, tactile, strain, etc.) to facilitate the manipulation and/or to determine the properties of the objects. Key technologies for nanomanipulation include observation, actuation, measurement, system design and fabrication, calibration and control, communication, and the human–machine interface.

Strategies for nanomanipulation are basically determined by the environment – air, liquid or vacuum – which is further decided by the properties and size of the objects and observation methods. Figure 27.14 depicts the microscopes, environments, and strategies of nanomanipulation. In order to observe manipulated objects, STMs can provide sub-Ångström imaging resolution, whereas AFMs can provide atomic resolutions. Both can obtain 3-D surface topology. Because AFMs can be used in an ambient environment, they provide a powerful tool for biomanipulation that may require a liquid environment. The resolution of SEM is limited to about 1 nm, whereas field-emission SEM (FESEM) can achieve higher resolutions. SEM/FESEM can be used for real-time 2-D observation for both the objects and end-effectors of manipulators, and large ultrahigh-vacuum (UHV) sample chambers provide enough space to contain an NRM with many degrees of freedom

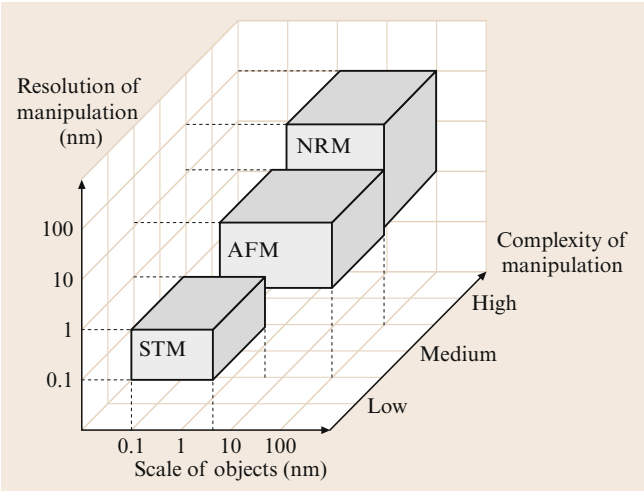


Fig. 27.13 Comparison of nanomanipulators

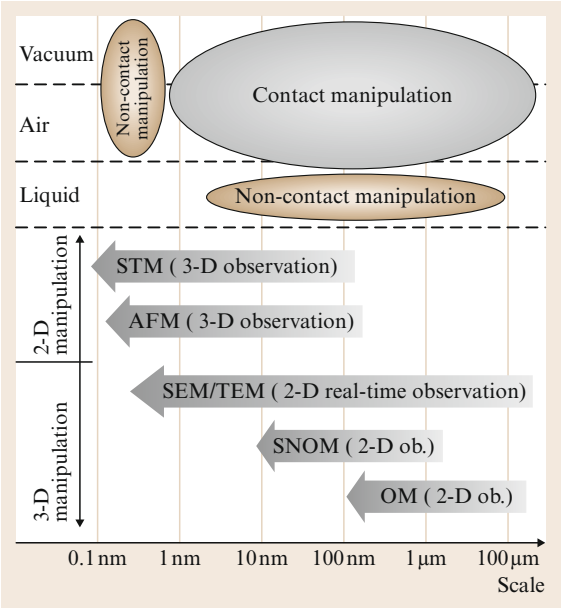


Fig. 27.14 Microscopes, environments, and strategies of nanomanipulation

(DOFs) for 3-D nanomanipulation. However, the 2-D nature of the observation makes positioning along the electron-beam direction difficult. High-resolution transmission electron microscopes (HRTEM) can provide atomic resolution. However, the narrow UHV specimen chamber makes it difficult to incorporate large manipulators. In principle, optical microscopes (OMs) cannot be used for nanoscale (smaller than the wavelength of visible lights) observation because of diffraction limits. Scanning near-field OMs (SNOMs) break this limitation and are promising as a real-time observation device for nanomanipulation, especially for ambient environments. SNOMs can be combined with AFMs, and potentially with NRMs for nanoscale biomanipulation.

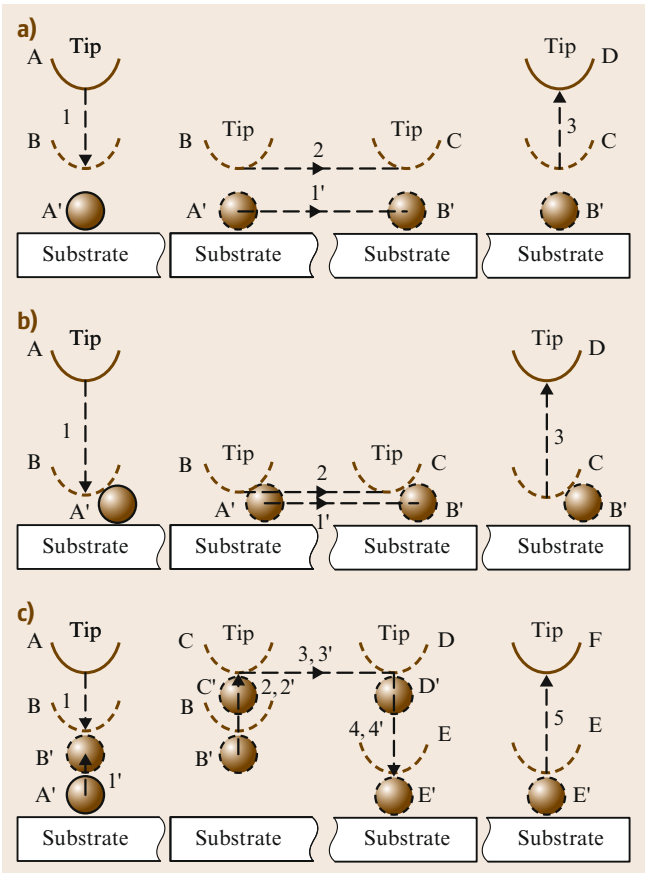


Fig.27.15a–c Basic strategies of nanomanipulation. In the figure, A, B, C,... represent the positions of end-effector (e.g., a tip); A′, B′, C′,... the positions of objects; 1, 2, 3,... the motions of end-effector; and 1′, 2′, 3′,... the motions of objects. Tweezers can be used in pick-and-place to facilitate the picking-up, but are generally not necessarily helpful for placing. **(a)** Lateral noncontact nanomanipulation (sliding). **(b)** Lateral contact nanomanipulation (pushing/pulling). **(c)** Vertical nanomanipulation (picking and placing)

Nanomanipulation processes can be broadly classified into three types:

1. Lateral noncontact
2. Lateral contact
3. Vertical manipulation.

Generally, lateral noncontact nanomanipulation is mainly applied for atoms and molecules in UHV with an STM or bio-object in liquid using optical or magnetic tweezers. Contact nanomanipulation can be used in almost any environment, generally with an AFM, but is difficult for atomic manipulation. Vertical manipulation can be performed by NRMs. Figure 27.15 shows the processes of the three basic strategies.

Motion of the lateral noncontact manipulation processes is shown in Fig. 27.15a. Applicable effects [27.144] able to cause the motion include long-range van der Waals (vdW) forces (attractive) generated by the proximity of the tip to the sample [27.145], electric-field-induced fields by the voltage bias between the tip and the sample [27.146, 147], tunneling current local heating or inelastic tunneling vibration [27.148, 149]. With these methods, some nanodevices and molecules have been assembled [27.150, 151]. Laser trapping (optical tweezers) and magnetic tweezers are possible for noncontact manipulation of nanoscale biosamples, e.g., DNA [27.152, 153].

Noncontact manipulation combined with STMs has revealed many possible strategies for manipulating atoms and molecules. However, for the manipulation of carbon nanotubes (CNTs) no examples have been demonstrated.

Pushing or pulling nanometer objects on a surface with an AFM is a typical manipulation using this method as shown in Fig. 27.15b. Early work showed the effectiveness of this method for the manipulation of nanoparticles [27.154–158]. This method has also been shown in nanoconstruction [27.159] and biomanipulation [27.160]. A virtual-reality interface facilitates such manipulation [27.161–163] and may create an opportunity for other types of manipulation. This technique has been used in the manipulation of nanotubes on a surface, and some examples will be introduced later in this chapter.

The pick-and-place task as shown in Fig. 27.15c is especially significant for 3-D nanomanipulation since its main purpose is to assemble prefabricated building blocks into devices. The main difficulty is in achieving sufficient control of the interaction between the tool and object and between the object and the substrate. Two strategies have been presented for micromanipulation [27.164] and have also proven to be effective for nanomanipulation [27.142]. One strategy is to apply

a dielectrophoretic force between a tool and an object as a controllable additional external force by applying a bias between the tool and the substrate on which the object is placed. Another strategy is to modify the van der Waals and other intermolecular and surface forces between the object and the substrate. For the former, an AFM cantilever is ideal as one electrode to generate a nonuniform electrical field between the cantilever and the substrate.

27.8.2 Nanorobotic Manipulation Systems

Nanorobotic manipulators are the core components of nanorobotic manipulation systems. The basic requirements for a nanorobotic manipulation system for 3-D manipulation include nanoscale positioning resolution, a relative large working space, enough DOFs including rotational ones for 3-D positioning and orientation control of the end-effectors, and usually multiple end effectors for complex operations.

A commercially available nanomanipulator (MM3A from Kleindiek) installed inside an SEM (Carl Zeiss DSM962) is shown in Fig. 27.16. The manipulator has three degrees of freedom, and nanometer- to subnanometer-scale resolution (Table 27.4). Calculations show that when moving/scanning in the A/B direction at joint q_1/q_2, the additional linear motion in C is very small. For example, when the arm length is 50 mm, the additional motion in the C direction is only 0.25 nm to 1 nm when moving 5–10 μm; in the A direction; these errors can be ignored or compensated with an additional motion of the prismatic joint p_3, which has a 0.25 nm resolution.

Figure 27.17a shows a nanorobotic manipulation system that has 16 DOFs in total and can be equipped with three or four AFM cantilevers as end-effectors for both manipulation and measurement. The positioning resolution is sub-nanometer and strokes are on the order

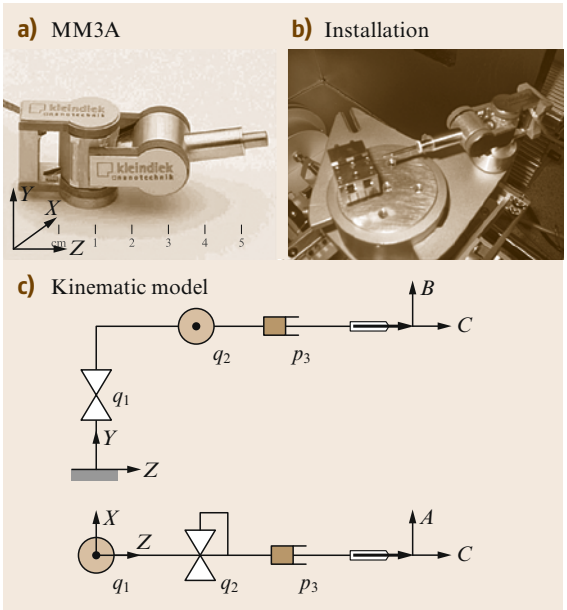


Fig. 27.16 (a) Nanomanipulator MM3A from Kleindiek (b) inside an SEM (c) kinematic model

Table 27.4 Specifications of MM3A

Item	Specification
Operating range q_1 and q_2	240°
Operating range Z	12 mm
Resolution A (horiz.)	10^{-7} rad (5 nm)
Resolution B (vert.)	10^{-7} rad (3.5 nm)
Resolution C (linear)	0.25 nm
Fine (scan) range A	20 μm
Fine (scan) range B	15 μm
Fine (scan) range C	1 μm
Speed A, B	10 mm/s
Speed C	2 mm/s

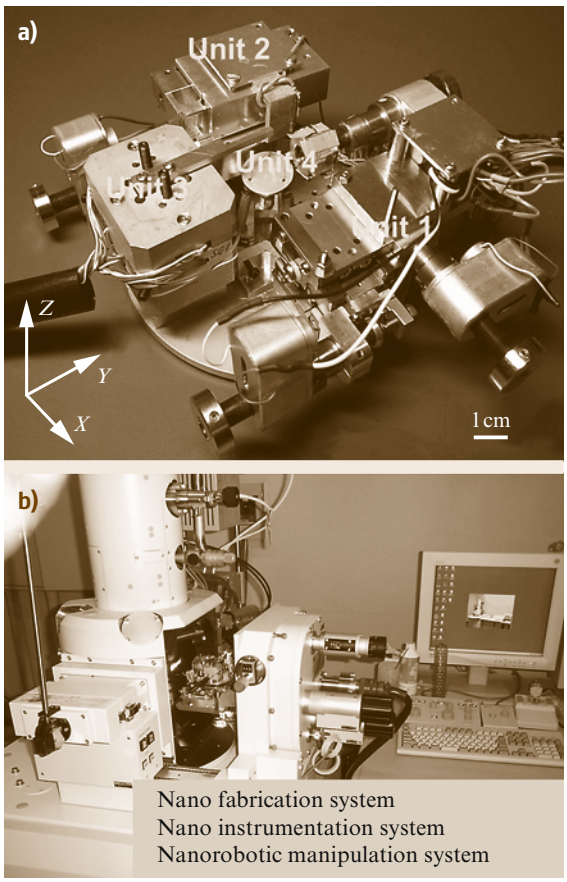


Fig.27.17a,b Nanorobotic system. (a) Nanorobotic manipulators. (b) System setup

of centimeters. The manipulation system is not only for nanomanipulation, but also for nanoassembly, nanoinstrumentation, and nanofabrication. Four-probe semiconductor measurements are perhaps the most complex manipulation that this system can perform, because it is necessary to actuate four probes independently by using four manipulators. With the advancement of nanotechnology, one could shrink the size of nanomanipulators and insert more DOFs inside the limited vacuum chamber of a microscope, and perhaps the molecular version of manipulators such as that dreamed of by *Drexler* could be realized [27.18].

For the construction of multi-walled carbon nanotube (MWNT)-based nanostructures, manipulators position and orient nanotubes for the fabrication of nanotube probes and emitters, for performing nanosoldering with electron-beam-induced deposition (EBID) [27.165], for the property characterization of single nanotubes for selection purposes and for characterizing junctions to test connection strength.

A nanolaboratory is shown in Fig. 27.17b. The nanolaboratory integrates a nanorobotic manipulation system with a nanoanalytical system and a nanofabrication system, and can be applied for manipulating nanomaterials, fabricating nanoscale building blocks, assembling nanodevices, and for in situ analysis of the properties of such materials, building blocks, and devices. Nanorobotic manipulation within the nanolaboratory has opened a new path for constructing nanosystems in 3-D space, and will create opportunities for new nanoinstrumentation and nanofabrication processes.

27.8.3 Nanorobotic Manipulation and Assembly

Nanomanipulation is a promising strategy for nanoassembly. Key techniques for nanoassembly include the structuring and characterization of nanoscale building blocks, the positioning and orientation control of the building blocks with nanoscale resolution, and effective connection techniques. Nanorobotic manipulation, which is characterized by multiple DOFs with both position and orientation controls, independently actuated multiprobes, and a real-time observation system, have been shown effective for assembling nanotube-based devices in 3-D space.

The well-defined geometries, exceptional mechanical properties, and extraordinary electric characteristics, among other outstanding physical properties of CNTs [27.166] qualify them for many potential applications, especially in nanoelectronics [27.167], NEMS, and other nanodevices [27.168]. For NEMS, some of the most important characteristics of nanotubes include their nanometer diameter, large aspect ratio (10−1000), TPa-scale Young's modulus [27.169–171], excellent elasticity [27.141], ultralow interlayer friction, excellent capability for field emission, various electric conductivities [27.172], high thermal conductivity [27.173], high current-carrying capability with essentially no heating [27.174], sensitivity of conductance to various physical or chemical changes, and charge-induced bond-length change.

Helical 3-D nanostructures, or nanocoils, have been synthesized from various materials, including helical carbon nanotubes [27.175] and zinc oxide nanobelts [27.176]. A new method of creating structures with nanoscale dimensions has recently been presented [27.177] and can be fabricated in a controllable way [27.178, 179]. The structures are created through a top-down fabrication process in which a strained nanometer-thick heteroepitaxial bilayer curls up to form 3-D structures with nanoscale features. Helical geometries and tubes with diameters between 10 nm and 10 μm have been achieved. Because of their interesting morphology, mechanical, electrical, and electromagnetic properties, potential applications of these nanostructures in NEMS include nanosprings [27.180], electromechanical sensors [27.181], magnetic field detectors, chemical or biological sensors, generators of magnetic beams, inductors, actuators, and high-performance electromagnetic wave absorbers.

NEMS based on individual carbon nanotubes and nanocoils are of increasing interest, indicating that capabilities for incorporating these individual building blocks at specific locations on a device must be developed. Random spreading [27.182], direct growth [27.183], self-assembly [27.184], dielectrophoretic assembly [27.185, 186], and nanomanipulation [27.187] have been demonstrated for positioning as-grown nanotubes on electrodes for the construction of these devices. However, for nanotube-based structures, nanorobotic assembly is still the only technique capable of in situ structuring, characterization, and assembly. Because the as-fabricated nanocoils are not free-standing from their substrate, nanorobotic assembly is virtually the only way to incorporate them into devices at present.

Nanorobotic Assembly of Carbon Nanotubes

Nanotube manipulation in two dimensions on a surface was first performed with an AFM by contact pushing on a substrate. Figure 27.18 shows the typical methods for 2-D pushing. Although similar to that shown in Fig. 27.15b, the same manipulation caused various results because nanotubes cannot be regarded as a 0-D point. The first demonstration was given by *Wong* et al. for measuring the mechanical properties of

a nanotube [27.188]. They adopt the method shown in Fig. 27.18b, i. e., to bend a nanotube by pushing one end of it and fixing the other end. The same strategy was used for the investigation of the behavior of nanotubes under large strain [27.189]. *Postma* et al. applied the strategies shown in Fig. 27.18c,d to obtain a kinked junction and crossed nanotubes [27.190]. *Hertel* et al. combined this technique with an inverse process, namely straightening by pushing along a bent tube, and realized the translation of the tube to another location [27.191] and between two electrodes to measure the conductivity [27.192]. This technique was also used to place a tube on another tube to form a single-electron transistor (SET) with a cross-junction of nanotubes [27.193]. Pushing-induced breaking (Fig. 27.18d) has also been demonstrated for a nanotube [27.191]. The simple assembly of two bent tubes and a straight one formed a Greek letter θ. To investigate the dynamics of rolling at the atomic level, rolling and sliding of a nanotube (as shown in Fig. 27.18e,f) are performed on graphite surfaces using an AFM [27.194]. Besides pushing and pulling, another important process is indentation. By indenting a surface, mechanical property characterization [27.195] and data storage [27.196] can be realized.

Manipulation of CNTs in 3-D space is important for assembling CNTs into structures and devices. Basic techniques for the nanorobotic manipulation of carbon nanotubes are shown in Fig. 27.19 [27.197]. These serve as the basis for handling, structuring, characterizing, and assembling NEMS.

The basic procedure is to pick up a single tube from nanotube soot (Fig. 27.19a). This has been shown first by using dielectrophoresis [27.142] through nanorobotic manipulation (Fig. 27.19b). By applying a bias between a sharp tip and a plane substrate, a nonuniform electric field can be generated between the tip and the substrate with the strongest field near the tip. This field can cause a tube to orient along the field or further *jump* to the tip by electrophoresis or dielectrophoresis (determined by the conductivity of objective tubes). Removing the bias, the tube can be placed at other locations at will. This method can be used for free-standing tubes on nanotube soot or on a rough surface on which surface van der Waals forces are generally weak. A tube strongly rooted in CNT soot or lying on a flat surface cannot be picked up in this way. The interaction between a tube and the atomic flat surface of AFM cantilever tip has been shown to be strong enough for picking up a tube onto the tip [27.198] (Fig. 27.19c). By using EBID, it is possible to pick up and fix a nanotube onto a probe [27.199] (Fig. 27.19d). For handling a tube, weak connection between the tube and the probe is desired.

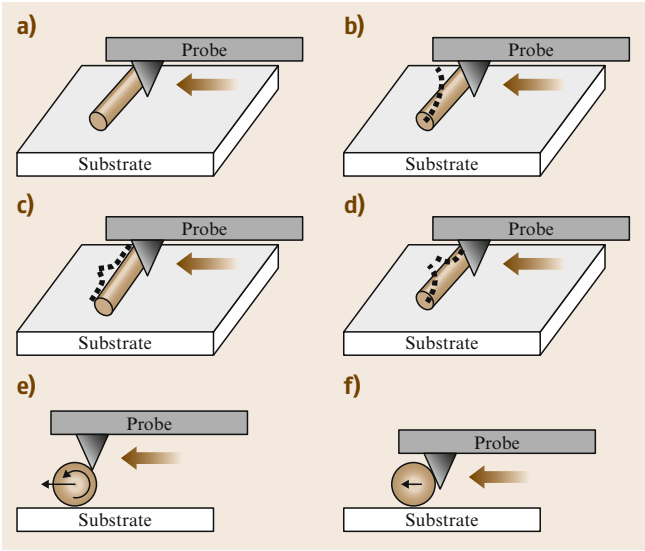


Fig.27.18a–f Two-dimensional manipulation of CNTs. Starting from the original state shown in **(a)**, pushing the tube at different site with different force may cause the tube to deform as in **(b)** and **(c)**, to break as in **(d)**, or to move as in **(e)** and **(f)**. **(a)** Original state. **(b)** Bending. **(c)** Kinking. **(d)** Breaking. **(e)** Rolling. **(f)** Sliding

Bending and buckling a CNT as shown in Fig. 27.19e,f are important for in situ property characterization of a nanotube [27.200], which is a simple way to obtain the Young's modulus of a nanotube without damaging the tube (if performed within its elastic range) and hence can be used for the selection of a tube with desired properties. By buckling an MWNT over its elastic limit, a kinked structure can be obtained [27.201]. To obtain any desired angle for a kinked junction it is possible to fix the shape of a buckled nanotube within its elastic limit by using EBID [27.202]. For a CNT, the maximum angular displacement will appear at the fixed left end under pure bending or at the middle point under pure buckling. A combination of these two kinds of loads will achieve a controllable position of the kinked point and a desired kink angle. If the deformation is within the elastic limit of the nanotube, it will recover as the load is released. To avoid this, EBID can be applied at the kinked point to fix the shape.

Stretching a nanotube between two probes or a probe and a substrate has generated several interesting results (Fig. 27.19g). The first demonstration of 3-D nanomanipulation of nanotubes took this as an example to show the breaking mechanism and to measure the tensile strength of CNTs [27.141]. By breaking an MWNT in a controlled manner, interesting nanodevices have been fabricated. This technique – destructive fabrication – has been presented to get sharpened and lay-

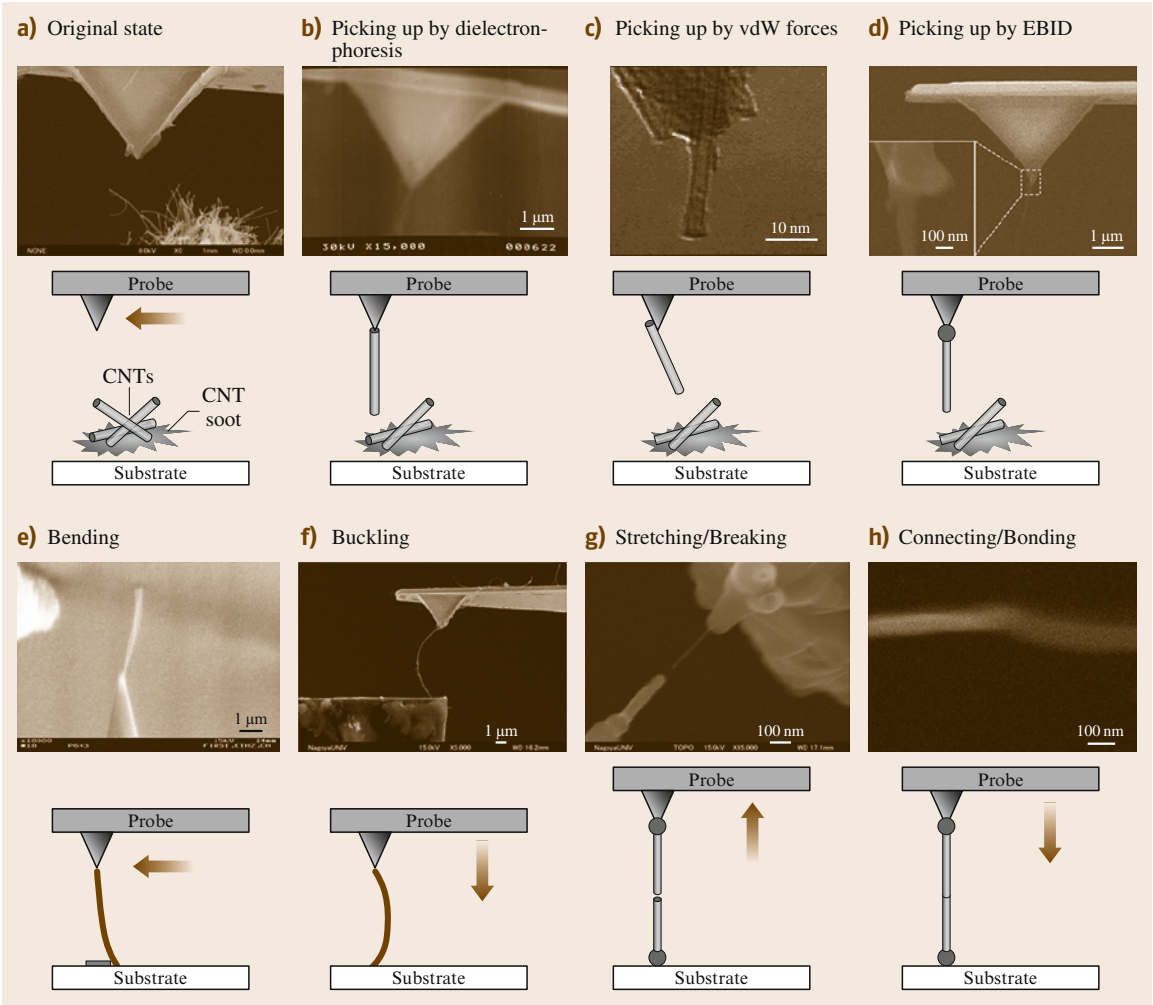


Fig.27.19a–h Nanorobotic manipulation of CNTs. The basic technique is to pick up an individual tube from CNT soot (as in **(a)**) or from an oriented array; **(b)** shows a free-standing nanotube picked up by dielectrophoresis generated by a nonuniform electric field between the probe and substrate, **(c)** (after [27.196]) and **(d)** show the same manipulation by contacting a tube with the probe surface or fixing (e.g., with EBID) a tube to the tip (inset shows the EBID deposit). Vertical manipulation of nanotubes includes bending **(e)**, buckling **(f)**, stretching/breaking **(g)**, and connecting/bonding **(h)**. All examples with the exception of **(i)** are from the authors' work

ered structures of nanotubes, and to improve the length control of a nanotube [27.201]. Typically, a layered and a sharpened structure can be obtained from this process, similar to that achieved from electric pulses [27.202]. Bearing motion has also been observed in an incompletely broken MWNT [27.201]. The interlayer friction has been shown to be very small [27.203, 204].

The reverse process, namely the connection of broken tubes (Fig. 27.19h), has been demonstrated recently, and the mechanism is revealed as rebonding of unclosed dangling bonds at the ends of broken tubes [27.205]. Based on this interesting phenomenon, mechanochemical nanorobotic assembly has been performed [27.206].

Assembly of nanotubes is a fundamental technology for enabling nanodevices. The most important tasks include the connection of nanotubes and placing of nanotubes onto electrodes. Pure nanotube circuits [27.206] created by interconnecting nanotubes of different diameters and chirality could lead to further size reductions in devices. Nanotube intermolecular and intramolecular junctions are basic elements for such systems [27.207]. Room-temperature (RT) single-electron transistors (SETs) [27.208] have been shown

with a short (≈ 20 nm) nanotube section that is created by inducing local barriers into the tube with an AFM, and Coulomb charging has been observed. With a cross-junction of two single-walled carbon nanotubes (SWNTs) (semiconducting/metallic), three- and four-terminal electronic devices have been made [27.209]. A suspended cross-junction can function as an electromechanical nonvolatile memory [27.210].

Although some kinds of junctions have been synthesized with chemical methods, there is no evidence yet showing that a self-assembly-based approach can provide more complex structures. SPMs were also used to fabricate junctions, but they are limited to a 2-D plane. We have presented 3-D nanorobotic manipulation-based nanoassembly, which is a promising strategy both for the fabrication of nanotube junctions and for the construction of more complex nanodevices with such junctions as well.

Nanotube junctions can be classified into different types by the kinds of components (SWNTs or MWNTs), geometric configurations (V kink, I, X cross, T, Y branch, and 3-D junctions) conductivity (metallic or semiconducting), and connection methods (intermolecular (connected with van der Waals force, EBID, etc.) or intramolecular (connected with chemical bonds) junctions). Here we show the fabrication of several kinds of MWNT junctions by emphasizing the connection methods. These methods will also be effective for SWNT junctions. Figure 27.20 shows CNT junctions constructed by connecting with van der Waals forces (a), joining by electron-beam-induced deposition (b), and bonding through mechanochemistry (c).

Figure 27.20a shows a T-junction connected with van der Waals forces, which is fabricated by positioning the tip of an MWNT onto another MWNT until they form a bond. The contact is checked by measuring the shear connection force.

EBID provides a soldering method to obtain stronger nanotube junctions than those connected through van der Waals forces. Hence, if the strength of nanostructures is emphasized, EBID can be applied. Figure 27.20b shows an MWNT junction connected through EBID, in which the upper MWNT is a single one with 20 nm in diameter and the lower one is a bundle of MWNTs with an extruded single CNT with Ø 30 nm. The development of conventional EBID has been limited by the expensive electron filament used and low productivity. We have presented a parallel EBID system by using CNTs as emitters because of their excellent field-emission properties [27.211]. The feasibility of parallel EBID is presented. It is a promising strategy for large-scale fabrications of nanotube junctions. As in its macroscale counterpart, welding, EBID works by adding materials to obtain stronger

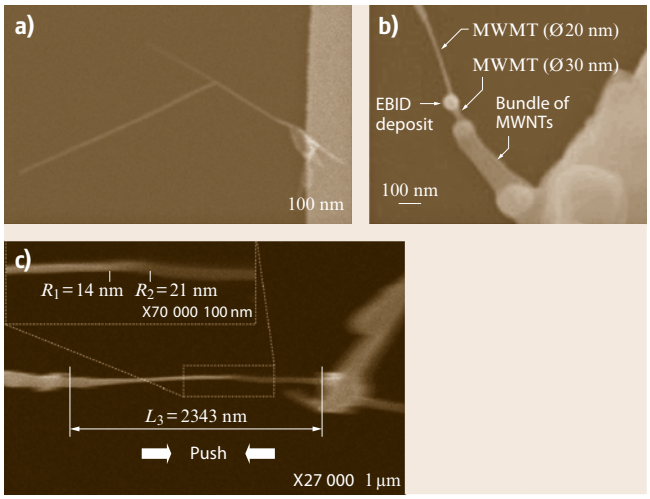


Fig.27.20a–c MWNT junctions. **(a)** MWNTs connected with van der Waals forces. **(b)** MWNTs joined with EBID. **(c)** MWNTs bonded with a mechanochemical reaction

connections, but in some cases, the added material might influence normal functions for nanosystems. So, EBID is mainly applied to nanostructures rather than nanomechanisms.

To construct stronger junctions without adding additional materials, mechanochemical nanorobotic assembly is an important strategy. Mechanochemical nanorobotic assembly is based on solid-phase chemical reactions, or mechanosynthesis, which is defined as chemical synthesis controlled by mechanical systems operating with atomic-scale precision, enabling direct positional selection of reaction sites [27.18]. By picking up atoms with dangling bonds rather than natural atoms only, it is easier to form primary bonds, which provides a simple but strong connection. Destructive fabrication provides a way to form dangling bonds at the ends of broken tubes. Some of the dangling bonds may close with neighboring atoms, but generally a few bonds will remain dangling. A nanotube with dangling bonds at its end will bind more easily to another to form intramolecular junctions. Figure 27.20c shows such a junction.

Three-dimensional nanorobotic manipulation has opened a new route for the structuring and assembly of nanotubes into nanodevices. However, at present nanomanipulation is still performed in a serial manner with master–slave control, which is not a large-scale production-oriented technique. Nevertheless, with advances in the exploration of mesoscopic physics, better control on the synthesis of nanotubes, more accurate actuators, and effective tools for manipulation, high-speed and automatic nanoassembly will be possible. Another approach might be parallel assembly by positioning building blocks with an array of probes [27.212] and

joining them together simultaneously, e.g., with the parallel EBID [27.199] we presented. Further steps might progress towards exponential assembly [27.213], and in the far future to self-replicating assembly [27.18].

Nanorobotic Assembly of Nanocoils

The construction of nanocoil-based NEMS involves the assembly of as-grown or as-fabricated nanocoils, which is a significant challenge from a fabrication standpoint. Focusing on the unique aspects of manipulating nanocoils due to their helical geometry, high elasticity, single-end fixation, and strong adhesion of the coils to the substrate from wet etching, a series of new processes has been presented using a manipulator (MM3A, Kleindiek) installed in an SEM (Zeiss DSM962). As-fabricated SiGe/Si bilayer nanocoils (thickness 20 nm without Cr layer or 41 nm with Cr layer; diameter $D = 3.4\,\mu m$) are manipulated. Special tools have been fabricated including a nanohook prepared by controlled *tip-crashing* of a commercially available tungsten sharp probe (Picoprobe T-4-10–1 mm and T-4-10) onto a substrate, and a *sticky* probe prepared by tip dipping into a double-sided SEM silver conductive tape (Ted Pella, Inc.). As shown in Fig. 27.21, experiments demonstrate that nanocoils can be released from a chip by lateral pushing, picked up with a nanohook or a *sticky* probe, and placed between the probe/hook and another probe or an AFM cantilever (Nano-probe, NP-S). Axial pulling/pushing, radial compressing/releasing, and bending/buckling have also been demonstrated. These processes have shown the effectiveness of manipulation for the characterization of coil-shaped nanostructures and their assembly for NEMS, which have been otherwise unavailable.

Configurations of nanodevices based on individual nanocoils are shown in Fig. 27.22. Cantilevered nanocoils as shown in Fig. 27.22a can serve as nanosprings. Nanoelectromagnets, chemical sensors, and nanoinductors involve nanocoils bridged between two electrodes as shown in Fig. 27.22b. Electromechanical sensors can use a similar configuration but with one end connected to a moveable electrode as shown in Fig. 27.22c. Mechanical stiffness and electric conductivity are fundamental properties for these devices that must be further investigated.

As shown in Fig. 27.21h, axial pulling is used to measure the stiffness of a nanocoil. A series of SEM images are analyzed to extract the AFM tip displace-

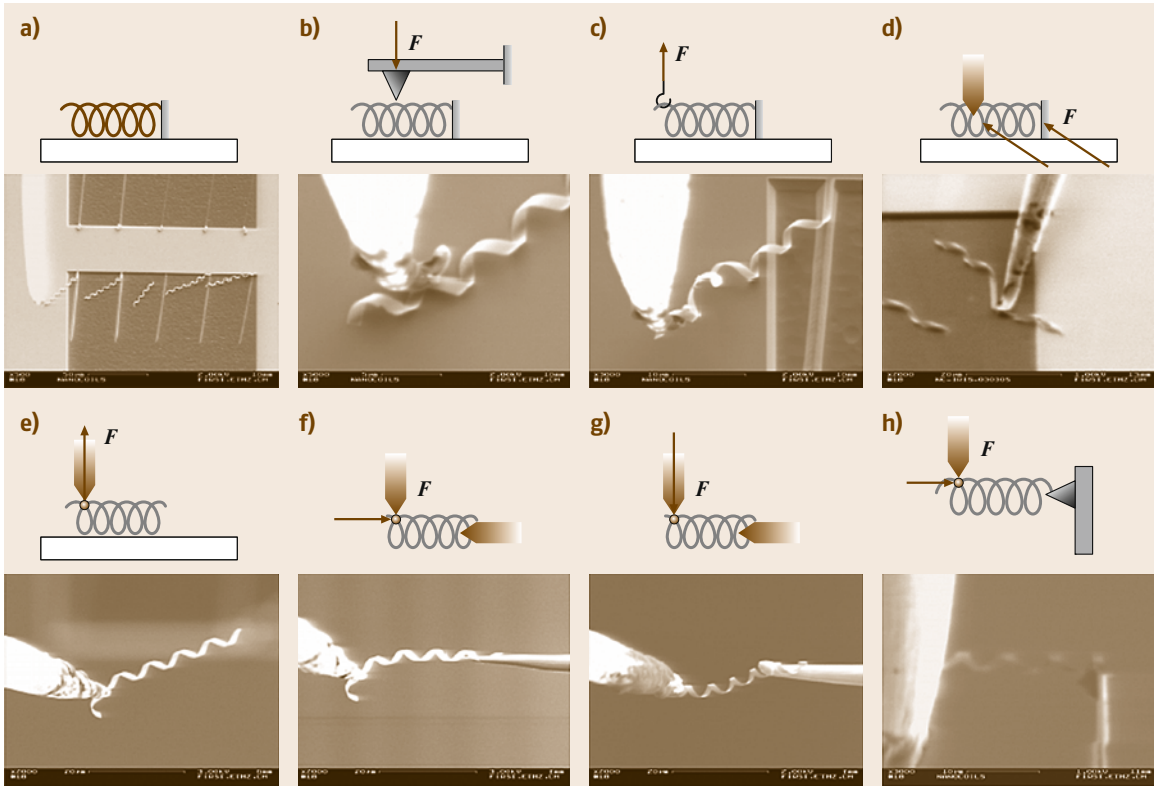


Fig.27.21a–h Nanorobotic manipulation of nanocoils **(a)** original state, **(b)** compressing/releasing, **(c)** hooking, **(d)** lateral pushing/breaking, **(e)** picking, **(f)** placing/inserting, **(g)** bending, and **(h)** pushing and pulling

ment and the nanospring deformation, i. e., the relative displacement of the probe from the AFM tip. From this displacement data and the known stiffness of the AFM cantilever, the tensile force acting on the nanospring versus the nanospring deformation was plotted. The deformation of the nanospring was measured relative to the first measurement point. This was necessary because the proper attachment of the nanospring to the AFM cantilever must be verified. Afterwards, it was not possible to return to the point of zero deformation. Instead, the experimental data as presented in Fig. 27.22d has been shifted such that with the calculated linear elastic spring stiffness the line begins at zero force and zero deformation. From Fig. 27.22d, the stiffness of the spring was estimated to be 0.0233 N/m. The linear elastic region of the nanospring extends to a deformation of 4.5 μm. An exponential approximation was fitted to the nonlinear region. When the applied force reached 0.176 μN, the attachment between the nanospring and the AFM cantilever broke. Finite-element simulation (ANSYS 9.0) was used to validate the experimental data [27.181]. Since the exact region of attachment cannot be identified from the SEM images, simulations were conducted for 4, 4.5, and 5 turns to estimate the possible range according to the apparent number of turns of the nanospring. The nanosprings in the simulations were fixed at one end and had an axial load of 0.106 μN applied at the other end. The simulation results for the spring with four turns yield a stiffness of 0.0302 N/m; for the nanospring with 5 turns it is 0.0191 N/m. The measured stiffness falls within this range with 22.0% above the minimum value and 22.8% below the maximum value, and very close to the stiffness of a 4.5-turn nanospring, which has a stiffness of 0.0230 N/m according to the simulation.

Figure 27.22e shows the results from electrical characterization experiments on a nanospring with 11 turns using the configuration shown in Fig. 27.21g. The *I–V* curve is nonlinear, which may be caused by the resistance change of the semiconductive bilayer due to ohmic heating. Another possible reason is the decrease in contact resistance caused by thermal stress. The maximum current was found to be 0.159 mA under an 8.8 V bias. Higher voltage causes the nanospring to *blow off*. From the fast scanning screen of the SEM, an extension of the nanospring on probes was observed around the peak current so that the current does not drop abruptly. At 9.4 V, the extended nanospring is broken down, causing an abrupt drop in the *I–V* curve.

From fabrication and characterization results, the helical nanostructures appear to be suitable to function as inductors. They would allow further miniaturization

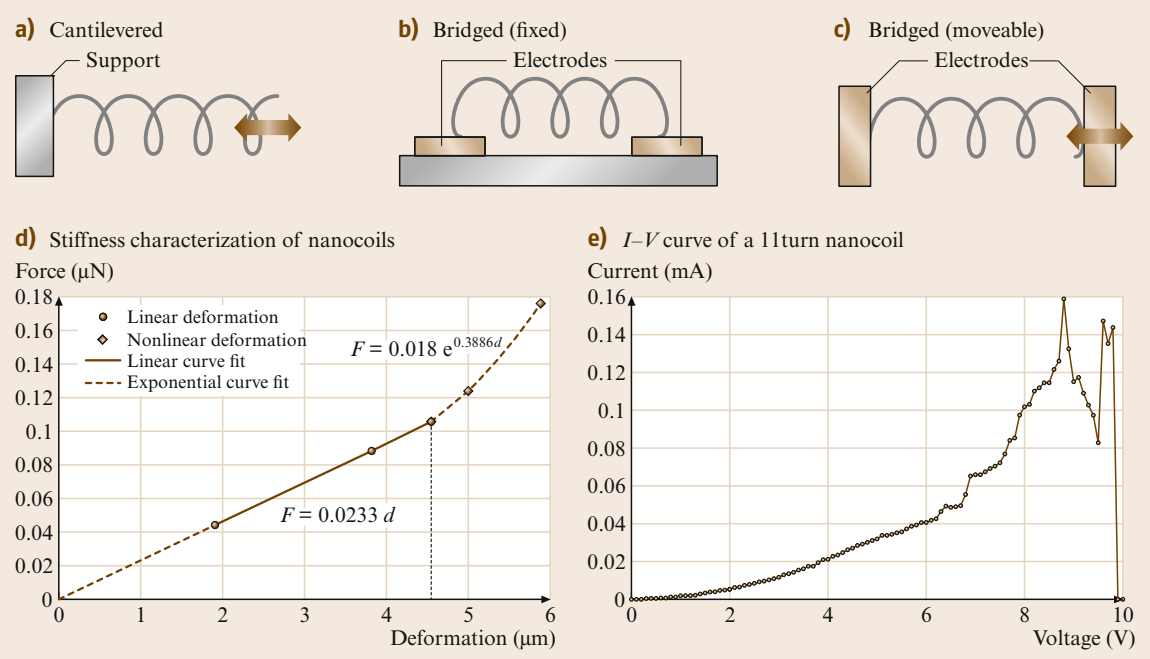


Fig.27.22a–e Nanocoil-based devices. Cantilevered nanocoils **(a)** can serve as nanosprings. Nanoelectromagnets, chemical sensors, and nanoinductors involve nanocoils bridged between two electrodes **(b)**. Electromechanical sensors can use a similar configuration but with one end connected to a moveable electrode **(c)**. Mechanical stiffness **(d)** and electric conductivity **(e)** are basic properties of interest for these devices

compared to state-of-the-art microinductors. For this purpose, higher doping of the bilayer and an additional metal layer would result in the required conductance. Conductance, inductance, and quality factor can be further improved if, after curling up, additional metal is electroplated onto the helical structures. Moreover, a semiconductive helical structure, when functionalized with binding molecules, can be used for chemical sensing under the same principle as demonstrated with other types of nanostructures. With bilayers in the range of a few monolayers, the resulting structures would exhibit very high surface-to-volume ratio with the whole surface exposed to an incoming analyst.

27.8.4 Nanorobotic Systems

Nanorobotic devices involve tools, sensors, and actuators at the nanometer scale. Shrinking device size makes it possible to manipulate nanosized objects with nanosized tools, measure mass in femtogram ranges, sense force at piconewton scales, and induce GHz motion, among other amazing advancements.

Top-down and bottom-up strategies for manufacturing such nanodevices have been independently investigated by a variety of researchers. Top-down strategies are based on nanofabrication and include technologies such as nanolithography, nanoimprinting, and chemical etching. Presently, these are 2-D fabrication processes with relatively low resolution. Bottom-up strategies are assembly-based techniques. At present, these strategies include such techniques as self-assembly, dip-pen lithography, and directed self-assembly. These techniques can generate regular nanopatterns at large scales. With the ability to position and orient nanoscale objects, nanorobotic manipulation is an enabling technology for structuring, characterizing and assembling many types of nanosystems [27.197]. By combining bottom-up and top-down processes, a hybrid nanorobotic approach based on nanorobotic manipulation provides a third way to fabricate NEMS by structuring as-grown nanomaterials or nanostructures. This new nanomanufacturing

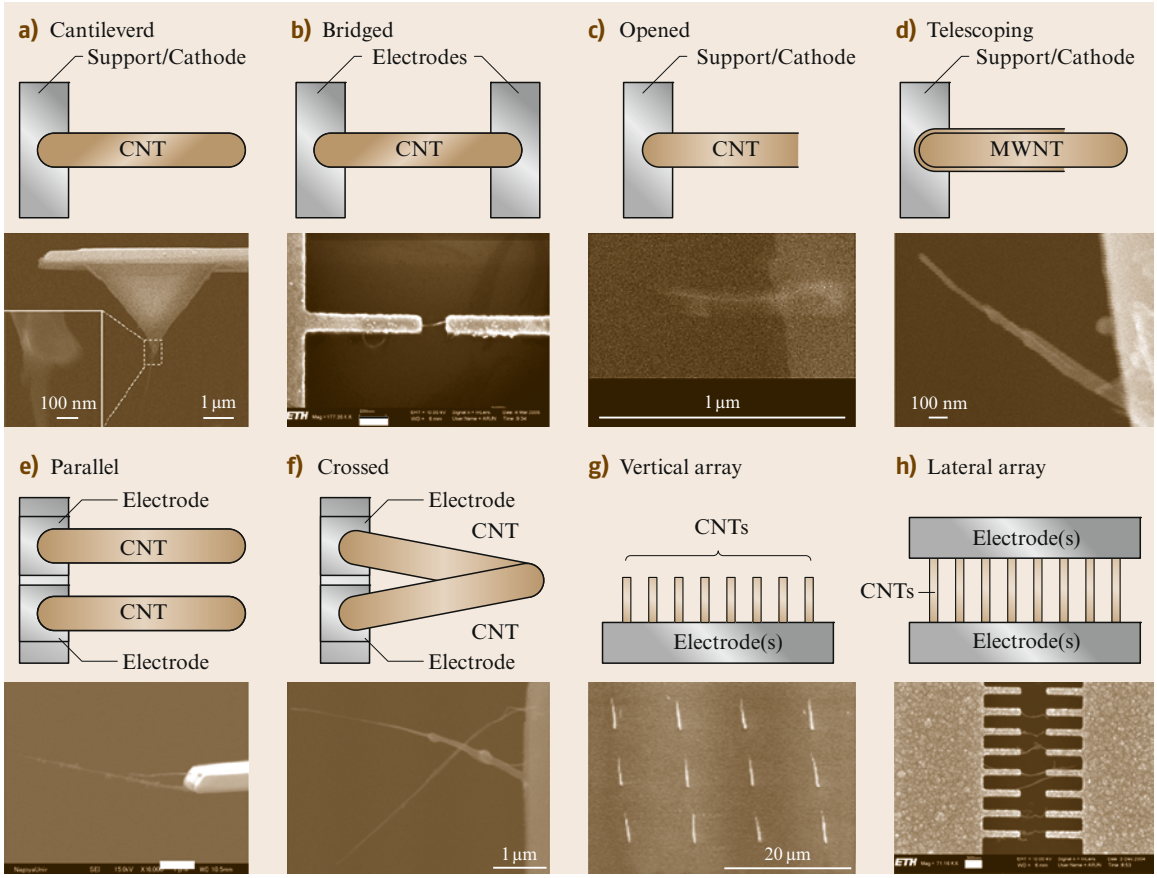


Fig.27.23a–h Configurations of individual nanotube-based NEMS. *Scale bars*: **(a)** 1 µm (*inset*: 100 nm), **(b)** 200 nm, **(c)** 1 µm, **(d)** 100 nm, **(e)** and **(f)** 1 µm, **(g)** 20 µm, and **(h)** 300 nm. All examples are from the authors' work

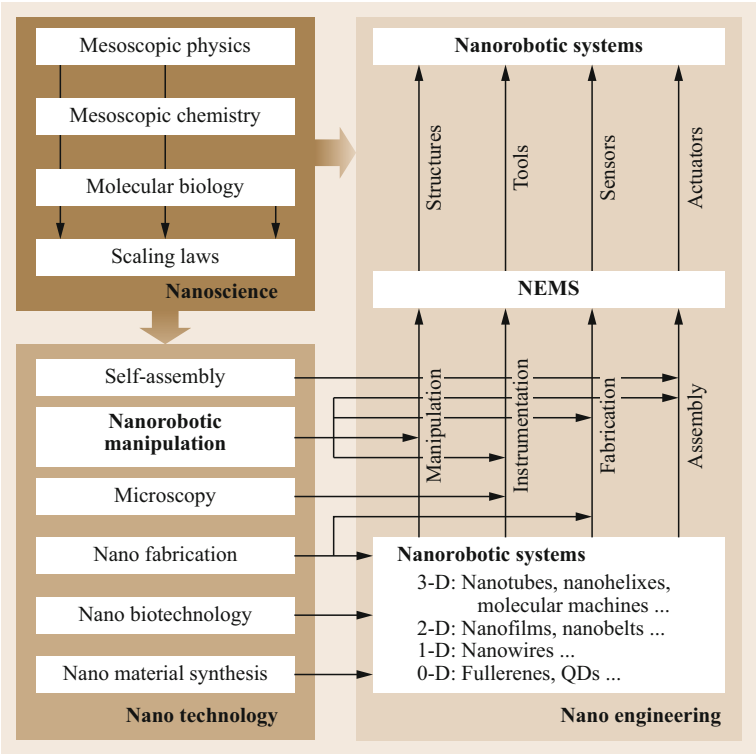


Fig. 27.24 A roadmap for nanorobotic systems

technique can be used to create complex 3-D nanodevices with such building blocks. Nanomaterial science, bio-nanotechnology, and nanoelectronics will also benefit from advances in nanorobotic assembly.

The configurations of nanotools, sensors, and actuators based on individual nanotubes that have been experimentally demonstrated are summarized as shown in Fig. 27.23.

For detecting deep and narrow features on a surface, cantilevered nanotubes (Fig. 27.23a, [27.199]) have been demonstrated as probe tips for an AFM [27.206], a STM, and other types of SPM. Nanotubes provide ultrasmall diameters, ultralarge aspect ratios, and excellent mechanical properties. Manual assembly [27.214], and direct growth [27.215] have proven effective for their construction. Cantilevered nanotubes have also been demonstrated as probes for the measurement of ultrasmall physical quantities, such as femtogram masses [27.170], piconewton-order force sensors, and mass flow sensors [27.197] on the basis of their static deflections or change of resonant frequencies detected within an electron microscope. Deflections cannot be measured from micrographs in real time, which limits the application of this kind of sensor. Interelectrode distance changes cause emission current variation of a nanotube emitter and may serve as a candidate to replace microscope images. Bridged individual nanotubes (Fig. 27.23b, [27.185]) have been the basis for electric characterization. Opened nanotubes (Fig. 27.23c, [27.216]) can serve as atomic or molecular containers, thermometers [27.217], or spot welders [27.218]. The electrostatic deflection of a nanotube has been used to construct a relay [27.219]. A new family of nanotube actuators can be constructed by taking advantage of the ultralow interlayer friction of a multiwalled nanotubes. Linear bearings based on telescoping nanotubes and a microactuator with a nanotube as a rotation bearing have been demonstrated [27.203, 220], and batch fabrication has been realized based on dielectrophoretically assembled arrays [27.221]. A preliminary experiment on a promising nanotube linear motor with field-emission current serving as position feedback has been shown with nanorobotic manipulation (Fig. 27.23d, [27.216]). Cantilevered dual nanotubes have been demonstrated as nanotweezers [27.222] and nanoscissors (Fig. 27.23e) [27.179] by manual and nanorobotic assembly, respectively. Based on electric resistance change under different temperatures, nanotube thermal probes (Fig. 27.23f) have been demonstrated for measuring the temperature at precise locations. These thermal probes are more advantageous than nanotube-based thermometers because the latter

require TEM imaging. The integration of the aforementioned devices can be realized using the configurations shown in Fig. 27.23g,h [27.183]. The arrays of individual nanotubes can also be used to fabricate nanosensors, such as position encoders [27.223].

Nanorobotic systems remains a rich research field with a large number of open problems. New materials such as nanowires, nanobelts, graphene, and polymer at the nanoscale will enable a new family of sensors and actuators for the detection and actuation of ultrasmall quantities or objects with ultrahigh precision and frequencies. Through random spreading, direct growth, optical tweezers, and nanorobotic manipulation, prototypes have been demonstrated. However, for integration into nanosystems, self-assembly processes will become increasingly important. Among them, we believe that dielectrophoretic nanoassembly will play a significant role for large-scale production of regular 2-D structures.

A roadmap for nanorobotic systems [27.224] is shown in Fig. 27.24. Knowledge from mesoscopic physics, mesoscopic/supramolecular chemistry, and molecular biology at the nanometer scale converges to form the field. Various disciplines contribute to nanorobotics, including nanomaterial synthesis, nanobiotechnology, and microscopy for imaging and characterization. Such topics as self-assembly, nanorobotic assembly, and hybrid nanomanufacturing approaches for assembling nano building blocks into structures, tools, sensors, and actuators are considered areas of nanorobotic study. A current focus of nanorobotics is on the fabrication of NEMS and other nanosystems, which may serve as components for future nanorobots. The main goals of nanorobotics are to provide effective tools for the experimental exploration of the nanoworld, and to push the boundaries of this exploration from a robotics research perspective.

27.9 Conclusions

Despite the claims of many *futurists*, such as Issac *Asimov*'s legendary submarine Proteus inside the human body [27.225] and *Robert A. Freitas's* nanomedical robots [27.226], the form that micro/nanorobots of the future will take and the tasks they will actually perform remain unclear. However, it is clear that technology is progressing towards the construction of intelligent sensors, actuators, and systems on small scales. These will serve as both the tools to be used for fabricating future micro/nanorobots as well as the components from which these robots may be developed. Shrinking device size to these dimensions presents many fascinating opportunities such as manipulating nano-objects with nanotools, measuring mass in femtogram ranges, sensing forces at piconewton scales, and inducing GHz motion, among other new possibilities waiting to be discovered. These capabilities will, of course, drive the tasks that future micro/nanorobots constructed by and with MEMS/NEMS will perform.

Video-References

VIDEO 11 Artificial bacterial flagella
available from http://handbookofrobotics.org/view-chapter/27/videodetails/11

VIDEO 12 The electromagnetic control of an untethered microrobot
available from http://handbookofrobotics.org/view-chapter/27/videodetails/12

VIDEO 13 A transversely magnetized rod-shaped microrobot
available from http://handbookofrobotics.org/view-chapter/27/videodetails/13

VIDEO 489 Attogram mass delivery from a carbon nanotube
available from http://handbookofrobotics.org/view-chapter/27/videodetails/489

VIDEO 490 Multi-beam bilateral teleoperation of holographic optical tweezers
available from http://handbookofrobotics.org/view-chapter/27/videodetails/490

VIDEO 491 High-speed magnetic microrobot actuation in a microfluidic chip by a fine V-groove surface
available from http://handbookofrobotics.org/view-chapter/27/videodetails/491

VIDEO 492 Linear-to-rotary motion converters for three-dimensional microscopy
available from http://handbookofrobotics.org/view-chapter/27/videodetails/492

References

27.1 R.P. Feynman: There's plenty of room at the bottom, Caltech Eng. Sci. **23**, 22–36 (1960)

27.2 R.S. Muller: Microdynamics, Sens. Actuators A **21**(1), 1–8 (1990)

27.3 A.M. Flynn, R.A. Brooks, W.M. Wells, D.S. Barrett: The world's largest one cubic inch robot, IEEE Micro Electro Mech. Syst. (MEMS) (1989) pp. 98–101

27.4 W. Trimmer, R. Jebens: Actuators for micro robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 1547–1552

27.5 S. Fatikow, U. Rembold: An automated microrobot-based desktop station for micro assembly and handling of micro-objects, IEEE Conf. Emerg. Technol. Fact. Autom. (EFTA'96) (1996) pp. 586–592

27.6 B.J. Nelson, Y. Zhou, B. Vikramaditya: Sensor-based microassembly of hybrid MEMS devices, IEEE Control Syst. Mag. **18**, 35–45 (1998)

27.7 K. Suzumori, T. Miyagawa, M. Kimura, Y. Hasegawa: Micro inspection robot for 1-in pipes, IEEE/ASME Trans. Mechatron. **4**, 286–292 (1999)

27.8 M. Takeda: Applications of MEMS to industrial inspection, Proc. 14th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (2001) pp. 182–191

27.9 T. Frank: Two-Axis electrodynamic micropositioning devices, J. Micromech. Microeng. **8**, 114–118 (1989)

27.10 N. Kawahara, N. Kawahara, T. Suto, T. Hirano, Y. Ishikawa, T. Kitahara, N. Ooyama, T. Ataka: Microfactories: New applications of micromachine technology to the manufacture of small products, Res. J. Microsyst. Technol. **3**, 37–41 (1997)

27.11 Y. Sun, B.J. Nelson: Microrobotic cell injection, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 620–625

27.12 P. Dario, M.C. Carrozza, L. Lencioni, B. Magnani, S. Dapos Attanasio: A micro robotic system for colonoscopy, Proc. Int. Conf. Robotics Autom. (ICRA) (1997) pp. 1567–1572

27.13 F. Tendick, S.S. Sastry, R.S. Fearing, M. Cohn: Application of micromechatronics in minimally invasive surgery, IEEE/ASME Trans. Mechatron. **3**, 34–42 (1998)

27.14 G. Iddan, G. Meron, A. Glukhovsky, P. Swain: Wireless capsule endoscopy, Nature **405**, 417 (2000)

27.15 K.B. Yesin, K. Vollmers, B.J. Nelson: Analysis and design of wireless magnetically guided microrobots in body fluids, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 1333–1338

27.16 M.C. Roco, R.S. Williams, P. Alivisatos: *Nanotechnology Research Directions. Vision for Nanotechnology in the Next Decade* (Kluwer, Dordrecht 2000)

27.17 M.L. Downey, D.T. Moore, G.R. Bachula, D.M. Etter, E.F. Carey, L.A. Perine: *National Nanotechnology Initiative: Leading to the Next Industrial Revolution, A Report by the Interagency Working Group on Nanoscience, Engineering and Technology* (Committee on Technology, National Science and Technology Council, Washington 2000)

27.18 K. Drexler: *Nanosystems: Molecular Machinery, Manufacturing and Computation* (Wiley, New York 1992)

27.19 G. Binnig, H. Rohrer, C. Gerber, E. Weibel: Surface studies by scanning tunneling microscopy, Phys. Rev. Lett. **49**, 57–61 (1982)

27.20 W.F. Degrado: Design of peptides and proteins, Adv. Protein Chem. **39**, 51–124 (1998)

27.21 G.M. Whitesides, B. Grzybowski: Self-assembly at all scales, Science **295**, 2418–2421 (2002)

27.22 R. Fearing: Survey of sticking effects for micro-parts, Proc. IEEE/RSJ Int. Conf. Int. Robots Syst. (1995) pp. 212–217

27.23 E.L. Wolf: *Nanophysics and Nanotechnology* (Wiley-VCH, Weinheim 2004)

27.24 T. Ebefors, G. Stemme: Microrobotics. In: *The MEMS Handbook*, ed. by M. Gad-el-Hak (CRC, Boca Raton 2002)

27.25 C.-J. Kim, A.P. Pisano, R.S. Muller: Silicon-processed overhanging microgripper, IEEE/ASME J. Microelectromechanical Syst. **1**, 31–36 (1992)

27.26 C. Liu, T. Tsao, Y.-C. Tai, C.-M. Ho: Surface micromachined magnetic actuators, Proc. 7th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1994) pp. 57–62

27.27 J. Judy, D.L. Polla, W.P. Robbins: A linear piezoelectric stepper motor with submicron displacement and centimeter travel, IEEE Trans. Ultrason. Ferroelectr. Freq. Control **37**, 428–437 (1990)

27.28 K. Nakamura, H. Ogura, S. Maeda, U. Sangawa, S. Aoki, T. Sato: Evaluation of the micro wobbler motor fabricated by concentric build-up process, Proc. 8th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1995) pp. 374–379

27.29 A. Teshigahara, M. Watanabe, N. Kawahara, I. Ohtsuka, T. Hattori: Performance of a 7-mm microfabricated car, IEEE/ASME J. Microelectromechanical Syst. **4**, 76–80 (1995)

27.30 K.R. Udayakumar, S.F. Bart, A.M. Flynn, J. Chen, L.S. Tavrow, L.E. Cross, R.A. Brooks, D.J. Ehrlich: Ferroelectric thin film ultrasonic micromotors, Proc. 4th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1991) pp. 109–113

27.31 P. Dario, R. Valleggi, M.C. Carrozza, M.C. Montesi, M. Cocco: Review – Microactuators for microrobots: A critical survey, J. Micromech. Microeng. **2**, 141–157 (1992)

27.32 I. Shimoyama: Scaling in microrobots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (1995) pp. 208–211

27.33 R.S. Fearing: Powering 3-dimensional microrobots: power density limitations, tutorial on Micro Mechatronics and Micro Robotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998)

27.34 R.G. Gilbertson, J.D. Busch: A survey of micro-actuator technologies for future spacecraft missions, J. Br. Interplanet. Soc. **49**, 129–138 (1996)

27.35 M. Mehregany, P. Nagarkar, S.D. Senturia, J.H. Lang: Operation of microfabricated harmonic and ordinary side-drive motors, Proc. 3rd IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1990) pp. 1–8

27.36 Y.C. Tai, L.S. Fan, R.S. Mulle: IC-processed micromotors: design, technology, and testing, Proc. 2nd IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1989) pp. 1–6

27.37 T. Ohnstein, T. Fukiura, J. Ridley, U. Bonne: Micromachined silicon microvalve, Proc. 3rd IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1990) pp. 95–99

27.38 L.Y. Chen, S.L. Zhang, J.J. Yao, D.C. Thomas, N.C. MacDonald: Selective chemical vapor deposition of tungsten for microdynamic structures, Proc. 2nd IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1989) pp. 82–87

27.39 K. Yanagisawa, H. Kuwano, A. Tago: An electromagnetically driven microvalve, Proc. 7th Int. Conf. Solid-State Sens. Actuators (1993) pp. 102–105

27.40 M. Esashi, S. Shoji, A. Nakano: Normally close microvalve and micropump fabricated on a silicon wafer, Proc. 2nd IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1989) pp. 29–34

27.41 R. Petrucci, K. Simmons: An introduction to piezoelectric crystals, Sens. J. Appl.Sens. Technol. **11**(5), 26–31 (1994)

27.42 J. Goldstein, D. Newbury, D. Joy, C. Lyman, P. Echlin, E. Lifshin, L. Sawyer, J. Michael: *Scanning Electron Microscopy and X-ray Microanalysis* (Kluwer, New York 2003)

27.43 G. Binnig, H. Rohrer: In touch with atoms, Rev. Mod. Phys. **71**, S324–S330 (1999)

27.44 G. Binnig, C.F. Quate, C. Gerber: Atomic force microscope, Phys. Rev. Lett. **56**, 93–96 (1986)

27.45 M.J. Doktycz, C.J. Sullivan, P.R. Hoyt, D.A. Pelletier, S. Wu, D.P. Allison: AFM imaging of bacteria in liquid media immobilized on gelatin coated mica surfaces, Ultramicroscopy **97**, 209–216 (2003)

27.46 S.A. Campbell: *The Science and Engineering of Microelectronic Fabrication* (Oxford Univ. Press, New York 2001)

27.47 C.J. Jaeger: *Introduction to Microelectronic Fabrication* (Prentice Hall, Upper Saddle River 2002)

27.48 J.D. Plummer, M.D. Deal, P.B. Griffin: *Silicon VLSI Technology* (Prentice Hall, Upper Saddle River 2000)

27.49 M. Gad-el-Hak (Ed.): *The MEMS Handbook* (CRC, Boca Raton 2002)

27.50 T.-R. Hsu: *MEMS and Microsystems Design and Manufacture* (McGraw-Hill, New York 2002)

27.51 G.T.A. Kovacs: *Micromachined Transducers Sourcebook* (McGraw-Hill, New York 1998)

27.52 G.T.A. Kovacs, N.I. Maluf, K.A. Petersen: Bulk micromachining of silicon, Proc. IEEE Int. Conf. Robotics Autom. (1998) pp. 1536–1551

27.53 P. Rai-Choudhury (Ed.): *Handbook of Microlithography, Micromachining and Microfabrication* (SPIE, Bellingham 1997)

27.54 S.Y. Chou: Nano-imprint lithography and lithographically induced self-assembly, MRS Bulletin **26**, 512–517 (2001)

27.55 M.A. Herman: *Molecular Beam Epitaxy: Fundamentals and Current Status* (Springer, New York 1996)

27.56 J.S. Frood, G.J. Davis, W.T. Tsang: *Chemical Beam Epitaxy and Related Techniques* (Wiley, New York 1997)

27.57 S. Mahajan, K.S.S. Harsha: *Principles of Growth and Processing of Semiconductors* (McGraw-Hill, New York 1999)

27.58 C.A. Mirkin: Dip-pen nanolithography: automated fabrication of custom multicomponent, sub-100 nanometer surface architectures, MRS Bulletin **26**, 535–538 (2001)

27.59 C.A. Harper: *Electronic Packaging and Interconnection Handbook* (McGraw-Hill, New York 2000)

27.60 K.F. Bohringer, R.S. Fearing, K.Y. Goldberg: Microassembly. In: *Handbook of Industrial Robotics*, ed. by S. Nof (Wiley, New York 1999) pp. 1045–1066

27.61 G. Yang, J.A. Gaines, B.J. Nelson: A supervisory wafer-level 3D microassembly system for hybrid MEMS fabrication, J. Intell. Robotics Syst. **37**, 43–68 (2003)

27.62 P. Dario, M. Carrozza, N. Croce, M. Montesi, M. Cocco: Non-traditional technologies for microfabrication, J. Micromech. Microeng. **5**, 64–71 (1995)

27.63 W. Benecke: Silicon microactuators: activation mechanisms and scaling problems, Proc. IEEE Int. Conf. Solid-State Sens. Actuators (1991) pp. 46–50

27.64 A. Menciassi, A. Eisinberg, M. Mazzoni, P. Dario: A sensorized electro discharge machined superelastic alloy microgripper for micromanipulation: simulation and characterization, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002) pp. 1591–1595

27.65 T.R. Hsu: Packaging design of microsystems and meso-scale devices, IEEE Trans. Adv. Packag. **23**, 596–601 (2000)

27.66 L. Lin: MEMS post-packaging by localized heating and bonding, IEEE Trans. Adv. Packag. **23**, 608–616 (2000)

27.67 A. Tixier, Y. Mita, S. Oshima, J.P. Gouy, H. Fujita: 3-D microsystem packaging for interconnecting electrical, optical and mechanical microdevices to the external world, Proc. 13th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (2000) pp. 698–703

27.68 M.J. Madou: *Fundamentals of Microfabrication* (CRC, Boca Raton 2002)

27.69 I. Shimoyama, O. Kano, H. Miura: 3D microstructures folded by Lorentz force, Proc. 11th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1998) pp. 24–28

27.70 K.F. Bohringer, B.R. Donald, L. Kavraki, F.L. Lamiraux: Part orientation with one or two stable equilibria using programmable vector fields, IEEE Trans. Robot. Autom. **16**, 157–170 (2000)

27.71 V. Kaajakari, A. Lal: An electrostatic batch assembly of surface MEMS using ultrasonic triboelectric-

ity, Proc. 14th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (2001) pp. 10–13
27.72 G. Yang, B.J. Nelson: Micromanipulation contact transition control by selective focusing and microforce control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3200–3206
27.73 G. Morel, E. Malis, S. Boudet: Impedance based combination of visual and force control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 1743–1748
27.74 F. Arai, D. Andou, T. Fukuda: Adhesion forces reduction for micro manipulation based on micro physics, Proc. 9th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1996) pp. 354–359
27.75 Y. Zhou, B.J. Nelson: The effect of material properties and gripping force on micrograsping, Proc. IEEE Int. Conf. Robotics Autom (ICRA) (2000) pp. 1115–1120
27.76 K. Kurata: Mass production techniques for optical modules, Proc. 48th IEEE Electronic Components and Technology Conf. (1998) pp. 572–580
27.77 V.T. Portman, B.-Z. Sandler, E. Zahavi: Rigid 6 ×6 parallel platform for precision 3-D micromanipulation: theory and design application, IEEE Trans. Robotics Autom. **16**, 629–643 (2000)
27.78 R.M. Haralick, L.G. Shapiro: *Computer and Robot Vision* (Addison-Wesley, Reading 1993)
27.79 A. Khotanzad, H. Banerjee, M.D. Srinath: A vision system for inspection of ball bonds and 2-D profile of bonding wires in integrated circuits, IEEE Trans. Semicond. Manuf. **7**, 413–422 (1994)
27.80 J.T. Feddema, R.W. Simon: CAD-driven microassembly and visual servoing, Proc. IEEE Int. Conf. Robotics Autom (ICRA) (1998), pp. 1212 -1219
27.81 E. Trucco, A. Verri: *Introductory Techniques for 3-D Computer Vision* (Prentice Hall, Upper Saddle River 1998)
27.82 S. Hutchinson, G.D. Hager, P.I. Corke: A tutorial on visual servo control, IEEE Trans. Robotics Autom. **12**, 651–670 (1996)
27.83 B. Siciliano, L. Villani: *Robot Force Control* (Kluwer, Dordrecht 2000)
27.84 T. Yoshikawa: Force control of robot manipulators, Proc. IEEE Int. Conf. Robotics Autom (ICRA) (2000) pp. 220–226
27.85 J.A. Thompson, R.S. Fearing: Automating microassembly with ortho-tweezers and force sensing, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 1327–1334
27.86 B.J. Nelson, P.K. Khosla: Force and vision resolvability for assimilating disparate sensory feedback, IEEE Trans. Robotics Autom. **12**, 714–731 (1996)
27.87 Y. Haddab, N. Chaillet, A. Bourjault: A microgripper using smart piezoelectric actuators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 659–664
27.88 D. Popa, B.H. Kang, J. Sin, J. Zou: Reconfigurable micro-assembly system for photonics applications, Proc. IEEE Int. Conf. Robotics Autom (ICRA) (2002) pp. 1495–1500
27.89 A.P. Lee, D.R. Ciarlo, P.A. Krulevitch, S. Lehew, J. Trevin, M.A. Northrup: A practical microgripper by fine alignment, eutectic bonding and SMA actuation, Proc. IEEE Int. conf. Solid-State Sens. Actuators (1995) pp. 368–371
27.90 H. Seki: Modeling and impedance control of a piezoelectric bimorph microgripper, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1992) pp. 958–965
27.91 W. Nogimori, K. Irisa, M. Ando, Y. Naruse: A laser-powered micro-gripper, Proc. 10th IEEE Int. Conf. Micro Electro Mech. Syst. (MEMS) (1997) pp. 267–271
27.92 S. Fatikow, U. Rembold: *Microsystem Technology and Microrobotics* (Springer, Berlin, Heidelberg 1997)
27.93 T. Hayashi: Micro mechanism, J. Robotics Mechatr. **3**, 2–7 (1991)
27.94 S. Johansson: Micromanipulation for micro- and nanomanufacturing, INRIA/IEEE Symp. Emerging Technologies and Factory Automation (ETFA'95), Paris (1995) pp. 3–8
27.95 K.-T. Park, M. Esashi: A multilink active catheter with polyimide-based integrated CMOS interface circuits, J. Microelectromechanical Syst. **8**, 349–357 (1999)
27.96 Y. Haga, Y. Tanahashi, M. Esashi: Small diameter active catheter using shape memory alloy, Proc. IEEE 11th Int. Workshop on Micro Electro Mechanical Systems, Heidelberg (1998) pp. 419–424
27.97 E.W.H. Jager, O. Inganas, I. Lundstrom: Microrobots for micrometer-size objects in aqueous media: Potential tools for single cell manipulation, Science **288**, 2335–2338 (2000)
27.98 E.W.H. Jager, E. Smela, O. Inganas: Microfabricating conjugated polymer actuators, Science **290**, 1540–1545 (2000)
27.99 J.W. Suh, S.F. Glander, R.B. Darling, C.W. Storment, G.T.A. Kovacs: Organic thermal and electrostatic ciliary microactuator array for object manipulation, Sens. Actuators A **58**, 51–60 (1997)
27.100 E. Smela, M. Kallenbach, J. Holdenried: Electrochemically driven polypyrrole bilayers for moving and positioning bulk micromachined silicon plates, J. Microelectromechanical Syst. **8**, 373–383 (1999)
27.101 S. Konishi, H. Fujita: A conveyance system using air flow based on the concept of distributed micro motion systems, IEEE J. Microelectromechanical Syst. **3**, 54–58 (1994)
27.102 M. Ataka, A. Omodaka, N. Takeshima, H. Fujita: Fabrication and operation of polyimide bimorph actuators for a ciliary motion system, J. Microelectromechanical Syst. **2**, 146–150 (1993)
27.103 G.-X. Zhou: Swallowable or implantable body temperature telemeter-body temperature radio pill, Proc. IEEE 15th Annual Northeast Bioengineering Conf. (1989) pp. 165–166
27.104 A. Uchiyama: Endoradiosonde needs micro machine technology, Proc. IEEE 6th Int. Symp. Micro Mach. Hum. Sci. (MHS) (1995) pp. 31–37

27.105 Y. Carts-Powell: *Tiny Camera in a Pill Extends Limits of Endoscopy*, OE-Rep. Aug., Vol. 200 (SPIE, Bellingham 2000)

27.106 R. Yeh, E.J.J. Kruglick, K.S.J. Pister: Surface-micromachined components for articulated microrobots, J. Microelectromechanical Syst. **5**, 10–17 (1996)

27.107 P.E. Kladitis, V.M. Bright, K.F. Harsh, Y.C. Lee: Prototype Microrobots for micro positioning in a manufacturing process and micro unmanned vehicles, Proc. IEEE 12th Int. Conf. Micro Electro Mech. Syst. (MEMS), Orlando (1999) pp. 570–575

27.108 D. Ruffieux, N.F. Rooij: A 3-DoF bimorph actuator array capable of locomotion, Proc. 13th Eur. Conf. Solid-State Transducers (Eurosensors), Hague (1999) pp. 725–728

27.109 J.-M. Breguet, P. Renaud: A 4 degrees-of-freedoms microrobot with nanometer resolution, Robotics **14**, 199–203 (1996)

27.110 A. Flynn, L.S. Tavrow, S.F. Bart, R.A. Brooks, D.J. Ehrlich, K.R. Udayakumar, L.E. Cross: Piezoelectric micromotors for microrobots, J. Microelectromechanical Syst. **1**, 44–51 (1992)

27.111 A. Teshigahara, M. Watanabe, N. Kawahara, Y. Ohtsuka, T. Hattori: Performance of a 7 mm microfabricated car, J. Microelectromechanical Syst. **4**, 76–80 (1995)

27.112 T. Ebefors, J. Mattson, E. Kalvesten, G. Stemme: A walking silicon micro-robot, 10th Int. Conf. Solid-State Sens. Actuators (Transducers), Sendai (1999) pp. 1202–1205

27.113 N. Miki, I. Shimoyama: Flight performance of micro-wings rotating in an alternating magnetic field, Proc. IEEE 12th Int. Conf. Micro Electro Mech. Syst. (MEMS), Orlando (1999) pp. 153–158

27.114 Mainz: Micro-motors: The World's Tiniest Helicopter, http://phys.org/news/2004-08-world-lightest-micro-flying-robot-built.html

27.115 K.I. Arai, W. Sugawara, T. Honda: Magnetic small flying machines, IEEE 8th Int. Conf. Solid-State Sens. Actuattors (1995) pp. 316–319

27.116 T. Fukuda, A. Kawamoto, F. Arai, H. Matsuura: Mechanism and swimming experiment of micro mobile robot in water, Proc. IEEE 7th Int. Workshop Micro Electro Mech. Syst. (MEMS), Oiso (1994) pp. 273–278

27.117 I. Shimoyama: Hybrid system of mechanical parts and living organisms for microrobots, Proc. IEEE 6th Int. Symp. Micro Mach. Hum. Sci. (MHS) (1995) p. 55

27.118 A. Ashkin: Acceleration and trapping of particles by radiation pressure, Phys. Rev. Lett. **24**, 156–159 (1970)

27.119 T.N. Bruican, M.J. Smyth, H.A. Crissman, G.C. Salzman, C.C. Stewart, J.C. Martin: Automated single-cell manipulation and sorting by light trapping, Appl. Opt. **26**, 5311–5316 (1987)

27.120 J. Conia, B.S. Edwards, S. Voelkel: The microrobotic laboratory: Optical trapping and scissing for the biologist, J. Clin. Lab. Anal. **11**, 28–38 (1997)

27.121 W.H. Wright, G.J. Sonek, Y. Tadir, M.W. Berns: Laser trapping in cell biology, IEEE J. Quant. Electron. **26**, 2148–2157 (1990)

27.122 F. Arai, K. Morishima, T. Kasugai, T. Fukuda: Bio-micromanipulation (new direction for operation improvement), Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (1997) pp. 1300–1305

27.123 M. Nishioka, S. Katsura, K. Hirano, A. Mizuno: Evaluation of cell characteristics by step-wise orientational rotation using optoelectrostatic micromanipulation, IEEE Trans. Ind. Appl. **33**, 1381–1388 (1997)

27.124 M. Washizu, Y. Kurahashi, H. Iochi, O. Kurosawa, S. Aizawa, S. Kudo, Y. Magariyama, H. Hotani: Dielectrophoretic measurement of bacterial motor characteristics, IEEE Trans. Ind. Appl. **29**, 286–294 (1993)

27.125 Y. Kimura, R. Yanagimachi: Intracytoplasmic sperm injection in the mouse, Biol. Reprod. **52**, 709–720 (1995)

27.126 M. Mischel, A. Voss, H.A. Pohl: Cellular spin resonance in rotating electric fields, J. Biol. Phys. **10**, 223–226 (1982)

27.127 W.M. Arnold, U. Zimmermann: Electro-Rotation: Development of a technique for dielectric measurements on individual cells and particles, J. Electrost. **21**, 151–191 (1988)

27.128 Y. Sun, B.J. Nelson: Autonomous injection of biological cells using visual servoing, Int. Symp. Experim. Robotics (ISER) (2000) pp. 175–184

27.129 Y. Sun, B.J. Nelson, D.P. Potasek, E. Enikov: A bulk microfabricated multi-axis capacitive cellular force sensor using transverse comb drives, J. Micromech. Microeng. **12**, 832–840 (2002)

27.130 Y. Sun, K. Wan, K.P. Roberts, J.C. Bischof, B.J. Nelson: Mechanical property characterization of mouse zona pellucida, IEEE Trans. Nanobiosci. **2**, 279–286 (2003)

27.131 E.M. Purcell: Life at low Reynolds-number, Am. J. Phy. **45**, 3–11 (1977)

27.132 R. Dreyfus, J. Baudry, M.L. Roper, M. Fermigier, H.A. Stone, J. Bibette: Microscopic artificial swimmers, Nature **437**, 862–865 (2005)

27.133 H.C. Berg, R.A. Anderson: Bacteria swim by rotating their flagellar filaments, Nature **245**, 380–382 (1973)

27.134 L. Zhang, J.J. Abbott, L.X. Dong, B.E. Kratochvil, D.J. Bell, B.J. Nelson: Artificial bacterial flagella: Fabrication and magnetic control, Appl. Phys. Lett. **94**, 064107 (2009)

27.135 L. Zhang, E. Deckhardt, A. Weber, C. Schonenberger, D. Grutzmacher: Controllable fabrication of SiGe/Si and SiGe/Si/Cr helical nanobelts, Nanotechnology **16**, 655–663 (2005)

27.136 V.Y. Prinz, V.A. Seleznev, A.K. Gutakovsky, A.V. Chehovskiy, V.V. Preobrazhenskii, M.A. Putyato, T.A. Gavrilova: Free-standing and overgrown InGaAs/GaAs nanotubes, nanohelices and their arrays, Physica E Low-Dimen. Syst. Nanostructures **6**, 828–831 (2000)

27.137 F.B. Hagedorn, E.M. Gyorgy: Magnetic-Shape Anisotropy in Polygonal Prisms, J. Appl. Phys. **39**, 995–997 (1968)

27.138 H.C. Berg, D.A. Brown: Chemotaxis in escherichia-coli analyzed by 3-dimensional tracking, Nature **239**, 500–504 (1972)

27.139 A. Ashkin, J.M. Dziedzic: Optical trapping and manipulation of viruses and bacteria, Science **235**, 1517–1520 (1987)

27.140 F.H.C. Crick, A.F.W. Hughes: The physical properties of cytoplasm: A study by means of the magnetic particle method, Part I, Exp. Cell Res. **1**, 37–80 (1950)

27.141 M.F. Yu, M.J. Dyer, G.D. Skidmore, H.W. Rohrs, X.K. Lu, K.D. Ausman, J.R.V. Ehr, R.S. Ruoff: Three-dimensional manipulation of carbon nanotubes under a scanning electron microscope, Nanotechnology **10**, 244–252 (1999)

27.142 L.X. Dong, F. Arai, T. Fukuda: 3D nanorobotic manipulation of nano-order objects inside SEM, Proc. Int. Symp. Micromechatron. Hum. Sci. (MHS) (2000) pp. 151–156

27.143 D.M. Eigler, E.K. Schweizer: Positioning single atoms with a scanning tunneling microscope, Nature **344**, 524–526 (1990)

27.144 P. Avouris: Manipulation of matter at the atomic and molecular levels, Acc. Chem. Res. **28**, 95–102 (1995)

27.145 M.F. Crommie, C.P. Lutz, D.M. Eigler: Confinement of electrons to quantum corrals on a metal surface, Science **262**, 218–220 (1993)

27.146 L.J. Whitman, J.A. Stroscio, R.A. Dragoset, R.J. Cellota: Manipulation of adsorbed atoms and creation of new structures on room-temperature surfaces with a scanning tunneling microscope, Science **251**, 1206–1210 (1991)

27.147 I.-W. Lyo, P. Avouris: Field-induced nanometer-scale to atomic-scale manipulation of silicon surfaces with the STM, Science **253**, 173–176 (1991)

27.148 G. Dujardin, R.E. Walkup, P. Avouris: Dissociation of individual molecules with electrons from the tip of a scanning tunneling microscope, Science **255**, 1232–1235 (1992)

27.149 T.-C. Shen, C. Wang, G.C. Abeln, J.R. Tucker, J.W. Lyding, P. Avouris, R.E. Walkup: Atomic-scale desorption through electronic and vibrational-excitation mechanisms, Science **268**, 1590–1592 (1995)

27.150 M.T. Cuberes, R.R. Schittler, J.K. Gimzewsk: Room-temperature repositioning of individual C60 molecules at Cu steps: operation of a molecular counting device, Appl. Phys. Lett. **69**, 3016–3018 (1996)

27.151 H.J. Lee, W. Ho: Single-bond formation and characterization with a scanning tunneling microscope, Science **286**, 1719–1722 (1999)

27.152 T. Yamamoto, O. Kurosawa, H. Kabata, N. Shimamoto, M. Washizu: Molecular surgery of DNA based on electrostatic micromanipulation, IEEE Trans. Ind. Appl. **36**, 1010–1017 (2000)

27.153 C. Haber, D. Wirtz: Magnetic tweezers for DNA micromanipulation, Rev. Sci. Instrum. **71**, 4561–4570 (2000)

27.154 D.M. Schaefer, R. Reifenberger, A. Patil, R.P. Andres: Fabrication of two-dimensional arrays of nanometer-size clusters with the atomic force microscope, Appl. Phys. Lett. **66**, 1012–1014 (1995)

27.155 T. Junno, K. Deppert, L. Montelius, L. Samuelson: Controlled manipulation of nanoparticles with an atomic force microscope, Appl. Phys. Lett. **66**, 3627–3629 (1995)

27.156 P.E. Sheehan, C.M. Lieber: Nanomachining, manipulation and fabrication by force microscopy, Nanotechnology **7**, 236–240 (1996)

27.157 C. Baur, B.C. Gazen, B. Koel, T.R. Ramachandran, A.A.G. Requicha, L. Zini: Robotic nanomanipulation with a scanning probe microscope in a networked computing environment, J. Vac. Sci. Tech. B **15**, 1577–1580 (1997)

27.158 A.A.G. Requicha: Nanorobots, NEMS, and nanoassembly, Proceedings IEEE **91**, 1922–1933 (2003)

27.159 R. Resch, C. Baur, A. Bugacov, B.E. Koel, A. Madhukar, A.A.G. Requicha, P. Will: Building and manipulating 3-D and linked 2-D structures of nanoparticles using scanning force microscopy, Langmuir **14**, 6613–6616 (1998)

27.160 J. Hu, Z.-H. Zhang, Z.-Q. Ouyang, S.-F. Chen, M.-Q. Li, F.-J. Yang: Stretch and align virus in nanometer scale on an atomically flat surface, J. Vac. Sci. Tech. B **16**, 2841–2843 (1998)

27.161 M. Sitti, S. Horiguchi, H. Hashimoto: Controlled pushing of nanoparticles: modeling and experiments, IEEE/ASME Trans. Mechatron. **5**, 199–211 (2000)

27.162 M. Guthold, M.R. Falvo, W.G. Matthews, S. Paulson, S. Washburn, D.A. Erie, R. Superfine, J.F.P. Brooks, I.R.M. Taylor: Controlled manipulation of molecular samples with the nanoManipulator, IEEE/ASME Trans. Mechatron. **5**, 189–198 (2000)

27.163 G.Y. Li, N. Xi, M.M. Yu, W.K. Fung: Development of augmented reality system for AFM-based nanomanipulation, IEEE/ASME Trans. Mechatron. **9**, 358–365 (2004)

27.164 F. Arai, D. Andou, T. Fukuda: Micro manipulation based on micro physics–strategy based on attractive force reduction and stress measurement, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (1995) pp. 236–241

27.165 H.W.P. Koops, J. Kretz, M. Rudolph, M. Weber, G. Dahm, K.L. Lee: Characterization and application of materials grown by electron-beam-induced deposition, Jpn. J. Appl. Phys. **33**, 7099–7107 (1994)

27.166 S. Iijima: Helical microtubules of graphitic carbon, Nature **354**, 56–58 (1991)

27.167 S.J. Tans, A.R.M. Verchueren, C. Dekker: Room-temperature transistor based on a single carbon nanotube, Nature **393**, 49–52 (1998)

27.168 R.H. Baughman, A.A. Zakhidov, W.A. de Heer: Carbon nanotubes–the route toward applications, Science **297**, 787–792 (2002)

27.169 M.J. Treacy, T.W. Ebbesen, J.M. Gibson: Exceptionally high Young's modulus observed for individual carbon nanotubes, Nature **381**, 678–680 (1996)

27.170 P. Poncharal, Z.L. Wang, D. Ugarte, W.A. de Heer: Electrostatic deflections and electromechanical resonances of carbon nanotubes, Science **283**, 1513–1516 (1999)

27.171 M.F. Yu, O. Lourie, M.J. Dyer, K. Moloni, T.F. Kelley, R.S. Ruoff: Strength and breaking mechanism of multiwalled carbon nanotubes under tensile load, Science **287**, 637–640 (2000)

27.172 T.W. Ebbesen, H.J. Lezec, H. Hiura, J.W. Bennett, H.F. Ghaemi, T. Thio: Electrical conductivity of individual carbon nanotubes, Nature **382**, 54–56 (1996)

27.173 P. Kim, L. Shi, A. Majumdar, P.L. McEuen: Thermal transport measurements of individual multiwalled nanotubes, Phys. Rev. Lett. **87**, 215502 (2001)

27.174 W.J. Liang, M. Bockrath, D. Bozovic, J.H. Hafner, M. Tinkham, H. Park: Fabry-Perot interference in a nanotube electron waveguide, Nature **411**, 665–669 (2001)

27.175 X.B. Zhang, D. Bernaerts, G.V. Tendeloo, S. Amelincks, J.V. Landuyt, V. Ivanov, J.B. Nagy, P. Lambin, A.A. Lucas: The texture of catalytically grown coil-shaped carbon nanotubules, Europhys. Lett. **27**, 141–146 (1994)

27.176 X.Y. Kong, Z.L. Wang: Spontaneous polarization-induced nanohelixes, nanosprings, and nanorings of piezoelectric nanobelts, Nano Lett. **3**, 1625–1631 (2003)

27.177 S.V. Golod, V.Y. Prinz, V.I. Mashanov, A.K. Gutakovsky: Fabrication of conducting GeSi/Si micro- and nanotubes and helical microcoils, Semicond. Sci. Technol. **16**, 181–185 (2001)

27.178 L. Zhang, E. Deckhardt, A. Weber, C. Schönenberger, D. Grützmacher: Controllable fabrication of SiGe/Si and SiGe/Si/Cr helical nanobelts, Nanotechnology **16**, 655–663 (2005)

27.179 L. Zhang, E. Ruh, D. Grützmacher, L.X. Dong, D.J. Bell, B.J. Nelson, C. Schönenberger: Anomalous coiling of SiGe/Si and SiGe/Si/Cr helical nanobelts, Nano Lett. **6**, 1311–1317 (2006)

27.180 D.J. Bell, L.X. Dong, B.J. Nelson, M. Golling, L. Zhang, D. Grützmacher: Fabrication and characterization of three-dimensional InGaAs/GaAs nanosprings, Nano Lett. **6**, 725–729 (2006)

27.181 D.J. Bell, Y. Sun, L. Zhang, L.X. Dong, B.J. Nelson, D. Grutzmacher: Three-dimensional nanosprings for electromechanical sensors, Sens. Actuators A Phys. **130**, 54–61 (2006)

27.182 R. Martel, T. Schmidt, H.R. Shea, T. Herte, P. Avouris: Single- and multi-wall carbon nanotube field-effect transistors, Appl. Phys. Lett. **73**, 2447–2449 (1998)

27.183 N.R. Franklin, Y.M. Li, R.J. Chen, A. Javey, H.J. Dai: Patterned growth of single-walled carbon nanotubes on full 4-inch wafers, Appl. Phys. Lett. **79**, 4571–4573 (2001)

27.184 T. Rueckes, K. Kim, E. Joselevich, G.Y. Tseng, C.-L. Cheung, C.M. Lieber: Carbon nanotube-based non-volatile random access memory for molecular computing science, Science **289**, 94–97 (2000)

27.185 A. Subramanian, B. Vikramaditya, L.X. Dong, D.J. Bell, B.J. Nelson: Micro and nanorobotic assembly using dielectrophoresis. In: *Robotics Sci. Syst*, ed. by S. Thrun, G.S. Sukhatme, S. Schaal, O. Brock (MIT Press, Cambridge 2005) pp. 327–334

27.186 C.K.M. Fung, V.T.S. Wong, R.H.M. Chan, W.J. Li: Dielectrophoretic batch fabrication of bundled carbon nanotube thermal sensors, IEEE Trans. Nanotechnol. **3**, 395–403 (2004)

27.187 T. Fukuda, F. Arai, L.X. Dong: Assembly of nanodevices with carbon nanotubes through nanorobotic manipulations, Proceedings IEEE **91**, 1803–1818 (2003)

27.188 E.W. Wong, P.E. Sheehan, C.M. Lieber: Nanobeam mechanics: elasticity, strength, and toughness of nanorods and nanotubes, Science **277**, 1971–1975 (1997)

27.189 M.R. Falvo, G.J. Clary, R.M. Taylor, V. Chi, F.P. Brooks, S. Washburn, R. Superfine: Bending and buckling of carbon nanotubes under large strain, Nature **389**, 582–584 (1997)

27.190 H.W.C. Postma, A. Sellmeijer, C. Dekker: Manipulation and imaging of individual single-walled carbon nanotubes with an atomic force microscope, Adv. Mater. **12**, 1299–1302 (2000)

27.191 T. Hertel, R. Martel, P. Avouris: Manipulation of individual carbon nanotubes and their interaction with surfaces, J. Phys. Chem. B **102**, 910–915 (1998)

27.192 P. Avouris, T. Hertel, R. Martel, T. Schmidt, H.R. Shea, R.E. Walkup: Carbon nanotubes: nanomechanics, manipulation, and electronic devices, Appl. Surf. Sci. **141**, 201–209 (1999)

27.193 M. Ahlskog, R. Tarkiainen, L. Roschier, P. Hakonen: Single-electron transistor made of two crossing multiwalled carbon nanotubes and its noise properties, Appl. Phys. Lett. **77**, 4037–4039 (2000)

27.194 M.R. Falvo, R.M.I. Taylor, A. Helser, V. Chi, F.P.J. Brooks, S. Washburn, R. Superfine: Nanometre-scale rolling and sliding of carbon nanotubes, Nature **397**, 236–238 (1999)

27.195 B. Bhushan, V.N. Koinkar: Nanoindentation hardness measurements using atomic-force microscopy, Appl. Phys. Lett. **64**, 1653–1655 (1994)

27.196 P. Vettiger, G. Cross, M. Despont, U. Drechsler, U. Durig, B. Gotsmann, W. Haberle, M.A. Lantz, H.E. Rothuizen, R. Stutz, G.K. Binnig: The *millipede* – Nanotechnology entering data storage, IEEE Trans. Nanotechnol. **1**, 39–55 (2002)

27.197 L.X. Dong: *Nanorobotic manipulations of carbon nanotubes. Ph.D. Thesis Ser* (Nagoya Univ., Nagoya 2003)

27.198 J.H. Hafner, C.-L. Cheung, T.H. Oosterkamp, C.M. Lieber: High-yield assembly of individual

single-walled carbon nanotube tips for scanning probe microscopies, J. Phys. Chem. B **105**, 743–746 (2001)
27.199 L.X. Dong, F. Arai, T. Fukuda: Electron-beam-induced deposition with carbon nanotube emitters, Appl. Phys. Lett. **81**, 1919–1921 (2002)
27.200 L.X. Dong, F. Arai, T. Fukuda: 3D nanorobotic manipulations of multi-walled carbon nanotubes, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 632–637
27.201 L.X. Dong, F. Arai, T. Fukuda: Destructive constructions of nanostructures with carbon nanotubes through nanorobotic manipulation, IEEE/ASME Trans. Mechatron. **9**, 350–357 (2004)
27.202 J. Cumings, P.G. Collins, A. Zettl: Peeling and sharpening multiwall nanotubes, Nature **406**, 58 (2000)
27.203 J. Cumings, A. Zettl: Low-friction nanoscale linear bearing realized from multiwall carbon nanotubes, Science **289**, 602–604 (2000)
27.204 A. Kis, K. Jensen, S. Aloni, W. Mickelson, A. Zettl: Interlayer forces and ultralow sliding friction in multiwalled carbon nanotubes, Phys. Rev. Lett. **97**, 025501 (2006)
27.205 L.X. Dong, F. Arai, T. Fukuda: Nanoassembly of carbon nanotubes through mechanochemical nanorobotic manipulations, Jpn. J. Appl. Phys. **42**, 295–298 (2003)
27.206 L. Chico, V.H. Crespi, L.X. Benedict, S.G. Louie, M.L. Cohen: Pure carbon nanoscale devices: Nanotube heterojunctions, Phys. Rev. Lett. **76**, 971–974 (1996)
27.207 Z. Yao, H.W.C. Postma, L. Balents, C. Dekker: Carbon nanotube intramolecular junctions, Nature **402**, 273–276 (1999)
27.208 H.W.C. Postma, T. Teepen, Z. Yao, M. Grifoni, C. Dekker: Carbon nanotube single-electron transistors at room temperature, Science **293**, 76–79 (2001)
27.209 M.S. Fuhrer, J. Nygård, L. Shih, M. Forero, Y.-G. Yoon, M.S.C. Mazzoni, H.J. Choi, J. Ihm, S.G. Louie, A. Zettl, P.L. McEuen: Crossed nanotube junctions, Science **288**, 494–497 (2000)
27.210 T. Rueckes, K. Kim, E. Joselevich, G.Y. Tseng, C.-L. Cheung, C.M. Lieber: Carbon nanotube-based nonvolatile random access memory for molecular computing science, Science **289**, 94–97 (2000)
27.211 A.G. Rinzler, J.H. Hafner, P. Nikolaev, L. Lou, S.G. Kim, D. Tománek, P. Nordlander, D.T. Colbert, R.E. Smalley: Unraveling nanotubes: field emission from an atomic wire, Science **269**, 1550–1553 (1995)
27.212 S.C. Minne, G. Yaralioglu, S.R. Manalis, J.D. Adams, J. Zesch, A. Atalar, C.F. Quate: Automated parallel high-speed atomic force microscopy, Appl. Phys. Lett. **72**, 2340–2342 (1998)
27.213 G.D. Skidmore, E. Parker, M. Ellis, N. Sarkar, R. Merkle: Exponential assembly, Nanotechnology **11**, 316–321 (2001)
27.214 H.J. Dai, J.H. Hafner, A.G. Rinzler, D.T. Colbert, R.E. Smalley: Nanotubes as nanoprobes in scanning probe microscopy, Nature **384**, 147–150 (1996)
27.215 J.H. Hafner, C.L. Cheung, C.M. Lieber: Growth of nanotubes for probe microscopy tips, Nature **398**, 761–762 (1999)
27.216 L.X. Dong, B.J. Nelson, T. Fukuda, F. Arai: Towards Nanotube Linear Servomotors, IEEE Trans. Autom. Sci. Eng. **3**, 228–235 (2006)
27.217 Y.H. Gao, Y. Bando: Carbon nanothermometer containing gallium, Nature **415**, 599 (2002)
27.218 L.X. Dong, X.Y. Tao, L. Zhang, B.J. Nelson, X.B. Zhang: Nanorobotic spot welding: Controlled metal deposition with attogram precision from Copper-filled carbon nanotubes, Nano Lett. **7**, 58–63 (2007)
27.219 S.W. Lee, D.S. Lee, R.E. Morjan, S.H. Jhang, M. Sveningsson, O.A. Nerushev, Y.W. Park, E.E.B. Campbell: A three-terminal carbon nanorelay, Nano Lett. **4**, 2027–2030 (2004)
27.220 A.M. Fennimore, T.D. Yuzvinsky, W.-Q. Han, M.S. Fuhrer, J. Cumings, A. Zettl: Rotational actuators based on carbon nanotubes, Nature **424**, 408–410 (2003)
27.221 A. Subramanian, L.X. Dong, J. Tharian, U. Sennhauser, B.J. Nelson: Batch fabrication of carbon nanotube bearings, Nanotechnology **18**, 075703 (2007)
27.222 P. Kim, C.M. Lieber: Nanotube nanotweezers, Science **286**, 2148–2150 (1999)
27.223 L.X. Dong, A. Subramanian, D. Hugentobler, B.J. Nelson, Y. Sun: Nano Encoders based on Vertical Arrays of Individual Carbon Nanotubes, Adv. Robotics **20**, 1281–1301 (2006)
27.224 L.X. Dong, B.J. Nelson: Robotics in the small, Part II: Nanorobotics, IEEE Robotics Autom. Mag. **14**, 111–121 (2007)
27.225 I. Asimov: *Fantastic Voyage* (Bantam Books, New York 1966)
27.226 R.A. Freitas: *Nanomedicine, Volume I: Basic Capabilities* (Landes Bioscience, Austin 1999)

Multimedia Contents

Part C Sensing

Part C Sensing and Perception

Ed. by Henrik I. Christensen

28 Force and Tactile Sensing
Mark R. Cutkosky, Stanford, USA
William Provancher, Salt Lake City, USA

29 Inertial Sensing, GPS and Odometry
Gregory Dudek, Montreal, Canada
Michael Jenkin, Toronto, Canada

30 Sonar Sensing
Lindsay Kleeman, Melbourne, Australia
Roman Kuc, New Haven, USA

31 Range Sensing
Kurt Konolige, Mountain View, USA
Andreas Nüchter, Würzburg, Germany

32 3-D Vision for Navigation and Grasping
Danica Kragic, Stockholm, Sweden
Kostas Daniilidis, Philadelphia, USA

33 Visual Object Class Recognition
Michael Stark, Saarbrücken, Germany
Bernt Schiele, Saarbrücken, Germany
Aleš Leonardis, Birmingham, UK

34 Visual Servoing
François Chaumette, Rennes, France
Seth Hutchinson, Urbana-Champaign, USA
Peter Corke, Brisbane, Australia

35 Multisensor Data Fusion
Hugh Durrant-Whyte, Sydney, Australia
Thomas C. Henderson, Salt Lake City, USA

Part C covers material related to sensing and perception. The section covers all the aspects of sensing for basic measurement of physical parameters in the world to making sense of such data for the purpose of enabling a robot to perform its tasks. Right now robotics is seeing a revolution in use of sensors. Traditionally robots have been designed to have maximum stiffness and applications have been designed to be predictable in their operation. As robots emerge from the fences areas and we deploy robots for a wider range of applications from collaborative robotics to autonomously driving cars it is essential to have perception capabilities that allow estimation of the state of the robot but also the state the surrounding environment. Due to these new requirements the importance of sensing and perception has increased significantly over the last decade and will without doubt to continue to grow in the future.

The topics addressed in this section include all aspects from detection and processing physical contact with the world through force and tackle sensing over augmented environments for detection of position and motion in the world to image based methods for mapping, detection and control in structured and unstructured environments. In many settings a single sensory modality/sensor is inadequate to give robust estimate of the state of the environment. Consequently a chapter on multi-sensory fusion is also included in this part. Part C considers primarily the sensing and perception aspects of robotics with a limited view to many of the other aspects. Part A is providing the basic, Part B provides the kinematic structures that we need to design control methods for interaction, while Part D will cover the grasping and manipulation of objects. The remaining parts of the handbook covers application verticals where sensors and perception plays a key role.

With this brief overview of Part C, we provide a brief synopsis of each chapter.

Chapter 28 covers force and tactile sensing, which is essential for physical interaction with the world. Tactile sensing addresses the issue of detecting and handling contact with objects and structures in the world. Tactile sensing is essential both for safety and manipulation applications as the control of a robot changes due to contact which changes the kinematic configurations. Force sensing addresses estimation of force and torque as part of dynamic motion but also for interaction with physical objects such as inserting an object or turning a valve. The model chosen for force estimation also impacts the control of the robot.

Chapter 29 covers use of odometry, inertial and GPS for estimation of motion and position in the world. Odometry is the ego-estimation of your position/motion based on robot mounted sensors. With the introduction of sensors for estimation of accelerator and rotational velocity it is possible improve the estimation of position and motion. With the introduction of inexpensive inertial measurement units (IMU), driven by the game and phone industry, the use of IMU has increased significantly. For outdoor operation it is frequently possible to utilize Global Navigation Systems such as the global positioning systems (GPS). The integration of odometry, IMU and GPS information allow design of systems for precision agriculture, autonomous driving, etc. The chapter provides a description of the core techniques to make this possible.

One of the first sensors to be used for large scale estimation of distance to external objects was sonars. Chapter 30 covers the basic methods for sound based ranging and localization. Sonars are widely used for underwater mapping and localization, but is also used as an inexpensive modality for localization and ground vehicles and landing of Unmanned Aerial Vehicles. Typically a single sensor has limited precision but through used of phased array techniques and multiple sensors it is possible to achieve high-fidelity ranging. The chapter covers both the basic physics, sensing, and estimation methods.

Over the last two decades we have seen tremendous progress on laser based ranging as a complement to stereo/multi-ocular methods for distance estimation. Light detection and ranging (LIDAR) is today used as an effective modality for mapping, localization and tracking of external objects in the environment. In Chapter 31 the basic techniques for range estimation using light based ranging is described and fundamental methods for three-dimensional (3-D) modeling of the environment are described. Recently there has been a renaissance in range based sensing due to the introduction of RGB-D sensors, that utilize structured light and cameras to generate a dense range maps at very-low prices.

Chapter 32 covers the general topics of 3-D vision which covers estimation of distance from the parallax between two or more cameras and/or the motion of a camera over time. The parallax between two or more images allows estimation of the distance to external objects, which can be leveraged for localization and navigation, but also for grasping and interacting with objects in the environment. The chapter covers both the basic aspects of 3-D estimation and some of the common applications of 3-D vision for robot control.

Chapter 33 covers the topic of object recognition. Computer based object recognition has been studied for more than 50 years. Over the last decade we have

seen tremendous progress on object recognition due to improved cameras and availability of much better computers and increased memory. We are seeing both image based recognition techniques and methods for recognition based on the 3-D structure of objects. More recently we have seen a renewed interesting use of neural nets for detection of objects due to new Bayesian methods and the problem of object categorization – recognizing categories of objects such as car, motorcycle, traffic sign, people, etc. has become an important problem. The chapter covers both view and 3-D based methods for recognition and discusses also categorization of objects.

Chapter 34 covers the problem of image servoing. When trying to interact with an object we can use image data to drive the end-effector to a goal location. Two common configurations are eye-in-hand and hand-to-eye. In addition the control can be performed directly in image coordinates or through recovery of two-and-a-half-dimensional (2.5-D) or 3-D pose for an object. To enable all of this an impact aspect is to derived the relation between changes in robot motion and changes in the image/pose, which is derivation of the Jacobian for the system. This chapter discusses both eye-in-hand and hand-to-eye visual servoing and image/pose based control of the process and provides examples of use of visual servoing in real scenarios.

Finally, the part is concluded by Chapter 35 which discusses multi-sensory data fusion. As mentioned earlier most applications require use of multiple sensors to generate robust/complete methods for control. Data fusion involves a number of different aspects from time synchronization to transformation into a common reference frame to integration of data over time/space to generate more robust/accurate estimate of the world state. Recently a number of new methods for Bayesian data fusion have emerged. In this chapter the fundamental of data fusion are reviewed and a number of the most common techniques for multi-sensory fusion are introduced.

Multimedia Contents

28. Force and Tactile Sensing

Mark R. Cutkosky, William Provancher

This chapter provides an overview of force and tactile sensing, with the primary emphasis placed on tactile sensing. We begin by presenting some basic considerations in choosing a tactile sensor and then review a wide variety of sensor types, including proximity, kinematic, force, dynamic, contact, skin deflection, thermal, and pressure sensors. We also review various transduction methods, appropriate for each general sensor type. We consider the information that these various types of sensors provide in terms of whether they are most useful for manipulation, surface exploration or being responsive to contacts from external agents.

Concerning the interpretation of tactile information, we describe the general problems and present two short illustrative examples. The first involves intrinsic tactile sensing, i. e., estimating contact locations and forces from force sensors. The second involves contact pressure sensing, i. e., estimating surface normal and shear stress distributions from an array of sensors in an elastic skin. We conclude with a brief discussion of the challenges that remain to be solved in packaging and manufacturing damage-tolerant tactile sensors.

28.1 **Overview** 717
28.2 **Sensor Types** 718
28.2.1 Proprioceptive and Proximity Sensing 718
28.2.2 Other Contact Sensors 720
28.2.3 Kinematic Sensors 721
28.2.4 Force and Load Sensing 721
28.2.5 Dynamic Tactile Sensors 722
28.2.6 Array Sensors 722
28.3 **Tactile Information Processing** 725
28.3.1 Tactile Information Flow: Means and Ends of Tactile Sensing 725
28.3.2 Solid Mechanics and Deconvolution 727
28.3.3 Curvature and Shape Information 729
28.3.4 Object and Surface Identification 729
28.3.5 Active Sensing Strategies 730
28.3.6 Dynamic Sensing and Event Detection 730
28.3.7 Integration of Thermal and Other Sensors 730
28.4 **Integration Challenges** 730
28.5 **Conclusions and Future Developments** .. 731
Video-References 731
References 731

28.1 Overview

Tactile sensing has been a component of robotics for roughly as long as vision. However, in comparison to vision, for which great strides have been made in terms of hardware and software and which is now widely used in industrial and mobile robot applications, tactile sensing always seems to be *a few years away* from widespread utility. So before reviewing the technologies and approaches available, it is worthwhile to ask some basic questions:

- How important is tactile sensing?
- What is it useful for?
- Why does it remain comparatively undeveloped?

In nature, tactile sensing is an essential survival tool. Even the simplest creatures are endowed with large numbers of mechanoreceptors for exploring and responding to various stimuli. In humans, tactile sensing is indispensable for three distinct kinds of activity: ma-

nipulation, exploration and response. The importance of tactile sensing for manipulation is most evident in fine motor tasks. When we are chilled, tasks like buttoning a shirt can become an exercise in frustration. The problem is primarily a lack of sensing; our muscles, snug in our coat sleeves, are only slightly affected but our cutaneous mechanoreceptors are anesthetized and we become clumsy. For exploration, we continually assimilate information about materials and surface properties (e.g., hardness, thermal conductivity, friction, roughness) to help us identify objects. We may have difficulty distinguishing real leather from synthetic leather by sight, but not by touch. Finally, the importance of tactile response, whether to a gentle touch or an impact, is seen in the damage that patients with peripheral neuropathy (e.g., as a complication of diabetes) accidentally do to themselves.

As Fig. 28.1 indicates, the same functional categories apply to robots. However, in comparison to animals, with hundreds or thousands of mechanoreceptors per square centimeter of skin, even the most sophisticated robots are impoverished. One reason for the slow development of tactile sensing technology as compared to vision is that there is no tactile analog to the charge-coupled device (CCD) or complementary metal-oxide semiconductor (CMOS) optical array. Instead, tactile sensors elicit information through physical interaction. They must be incorporated into skin surfaces with compliance, for conforming locally to surfaces, and with adequate friction for handling objects securely. The sensors and skin must also be robust enough to survive repeated impacts and abrasions. And unlike the image plane in a camera, tactile sensors must

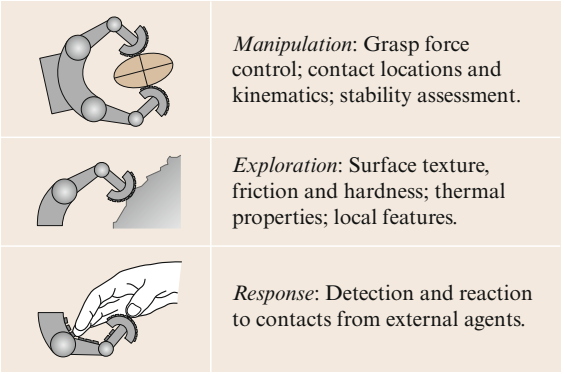


Fig. 28.1 Uses of tactile sensing in robotics

be distributed over the robot appendages, with particularly high concentrations in areas like the fingertips. The wiring of tactile sensors is consequently another formidable challenge.

Nonetheless, considerable progress in tactile sensor design and deployment has been made over the last couple of decades. In the following sections we review the main functional classes of tactile sensors and discuss their relative strengths and limitations. Looking ahead, new fabrication techniques offer the possibility of artificial skin materials with integrated sensors and local processing for interpreting sensor signals and communicating over a common bus to reduce wiring.

There is an extensive literature describing touch sensing research. Recent general reviews include [28.1–4] and these cite a number of useful older reviews including [28.5–7].

28.2 Sensor Types

This section outlines five main types of sensors: proprioceptive, kinematic, force, dynamic tactile, and array tactile sensors, as well as sensors that provide thermal or material composition data. However, the emphasis is on sensors that provide mechanoreception, as summarized in Table 28.1. The most important quantities measured with tactile sensors are *shape* and *force*. Each of these may be measured as an average quantity for some part of the robot or as a spatially resolved, distributed quantity across a contact area. In this chapter we follow the convention of studies of the human sense of touch and use the term *touch sensing* to refer to the combination of these two modes. Devices that measure an average or resultant quantity are sometimes referred to as *internal* or *intrinsic* sensors. The basis for these sensors is force sensing, which precedes the discussion of tactile array sensors.

28.2.1 Proprioceptive and Proximity Sensing

Proprioceptive sensing refers to sensors that provide information about the net force or motion of an appendage, analogous to receptors that provide information in humans about tendon tensions or joint movements. Generally speaking, the primary source for spatial proprioceptive information on a robot is provided by joint angle and force–torque sensors. Since joint angle sensors such as potentiometers, encoders, and resolvers are well established technologies, they do not warrant discussion here. Instead, a brief review of proximity sensing via whiskers and antennae as well as noncontact proximity sensing is provided. Force–torque sensors are discussed in greater detail in Sect. 28.2.4.

Table 28.1 Tactile sensor modalities and common transduction types

Sensor modality	Sensor type	Sensor attributes	Advantages	Disadvantages
Normal pressure	Piezoresistive array [28.8–12]	Array of piezoresistive junctions Embedded in a elastomeric skin Cast or screen printed	Suitable for mass production Simple design Simple signal conditioning	Low force thresholds/high sensitivity Frail Signal drift and hysteresis
	Capacitive array [28.13–17]	Array of capacitive junctions Row and column electrodes separated by elastomeric dielectric	Good sensitivity Moderate hysteresis, depending on construction	Complex circuitry
	Piezoresistive microelectromechanical system (MEMS) array [28.18–20]	Silicon micromachined array with doped silicon strain gauged flexures	Suitable for to mass production	Frail
	Optical [28.21, 22]	Combined tracking of optical markers with a constitutive model	No interconnects to break	Requires personal computer (PC) for computing applied forces
Skin deformation	Optical [28.23, 24]	Fluid-filled elastomeric membrane Tracking of optical markers inscribed on membrane coupled with energy minimization algorithm	Compliant membrane No electrical interconnects to be damaged	Complex computations Hard to customize sensor
	Magnetic [28.25]	Array of hall effect sensors		Complex computations Hard to customize sensor
	Resistive tomography [28.26]	Array of conductive rubber traces as electrodes	Robust construction	
	Piezoresistive (curvature) [28.27, 28]	Employs an array of strain gauges	Directly measure curvature	Frailty of electrical interconnects Hysteresis
Dynamic tactile sensing	Piezoelectric (stress rate) [28.19, 29, 30]	polyvinylidene fluoride (PVDF) embedded in elastomeric skin	High bandwidth	Frailty of electrical junctions
	Skin acceleration [28.31, 32]	Commercial accelerometer afixed to robot skin	Simple	No spatially distributed content Sensed vibrations tend to be dominated by structural resonant frequency

Whisker and Antenna Sensors

For many animals whiskers or antennae provide an extremely accurate combination of contact sensing and proprioceptive information. For example, cockroaches can steer themselves along curved walls using only position and rate information from their antennae [28.33]. Other insects and arthropods use numerous small hair sensors on the exoskeleton to localize contacts. Rats perform highly accurate whisking to explore the shapes and textures of objects in their vicinity, with sensor processing performed by a specialized barrel cortex [28.34].

In robotics, this potentially very useful hybrid of proprioceptive and tactile sensing has received comparatively little attention, although examples date at least to the early 1990s [28.35, 36]. In more recent work, *Clements* and *Rahn* [28.37] demonstrated an active whisker with a sweeping motion and *Lee* et al. [28.33] used a passive, flexible antenna to steer a running cockroach-inspired robot. *Prescott* et al. [28.34, 38] have conducted extensive work on active robotic whisking for object exploration and identification, inspired by mammalian models.

Proximity

While proximity sensing does not strictly fall under the category of tactile sensing, a number of researchers have employed proximity sensors for collision detection between a robot arm and the environment; hence, we briefly review these technologies here. Three primary sensor technologies which include capaciflective, infrared (IR) optical, and ultrasonic sensors have been used in this application. *Vranish* et al. developed an early capaciflective sensor for collision avoidance between the environment and a grounded robot arm [28.39]. Early examples of IR emitter/detector pairs in an artificial skin include [28.40, 41]. A more recent design using optical fibers is reported in [28.42], and an adaptation for a prosthetic hand is reported in [28.43]. Other researchers have developed robot skin that includes ultrasonic and IR optical sensors for collision avoidance [28.44]. *Wegerif* and *Rosinski* provide a comparison of the performance of all 3 of these proximity sensing technologies [28.45]. For a further review of some of these sensors, see Chap. 31 on range sensing.

28.2.2 Other Contact Sensors

There are a variety of other contact-based sensors that are capable of discerning object properties such as electromagnetic characteristics, density (via ultrasound), or chemical composition (cf. animals' senses of taste and smell). While this is beyond the scope of the current chapter, Chap. 75, on biologically inspired robots, briefly discusses biologically inspired chemical sensors related to smell and taste. For completeness, thermal sensors and material composition sensors are also briefly discussed below.

Thermal Sensors

Thermal sensing is an important element of human tactile sensing, useful for determining the material composition of an object as well as to measure surface temperatures. Since most objects in the environment are at about the same (*room*) temperature, a temperature sensor that contains a heat source can detect the rate at which heat is absorbed by an object. This provides information about the heat capacity of the object and the thermal conductivity of the material from which it is made, making it easy, for example, to distinguish metals from plastics.

Buttazzo et al. [28.46] note that the piezoelectric polymer used in their tactile sensing system is also strongly pyroelectric, and use a superficial layer as a thermal sensor. Other sensors use thermistors as transducers [28.47–49]. Some systems purposely provide an internal temperature reference and use the temperature differential from the environment to detect contacts [28.50, 51]; however, objects with a temperature the same as the reference will not be detected. Most of these sensors have a relatively thick outer skin covering the heat sensitive elements, thus protecting delicate components and providing a conformal surface at the expense of slower response time.

As a more recent example of thermal sensing, *Engel* et al. [28.52] present a flexible tactile sensor design that includes integrated gold film heaters and RTDs (resistance temperature devices) on a polymer micromachined substrate. *Lin* et al. [28.24] include a thermistor as part of the sensing suite in their artificial fingertip. While there is a high level of integration in these sensors, tradeoffs concerning construction, performance, and protection of sensing elements remain an ongoing challenge.

Material Composition Sensors

There has been some work on sensors for material composition. In analogy with the human senses of taste and smell, liquid and vapor phase chemical sensors could potentially determine the chemical composition of a surface [28.53, 54]. However, the large majority of robotic chemical sensing has involved non-contact sensing of airborne plumes [28.55, 56]. Another sensing modality which provides information about material properties is electromagnetic field sensing, using devices such as eddy current or Hall effect probes to measure ferromagnetism or conductivity [28.57, 58].

28.2.3 Kinematic Sensors

Although they are not generally regarded as tactile sensors, sensors that detect the position of a limb can provide the robot with geometric information for manipulation and exploration, particularly when the limb also includes sensors that register contact events. Examples of combining joint angle sensing and contact sensing with compliant fingers to learn about the configuration of a grasp include [28.59, 60].

28.2.4 Force and Load Sensing

Actuator Effort Sensors

For some actuators such as electric servo motors, a measure of the actuator effort can be obtained directly by measuring the motor current (typically using a sensing resistor in series with the motor). However, because motors are typically connected to robot limbs via gearboxes with output/input efficiencies of 60% or less, it is usually much more accurate to measure the torque at the output of the gearbox. Solutions to this problem include torque load cells and mechanical structures at the robot joints whose deflections can be measured using electromagnetic or optical sensors. For cable or tendon-driven arms and hands it is useful to measure the cable tension – both for purposes of compensating for friction in the drive-train and as a way of measuring the loads upon the appendage [28.61, 62]. When fingers or arms make contact with objects in the environment, cable tension sensing becomes an alternative to endpoint load sensing for measuring components of the contact forces. Of course, only those components that produce significant torques can be measured with accuracy. Chapter 19 on multifingered hands contains more details concerning tendon tension measurement.

Force Sensors

When actuator effort sensors are not sufficient to measure the forces exerted by or on a robot appendage, discrete force sensors are typically utilized. These sensors are found most often at the base joint or wrist of a robot but could be distributed throughout the links of a robot.

In principle, any type of multi-axis load cell could be used for manipulator force–torque sensing. However, the need for small, lightweight units with good static response eliminates many commercial sensors. The design of force sensors for mounting above the gripper at the wrist has received the most attention [28.63, 64], but fingertip sensors for dextrous hands have also been devised. Often these sensors are based on strain gauges mounted on a metal flexure [28.65–67] which can be fairly stiff and robust. *Sinden* and *Boie* [28.68] propose a planar six axis force–torque sensor based on capacitive measurements with an elastomer dielectric. Design considerations for force sensors include stiffness, hysteresis, calibration, amplification, robustness, and mounting. *Dario* et al. present an integrated fingertip for robotic hands integrated FSR (force sensing resistor) pressure array, piezoceramic bimorph dynamic sensor, and force–torque sensor [28.29]. More recently *Edin* et al. [28.69] have developed a miniature a multi-axis fingertip force sensor (Fig. 28.2). For applications where immunity to electromagnetic noise is desirable, *Park* et al. [28.70] present a robot fingertip with embedded fiber optic Bragg gratings, used as optical strain gages. *Bicchi* [28.71] and *Uchiyama* et al. [28.72] consider the optimal design of multi-axis force sensors in general.

Information from the force sensors can also be combined with knowledge of fingertip geometry to estimate contact location, as implied in Fig. 28.3. This method of contact sensing is referred to as *intrinsic tactile sensing* and was first presented by *Bicchi* et al. [28.73]. A comparison between intrinsic and extrinsic contact sensing

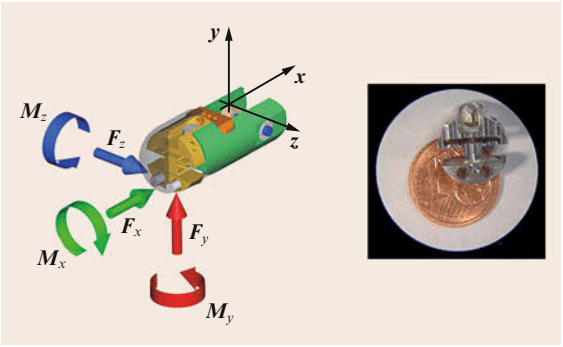


Fig. 28.2 Miniature fingertip force–torque sensor for a prosthetic hand (after [28.69])

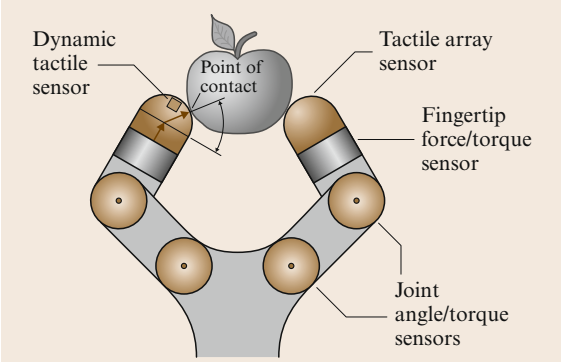


Fig. 28.3 Robot hand with fingertip force and tactile sensing. Information from the force sensors can be combined with knowledge of fingertip geometry to estimate contact location, referred to as *intrinsic tactile sensing*

(i. e., using distributed contact sensors) is presented by *Son* et al. [28.74]. This topic is discussed further in Sect. 28.3.1.

28.2.5 Dynamic Tactile Sensors

Taking a cue from the role of fast-acting or dynamic tactile sensors in human manipulation [28.75], researchers have developed dynamic sensors for slip detection and for sensing textures and fine features.

Early slip sensors based on displacement detected the motion of a moving element such as a roller or needle in the gripper surface [28.76, 77]. Subsequent work has typically used accelerometers or other sensors that are inherently sensitive to small, transient forces or motions. Early examples include [28.31, 78–80]. Many subsequent contributions are reviewed in [28.81].

For the case of hard objects held in metal grippers, acoustic emissions may reveal slip [28.82]. Because these signals are at very high frequency (over 100 kHz) they can be useful for distinguishing among different kinds of contact events in noisy environments [28.83]. Another approach for improving the signal/noise robustness of dynamic sensing is to use an actively stimulated sensor and measure the change in response as contact conditions change [28.84, 85].

For grasp force control, it is especially useful to detect incipient slips, which are accompanied by small, localized vibrations, or micro-slips, at the periphery of a finger/object contact region before gross sliding occurs. The challenge is to distinguish these from other events that produce vibrations [28.86–89].

For detecting features and fine surface features it can be effective to use small fibers, a skin with fingerprint-like ridges, or a stylus that drags over the surface like a fingernail [28.46, 90, 91].

Because many dynamic tactile sensors only produce a transient response, they are often combined with pressure sensing arrays or force sensors to provide a combination of low frequency and high frequency tactile sensing [28.24, 29, 32, 46, 89, 92, 93]. An alternative, which can be simpler from the standpoint of sensor integration, is to use conventional pressure sensing arrays for slip detection. In this case, the array resolution and scanning rate must be sufficient to detect motion or incipient motion quickly, to prevent grasp failures. Fortunately, this is becoming increasingly feasible [28.19, 81, 94, 95]. VIDEO 14 shows an example of dynamic tactile sensing.

28.2.6 Array Sensors

Hundreds of designs for tactile array sensors have appeared in the literature in the last 25 years, and many of them are designed for use with dextrous hands. In terms of transducers, the fundamental requirement is to unambiguously recover either the shape or pressure distribution across the contact. Shape sensing requires a compliant skin, which can also have advantages for grasp stability (see Chaps. 37 and 38 on contact modeling and grasping). Examples of shape-sensing tactile arrays include [28.23, 24]. However, the far more common approach is to measure subsurface strains, which can be correlated with surface pressure distributions, as presented by *Fearing* and *Hollerbach* [28.96] and discussed further in Sect. 28.3.2.

Contact Location Sensors

The simplest and perhaps most robust tactile arrays provide measurements solely of contact location. Some such sensors utilize a membrane switch design like that found in keyboards [28.97]. As another example, a robust two-dimensional (2-D) switch array can be embedded in a prosthetic hand [28.69]. Some optical tactile sensors have also been used primarily as contact location sensors. *Maekawa* et al. [28.98] used a single optical position sensing device (PSD) or a CCD camera array to detect the position of scattered light off of a hemispherical optical wave guide fingertip with a silicone rubber cover. Light is scattered at the locations of contacts. With a textured skin, the magnitude of the force can also be estimated, as the contact area grows in proportion to the pressure. However, an issue with fingertips that use a compliant skin covering a hard substrate is that adhesion between the two materials results in hysteresis. In addition, when the fingertip is dragged over a surface, the friction can produce a shift in the estimated contact position.

Pressure Sensing Arrays

Capacitive Pressure Sensing Arrays. Capacitive pressure sensing arrays are one of the oldest and most common tactile sensor types. Some of the earliest analysis of such sensors is presented by *Fearing* [28.99] for capacitive arrays embedded into a soft robot fingertip. The arrays consist of overlapping row and column electrodes separated by an elastomeric dielectric, forming an array of capacitors. A change in capacitance results from compressing the dielectric between row-column plates at a particular intersection. The equation for capacitance, based on the physical parameters, is expressed as $C \approx \epsilon A/d$, where ϵ is the permittivity of the dielectric between the plates of the capacitor, A is the area of the plates, and d is the spacing between them. Pressing on the skin reduces the plate spacing d, thus providing a linear response with displacement. Through appropriate switching circuitry, a region of a sensor array can be isolated at a particular

row/column intersection. Examples of similar capacitive tactile arrays can be found in [28.100, 101] and commercially [28.14]. Large capacitive arrays have also been developed to cover the arms of a robot [28.15–17]. A woven fabric with capacitive sensing junctions is reported in [28.102].

With a suitable dielectric and plate design, capacitive arrays can be robust, have a large dynamic range (ratio between minimum and maximum detectable pressure) and low hysteresis, which is desirable for fast response. However, a common problem is the need to shield the array from stray capacitance effects (e.g., from an approaching metallic surface) and to minimize *parasitic* capacitance associated with wiring to and from the active elements. For these reasons, the recent trend is to use local microprocessors specialized for capacitance measurement as part of an integrated sensing system (Sect. 28.4). An example is shown in Fig. 28.4. VIDEO 15 actually shows results taken with the sensor depicted in Fig. 28.4.

Piezoresistive Pressure Sensing Arrays. Many researchers have produced tactile sensor arrays that are piezoresistive. In most cases, these sensors utilize a conductive rubber that is bulk molded or a piezoresistive ink that is patterned via screen printing or stamping. Each of these approaches employs a conductive additive (typically carbon black or silver) to create its conductive/piezoresistive behavior. As a more flexible and durable alternative, some researchers have developed fabric-based piezoresistive sensors, discussed separately below.

Russell [28.27] presented one of the first molded conductive rubber tactile sensor arrays composed of conductive rubber column and row electrodes with piezoresistive junctions. However, this sensor exhibited significant drift and hysteresis, which fueled research to minimize these effects through proper selection of molding material [28.8]. These issues were never completely solved due to the hysteretic nature of elastomers, but this sensing approach remains attractive due to its ease of manufacturing. Hence, it has continued to find applications, for example, in the appendages of humanoid robots where extreme accuracy is not required [28.104]. Significant improvements in dynamic range and robustness are also possible using a treated elastomer that exploits a quantum tunneling effect [28.105], and commercial versions are now available [28.106].

A number of researchers and companies have developed tactile sensors that utilize conductive (piezoresistive) ink, generally referred to as FSRs (force sensitive resistors). This is by far the most common and economical means to incorporate tactile sensing via off-the-shelf sensors. However, to make highly integrated, dense sensor arrays, custom fabrication is necessary. Examples of such sensors include [28.9, 29]. To take this basic concept a step further, *Someya* [28.107] has produced robotic skin that employs patterned organic semiconductors for local amplification of the piezoresistive sensor array, printed on flexible polyimide film. Although fabricated on a flexible substrate, these sensor arrays are somewhat susceptible to bending fatigue.

Piezoresistive fabrics have been developed to address fatigue and fragility issues found in tactile arrays. Examples of these sensors are presented in [28.10–12, 108] and are utilized in applications such as the arms or legs of humanoid robots. Because this technology has the potential to replace ordinary cloth, it is a promising technology for applications in wearable computing or even smart clothing. A different solution to the problem of making a sensor that will work on joints and locations with extensive flexing is to employ fine channels of liquid metal in a stretchy elastomer [28.109].

One final design that does not fall under the above fabrication categories is a sensor designed by *Kageyama* et al. [28.110] that utilizes a piezoresistive conductive gel pressure array along with a multi-level contact switch array via variable contact resistance within sensor layers that they developed for use in humanoid robots.

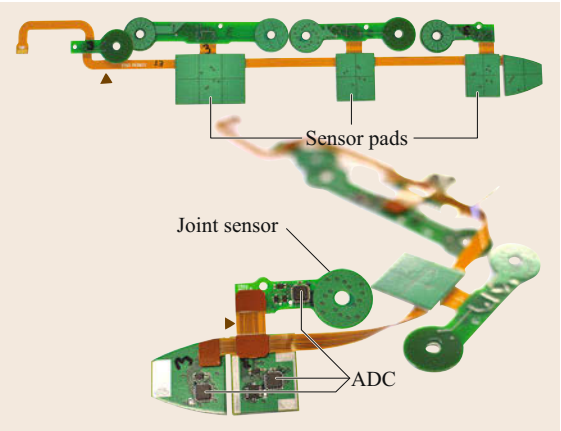


Fig. 28.4 Capacitive touch and joint angle sensors on a flexible circuit for incorporation in a robotic hand, sharing the same microprocessors for signal processing and communication (after [28.103])

MEMS arrays. MEMS technology is attractive for producing dense sensor arrays with integrated packaging, signal processing and communications electronics. Early devices were produced in silicon through standard silicon micro-machining techniques, allowing high spatial resolution and dedicated hardware for multi-

plexing, etc. [28.111, 112]. However, such silicon-only devices can be brittle and difficult to integrate into a robust, compliant skin. More recent efforts have addressed these problems by using combinations of semiconductors and other materials, including organic semiconductors [28.18, 19, 52, 113, 114].

Returning to the idea of using silicon load cells for tactile sensing, *Valdastri* et al. [28.115] developed a miniature MEMS silicon-based three-axis load cell that resembles a joystick and is appropriate for embedding within an elastomeric skin for detecting shear and normal stresses (Fig. 28.5) [28.93]. Other recent efforts include flexible arrays with high spatial resolution using capacitive or piezoresistive technology [28.20, 116].

Skin Deflection Sensing

Brocket [28.117] was one of the first to propose the idea of using deformable membranes as robot fingertips. As noted by *Shimoga* and *Goldenberg* [28.118], there are several advantages to using deformable fingertips over more rigid robot fingertips, which include: 1) improved grasp stability, 2) reduced shock, and 3) reduced fatigue for embedded sensor elements. Other early work includes a deformable fingertip with skin covering polyurethane foam and instrumented with elastomeric strain gages [28.27].

Nowlin [28.25] used Bayesian algorithms to improve data interpretation of a deformable tactile sensor that used magnetic field sensing. A 4×4 array of magnets were supported above paired hall-effect sensors off a rigid base by individual fluid filled balloons. Hall-effect sensors measure the strength of the local magnetic field, which should increase with proximity to the magnets; however, this is complicated by neighboring magnets in the array. Hence, noisy data were combined using Bayesian algorithms to predict membrane deformation for a deformable fingertip.

Later, *Russell* and *Parkinson* [28.26] developed an impedance tomographic tactile sensor, capable of measure skin deformation over an 8×5 array. This sensor was constructed with of neoprene rubber and filled with distilled water, similar to the more recent BioTac sensor [28.24]. Row and column electrodes were made from copper and conductive rubber for the rigid substrate and neoprene skin, respectively. Like the capacitive tactile sensors described above, this sensor utilized multiplexing electronics to reduce the number of electrical interconnects. Square waveform driver electronics are used to estimate the resistance of a column of water formed between row and column elements providing a signal that is proportional to the current skin height.

Ferrier and *Brocket* [28.23] implemented a tactile sensor which used optical tracking in combination with models of the skin to predict sensor fingertip deformations. The fingertip sensor consists of a tiny CCD camera focused on a 7×7 array of dots marked on the inside of a gel-filled silicone membrane. An algorithm is used to construct a 13×13 grid over the array of dots. This algorithm uses a combination of the position that is sensed by the CCD camera, which provides the location along a line radially outward from the focal point, in combination with a mechanical model used to solve for the radial distance from the camera focal point, based on energy minimization.

A more recent example of a shape sensing array is the commercial Syntouch BioTac that grew out of research on a multimodal fingertip tactile sensor [28.24]. In this design, measuring the changes in resistance between terminals in a water-filled skin provides an estimate of the skin deformation.

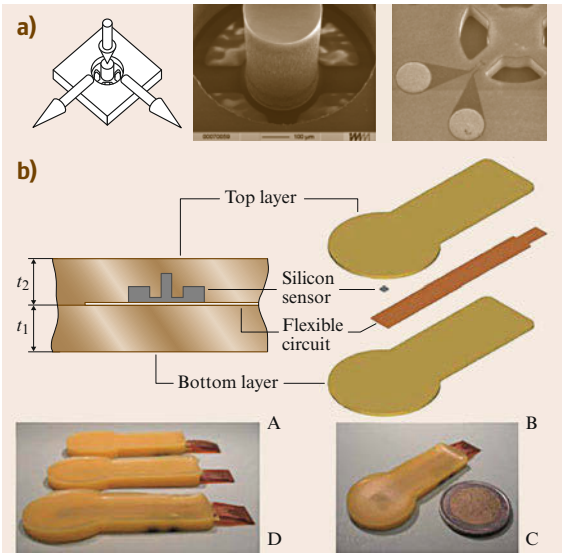


Fig. 28.5 **(a)** Scanning electron microscope (SEM) micrographs of a MEMS 3-axis tactile force sensor. **(b)** MEMS force sensor wirebonded to a flex-circuit and embedded within a silicone rubber skin (after [28.115])

Other Array Tactile Array Sensors

While less common than capacitive or piezoresistive sensors, optical tactile sensors have consistently been attractive for their immunity to electromagnetic interference. The basic approaches include designs that use a small camera to measure skin deformations and arrays of optical emitters and detectors. Optical fibers can also be used to separate the transducer from the contact site [28.42, 119].

An interesting tactile sensor uses vision to track an array of spherical markers embedded in a transparent elastomer to infer the stress state of the skin material

due to applied forces [28.21]. This sensor is currently being commercialized under the tradename GelForce.

Miniaturization of optical components has also made it possible to construct a tactile array of surface-mounted optical devices. In a physically robust design that can be adapted for shear and normal sensing, emitters and detector pairs are covered by a thin translucent layer of silicone rubber followed by an opaque outer layer. As the silicone skin is depressed, there is a variation in the amount of light reflected from the emitter to the detector [28.22, 120].

The concept of reflection off the underside of a compliant skin has also been used with acoustic or ultrasonic sensors. *Shinoda* et al. [28.121] present a sensor that looks at the change in reflected accoustic energy from a resonator chamber near the surface of the skin, with application to friction measurement [28.122]. *Ando* et al. [28.123] present a more sophisticated ultrasound sensor that achieves 6-DOF (degree of freedom) displacement sensing via paired plate elements that utilize 4 ultrasound transducers per plate.

Multimodal Arrays

Human sensing is inherently multi-modal, with combinations of fast- and slow-acting mechanoreceptors as well as specialized thermal and pain receptors. In an analogous approach, a number of researchers have developed multi-modal tactile sensing suites that combine mechanical and thermal sensing [28.24, 49, 52, 124, 125]. As noted earlier, other work has combined static and dynamic sensing [28.24, 29, 32, 46, 92, 93, 126].

28.3 Tactile Information Processing

In discussing the processing of tactile information, we return first to the three main uses depicted in Fig. 28.1. For manipulation, we require, foremost, information about contact locations and forces so that we can grasp objects securely and impart desired forces and motions to them. For exploration, we are concerned with obtaining and integrating information about the object, including the local geometry, hardness, friction, texture, thermal conductivity, etc. For response, we are concerned especially with the detection of events, such as contacts produced by an external agent, and in assessing their types and magnitudes. The uses of information are often coupled. For example, we manipulate objects in order to explore them, and we use the information obtained through object exploration to improve our ability to control forces and motions in manipulation. Recognizing contact events is also important for manipulation and exploration, as it is for response.

28.3.1 Tactile Information Flow: Means and Ends of Tactile Sensing

Figure 28.6 summarizes the general flow of information from each type of sensor reviewed in the previous section through primary sensed quantities to information provided for manipulation, exploration and response. A useful thought exercise is to consider exactly what information we use to perform a task such as turning a pen end-over-end between the fingers. We can easily perform this task with our eyes closed. What information are we using? We need to track the position and orientation of the pen and to monitor the forces that we impose on it to maintain stable manipulation. In other words, we need to know the *configuration* of our grasp, the *locations and movements of contacts* over the surfaces of our fingers, the *magnitudes of grasp forces* and the *contact conditions* with respect to friction limits, etc. The same requirements apply for robots and are provided by the information flow in Fig. 28.6.

At the upper left corner of the figure, joint angles, combined with the forward kinematic model of the manipulator and knowledge of external link geometries, establish the positions and orientations of coordinate frames embedded in the fingertips. This information is needed to integrate local information about object shape, surface normal orientation, etc. so that the overall geometry and pose of the object can be determined.

Actuator effort sensors provide information about the resultant forces, using the Jacobian transpose: $\mathbf{J}^{\mathrm{T}}\boldsymbol{f} = \boldsymbol{\tau}$, where $\boldsymbol{f}$ is an $n \times 1$ vector of external forces and moments taken with respect to a coordinate frame embedded in the appendage. $\mathbf{J}^{\mathrm{T}}$ is the Jacobian transpose, mapping external forces and moments to joint torques, and $\boldsymbol{\tau}$ is an $m \times 1$ vector of joint torques for a serial kinematic chain with m degrees of freedom. We require that the k-th column of $\mathbf{J}^{\mathrm{T}}$ have elements that are relatively large compared to the overall condition number of $\mathbf{J}$ to provide an accurate measurement of the k-th element of $\boldsymbol{f}$. *Eberman* and *Salisbury* [28.127] show that it is possible to measure contact force and location using only joint torque measurements if the manipulator has clean dynamics.

Alternatively, we can use a multi-axis force–torque sensor in the fingers, as shown in Fig. 28.3, or robot wrist to obtain contact forces. This approach has the advantage of providing dynamic force signals with

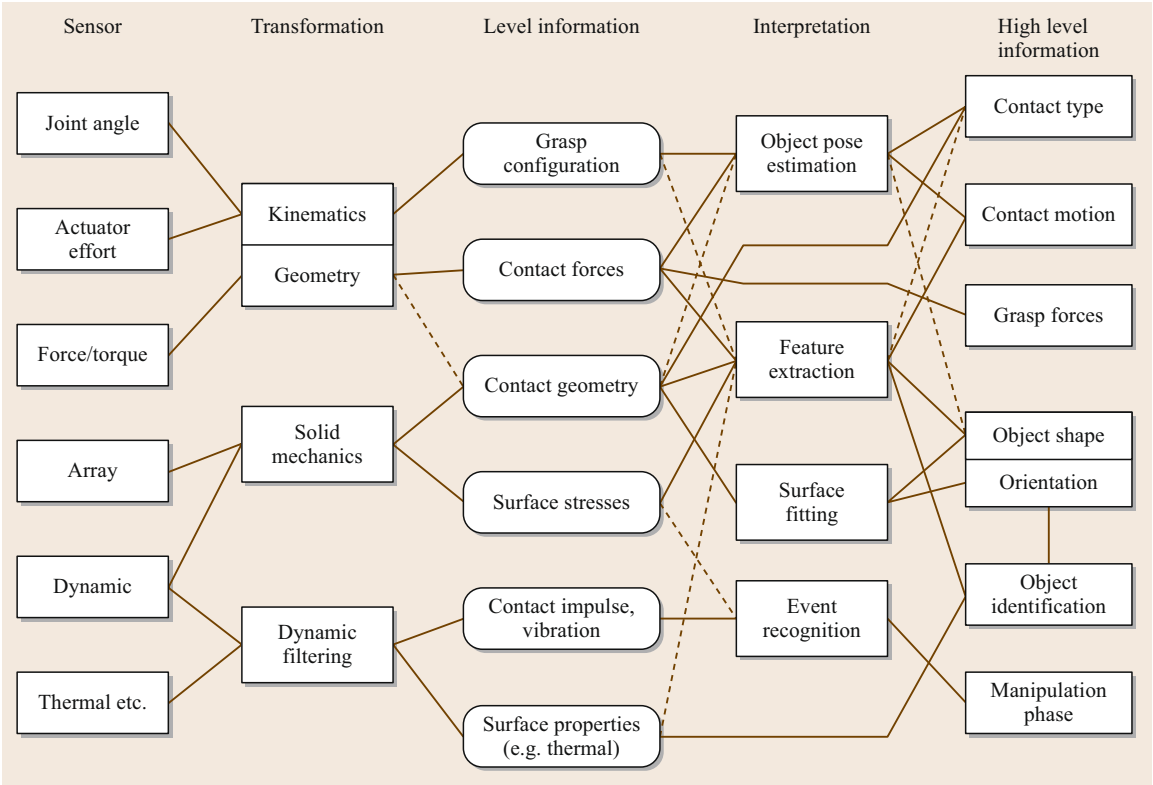


Fig. 28.6 Force and tactile sensor information flow and signal processing

a higher signal/noise ratio because they are not masked by the inertias of the robot arm or fingers and their transmissions. If the geometry of the fingertip is known, one can use *intrinsic tactile sensing* [28.65, 128] to compute the contact location as well as the contact force by examining ratios of resultant forces and torques at the sensor.

When the contacts are small compared to the fingertips (so that a point-contact approximation applies) and the fingertips are convex shapes, the contact location is easily computed. Fig. 28.7 shows a contact force $\boldsymbol{f}$, contacting the fingertip surface at a location $\boldsymbol{r}$. A force–torque sensor such as that in Fig. 28.2 measures the moment, $\boldsymbol{\tau} = \boldsymbol{r} \times \boldsymbol{f}$, with respect to the origin. If we consider the lever arm $\boldsymbol{h}$, perpendicular to the line of action of $\boldsymbol{f}$, then $\boldsymbol{h}/h = \boldsymbol{f}/f \times \boldsymbol{\tau}/\tau$, where $h = \tau/f$ is the magnitude of $\boldsymbol{h}$. We can then write that $\boldsymbol{r} = \boldsymbol{h} - \alpha \boldsymbol{f}$, where α is a constant obtained by solving for the intersection of the line of action and the fingertip surface. For a convex fingertip, there will be two such points, of which only one corresponds to a positive (inward) contact force.

From the contact location one can deduce the local contact normal and contact kinematic type from a small number of force measurements. *Bicchi* presents algorithms for extending these methods to soft fingers [28.128]. *Brock* and *Chiu* [28.66] describe the use of force sensors for the perception of object shape using this approach and for measuring the mass and center of mass of a grasped object.

For precision tasks involving small objects or small forces and motions, cutaneous sensors provide the most sensitive measurements. In general, as task require-

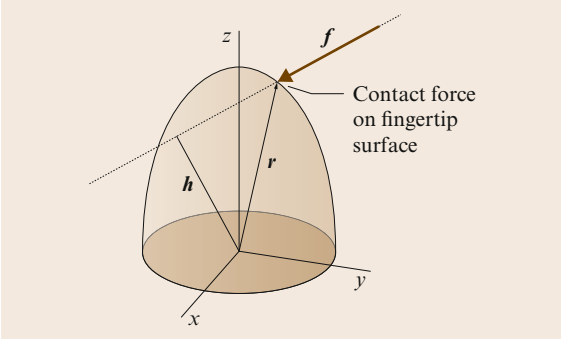


Fig. 28.7 Intrinsic tactile sensing: a contact produces a unique line of action and moment about the origin of a coordinate system in the fingertip. The contact location can be obtained by solving for the intersection of the line of action and the fingertip surface

ments get smaller, the sensor must be located closer to the contact so that the compliance and inertia of the intervening parts of the manipulator do not interfere with the measurement. *Dario* [28.129] suggests that finger tip force sensors are useful for forces of 0.1–10.0 N while array sensors can measure distributed forces of 0.01–1.0 N. *Son* et al. [28.74] find that intrinsic tactile sensing and array sensors can both provide accurate (within 1 mm) estimates of contact location; however, the intrinsic tactile sensing method is inherently sensitive to the accuracy of the force–torque sensor calibration and can produce transient errors due to unmodeled dynamic forces.

Proceeding down the left side of Fig. 28.6 we come to the large category of cutaneous array sensors. The interpretation of information from array sensors depends initially on the transducer type. For arrays of binary contact or proximity sensors, interpretation amounts mainly to establishing the location and general shape of the contact area. Techniques common to binary vision can be used to obtain sub-pixel resolution and to identify contact features. This information, in combination with measurements of the grasp forces from actuator effort or force–torque sensors, is sufficient for basic manipulation tasks [28.69].

28.3.2 Solid Mechanics and Deconvolution

A basic problem associated with tactile array sensors is to reconstruct what is happening at the surface of the skin from a finite set of measurements obtained beneath the surface. Typically we are interested in determining pressure and perhaps shear stress distributions associated with contacts on the skin. In other cases, as when the fingertips consist of a gel or soft foam covered by a thin membrane so that the pressure is nearly constant, the local geometry of the contact is of interest.

In the following example we consider the case of an array of elements located at a depth d below the surface of an elastomeric skin. A contact has resulted in a pressure distribution over the region of interest. We establish a coordinate system with z pointing in the inward normal direction and, for simplicity, we examine a one-dimensional loading case, $p(y)$, in which the pressure distribution is unchanging in the x direction. We further assume that the extent of the skin in the x direction is large compared to the skin thickness so that strains in the x direction are inhibited, leading to a plane strain elasticity problem. We assume that the skin is a homogenous, isotropic material and the strains are small enough that linear elasticity theory can be applied. Of course, none of these assumptions is entirely valid in practical cases; however, the results do agree qualitatively with the measurements obtained with actual robot fingers and tactile arrays. A thorough discussion of the general approach and of the accuracy of the linear elastic models can be found in [28.91, 96, 99, 130, 131].

Figure 28.8 illustrates the case of two line loads, or knife edges, pressed against the surface of the skin (akin to a planar version of the two-point discrimination test for human tactile acuity). The solution for a single line load, or impulse response, was derived by Boussinesq in 1885. For the case of plane strain the principal stresses in the (y, z) plane from a normal unit impulse can be expressed in cartesian coordinates as [28.132]

$$\sigma_z(y,z) = \left(\frac{-2}{\pi z}\right)\frac{1}{[1+(y/z)^2]^2}\,, \tag{28.1}$$

$$\sigma_y(y,z) = \left(\frac{-2}{\pi z}\right)\frac{(y/z)^2}{[1+(y/z)^2]^2}\,, \tag{28.2}$$

$$\sigma_x(y,z) = \nu(\sigma_y+\sigma_z)\,, \tag{28.3}$$

where ν is Poisson's ratio for the material (typically 0.5 for elastomeric materials).

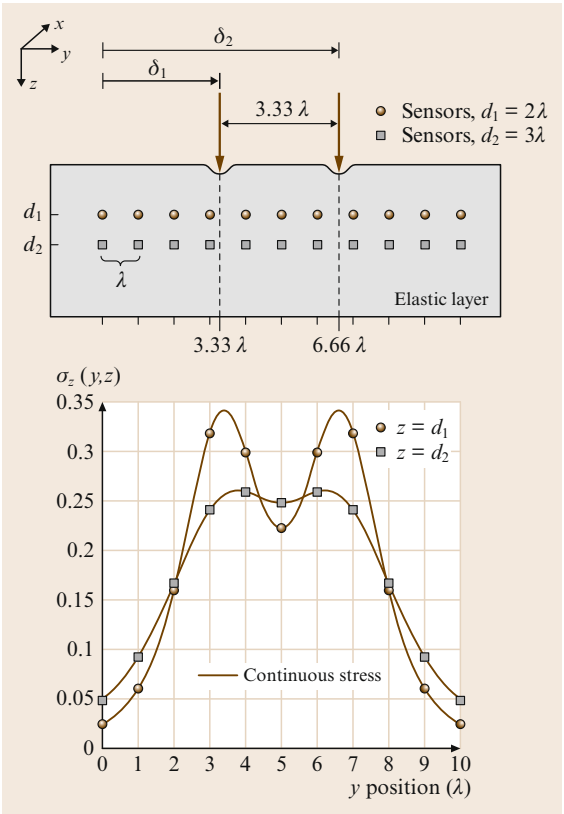


Fig. 28.8 Plane strain stress response for two (unit magnitude) line loads. Note the *blurring* that occurs with greater depth

For two such line loads located at distances δ_1 and δ_2 from the origin, the solution can be obtained by superposition

$$\sigma_z(y,z) = \left(\frac{-2}{\pi z}\right)\left\{\frac{1}{\left[1+\left(\frac{y-\delta_1}{z}\right)^2\right]^2} + \frac{1}{\left[1+\left(\frac{y-\delta_2}{z}\right)^2\right]^2}\right\}, \tag{28.4}$$

$$\sigma_y(y,z) = \left(\frac{-2}{\pi z}\right)\left\{\frac{\left(\frac{y-\delta_1}{z}\right)^2}{\left[1+\left\{\frac{y-\delta_1}{z}\right\}^2\right]^2} + \frac{\left(\frac{y-\delta_2}{z}\right)^2}{\left[1+\left\{\frac{y-\delta_2}{z}\right\}^2\right]^2}\right\}. \tag{28.5}$$

For more general pressure distributions the stresses can be found by convolution of the pressure distribution, $p(y)$, and the impulse response $G_i(y,z)$

$$\sigma_i = \int_{\tau=-\infty}^{\tau=y} [p(\tau)\mathrm{d}\tau] G_i(y-\tau,z)\,. \tag{28.6}$$

Also plotted in Fig. 28.8 are curves corresponding to the vertical stress components, σ_z, at two different depths $d_1 = 2\lambda$ and $d_2 = 3\lambda$, where λ is the sensor spacing. As we go deeper beneath the skin, the stresses become smoothed or blurred, and the ability to distinguish between closely spaced impulses diminishes. However, the blurring of concentrated pressure distributions can also provide an advantage when we have a limited number of sensors because the stresses and strains spread over a larger area and are more likely to affect at least one sensor. The elastic skin also provides a kind of automatic *edge enhancement* because stresses are high at the transitions between loaded and unloaded regions of the skin.

In most cases, for example in the case of capacitive or magnetic sensors, the sensing elements will measure strains or local deformations of the skin material in the vertical direction. In a few cases, such as pieces of piezoelectric film embedded in an elastomeric skin [28.91], the sensors are sufficiently stiff compared to the surrounding material that they can be considered to measure stresses directly.

For the case of elastic plane strain, the strains are related to the stresses by [28.133]

$$\epsilon_y = \frac{1}{E}\left[\sigma_y - \nu(\sigma_x+\sigma_z)\right]\,, \tag{28.7}$$

$$\epsilon_z = \frac{1}{E}\left[\sigma_z - \nu(\sigma_x+\sigma_y)\right]\,, \tag{28.8}$$

where E is the Young's modulus and ν is the Poisson's ratio, which we assume is 0.5 for an elastomeric skin.

Figure 28.9 shows the typical measurements that may be obtained from a row of sensing elements from the two line loads applied in Fig. 28.8. Each bar corresponds to the strain, ϵ_{zi} measured by a corresponding element and computed using (28.8), with stresses obtained from (28.3)–(28.5).

The problem at this point is to produce a best estimate of the surface pressure distribution, $p(y)$, from this finite set of subsurface strain measurements. The problem is a classic example of estimating a signal from a sparse set of remote measurements. One approach to this process is based on deconvolution techniques [28.91, 96, 99]. The measured signal from the sensors $\boldsymbol{\epsilon}_z$ is convolved with the inverse of the impulse strain response $H(y)$ to find an estimate of the surface pressure that produced the signals. The inversion tends to amplify high frequency noise, and the inverse filter bandwidth must be limited according to the spatial density of the sensors and their depth below surface.

Another approach [28.21, 130] is to assume that the surface pressure distribution can be approximated by a finite set of impulses $\boldsymbol{p} = [p_1, p_2 \ldots p_n]^t$. The sensor readings form a vector, $\boldsymbol{\epsilon} = [\epsilon_1, \epsilon_2 \ldots \epsilon_m]^t$, where $m > n$ for the bandwidth limitations discussed above. The strain response can then be written as a matrix

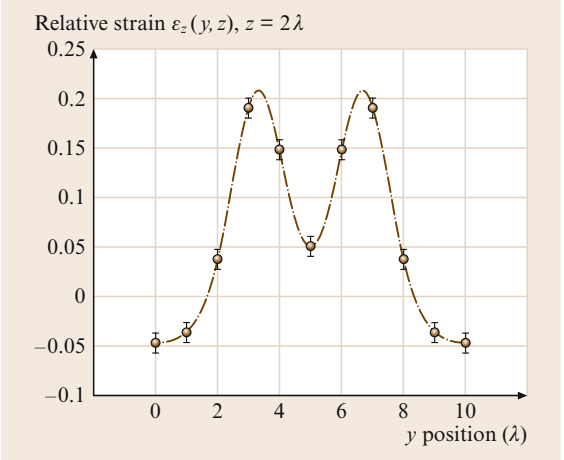


Fig. 28.9 Measured strain with assumed 10% noise

equation

$$\boldsymbol{\epsilon} = \mathbf{H}\boldsymbol{p} \,. \tag{28.9}$$

Each element of $\mathbf{H}$ is computed using (28.8) with σ_z and σ_y computed using equations similar to (28.4) and (28.5) and with σ_x from (28.3). The estimated discrete pressure distribution is then found by taking the pseudoinverse of $\mathbf{H}$

$$\hat{\boldsymbol{p}} = \mathbf{H}^{+}\boldsymbol{\epsilon} \,. \tag{28.10}$$

Using the strain measurements from Fig. 28.9, at a depth $d = 2\lambda$, the estimated pressure distribution using the pseudoinverse method is seen in Fig. 28.10. In this example, because the assumed set of seven impulses matches fortuitously with the actual loading, the reconstruction is fairly accurate despite the assumed noise.

An alternative approach to constructing soft robot finger tips is to enclose a compliant medium such as foam rubber or fluid in a thin elastic membrane [28.25, 27, 117, 134–137]. Some of the tactile array sensors developed for these fingers are able to measure directly the shape of the membrane, so that a physical model is not needed for interpretation of the signal [28.25]. Another sensing scheme uses an array of magnetic sensors at the center of the finger to measure the changes in the magnetic field generated by deformations of the magnet-loaded membrane [28.137]. A statistical algorithm has been developed that can robustly determine the membrane shape from the sensor signals [28.25]. However, a mechanical model is still required to find the pressure distribution across the contact from the shape information provided by all of these sensors.

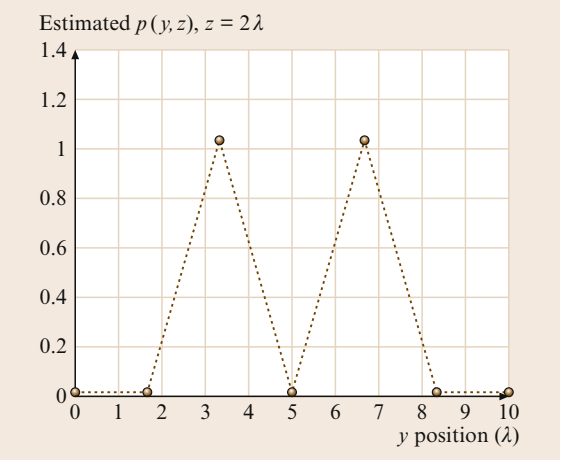


Fig. 28.10 Estimated surface pressure distribution using pseudoinverse method for 11 sensors and 7 assumed impulses

28.3.3 Curvature and Shape Information

Another alternative to measuring subsurface strains or deflections is to measure directly the local curvature at each element of an array of sensors. The curvature information can be applied directly toward identifying contact type and centroid location or integrated to obtain the local shape of the contact [28.28, 138].

Returning to Fig. 28.6, once the local contact shape or geometry has been established, the next steps typically include feature identification (e.g., identifying corners or ridges on an object) and determining the overall shape and pose of the object in the hand.

Often the object shape is at least partially known a priori in which case a variety of surface or data fitting methods can be used. For example, *Fearing* [28.139] developed a method for calculation of the radius of curvature and orientation of a generalized cylinders from tactile array data and [28.140] developed a neural network that performs a similar calculation. Other schemes use contact locations, surface normals and contact forces to determine information about object shape and orientation with respect to the hand [28.141–144].

Allen [28.145] uses several different primitive representations for object shape attributes based on the particular exploratory procedure used to sense the object. Object volume and approximate shape are perceived with enclosure grasping, and the resulting shape is modeled using superquadric surfaces. Similarly, measurement of the lateral extent of object faces leads to a face-edge-vertex model and contour following to a generalized cylinder representation.

The question of what constitutes an appropriate set of features is not well understood, although it clearly depends on the intended application. *Ellis* [28.146] considers appropriate feature sets and methodologies for acquiring the needed data. *Lederman* and *Browse* [28.147] suggest that surface roughness, surface curvature, and oriented edges are used in human haptic perception.

28.3.4 Object and Surface Identification

The most common application of touch information has been in object recognition and classification. In object recognition the goal is to identify one object from a set of known objects using information derived from touch. In classification the goal is to categorize objects according to preselected sensed properties. These systems are usually based on geometric information derived

from tactile array or force sensors, although some approaches use additional information (e.g., compliance, texture, thermal properties and the kinematic pose of the hand) [28.148–154].

28.3.5 Active Sensing Strategies

Because touch provides only local information, movement is an integral part of touch sensing for recognition and exploration. A review of tactile sensing focusing on exploration and manipulation is found in [28.2]. Several researchers have developed strategies for scheduling sensor movements so that each additional observation decreases the number of objects which are consistent with prior observations. This is sometimes described as a *hypothesize and test* approach. Early examples include [28.141, 155–157].

Klatzky et al. [28.149] have suggested that robotics systems can employ the same exploratory procedures used by humans in haptic exploration. These procedures prescribe the finger motions needed for tasks such as tracing object contours, measuring compliance, and determining the lateral extent of object surfaces. Early work on exploratory tactile sensing includes [28.145, 158, 159]. More recent examples include [28.32, 160, 161].

Edge tracking and surface following strategies have received particular attention. Early examples include [28.162–166]; a more recent example is found in [28.167].

28.3.6 Dynamic Sensing and Event Detection

For dynamic tactile sensors used to detect such events as gentle contacts or slippage between the fingertips and an object, the main challenge is to reliably detect the event in question without false positives. The dynamic tactile sensors that produce large signals in response to contact events are also prone to producing large signals in response to vibrations from the robot drive train and to rapid accelerations of the robot hand. Solutions for more robustly detecting contact events include comparing the signals from dynamic tactile sensors at and away from the contact regions and statistical pattern recognition methods to identify the *signature* of true contact events. Early examples include [28.86, 127, 168]. More recent examples have taken advantage of the much greater realtime processing capability that is now available for signal processing and event classification [28.32, 161, 169, 170].

28.3.7 Integration of Thermal and Other Sensors

Sensors such as thermal contact sensors are rarely used in isolation; their signals are generally integrated with those from tactile arrays and other sensors to produce additional information for identifying objects. Examples of integrated thermal and mechanical sensors for surface characterization include [28.24, 52, 171].

28.4 Integration Challenges

A critical problem that we have not yet addressed is the difficulty of connecting to a large and diverse array of tactile sensors. In 1987 *Jacobsen* et al. [28.172] cited the routing of wires as perhaps the most difficult problem in dexterous hand design and, to a large extent, this remains true today. However, some solutions to this problem have been presented in recent years using wireless sensor arrays or digital busses for power and signal connections.

In early work, *Shinoda* and *Oasa* [28.173] embedded tiny wireless sensing elements in an elastic skin that uses an inductive base coil to provide power and signal transmission. *Hakozaki* and *Shinoda* [28.174] embed tactile sensor chips between 2 layers of conductive rubber provide power and serial communication. In other work, [28.18, 20] produce large scale printed arrays with built-in multiplexing and communications.

In a different approach, *Yamada* et al. [28.168] use wireless sensor chips and light transmitted through a transparent elastomer both for power and to communicate to a power-receiver chip. *Ascari* et al. [28.175] propose a communications scheme with all information transmitted along optical fibers.

In recent years, the proliferation of microprocessors that can perform local signal conditioning, multiplexing and digital communications has been particularly helpful.These devices, adapted from touch-screens, are small enough to be located at the fingertips of a hand. One solution is to modify the barometric pressure sensing chips common in cell phones, converting them into pressure sensing elements [28.95, 176]. Another is to use processors adapted for capacitive touch screens [28.94, 103]. Still another involves arrays of optical tactile devices [28.22].

28.5 Conclusions and Future Developments

In comparison to computer vision, tactile sensing always seems to be *a few years away* from widespread adoption. As explained in the introduction to this chapter, the reasons include physical problems (placement and robustness of sensors, wiring challenges) and the diversity of sensor types for detecting forces, pressures, local geometries, vibrations, etc. As we have seen, the transduction and interpretation methods are typically different for each of these tactile quantities. However, there are some basic issues that apply to tactile sensing in general. For example, sensors are generally located within or beneath a compliant skin, which affects the quantities that they sense in comparison to pressures, stresses, thermal gradients or displacements applied to the skin surface.

When choosing tactile sensors for a robot arm or hand, it is effective to begin with a consideration of which tactile quantities are most desired and for what purpose. For example, if the main concern is to obtain accurate measurements of loads or contact forces at sufficient data rates for force servoing, then intrinsic tactile sensing may make the most sense. If manipulating objects with soft contacts and with sliding or rolling, curved array sensors for measuring pressure distributions, or perhaps local skin deflections, may be desirable. If exploring objects to learn about their texture and material composition, dynamic tactile sensors and thermal sensors may be effective.

In an ideal world, one would incorporate all these tactile sensors in a robotic end-effector without regard to cost, signal processing or wiring complexity. Fortunately, the cost and size of transducers suitable for tactile sensing are steadily dropping and the ability to perform localized processing is improving with surface-mounted devices on flexible circuits. In the near future it will be increasingly possible to fabricate dense arrays of transducers in-situ on contoured surfaces, using material deposition and laser machining techniques. In this way, robots may finally start to approach the tactile sensitivity and responsiveness of animals.

Video-References

VIDEO 14 The effect of twice dropping, and then gently placing, a two gram weight on a small capacitive tactile array
available from http://handbookofrobotics.org/view-chapter/28/videodetails/14

VIDEO 15 Capacitive tactile sensing
available from http://handbookofrobotics.org/view-chapter/28/videodetails/15

References

28.1 M.I. Tiwana, S.J. Redmond, N.H. Lovell: A review of tactile sensing technologies with applications in biomedical engineering, Sens. Actuators A Phys. **179**, 17–31 (2012)

28.2 H. Yousef, M. Boukallel, K. Althoefer: Tactile sensing for dexterous in-hand manipulation in robotics – A review, Sens. Actuators A Phys. **167**(2), 171–187 (2011)

28.3 R.S. Dahiya, G. Metta, M. Valle, G. Sandini: Tactile sensing – From humans to humanoids, IEEE Trans. Robotics **26**(1), 1–20 (2010)

28.4 C. Lucarotti, C.M. Oddo, N. Vitiello, M.C. Carrozza: Synthetic and bio-artificial tactile sensing: A review, Sensors **13**(2), 1435–1466 (2013)

28.5 M.H. Lee: Tactile sensing: new directions, new challenges, Int. J. Robotic Res. **19**(7), 636–643 (2000)

28.6 M.H. Lee, H.R. Nicholls: Tactile sensing for mechatronics-a state of the art survey, Mechatronics **9**(1), 1–31 (1999)

28.7 L.D. Harmon: Automated tactile sensing, Int. J. Robotics Res. **1**(2), 3–32 (1982)

28.8 J.-P. Uldry, R.A. Russell: Developing conductive elastomers for applications in robotic tactile sensing, Adv. Robotics **6**(2), 255–271 (1992)

28.9 T.V. Papakostas, J. Lima, M. Lowe: A large area force sensor for smart skin applications, Proc. IEEE Sens., Vol. 2 (2002) pp. 1620–1624

28.10 M. Shimojo, A. Namiki, M. Ishikawa, R. Makino, K. Mabuchi: A tactile sensor sheet using pressure conductive rubber with electrical-wires stitched method, IEEE Sens. J. **4**(5), 589–596 (2004)

28.11 D. De Rossi, A. Della Santa, A. Mazzoldi: Dressware: wearable piezo- and thermoresistive fabrics for ergonomics and rehabilitation, Proc. 19th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc., Vol. 5 (1997) pp. 1880–1883

28.12 A. Tognetti, F. Lorussi, M. Tesconi, D. De Rossi: Strain sensing fabric characterization, Proc. IEEE Sens., Vol. 1 (2004) pp. 527–530

28.13 R.S. Fearing, T.O. Binford: Using a cylindrical tactile sensor for determining curvature, IEEE Trans. Robotics Autom. **7**(6), 806–817 (1991)

28.14 Pressure Profile Systems: http://www.pressureprofile.com/

28.15 H.-K. Lee, S.-I. Chang, E. Yoon: A flexible polymer tactile sensor: Fabrication and modular expandability for large area deployment, J. Microelectromechanical Syst. **15**(6), 1681–1686 (2006)

28.16 T. Hoshi, H. Shinoda: A sensitive skin based on touch-area-evaluating tactile elements, Proc. 14th Symp. Haptic Interfaces Virtual Env. Teleoperator Syst. (2006) pp. 89–94

28.17 P. Maiolino, M. Maggiali, G. Cannata, G. Metta, L. Natale: A flexible and robust large scale capacitive tactile system for robots, IEEE Sens. J. **13**(10), 3910–3917 (2013)

28.18 T. Sekitani, M. Takamiya, Y. Noguchi, S. Nakano, Y. Kato, T. Sakurai, T. Someya: A large-area wireless power-transmission sheet using printed organic transistors and plastic MEMS switches, Nat. Mater. **6**(6), 413–417 (2007)

28.19 R.S. Dahiya, D. Cattin, A. Adami, C. Collini, L. Barboni, M. Valle, L. Lorenzelli, R. Oboe, G. Metta, F. Brunetti: Towards tactile sensing system on chip for robotic applications, IEEE Sens. J. **11**(12), 3216–3226 (2011)

28.20 K. Takei, T. Takahashi, J.C. Ho, H. Ko, A.G. Gillies, P.W. Leu, R.S. Fearing, A. Javey: Nanowire active-matrix circuitry for low-voltage macroscale artificial skin, Nat. Mater. **9**(10), 821–826 (2010)

28.21 K. Kamiyama, H. Kajimoto, N. Kawakami, S. Tachi: Evaluation of a vision-based tactile sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2004) pp. 1542–1547

28.22 M. Quigley, C. Salisbury, A.Y. Ng, J.K. Salisbury: Mechatronic design of an integrated robotic hand, Int. J. Robobotics Res. **33**(5), 706–720 (2014)

28.23 N.J. Ferrier, R.W. Brockett: Reconstructing the shape of a deformable membrane from image data, Int. J. Robotics Res. **19**(9), 795–816 (2000)

28.24 C.H. Lin, T.W. Erickson, J.A. Fishel, N. Wettels, G.E. Loeb: Signal processing and fabrication of a biomimetic tactile sensor array with thermal, force and microvibration modalities, Proc. IEEE Int. Conf. Robotics Biomim. (ROBIO) (2009) pp. 129–134

28.25 W.C. Nowlin: Experimental results on Bayesian algorithms for interpreting compliant tactile sensing data, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (1991) pp. 378–383

28.26 R.A. Russell, S. Parkinson: Sensing surface shape by touch, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (1993) pp. 423–428

28.27 R.A. Russell: Compliant-skin tactile sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 1645–1648

28.28 W.R. Provancher, M.R. Cutkosky: Sensing local geometry for dexterous manipulation, Proc. Intl. Symp. Exp. Robotics (2002) pp. 507–516

28.29 P. Dario, R. Lazzarini, R. Magni, S.R. Oh: An integrated miniature fingertip sensor, Proc. 7th Int. Symp. Micro Mach. Hum. Sci. (1996) pp. 91–97

28.30 R.D. Howe, M.R. Cutkosky: Dynamic tactile sensing: perception of fine surface features with stress rate sensing, IEEE Trans. Robotics Autom. **9**(2), 140–151 (1993)

28.31 R.D. Howe, M.R. Cutkosky: Sensing skin acceleration for texture and slip perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (1989) pp. 145–150

28.32 J.M. Romano, K. Hsiao, G. Niemeyer, S. Chitta, K.J. Kuchenbecker: Human-inspired robotic grasp control with tactile sensing, IEEE Trans. Robotics **27**(6), 1067–1079 (2011)

28.33 J. Lee, S.N. Sponberg, O.Y. Loh, A.G. Lamperski, R.J. Full, N.J. Cowan: Templates and anchors for antenna-based wall following in cockroaches and robots, IEEE Trans. Robotics **24**(1), 130–143 (2008)

28.34 T.J. Prescott, M.J. Pearson, B. Mitchinson, J.C. Sullivan, A. Pipe: Whisking with robots: from rat vibrissae to biomimetic technology for active touch, IEEE Robotics Autom. Mag. **16**(3), 42–50 (2009)

28.35 R.A. Russell: Using tactile whiskers to measure surface contours, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1992) pp. 1295–1299

28.36 M. Kaneko, N. Kanayama, T. Tsuji: Active antenna for contact sensing, IEEE Trans. Robotics Autom. **14**(2), 278–291 (1998)

28.37 T.N. Clements, C.D. Rahn: Three-dimensional contact imaging with an actuated whisker, IEEE Trans. Robotics **22**(4), 844–848 (2006)

28.38 T.J. Prescott, M.J. Pearson, B. Mitchinson, J.C. Sullivan, A. Pipe: Tactile discrimination using active whisker sensors, IEEE Sens. J. **12**(2), 350–362 (2012)

28.39 J.M. Vranish, R.L. McConnell, S. Mahalingam: Capaciflector collision avoidance sensors for robots, Comput. Electr. Eng. **17**(3), 173–179 (1991)

28.40 E. Cheung, V. Lumelsky: A sensitive skin system for motion control of robot arm manipulators, Robotics Auton. Syst. **10**(1), 9–32 (1992)

28.41 D. Um, V. Lumelsky: Fault tolerance via component redundancy for a modularized sensitive skin, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 722–727

28.42 S. Walker, K. Loewke, M. Fischer, C. Liu, J.K. Salisbury: An optical fiber proximity sensor for haptic exploration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 473–478

28.43 P. Wei, L. Zhizeng: A design of miniature strong anti-jamming proximity sensor, Proc. Int. Conf. Comp. Sci. Electron. Eng. (ICCSEE) (2012) pp. 327–331

28.44 E. Guglielmelli, V. Genovese, P. Dario, G. Morana: Avoiding obstacles by using a proximity US/IR sensitive skin, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1993) pp. 2207–2214

28.45 D. Wegerif, D. Rosinski: Sensor based whole arm obstacle avoidance for kinematically redundant robots, Proc. SPIE – Int. Soc. Opt. Eng. **1828**, 417–426 (1992)

28.46 G. Buttazzo, P. Dario, R. Bajcsy: Finger based explorations, Proc. SPIE 0726, Intell. Robots Comput. Vis. V, ed. by D.P. Casadent (1986) pp. 338–345

28.47 D. Siegel, I. Garabieta, J. Hollerbach: An integrated tactile and thermal sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 1286–1291

28.48 R.A. Russell: A thermal sensor array to provide tactile feedback for robots, Int. J. Robotics Res. **5**(3), 35–39 (1985)

28.49 F. Castelli: An integrated tactile-thermal robot sensor with capacitive tactile array, IEEE Trans. Ind. Appl. **38**(1), 85–90 (2002)

28.50 D.G. Caldwell, C. Gosney: Enhanced tactile feedback (Tele-taction) using a multi-functional sensory system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) **1**, 955–960 (1993)

28.51 G.J. Monkman, P.M. Taylor: Thermal tactile sensing, IEEE Trans. Robotics Autom. **9**(3), 313–318 (1993)

28.52 J. Engel, J. Chen, X. Wang, Z. Fan, C. Liu, D. Jones: Technology development of integrated multimodal and flexible tactile skin for robotics applications, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 3 (2003) pp. 2359–2364

28.53 P. Bergveld: Development and application of chemical sensors in liquids. In: *Sensors and Sensory Systems for Advanced Robots*, NATO ASI Series, Vol. 43, ed. by P. Dario (Springer, Berlin, Heidelberg 1988) pp. 397–414

28.54 T. Nakamoto, A. Fukuda, T. Moriizumi: Perfume and flavor identification by odor sensing system using quartz-resonator sensor array and neural-network pattern recognition, Proc. 6th Int. Conf. Solid-State Sens. Actuators (TRANSDUCERS '91) (1991)

28.55 R.A. Russell: Survey of robotic applications for odor-sensing technology, Int. J. Robotics Res. **20**(2), 144–162 (2001)

28.56 A.J. Lilienthal, A. Loutfi, T. Duckett: Airborne chemical sensing with mobile robots, Sensors **6**(11), 1616–1678 (2006)

28.57 B.A. Auld, A.J. Bahr: A novel multifunction robot sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 1791–1797

28.58 H. Clergeot, D. Placko, J.M. Detriche: Electrical proximity sensors. In: *Sensors and Sensory Systems for Advanced Robots*, NATO ASI Series, Vol. 43, ed. by P. Dario (Springer, Berlin, Heidelberg 1988) pp. 295–308

28.59 M. Kaneko, K. Tanie: Contact point detection for grasping of an unknown object using self-posture changeability (SPC), IEEE Trans. Robotics Autom., Vol. 10 (1994) pp. 355–367

28.60 A.M. Dollor, L.P. Jentoft, J.H. Cao, R.D. Howe: Contact sensing and grasping performance of compliant hands, Auton. Robots **28**(1), 65–75 (2010)

28.61 J.K. Salisbury: Appendix to kinematic and force analysis of articulated hands. In: *Robot Hands and the Mechanics of manipulation*, ed. by M.T. Mason, J.K. Salisbury (MIT Press, Cambridge 1985)

28.62 G. Palli, C. Melchiorri, G. Vassura, U. Scarcia, L. Moriello, G. Berselli, A. Cavallo, G. De Maria, C. Natale, S. Pirozzi, C. May, F. Ficuciello, B. Siciliano: The DEXMART hand: Mechatronic design and experimental evaluation of synergy-based control for human-like grasping, Int. J. Robotics Res. **33**(5), 799–824 (2014)

28.63 A. Pugh (Ed.): *Robot Sensors, Volume 2: Tactile and Non-Vision* (IFS Publ./Springer, New York 1986)

28.64 J.G. Webster: *Tactile Sensors for Robotics and Medicine* (Wiley, New York 1988)

28.65 J.K. Salisbury: Interpretation of contact geometries from force measurements. In: *Robotics Res. First Int. Symp*, ed. by M. Brady, R.P. Paul (MIT Press, Cambridge 1984)

28.66 D. Brock, S. Chiu: Environment perception of an articulated robot hand using contact sensors, ASME Winter Annu. Meet. Robotics Manuf. Automa., Vol. 15 (1985) pp. 89–96

28.67 J. Butterfass, M. Grebenstein, H. Liu, G. Hirzinger: DLR-Hand II: next generation of a dextrous robot hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (2001) pp. 109–114

28.68 F.W. Sinden, R.A. Boie: A planar capacitive force sensor with six degrees of freedom, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 1806–1813

28.69 B.B. Edin, L. Beccai, L. Ascari, S. Roccella, J.J. Cabibihan, M.C. Carrozza: A bio-inspired approach for the design and characterization of a tactile sensory system for a cybernetic prosthetic hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1354–1358

28.70 Y.-L. Park, S.C. Ryu, R.J. Black, K.K. Chau, B. Moslehi, M.R. Cutkosky: Exoskeletal force-sensing end-effectors with embedded optical fiber-bragg-grating sensors, IEEE Trans. Robotics **25**(6), 1319–1331 (2009)

28.71 A. Bicchi: A criterion for optimal design of multiaxis force sensors, Robotics Auton. Syst. **10**(4), 269–286 (1992)

28.72 M. Uchiyama, E. Bayo, E. Palma-Villalon: A mathematical approach to the optimal structural design of a robot force sensor, Proc. USA-Japan Symp. Flexible Automation (1998) pp. 539–546

28.73 A. Bicchi, J.K. Salisbury, P. Dario: Augmentation of grasp robustness using intrinsic tactile sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (1989) pp. 302–307

28.74 J.S. Son, M.R. Cutkosky, R.D. Howe: Comparison of contact sensor localization abilities during manipulation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 2 (1995) pp. 96–103

28.75 R.S. Johansson, J.R. Flanagan: Coding and use of tactile signals from the fingertips in object manipulation tasks, Nat. Rev. Neurosci. **10**(5), 345–359 (2009)

28.76 M. Ueda: Tactile sensors for an industrial robot to detect a slip, Proc. 2nd Int. Symp. Ind. Robots (1972) pp. 63–70

28.77 R. Matsuda: Slip sensor of industrial robot and its application, Electric. Eng. Jap. **96**(5), 129–136 (1976)

28.78 J. Rebman, J.-E. Kallhammer: A Search for Precursors of Slip in Robotic Grasp, Intelligent Robots

and Computer Vision: Fifth in a Series, Cambridge, ed. by E. Casaent (1986) pp. 329–337
28.79 P. Dario, D. De Rossi: Tactile sensors and the gripping challenge, IEEE Spectrum **22**(8), 46–52 (1985)
28.80 R.W. Patterson, G.E. Nevill: The induced vibration touch sensor – A new dynamic touch sensing concept, Robotica **4**(01), 27–31 (1986)
28.81 M.R. Cutkosky, J. Ulmen: Dynamic Tactile Sensing. In: *The Human Hand as an Inspiration for Robot Hand Development*, Springer Tracts in Advanced Robotics 95, ed. by R. Balasubramanian, V.J. Santos (Springer, Cham 2014) pp. 389–403
28.82 D. Dornfeld, C. Handy: Slip detection using acoustic emission signal analysis, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1987) pp. 1868–1875
28.83 X.A. Wu, N. Burkhard, B. Heyneman, R. Valen, M.R. Cutkosky: Contact event detection for robotic oil drilling, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014) pp. 2255–2261
28.84 S. Omata: Real time robotic tactile sensor system for the determination of the physical properties of biomaterials, Sens. Actuators A Phys. **112**(2/3), 278–285 (2004)
28.85 S.B. Backus, A.M. Dollar: Robust resonant frequency-based contact detection with applications in robotic reaching and grasping, IEEE/ASME Trans. Mechatron. **19**(5), 1552–1561 (2014)
28.86 M.R. Tremblay, M.R. Cutkosky: Estimating friction using incipient slip sensing during a manipulation task, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (1993) pp. 429–434
28.87 E.G.M. Holweg, H. Hoeve, W. Jongkind, L. Marconi, C. Melchiorri, C. Bonivento: Slip detection by tactile sensors: Algorithms and experimental results, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (1996) pp. 3234–3239
28.88 I. Fujimoto, Y. Yamada, T. Maeno, T. Morizono, Y. Umetani: Identification of incipient slip phenomena based on the circuit output signals of PVDF film strips embedded in artificial finger ridges, Trans. Soc. Instrum. Control Eng. **40**(6), 648–655 (2004)
28.89 B. Choi, H.R. Choi, S. Kang: Development of tactile sensor for detecting contact force and slip, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 2638–2643
28.90 P.A. Schmidt, E. Maël, R.P. Würtz: A sensor for dynamic tactile information with applications in human–robot interaction and object exploration, Robotics Auton. Syst. **54**(12), 1005–1014 (2006)
28.91 R.D. Howe: Tactile sensing and control of robotic manipulation, Adv. Robotics **8**(3), 245–261 (1993)
28.92 C. Melchiorri: Slip detection and control using tactile and force sensors, IEEE/ASME Trans. Mechatron. **5**(3), 235–243 (2000)
28.93 C.M. Oddo, L. Beccai, G.G. Muscolo, M.C. Carrozza: A biomimetic MEMS-based tactile sensor array with fingerprints integrated in a robotic fingertip for artificial roughness encoding, Proc. IEEE Int. Conf. Robotics Biomim. (2009) pp. 894–900
28.94 A. Schmitz, M. Maggiali, L. Natale, B. Bonino, G. Metta: A tactile sensor for the fingertips of the humanoid robot iCub, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2010) pp. 2212–2217
28.95 L.P. Jentoft, Y. Tenzer, D. Vogt, R.J. Wood, R.D. Howe: Flexible, stretchable tactile arrays from MEMS barometers, Proc. 16th Int. Conf. Adv. Robotics (2013) pp. 1–6
28.96 R.S. Fearing, J.M. Hollerbach: Basic solid mechanics for tactile sensing, Int. J. Robotics Res. **4**(3), 40–54 (1985)
28.97 W. Griffin, W.M. Provancher, M.R. Cutkosky: Feedback strategies for telemanipulation with shared control of object handling forces, Presence Teleoperations Virtual Environ. **14**(6), 720–731 (2005)
28.98 H. Maekawa, K. Tanie, K. Komoriya, M. Kaneko, C. Horiguchi, T. Sugawara: Development of a finger-shaped tactile sensor and its evaluation by active touch, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (1992) pp. 1327–1334
28.99 R.S. Fearing: Tactile sensing mechanisms, Int. J. Robotics Res. **9**(3), 3–23 (1987)
28.100 G. Cannata, M. Maggiali, G. Metta, G. Sandini: An embedded artificial skin for humanoid robots, Proc. IEEE Int. Conf. Muiltisens. Fusion Integr. Intell. Syst. (2008) pp. 434–438
28.101 M.-Y. Cheng, X.-H. Huang, C.-W. Ma, Y.-J. Yang: A flexible capacitive tactile sensing array with floating electrodes, J. Micromechanics Microengineering **19**(11), 115001 (2009)
28.102 Y. Hasegawa, M. Shikida, D. Ogura, Y. Suzuki, K. Sato: Fabrication of a wearable fabric tactile sensor produced by artificial hollow fiber, J. Micromechanics Microengineering **18**(8), 085014 (2008)
28.103 D. McConnell Aukes, M.R. Cutkosky, S. Kim, J. Ulmen, P. Garcia, H. Stuart, A. Edsinger: Design and testing of a selectively compliant underactuated hand, Int. J. Robotics Res. **33**(5), 721–735 (2014)
28.104 O. Kerpa, K. Weiss, H. Worn: Development of a flexible tactile sensor system for a humanoid robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1–6
28.105 D. Bloor, A. Graham, E.J. Williams, P.J. Laughlin, D. Lussey: Metal–polymer composite with nanostructured filler particles and amplified physical properties, Appl. Phys. Lett. **88**(10), 102103 (2006)
28.106 Peratech: Peratech QTC, http://www.peratech.com/standard-products/ (2014)
28.107 T. Someya: Integration of organic field-effect transistors and rubbery pressure sensors for artificial skin applications, Proc. IEEE Int. Electron. Dev. Meet. (2003) pp. 8–14
28.108 H. Alirezaei, A. Nagakubo, Y. Kuniyoshi: A tactile distribution sensor which enables stable measurement under high and dynamic stretch, Proc. IEEE Symp. 3D User Interfaces (2009) pp. 87–93
28.109 Y.-L. Park, B.-R. Chen, R.J. Wood: Design and fabrication of soft artificial skin using embedded microchannels and liquid conductors, IEEE Sens. J. **12**(8), 2711–2718 (2012)
28.110 R. Kageyama, S. Kagami, M. Inaba, H. Inoue: Development of soft and distributed tactile sensors

and the application to a humanoid robot, Proc. IEEE Int. Conf. Syst. Man Cybern., Vol. 2 (1999) pp. 981–986
28.111 B.J. Kane, M.R. Cutkosky, G.T.A. Kovacs: A traction stress sensor array for use in high-resolution robotic tactile imaging, J. Microelectromechanical Syst. **9**(4), 425–434 (2000)
28.112 H. Takao, K. Sawada, M. Ishida: Monolithic silicon smart tactile image sensor with integrated strain sensor array on pneumatically swollen single-diaphragm structure, IEEE Trans. Electron. Dev. **53**(5), 1250–1259 (2006)
28.113 K. Noda, I. Shimoyama: A Shear stress sensing for robot hands -Orthogonal arrayed piezoresistive cantilevers standing in elastic material-, Proc. 14th Symp. Haptic Interfaces Virtual Env. Teleoperator Syst. (2006) pp. 63–66
28.114 M.-Y. Cheng, C.-L. Lin, Y.-J. Yang: Tactile and shear stress sensing array using capacitive mechanisms with floating electrodes, 2010 IEEE 23rd Int. Conf. Micro Electro Mech. Syst. (2010) pp. 228–231
28.115 P. Valdastri, S. Roccella, L. Beccai, E. Cattin, A. Menciassi, M.C. Carrozza, P. Dario: Characterization of a novel hybrid silicon three-axial force sensor, Sens. Actuators A **123/124**, 249–257 (2005)
28.116 S.C.B. Mannsfeld, B.C.-K. Tee, R.M. Stoltenberg, C.V.H.H. Chen, S. Barman, B.V.O. Muir, A.N. Sokolov, C. Reese, Z. Bao: Highly sensitive flexible pressure sensors with microstructured rubber dielectric layers, Nat. Mater. **9**(10), 859–864 (2010)
28.117 R. Brockett: Robotic hands with rheological surfaces, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1985) pp. 942–946
28.118 K.B. Shimoga, A.A. Goldenberg: Soft robotic fingertips. I. A comparison of construction materials, Int. J. Rob, Res. **15**(4), 320–350 (1996)
28.119 A. Mazid, R. Russell: A robotic opto-tactile sensor for assessing object surface texture, IEEE Conf. Robotics Autom. Mechatronics (2006) pp. 1–5
28.120 L.S. Lincoln, S.J.M. Bamberg, E. Parsons, C. Salisbury, J. Wheeler: An elastomeric insole for 3-axis ground reaction force measurement, Proc. IEEE RAS/EMBS Int. Conf. Biomedical Robotics Biomech. (2012) pp. 1512–1517
28.121 H. Shinoda, K. Matsumoto, S. Ando: Acoustic resonant tensor cell for tactile sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (1997) pp. 3087–3092
28.122 H. Shinoda, S. Sasaki, K. Nakamura: Instantaneous evaluation of friction based on ARTC tactile sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (2000) pp. 2173–2178
28.123 S. Ando, H. Shinoda, A. Yonenaga, J. Terao: Ultrasonic six-axis deformation sensing, IEEE Trans. Ultrason. Ferroelectr. Freq. Control. **48**(4), 1031–1045 (2001)
28.124 P. Dario, D. De Rossi, C. Domenici, R. Francesconi: Ferroelectric polymer tactile sensors with anthropomorphic features, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1984) pp. 332–340
28.125 D.M. Siegel: *Contact sensors for dextrous robotic hands*, MIT Artificial Intelligence Laboratory Tech. Rep., no. 900 (MIT Press, Cambridge 1986)
28.126 J.S. Son, E.A. Monteverde, R.D. Howe: A tactile sensor for localizing transient events in manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (1994) pp. 471–476
28.127 B.S. Eberman, J.K. Salisbury: Determination of Manipulator Contact Information from Joint Torque Measurements. In: *Experimental Robotics I, The First International Symposium*, ed. by V. Hayward, O. Khatib (Springer, Montreal 1990)
28.128 A. Bicchi: Intrinsic contact sensing for soft fingers, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1990) pp. 968–973
28.129 P. Dario: Tactile sensing for robots: Present and future. In: *The Robotics Review 1*, ed. by O. Khatib, J. Craig, T. Lozano-Perez (MIT Press, Cambridge 1989) pp. 133–146
28.130 J.R. Phillips, K.O. Johnson: Tactile spatial resolution III: A continuum mechanics model of skin predicting mechanoreceptor responses to bars, edges and gratings, J. Neurophysiol. **46**(6), 1204–1225 (1981)
28.131 T. Speeter: A tactile sensing system for robotic manipulation, Int. J. Robotics Res. **9**(6), 25–36 (1990)
28.132 K.L. Johnson: *Contact Mechanics* (Cambridge Univ. Press, Cambridge 1985)
28.133 S. Timoshenko, J.N.N. Goodier: *Theory of Elasticity* (McGraw-Hill, New York 1951)
28.134 G. Kenaly, M. Cutkosky: Electrorheological fluid-based fingers with tactile sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 132–136
28.135 R.D. Howe: Dynamic Tactile Sensing, Ph.D. Thesis (Stanford University, Stanford 1990)
28.136 R.M. Voyles, B.L. Stavnheim, B. Yap: Practical electrorheological fluid-based fingers for robotic applications, IASTED Int. Symp. Robotics Manuf. (1989)
28.137 J.J. Clark: A magnetic field based compliance matching sensor for high resolution, high compliance tactile sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 772–777
28.138 T.H. Speeter: Analysis and Control of Robotic Manipulation, Ph.D. Thesis (Case Western Reserve University, Cleveland 1987)
28.139 R. Fearing: Tactile sensing for shape interpretation. In: *Dextrous Robot Hands*, ed. by S.T. Venkataraman, T. Iberall (Springer, Berlin, Heidelberg 1990) pp. 209–238
28.140 A.J. Worth, R.R. Spencer: A neural network for tactile sensing: The hertzian contact problem, Proc. Int. Jt. Conf. Neural Netw. (1989) pp. 267–274
28.141 W.E.L. Grimson, T. Lozano-Perez: Model-based recognition and localization from sparse range or tactile data, Int. J. Robotics Res. **3**(3), 3–35 (1984)
28.142 P.C. Gaston, T. Lozano-Perez: Tactile recognition and localization using object models: The case of polyhedra on a plane, Proc. IEEE Trans. Pattern Anal. Mach. Intell. (1984) pp. 257–266

28.143 J.L. Schneiter: An objective sensing strategy for object recognition and localization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 1262–1267

28.144 R. Cole, C. Yap: Shape from probing, J. Algorithm. **8**(1), 19–38 (1987)

28.145 P.K. Allen: Mapping haptic exploratory procedures to multiple shape representations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1990) pp. 1679–1684

28.146 R.E. Ellis: Extraction of tactile features by passive and active sensing, Proc. SPIE 0521 (1985) p. 289

28.147 S.J. Lederman, R. Browse: The physiology and psychophysics of touch. In: *Sensors and Sensory Systems for Advanced Robotics*, ed. by P. Dario (Springer, Berlin, Heidelberg 1986) pp. 71–91

28.148 H. Ozaki, S. Waku, A. Mohri, M. Takata: Pattern recognition of a grasped object by unit-vector distribution, IEEE Trans. Syst. Man Cybern. **12**(3), 315–324 (1982)

28.149 R.L. Klatzky, R. Bajcsy, S.J. Lederman: Object exploration in one and two fingered robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 1806–1809

28.150 D. Siegel: Finding the pose of an object in the hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1991) pp. 406–411

28.151 D. Taddeucci, C. Laschi, R. Lazzarini, R. Magni, P. Dario, A. Starita: An approach to integrated tactile perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (1997) pp. 3100–3105

28.152 A. Schneider, J. Sturm, C. Stachniss, M. Reisert, H. Burkhardt, W. Burgard: Object identification with tactile sensors using bag-of-features, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 243–248

28.153 N. Gorges, S.E. Navarro, D. Göger, H. Wörn: Haptic object recognition using passive joints and haptic key features, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 2349–2355

28.154 Y. Bekiroglu, J. Laaksonen, J.A. Jorgensen, V. Kyrki, D. Kragic: Assessing grasp stability based on learning and haptic data, IEEE Trans. Robotics **27**(3), 616–629 (2011)

28.155 V.S. Gurfinkel: Tactile sensitizing of manipulators, Eng. Cybern. **12**(6), 47–56 (1974)

28.156 R. Ellis: Acquiring tactile data for the recognition of planar objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (1987) pp. 1799–1805

28.157 A. Cameron: Optimal tactile sensor placement, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 308–313

28.158 P. Dario: Sensing body structures by an advanced robot system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1988) pp. 1758–1763

28.159 S.A.A. Stansfield: Robotic grasping of unknown objects: A knowledge-based approach, Int. J. Robotics Res. **10**(4), 314–326 (1991)

28.160 A. Petrovskaya, O. Khatib: Global localization of objects via touch, IEEE Trans. Robotics **27**(3), 569–585 (2011)

28.161 N.F. Lepora, U. Martinez-Hernandez, H. Barron-Gonzalez, M. Evans, G. Metta, T.J. Prescott: Embodied hyperacuity from Bayesian perception: Shape and position discrimination with an iCub fingertip sensor, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 4638–4643

28.162 C. Muthukrishnan, D. Smith, D. Meyers, J. Rebman, A. Koivo: Edge detection in tactile images, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 1500–1505

28.163 A.D. Berger, P.K. Khosla: Using tactile data for real-time feedback, Int. J. Robotics Res. **10**(2), 88–102 (1991)

28.164 K. Pribadi, J.S. Bay, H. Hemami: Exploration and dynamic shape estimation by a robotic probe, IEEE Trans. Syst. Man Cybern. **19**(4), 840–846 (1989)

28.165 H. Zhang, N.N. Chen: Control of contact via tactile sensing, IEEE Trans. Robotics Autom. **16**(5), 482–495 (2000)

28.166 A.M. Okamura, M.R. Cutkosky: Feature detection for haptic exploration with robotic fingers, Int. J. Robotics Res. **20**(12), 925–938 (2001)

28.167 K. Suwanratchatamanee, M. Matsumoto, S. Hashimoto: Robotic tactile sensor system and applications, IEEE Trans. Ind. Electron. **57**(3), 1074–1087 (2010)

28.168 K. Yamada, K. Goto, Y. Nakajima, N. Koshida, H. Shinoda: A sensor skin using wire-free tactile sensing elements based on optical connection, Proc. 41st SICE Annu. Conf., Vol. 1 (2002) pp. 131–134

28.169 M. Schoepfer, C. Schuermann, M. Pardowitz, H. Ritter: Using a piezo-resistive tactile sensor for detection of incipient slippage, Proc. ISR/ROBOTIK 41st Int. Symp. Robotics (2010) pp. 14–20

28.170 B. Heyneman, M.R. Cutkosky: Slip interface classification through tactile signal coherence, IEEE/RSJ IEEE Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 801–808

28.171 P. Dario, P. Ferrante, G. Giacalone, L. Livaldi, B. Allotta, G. Buttazzo, A.M. Sabatini: Planning and executing tactile exploratory procedures, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 3 (1992) pp. 1896–1903

28.172 S.C. Jacobsen, J.E. Wood, D.F. Knutti, K.B. Biggers: The Utah/MIT dextrous hand: Work in progress. In: *First International Conference on Robotics Research*, ed. by M. Brady, R.P. Paul (MIT Press, Cambridge 1984) pp. 601–653

28.173 H. Shinoda, H. Oasa: Passive wireless sensing element for sensitive skin, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 2 (2000) pp. 1516–1521

28.174 M. Hakozaki, H. Shinoda: Digital tactile sensing elements communicating through conductive skin layers, Proc. IEEE Int. Conf. Robotics Autom. (ICRA'02), Vol. 4 (2002) pp. 3813–3817

28.175 L. Ascari, P. Corradi, L. Beccai, C. Laschi: A miniaturized and flexible optoelectronic sensing system for a tactile skin, Int. J. Micromechanics Microengineering **17**, 2288–2298 (2007)

28.176 M. Zillich, W. Feiten: A versatile tactile sensor system for covering large and curved surface areas, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 20–24

Multimedia Contents

29. Inertial Sensing, GPS and Odometry

Gregory Dudek, Michael Jenkin

This chapter examines how certain properties of the world can be exploited in order for a robot or other device to develop a model of its own motion or pose (position and orientation) relative to an external frame of reference. Although this is a critical problem for many autonomous robotic systems, the problem of establishing and maintaining an orientation or position estimate of a mobile agent has a long history in terrestrial navigation.

29.1 **Odometry** 737
29.2 **Gyroscopic Systems** 739
29.2.1 Mechanical Systems 739
29.2.2 Optical Systems 740
29.2.3 MEMS 741
29.3 **Accelerometers** 742
29.3.1 Mechanical Accelerometer 742
29.3.2 Piezoelectric Accelerometer 743
29.4 **IMU Packages** 743
29.4.1 Performance 744
29.5 **Satellite-Based Positioning (GPS and GNSS)** 744
29.5.1 Overview 745
29.5.2 Performance Factors 746
29.5.3 Enhanced GPS 748
29.5.4 Nationwide Differential GPS System (NDGPS) 748
29.5.5 GPS Receivers and Communications 749
29.6 **GPS-IMU Integration** 749
29.7 **Further Reading** 750
29.7.1 Odometry 750
29.7.2 Gyroscopic Systems 750
29.7.3 Accelerometers 750
29.7.4 GPS 750
29.8 **Currently Available Hardware** 750
29.8.1 Gyroscopic Systems 750
29.8.2 Accelerometers 750
29.8.3 IMU Packages 751
29.8.4 GPS Components 751
References 751

29.1 Odometry

The word *odometry* is a contraction of the Greek words *hodos* meaning *travel* or *journey*, and *metron* meaning measure. Given its importance to a wide variety of applications from civil engineering to military conquest, the basic concepts that underly odometry have been studied for over 2000 years. Perhaps the earliest reference to odometry appears in his *Ten Books on Architecture* where *Vitruvius* describes [29.1]

> *a useful invention of the greatest ingenuity, transmitted by our predecessors, which enables us, while sitting in a carriage on the road or sailing by sea, to know how many miles of a journey we have accomplished.*

In the context of autonomous vehicles, odometry usually refers to the use of data from the actuators (wheels, treads, etc.) to estimate the overall motion of the vehicle. The basic concept [29.2] is to develop a mathematical model of how commanded motions of the vehicles wheels/joints/etc. induce motion of the vehicle itself, and then to integrate these commanded motions over time in order to develop a model of the pose of the vehicle as a function of time. The use of odometry infor-

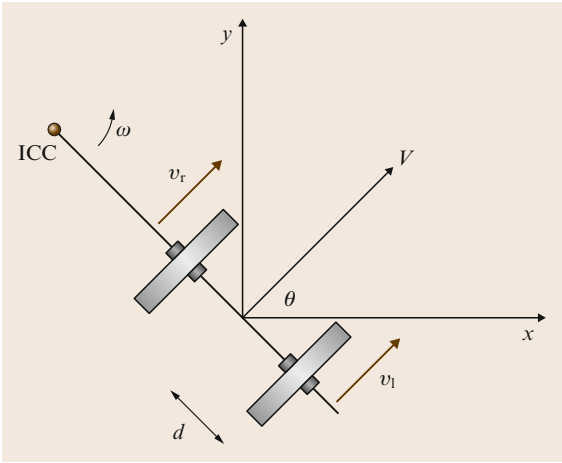


Fig. 29.1 Differential drive kinematics

mation to estimate the pose of the vehicle as a function of time is known as *Dead Reckoning* or *Deductive Reckoning* and finds wide application in navigation at sea [29.3].

The details of odometry estimation varies with vehicle design. In the context of mobile robots perhaps the simplest vehicle for odometry estimation is the differential drive vehicle (Fig. 29.1). A differential drive vehicle has two driveable wheels which are independently controllable and which are mounted along a common axis. Assuming that the location of the wheels are fixed on the vehicle, then for the wheels to remain in constant contact with the ground, the two wheels must describe arcs on the plane such that the vehicle rotates around a point (known as the ICC – instantaneous center of curvature) that lies on the wheels' common axis (Fig. 29.1). If the ground contact speeds of the left and right wheels are v_l and v_r respectively, and the wheels are separated by a distance $2d$, then

$$\omega(R+d) = v_r \,,$$
$$\omega(R-d) = v_l \,.$$

We can rearrange these two equations to solve for ω the rate of rotation about the ICC and R the distance from the center of the robot to the ICC

$$\omega = (v_r - v_l)/2d$$
$$R = d(v_r + v_l)/(v_r - v_l) \,.$$

The instantaneous velocity of the point midway between the robot's wheels is given by $V = \omega R$.

Now as v_l and v_r are functions of time we can generate the equation of motion for the differential drive robot. Using the point midway between the wheels as the origin of the robot, and writing θ as the orientation of the robot with respect to the x-axis of a global Cartesian coordinate system obtains

$$x(t) = \int V(t)\cos\left[\theta(t)\right] \mathrm{d}t \,,$$
$$y(t) = \int V(t)\sin\left[\theta(t)\right] \mathrm{d}t \,,$$
$$\theta(t) = \int \omega(t)\,\mathrm{d}t \,.$$

This is the solution for the odometry of a differential drive vehicle on the plane. Given control inputs (v_l and v_r) and some initial state estimate, we can estimate where an idealized robot using this motion model will be at any time t.

Given such a model and complete knowledge of the control inputs, we should, in principle, be able to estimate a robot's pose at any time. In a perfect world this would be all that is necessary to estimate accurately the robot's pose at any time in the future. Unfortunately errors in the modelling (incorrect estimations of wheel size, vehicle size), uncertainty about the control inputs, realities of the motor controller (errors between commanded wheel rotation and true rotation), errors in the physical modelling of the robot (wheel compaction, ground compaction, wheel slippage, non-zero tire width), etc., introduces an error between the dead reckoning estimate of the vehicle motion and its true motion. The problem of correcting for this error is the problem of *pose maintenance* for the vehicle, and requires the integration of the dead reckoning estimate with estimates obtained from other sensor systems.

Other chapters in this handbook examine sensors that rely on external events, visual and otherwise, that can provide information as to the robot's pose or changes in it's pose. Here we consider sensors that transduce physical properties of matter under the influence of external forces and properties of matter and the use of global positioning systems (GPS).

29.2 Gyroscopic Systems

The goal of gyroscopic systems is to measure changes in vehicle orientation by taking advantage of physical laws that produce predictable effects under rotation. A rotating frame is not an inertial frame, and thus many physical systems will appear to behave in an apparently *non-Newtonian* manner if the frame is rotating. By measuring these deviations from what would be expected in a Newtonian frame the underlying self-rotation can be determined.

29.2.1 Mechanical Systems

Mechanical gyroscopes and gyrocompasses have a long history in navigation, *Bohnenberger* is generally credited with the first recorded construction of a gyroscope [29.4]. The gyrocompass was patented in 1885 by Martinus Gerardus ven den Bos. In 1903 Herman Anschütz-Kaempfe constructed a working gyrocompass and obtained a patent on the design. In 1908 Elmer Sperry patented a gyrocompass in the US and then attempted to sell this device to the German Navy. A patent battle followed and Albert Einstein testified in the case. ([29.5–8] for more details on the history of the gyrocompass and its inventors.)

Gyroscopes and gyrocompasses rely on the principle of the conservation of angular momentum [29.9]. Angular momentum is the tendency of a rotating object to keep rotating at the same angular speed about the same axis of rotation in the absence of an external torque. The angular momentum $\boldsymbol{L}$ of an object with moment of inertia $\boldsymbol{I}$ rotating at angular speed ω is given by

$$\boldsymbol{L} = \boldsymbol{I} \times \boldsymbol{\omega} .$$

Consider a rapidly spinning wheel mounted on a shaft so that it is free to change its axis of rotation (Fig. 29.2a). Assuming no friction on the bearings or air resistance, the rotor axis will remain constant regardless of the motion of the external cage. This constancy of orientation can be exploited to maintain a bearing independently of the motion of the vehicle, although it is not usually desirable to utilize the principle of conservation of angular momentum via a gyroscope directly. To see this, suppose that a gyroscope is set on the equator, with its spinning axis aligned pointing along the equator (Fig. 29.2b). As the earth spins, the gyroscope will maintain a constant axis of orientation and thus to an earth-fixed observer will appear to rotate returning to its original orientation every 24 hours. Similarly, if the gyroscope were to be positioned on the equator such that its spinning axis was parallel to the axis of rotation of the earth, the gyroscope's axis of rotation would remain stationary and would appear to remain stationary to an earth-fixed observer as the planet rotates.

Although this global motion limits the mechanical gyroscope's ability to sense absolute orientation directly, gyroscopes can be used to measure local changes in orientation, and thus are well suited to vehicular robotic applications. Rate gyros (RGs) measure a vehicle's rotation rate (its *angular rate* of rotation). This is the fundamental measurement that is the basis of all gyroscopic systems. Rate-integrating gyros (RIGs) use embedded processing to internally integrate the angular rotation rate to produce an estimate of the absolute angular displacement of the vehicle.

In order to exploit a gyroscope for navigation with respect to an earth-stable frame, it is desirable for the rotational axis of the shaft to remain fixed within the earth frame, rather than remaining fixed with respect to an external frame. A gyrocompass obtains this by relying on precession. When a torque is applied to change the axis of rotation of a spinning object, conservation of angular momentum will cause a change in the spin direction that is perpendicular to the angular momentum and the direction in which the torque is applied. This is the effect that causes gyroscopes suspended at one end to *spin* around the end from which they have been suspended. Consider the *pendulus gyro* sketched in Fig. 29.3a. This is a standard gyroscope with a weight suspended below the rotational axis. As before, imagine this pendulus gyrocompass set spinning on the equator with the axis of rotation aligned with the axis of rotation of the planet, and with the weight hanging directly down. As the planet spins, the gyroscope's axis of rotation remains stationary and would appear to remain

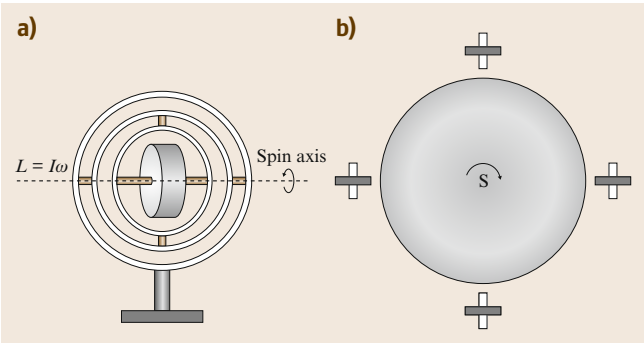


Fig.29.2a,b Mechanical gyroscope. **(a)** Traditionally gimballed gyroscope. The gimball provides the gyroscope the freedom to rotate about its axis as the base of the gyroscope is rotated **(b)** gyroscope as it is rotated around the planet. The wheel of the gyroscope (*grey*) remains in the same orientation as it revolves with the planet. To an observer on the planet the gyroscope will appear to rotate

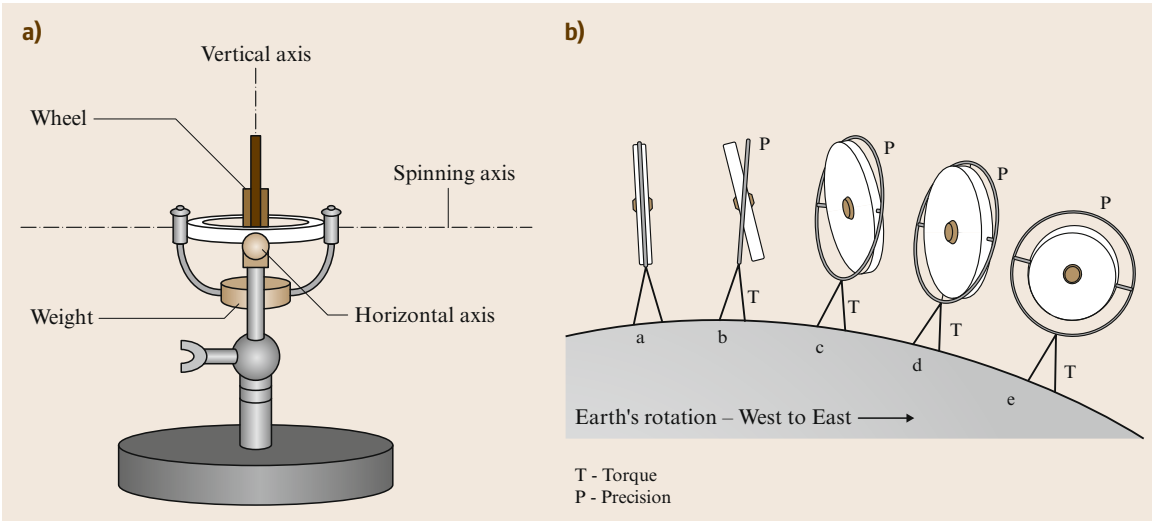


Fig.29.3a,b Simple gyrocompass: **(a)** Pendulus gyro; **(b)** Precessional motion

stationary as the planet rotates. Now suppose that instead of being aligned with the spin axis of the planet, the spin axis is aligned with the equator. As the planet spins, the spin axis is drawn out of the plane as it remains aligned with the original spin axis. As it becomes drawn out of the plane, the mass hanging below the gyrocompass is raised up and generates a torque down due to gravity. The direction perpendicular to the spin axis and the torque rotates the spin axis away from the equator (aka towards the true pole). This process is sketched in Fig. 29.3b.

Unfortunately the pendulus gyro is not an ideal device for navigation. Although its rotation axis will align with the rotation axis of the planet, it does not converge to this value directly but oscillates about it. The solution to this damping problem is to use oil reservoirs, rather than a solid weight, as the counterbalance, and to restrict the motion of the oil in the reservoir [29.10].

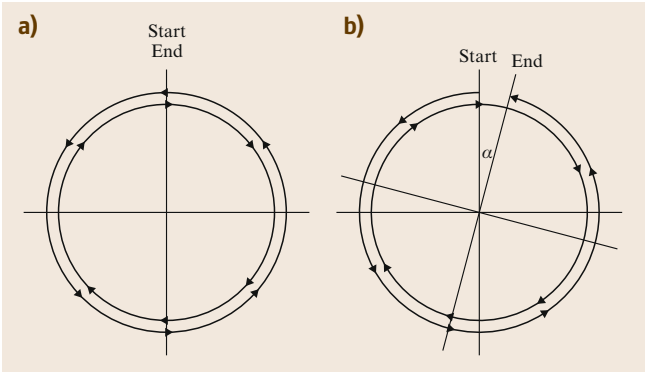


Fig.29.4a,b Circular light path. **(a)** Stationary path; **(b)** Moving path

The gyrocompass finds true north by controlling the precession of a gyroscope. In practice, the performance of a mechanical gyrocompass is impacted by external forces acting on the compass which also contribute to the precession of the gyroscope. This includes forces generated by motion of the device containing the gyrocompass, as well as any external forces acting on the vehicle. Another issue for mechanical gyrocompasses is that in latitudes away from the equator, the stable position of the gyrocompass is not horizontal, and accurate estimates of true north in such latitudes requires corrections to be applied to raw gyrocompass values. Finally, mechanical gyrocompasses require an external force to be applied to maintain the spin of the gyroscope. This process introduces unwanted forces into the system which can further corrupt the measurement process.

Given the complexity, cost, size, the delicate nature of gyrocompasses, and the availability of less expensive and more reliable technologies, mechanical gyrocompasses have given way to optical- and MEMS-based systems (MEMS or *microelectromechanical systems* are also known as *micromachines* or *micro systems technology* and refer to mechanisms at the scale of integrated circuits).

29.2.2 Optical Systems

Optical gyroscopes rely on the *Sagnac effect* rather than rotational inertia in order to measure (relative) heading. The mechanism is based on the behaviour of an optical standing wave in a rotating frame. Historically this was first produced using lasers and an arrangement of mirrors, but it is now typically obtained using fibre-

optic technology. The Sagnac Effect is named after its discoverer *Georges Sagnac* [29.11, 12]. The underlying concept can be traced back even earlier to the work of *Harress* [29.13], and perhaps finds its most famous application in terms of the measurement of the rotation of the earth [29.14].

Ignore relativistic effects and consider the circular light path shown in Fig. 29.4a. If two light pulses are sent in opposite directions around a stationary path of perimeter $D = 2\pi R$ they will travel the same distance at the same speed. They will arrive at the starting point simultaneously, taking time $t = D/c$ where c is the speed of light in the medium. Now let us suppose that instead of being stationary, this circular light path rotates clockwise about its center at rotational speed ω (Fig. 29.4b). The light travelling clockwise around the path must go *farther* to reach the starting point, while light travelling counterclockwise around the path goes a *shorter* distance. The clockwise path has distance

$$D_c = 2\pi R + \omega R t_c \,,$$

where t_c is the time taken in the clockwise direction, while the counterclockwise path has distance

$$D_a = 2\pi R - \omega R t_a \,,$$

where t_a is the time taken in the counterclockwise direction. But $D_c = ct_c$ and $D_a = ct_a$, so

$$t_c = 2\pi \frac{R}{c - \omega R}$$

and

$$t_a = 2\pi R(c + \omega R) \,.$$

The time difference $\Delta t = t_c - t_a$ is given by

$$\Delta t = 2\pi R \left(\frac{1}{c - \omega R} - \frac{1}{c + \omega R} \right) .$$

By measuring Δt, the rotational speed can be computed. Note that although the above derivation assumes classical mechanics and ignores relativistic effects, the derivation also applies when relativistic speeds are taken into account [29.15]. See [29.16] for an in depth review of the Sagnac effect and ring-lasers.

In optical gyroscopes lasers are typically used as the light source. Optical gyroscopes either utilize straight line light paths with mirror surfaces or prisms at the edges to direct the light beam (a ring laser gyroscope – RLG), or a polarization maintaining glass-fiber loop (fiber optic gyro – FOG). The glass-fiber may actually loop multiple times, thus extending the effective length of the light path. The time delay between the clockwise and counter-clockwise is detected by examining the phase interference between the clockwise and counter-clockwise light signals. Multiple optical gyroscopes with non-parallel axes can be ganged together in order to measure 3-D rotations.

Various techniques can be used to measure the time difference including examining the Doppler (frequency) shift of the laser light due to the motion of the gyro and an examination of the beat frequency of the interference pattern between the clockwise and counter-clockwise paths [29.17]. Ring interferometers typically consist of many windings of fiber optic lines that conduct light of a fixed frequency in opposite directions around the loop and measure the phase difference. A ring laser consists of a laser cavity in the shape of a ring. Light circulates in both directions around this cavity producing two standing waves with the same number of nodes in both directions. Since the optical path lengths are different in the two directions, the resonate frequencies differ. The difference between these two frequencies is measured. An unfortunate side effect of the ring laser approach is that the two signals will lock to each other for small rotations and it is typically necessary to physically rotate the device in a controlled manner in order to ensure that this lock-in effect can be avoided.

29.2.3 MEMS

Almost all MEMS gyroscopes are based on vibrating mechanical elements to sense rotation. Vibratory gyroscopes rely on the transfer of energy between vibratory modes based on Coriolis acceleration. Coriolis acceleration is the apparent acceleration that arises in a rotating frame of references. Suppose an object moves along a straight line in a rotating frame of reference. To an outside observer in an inertial frame the object's path is curved – thus there must be some force acting on the object to maintain the straight line motion as viewed by the rotating observer. An object moving in a straight line with local velocity $\boldsymbol{v}$ in a frame rotating at rate $\boldsymbol{\Omega}$ relative to an inertial frame will experience a Coriolis acceleration given by

$$\boldsymbol{a} = 2\boldsymbol{v} \times \boldsymbol{\Omega} \,.$$

Transducing acceleration in a MEMS gyroscope amounts to inducing some local linear velocity and measuring the resultant Coriolis forces.

Early MEMS gyroscopes utilized vibrating quartz crystals to generate the necessary linear motion. More

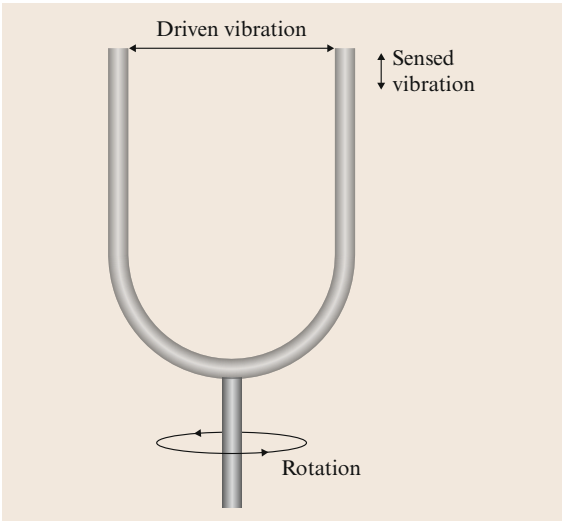


Fig. 29.5 MEMS gyroscope principle of operation

recent designs have replaced the vibrating quartz crystals with silicon based vibrators. Various MEMS structures have been developed including

Tuning Fork Gyroscopes

Tuning fork gyroscopes use a tuning fork-like structure (Fig. 29.5) as the underlying mechanism. As the tuning forks vibrate within a rotating frame, Coriolis forces cause the tines of the fork to vibrate out of the plane of the fork which is measured. This is the effect used by the InertiaCube sensor [29.18].

Vibrating Wheel Gyroscopes

Vibrating wheel gyroscopes use a wheel that oscillates about its rotational axis. External rotation of the frame causes the wheel to tilt which can be sensed.

Wine Glass Resonator Gyroscopes

Wine glass resonator gyroscopes use the effect of Coriolis forces on the position of nodal points on a resonating structure to estimate the external rotation. Sadly, no actual wine glasses are involved.

As MEMS gyroscopes have no rotating parts, have low power consumption requirements, and are very small, MEMS gyros are quickly replacing mechanical and optical gyroscope sensors in robotic applications.

With the exception of gyrocompasses, gyroscopes measure relative rotational motion around a single axis. They accomplish this measurement by exploiting physical properties of rotating frames of reference. Earlier technologies based on mechanical gyroscopes have given way to optical- and MEMS-based devices but the underlying principle remains unchanged – that rotating frames of reference show specific physical properties can be measured to estimate the relative rotation.

A problem common to all gyroscopes is that of drift. Each of the relative motion measurements is corrupted by an error process, and these errors accumulate over time. This, coupled with specific measurement errors associated with the individual gyroscope technologies means that unless the error is corrected through reference to some alternate (external) measurement, the drift will eventually exceed the required accuracy of the measurement.

As individual gyros only measure rotation about a single axis, it is common to gang multiple gyros together with orthogonal axes of sensitivity in order to measure 3-D rotations. These collections of gyros are often integrated with other sensors (compasses, accelerometers, etc.) in order to construct *inertial measurement units* (or IMUs). This is considered in Sect. 29.4.

29.3 Accelerometers

Just as gyroscopes can be used to measure changes in orientation of a robot, other inertial sensors – known as *accelerometers* – can be used to measure external forces acting on the vehicle. One important factor concerning accelerometers is that they are sensitive to all external forces acting upon them – including gravity. Accelerometers use one of a number of different mechanisms to transduce external forces into a computer readable signal.

29.3.1 Mechanical Accelerometer

A mechanical accelerometer (Fig. 29.6a) is essentially a spring-mass-damper system with some mechanism for external monitoring. When some force is applied (e.g., gravity), the force acts on the mass and displaces the spring. Assuming an ideal spring with a force proportional to its displacement, the external forces balance the internal ones

$$\begin{aligned} F_{\text{applied}} &= F_{\text{intertial}} + F_{\text{damping}} + F_{\text{spring}} \,, \\ &= m\ddot{x} + c\dot{x} + kx \,, \end{aligned}$$

where c is the damping coefficient. This equation can be solved to show that depending on the size of the damping coefficient relative to the expected external force and the mass, the system can be made to reach a stable final value in a reasonably short period of time when-

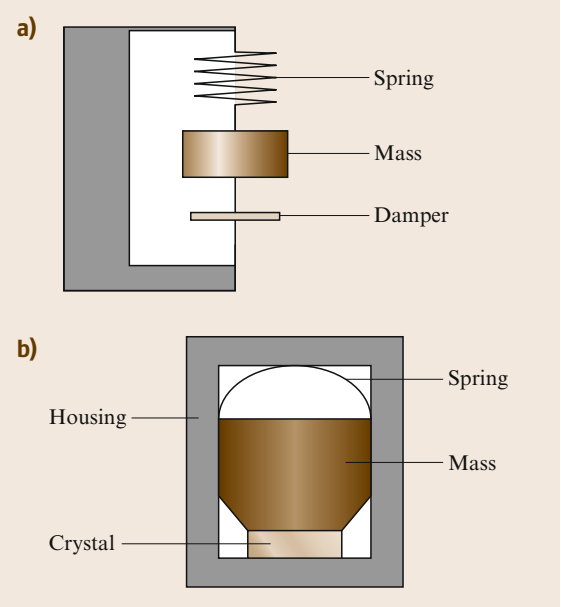


Fig. 29.6 Accelerometers. **(a)** Mechanical accelerometer; **(b)** piezoelectric accelerometer ◄

ever a static force is presented. This need to pre-estimate the expected force and the resulting (potentially long) time for the system to converge on a final measurement coupled with non-ideal performance of the spring limits the applicability of mechanical accelerometers. Another issue with mechanical accelerometers is that they are particularly sensitive to vibration.

29.3.2 Piezoelectric Accelerometer

Rather than relying on a direct mechanical measurement of external forces, piezoelectric accelerometers are based on a property exhibited by certain crystals which generate a voltage across them when stressed. A small mass can be positioned so that it is only supported by the crystal, and as forces cause the mass to act upon the crystal this induces a voltage that can be measured (Fig. 29.6b).

29.4 IMU Packages

An inertial measurement unit (IMU) is a device that utilizes measurement systems such as gyroscopes and accelerometers to estimate the relative position, velocity and acceleration of a vehicle in motion. First demonstrated in 1949 by C. S. Draper, IMU's have become a common navigational component of aircraft and ships. Historically an IMU is self-contained and provides this estimate without reference to external references, however the definition has become less precise in recent years and now it is common to have the term IMU also refer to systems which do include such external references.

IMU's come in two basic flavours, gimbaled systems and strap-down systems. As their name suggests, gimbaled IMU's are mounted within complex gimball structures in order to provide a stable platform from which measurements can be made. Gyroscopes are used to ensure that the gimball remains aligned with the initial reference frame at power up. The orientation of the gimbaled platform relative to the vehicle is used to map measurements taken within the IMU to the reference frame of the vehicle. Strap down IMU's, on the other hand, have the IMU rigidly connected to the vehicle (strapped down), so no such transformation is required. In either case estimating the motion relative to the initial frame requires integrating information from the sensors within the IMU (accelerometers, gyroscopes, etc.) in real time. This was a significant computational effort in the early days of IMU's, and thus historically (prior to the 1970s) the gimbaled IMU was more common. Given the low expense of such computation today, and the costs associated with manufacturing and operating gimbaled IMU's, strap down IMU's are much more common today [29.19].

The sensing technology of an IMU is typically integrated with an on-board computational unit to produce an attitude and heading reference unit (AHRS) which is a single device that maintains a 6-DOF estimate of the pose of the vehicle (position (x, y, z) and orientation (roll, pitch, yaw)). In practice, the distinction between an AHRS and an IMU is often unclear since the combinations of software and device outputs is quite varied, and most MEMS devices contain a substantial computational substrate and are able to directly offer up a position, velocity, and attitude (PVA) solution. In general, an AHRS includes gyroscopes, accelerometers and a magnetometer. These are often combined with a GPS as described in the next section. In addition to maintaining a 6-DOF pose of the vehicle, commercial IMU's also typically maintain estimates of velocity and acceleration and can usually provide a diverse set of lower-level measurements as well.

The basic computational task of an IMU is shown in Fig. 29.7. This IMU uses three orthogonal accelerometers and three orthogonal gyroscopes. The Gyroscope data $\boldsymbol{\omega}$ is integrated to maintain an ongoing estimate

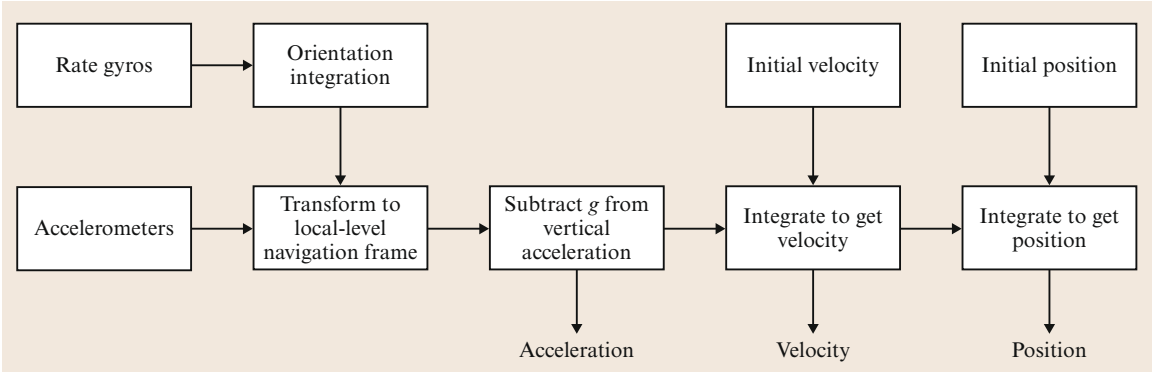


Fig. 29.7 IMU block diagram

of vehicle orientation $\boldsymbol{\theta}$. At the same time, three accelerometers are used to estimate the instantaneous vehicle acceleration $\boldsymbol{a}$. This data is then transformed via the current estimate of the vehicle orientation relative to gravity, so that the gravity vector can be estimated and extracted from the measurement. The resulting acceleration is then integrated to obtain vehicle velocity $\boldsymbol{v}$ and then integrated again to obtain position $\boldsymbol{r}$.

IMU's are extremely sensitive to measurement errors in the underlying gyroscopes and accelerometers. Drift in the gyroscopes leads to mis-estimates of the vehicle orientation relative to gravity, resulting in an incorrect cancellation of the gravity vector. As the accelerometer data is integrated twice, any residual gravity vector will result in a quadratic error in position [29.18]. As it is never possible to completely eliminate the gravity vector and this and any other error is integrated over time drift is a fundamental issue for any IMU. Given a sufficiently long period of operation all IMU's eventually drift and reference to some external measurement is required to correct this. For many field robots GPS has become an effective source for these external corrections.

29.4.1 Performance

Gyroscopes and Inertial measurements units today frequently include the required computational support to directly integrate the inertial signals and provide position, velocity, and attitude (PVA) estimates as direct output. The quality of these estimates is critically dependent not only on the sensing technology, but on the algorithms (almost universally including an extended Kalman filter) used in the estimation process and their calibration. While the extended Kalman filer is, of course, a well understood algorithm, the subtlety of its implementation remains a significant distinguishing factor among different brands of IMU.

Intertial measurement units can be evaluated with respect to various factors that determine performance. Several of these are enumerated below:

1. Bias repeatability. This is maximum deviation of the gyroscope under fixed inertial operation conditions with constant temperature, i. e., the drift of the reading under ideal conditions. This is measured over different time scales and leads to *short term* and *long term* bias repeatability.
2. Angle random walk. This measures the noise in the angular rate data coming from the gyro.
3. Scale factor ratio. This parameter is not specific to IMU's or gyros and is a general measurement of signal amplitude. It measures the ratio of the output analogue voltage to the sensor parameter of interest. For a gyroscope this is typically measured in $\mathrm{mV/(deg/s)}$ whereas for an accelerometer it is typically $\mathrm{mV/(m/s^2)}$.
4. Position accuracy. The accuracy in practice of position estimation depends on the sensors and the integration algorithm, both of which are intrinsic to the IMU, as well as the nature of the trajectory experienced by the IMU.

29.5 Satellite-Based Positioning (GPS and GLS)

The global navigation satellite system (GNSS) and it's most commonplace instance, the global positioning system (GPS), is the single most commonly used mechanism for location estimation. It provides a three dimensional position estimate in absolute coordinates as well as current time and date and is available any-

where on the earth's surface. Standard GPS provides a position estimate in the horizontal plane to within about 20 meters. It was original developed for military applications but has become widely adopted in civilian applications including automobile navigation systems, recreational orienteering, and inventory tracking for transportation companies. It is rapidly being superseded (via augmentation) by systems that add related technologies to basic GPS reception.

29.5.1 Overview

GPS is based on received radio signals transmitted by an ensemble of satellites orbiting the earth. By comparing the time delays from the different satellite signals, a position fix can be computed. The most widely accepted GPS system is based on the NAVSTAR satellite system deployed and maintained by the United States, specifically by the Air Force Space Command. As a US military service, the US government reserves the right to terminate or modify it's availability at their discretion. A similar system named GLONASS (globalnaya navigatsionnaya sputnikovaya sistema) is operated by the Russian government. Many modern GPS receivers and cell phones can access both the US GPS and Russian GLONASS systems. Another alternative system is being deployed by the European Union, named *Galileo*, with the explicit expectation that it not be under military control. It is expected to offer two different classes of service, an open service and an encrypted higher-quality commercial service. Other GPS systems such as the Chinese Beidou, the Japanese quasi-zenith satellite system (QZSS), the Indian regional navigational satellite system (IRNSS) system are intended to provide similar functionality. GPS receivers that are able to receive position data from multiple different satellite systems have become readily available and are referred to as global navigation satellite system (GNSS) systems.

Technically, the term GPS always refers to the NAVSTAR system although in common parlance it is sometime used to refer to any system that provides satellite-based positioning. Historically NAVSTAR provided two different services: the precise positioning system (PPS), reserved primarily for military users, and the standard position system (SPS) with lower accuracy. The difference between these was referred to as selective availability (SA). This difference in accuracy was artificially induced via pseudo-random noise in the SPS signal for strategic reasons, and was eliminated on May 2, 2000. Although this accuracy distinction could, in principle, be reinstated this seems unlikely, and recent GPS satellites (GPS Block III series, not launched at this time) are not equipped with the SA feature. US military users have access to a precise positioning service (PPS). This second frequency enables military users to correct for radio signal degradation caused by the Earth's atmosphere.

The GPS satellite network is based on a base constellation of 24 orbiting satellites along with up to six supplementary additional satellites that are also operational. These satellites are in almost-circular medium earth orbit. As opposed to being geostationary the orbits are semi-synchronous, meaning that their position relative to a ground observer is constantly changing, and that their orbital period is exactly half a sidereal day. The orbits are selected so that from almost any point of the earth's surface there will always be four or more satellites directly visible – a criterion for obtaining a GPS position estimate. The satellites are organized into six orbital planes with four satellites in each. The system is designed such that averaged over the entire earth's surface and over a 24 hour internal, the satellites should provide 99.9% coverage at the worst-covered location in any 24 hour interval, and the signal should be at least 83.92% available at the worst place on earth on the worst day over a 30 day measurement interval. Note that this criterion for availability takes into account transmitted operational factors above and beyond simple coverage. Of course, this criterion ignores the reality of topographic features like mountains, as well as other objects such as buildings that can obstruct the like-of-sight.

Each satellite repeatedly broadcasts a data packet known as the coarse-acquisition (C/A) code, which is received by the GPS receiver on the L1 channel at 1575.42 MHz. The simple principle is that if the receiver knows the absolute positions of the observed satellites, the receiver position can be directly determined. If the signal propagation time for the radio signals were known, the receiver position could be computed directly via trilateration (Fig. 29.8). This implies that an absolute timing reference is present on the receiver, which would be prohibitively costly. Instead, only the satellites have highly accurate atomic clocks. The receiver computes the difference in signal propagation times between the different satellites, and uses this to compute a range estimate referred to as a *pseudo-range* (to explicitly indicate that it is corrupted by several sources of measurement noise). The specific geometric problem is referred to as *multilateration* or *hyperbolic positioning* and the solution is computed using a sophistical Kalman filter within the GPS receiver. To avoid retaining an ephemeris (pose) table for the satellites and a very accurate clock in the receiver, each satellite broadcasts its own position and an accurate time signal as part of the data packet that it transmits.

GPS satellites broadcast at several different frequencies known as L1 through L5 although until recently only L1 (1575.42 MHz) and L2 (1227.6 MHz) were being using for civilian GPS receivers. Dual frequency L1/L5 GPS systems have become more broadly largely due to the emergence other GNSS systems in addition to the original NAVSTAR GPS.

The standard service offered by NAVSTAR and the performance criteria for it are determined by the L1 signal, which contains two unencrypted components: the acquisition message (coarse acquisition message C/A) and a navigation data message. It is also possible to use the encrypted L2 signal as well, even without the secret decryption keys, to provide augmented error correction (by observing the relative effects of ionospheric distortion as a function of frequency). The restricted access signal broadcast on both the L1 and L2 channels is the P-code (as well as a fairly recent M-code) which, is known as the Y-code or P(Y) or P/Y code once it is encrypted. Both the C/A and P(Y) code include the navigation message stream that specifies clock bias data, orbital information, ionospherical propagation corrections factors, ephemeris data, status information on all the satellites, universal time code and other information. The satellite performance is coordinated by the master control station located at the Schriever Air Force Base near Colorado Springs, (Colorado, USA) and is connected to a global network of five additional monitoring stations (Cape Canaveral, Ascension Island, Kwajalein Atoll (Marshall Islands), Diego Garcia Atoll, Hawaii) which are used to produce the measurements that are uplinked to generate the navigation message stream. Finally, it should be noted an additional signal is now available on the L2 frequency band. This L2C signal on satellites designated *block IIR-M* promises to provide much improved receiver sensitivity so that position fixes can be obtained in environments, such as in forests, where they are currently not readily available.

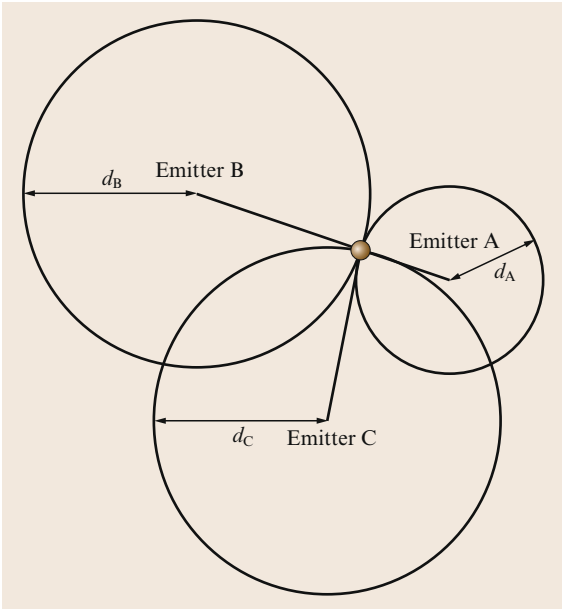


Fig. 29.8 GPS trilateration on the plane. Suppose that one receives signals from three transmitters (A, B, and C) with known locations. Knowledge of the signal delay from one emitter (say A) localizes the receiver to lie on a circle of known diameter (d_A) whose center is the emitter. The constraints from two emitters intersect at two points (maximum). A third emitter is required to disambiguate these two solutions. In 3-D, the signal propagation constraint from a single emitter constrains the receiver to lie on a sphere. The intersection of the constraints from two emitters constrains the receiver to lie on a circle. The intersection of the constraints from three emitters constrains the receiver to one of two points

29.5.2 Performance Factors

GPS performance depends on several factors: satellite transmission accuracy, environmental conditions, interactions with ground-based obstacles, and receiver properties.

In the context of robotics, factors that effect the performance of the satellites themselves and the atmospheric conditions are essentially uncontrollable. Nevertheless, it should be noted that these can be sources of error and that the GPS signal itself may not always be reliable. A *service failure* is defined as a set of circumstances where the positioning service exhibits atypical error behavior (i. e., incorrect signals). Such failures are classified into minor and major failures. Minor failures are those that have limited impact on a receiver and lead to ranging error of under 150 m. Major failures are those that lead either to larger errors or data processing overloads in the receiver. If a single satellite experiences an error that leads to a major failure, then within a 6 hour period, approximately 63% of the earth's surface will have the satellite in view a some point.

The controllable factors in using GPS for accurate localization are (i) it requires an unobstructed line of sight to the satellites, (ii) it depends on atmospheric conditions, and (iii) it depends the ability to receive (weak) radio-frequency communications. There is a potential for wildly incorrect estimates. Generally satellites that are directly overhead provide better signal than those near the horizon. In addition, since the basis of GPS position is differential signal analysis, it is best if the satellites used in the GPS computation are widely spaced in the sky.

GPS signals are in the microwave band and, as such, they can pass through plastic and glass, but are absorbed by water (wood, heavy foliage) and are reflected by many materials. As a consequence, GPS is unreliable in heavy forest, deep canyons, inside automobiles and boats, heavy snowfall or between tall buildings. In some cases, partial obstruction of the sky may not prevent a position estimate from being computed. Assuming the minimum number of satellites operating at any time is 24, then on average across the earth's surface 8 satellites are in view so that even partial occlusion of the sky can often be tolerated. On the other hand, partial occlusion can lead to reduced accuracy since the selection of available satellites used for computing position becomes limited and optimal accuracy is obtained by using as many satellites as possible (weighting them appropriately in the internal Kalman filter).

Secondary factors that differentiate different GPS receivers are the rate at which the signals are collected, the receiver sensitivity, the number of satellites used in the final computation, the number of factors taken into account in the estimator and the exploitation of supplementary positioning schemes such as WAAS (described later). A major factor in determining the rate at which estimates can be produced is the number of independent receiver elements in the GPS system. Sequential single-channel receivers are simpler and thus more economical (and potentially smaller), but they must sequentially lock onto each satellite being used. Parallel multi-channel receivers can lock onto to more than one satellite at once, and are generally faster and more costly; some degree of parallelism is the norm in good-quality consumer devices.

GPS computations are based on an estimation of the dilution of precision (DOP) and specifically for the dilution of precision of the positional parts of the system (i. e., positional dilution of precision PDOP). These correspond to partial derivatives of the error with respect to position and allow the most accurate ensemble of visible satellites to be determined at any time. The standard implementation for GPS systems specifies that PDOP values be recomputed every 5 minutes.

The minimum performance parameters for GPS receivers are based upon transforming instantaneous range residuals to a user position estimate using a linearized position solution from a stationary, surveyed location. Most GPS receivers use additional techniques such as range residual smoothing, velocity aiding, Kalman filtering, or multiple satellite (*all-in-view satellite*) solutions. That said, formal performance for the system is measured with respect to the minimum. The GPS position estimation algorithm is summarized as follows:

1. Select the best four satellites based upon minimum error measured in terms of PDOP.
2. Update every five minutes, or whenever a satellite being used in the solution sets.
3. Measure the pseudo range to each satellite. Each of the four measurements must have a reception timetag within 0.5 s of the solution time. The reception timetag is based upon measurement system time, and the transmission timetag is based upon satellite time.
4. Determine the ephemeris for each of the satellites being used, and compute the Earth-centred, Earth-fixed (ECEF) coordinates for each. Correct for the earth's rotation and thus compute an estimate pseudo-range measurement that should be obtained for each satellite.
5. Compute the *range residuals* as the differences between the actual and observed measurements.
6. Estimate the matrix $\mathbf{G}$ that determines the overall system solution, know as the *position solution geometry matrix*. The matrix can be described in terms of a collection of row vectors one for each of the satellites being used, each row being made up of the x, y, z and time coordinate direction cosines for the vector between the user and the satellite (the is with respect to a fixed reference frame for the planet called the World Geodetic System, WGS84). WGS84 represents position on the Earth in terms of latitude, longitude, and height above the surface, which is modeled as an ellipsoid. The WGS84 coorinate system represents longitude as a value from −180 to 180 degrees, with 0 degrees at the Earth's prime meridian, and longitude as a value from 90 degrees (at the North pole) to −90.
7. Compute the user position.

The standard implementation of GPS is based upon a position fix rate of once per second, although faster and slower rates are possible. Under typical operating conditions and without specialized enhancements GPS accuracy is roughly 20−25 m in the horizontal plane and 43 m in the vertical direction. The restricted PPS signal provides an accuracy of at least 22 m (typical values are 7−10 m) in the horizontal plane and 27.7 m in the vertical direction as well as UTC time accuracy within 200 ns based on a reference signal from the US Naval Observatory.

GPS signals can be affected by multipath issues, where the radio signals reflect off surrounding terrain – buildings, canyon walls, hard ground, etc. This delay in reaching the receiver causes inaccuracy. A variety of receiver techniques, most notably narrow correlator spacing [29.20], have been developed to mitigate multipath errors. For long delay multipath, the receiver itself

can recognize the wayward signal and discard it. To address shorter delay multipath due to the signal reflecting off the ground, specialized antennas may be used. This form of multipath is harder to filter out as it is only slightly delayed as compared to the direct signal, causing effects almost indistinguishable from routine fluctuations in atmospheric delay.

29.5.3 Enhanced GPS

A wide range of augmentation systems have been developed both by government agencies and private companies to enhance the basic GPS signal.

Wide Area Augmentation System (WAAS)

The wide area augmentation system (WAAS) is an augmentation system operated by the US Federal Aviation Administration (FAA) to provide better GPS and timing performance for aircraft operating over North America. The WAAS system is based around a supplementary signal that can be received by GPS receivers to improve their accuracy. WAAS increases the accuracy of horizontal position estimates from 10–12 m with GPS alone, to between 1 and 2 m. The WAAS signal contains corrections for the GPS signal that reduce the effects of errors due to timing errors, satellite position corrections, and local perturbations due to variations in the ionosphere. These correction terms are estimated by ground-based stations at fixed and accurately-known positions and uplinked to satellites which broadcast them to suitably enabled GPS receivers. The WAAS signal is only computed and available for North America, but similar correction signals are becoming available elsewhere as part of the standardization of satellite-based augmentation systems (SBASs). This includes Europe (where is it called EGNOS, the European Geostationary Navigation Overlay Service) and Japan and parts of Asia (where it is called MSAS, the multifunctional satellite augmentation system). Further enhancements to GPS and WAAS, in the form of global navigation satellite system landing system (GLS), are slated for completion in 2013.

Differential GPS

Differential GPS (DGPS) is a technique for correcting GPS signals by using a nearby GPS receiver located at a known accurately-surveyed position. In fact, several variations on this basic idea exist and are also known under the general rubric of ground based augmentation systems (GBASs). DGPS uses the same principles as WAAS but on a local scale without resorting to the use of satellite uplinks. The receiver at the known position computes the error in the GPS signal and transmits it to the nearby receiver at the unknown location. Since the error varies as a function of position on the earth, the effectiveness of the correction degrades with distance, typically with a maximum effective range of a couple of hundred miles. The method was especially desirable before the suspension of selective availability and the development of WAAS (which can be viewed as a form of DGPS). In the USA and Canada, a network of ground based DGPS transmitters are in place, send signals using radio frequencies between 285 and 325 kHz. Commercial DGPS solutions also exist akin to WAAS.

29.5.4 Nationwide Differential GPS System (NDGPS)

The (US) nationwide differential GPS system (NDGPS) is an augmentation system that integrates costal and ground transportation systems to provide 10–15 cm accuracy throughout most of the US. The system relies on a network of radio signals broadcast from base stations. Similar systems exist in a number of other countries including Canada.

Receiver Autonomous Integrity Monitor (RAIM)

Receiver autonomous integrity monitor (RAIM) is a technique by which multiple pseudo-range measurements are obtained (i. e., pose estimates) using different combinations of satellites. If inconsistent measurements are obtained, it indicates that a failure of some sort has taken place in the system. A position fix using at least satellites is needed to detect such an error which at least six satellite are needed to exclude the data from a single bad satellite and still obtain a reliable estimate.

Real-Time Kinematic Positioning (RTK)

One GPS enhancement technology with rapidly growing acceptance in robotics applications is real-time kinematic positioning or RTK [29.21]. RTK-GPS in a GNSS configuration also exploits multiple satellite constellations and is often regarded as the most high-performance GPS solution that is readily available. Because of it's high accuracy, it is often referred to as *survey grade*. In practice, commonplace RTK-GPS units can achieve horizontal positioning accuracy on the order of 1 cm. RTK-GPS augments standard differential GPS technology by enhancing the effective resolution of the pseudo-range distance estimates using corrections obtained by a base station. Specifically, RTK estimates the number of carrier cycles in the communication path between a GPS satellite and each of the base station and the receiver, and thus is essentially dependent on the phase of the signal. Such units typically ignore the actual data payload of the signal and exploit the phase of the GPS signal (typically the L1 band); this is a fundamentally different mode of operation from

the commonplace code phase computation that low cost receivers use. The base station that is used to provide a reference position for the estimation process can be part of a custom system being deployed, but various networks reference base stations accessible to the public now exist. When a network of base stations are used to provide correction signals over a wide region, the system is referred to as network real-time kinematic (NRTK) GPS and a class of networks as continuously operating reference station (CORS) networks. NRTK networks are evolving rapidly and a range computing protocols have been developed or employed including the Radio Technical Commission for Maritime Services Special Committee 104 (RTCMS C104) protocol for transmitting GNSS data.

If properly estimated, RTK-GPS can obtain position estimates that are several orders of magnitude more accurate than the multiple meter accuracy associated with conventional DGPS. Critical to RTK is estimating the number of whole signal cycles that are present in the path from the satellite to the station. GPS carrier waves have a wavelength of about 19 cm, and this coupled with an estimate of the phase of the carrier signal can result in pseudo-range estimates with errors in the 1 mm range.

29.5.5 GPS Receivers and Communications

GPS receivers are classified according to their performance and cost. The best receivers are referred to as *geodetic grade* (or survey grade) with economical models referred to as *resource grade* or *recreational*. In general the costs of these different models varies by several orders of magnitude, but the gap in performance is gradually narrowing.

Receivers come in two types: code phase and carrier phase. Code phase receivers use the satellite navigation message part of the data stream to provide the ephemeris data and produce real-time output. There is a delay for them to lock into the satellites, but then they produce output continuously without an initial position estimate. The C/A signal is a 1023 pseudo-random *noise* (PRN) bit string with a known key. The actual pseudo-range data is determined by finding the offset of this bit string, and computing the correct shift of this bit string is the essential computation that determined the delay before a lock is achieved.

Carrier phase receivers, on the other hand, use the phase of the raw GPS signal rather than the embedded (digital) C/A signal. The L1 and L2 signals have wavelengths of 19 and 24 cm respectively, and good quality phase measurements allow horizontal positioning accuracies on the order of millimeters in principle. These measurements, however, only provide relative position information within a neighborhood of some tens of kilometers.

Serial Protocols

Consumer GPS devices almost universally support some variant of the National Marine Electronics Association (NMEA) protocol as well as various proprietary protocols specific to each manufacturer. NMEA is a serial protocol sometimes encapsulated within other protocols. Several variants of the protocol exist, but NMEA 0183 is the most commonly supported while NMEA 2000 supports higher data rates and uses a very different communication protocol. While the protocol is proprietary and the official specification can only be purchased from the NMEA, there are several open-source descriptions of the protocol that have been reversed engineered and its basic message characteristics are very widely used and distributed.

The NMEA 0183 protocol supports an ASCII mode of communications based on a *talker* (the GPS receiver) and one or most *listeners* (computers) which receiver simple protocol strings called sentences. Anecdotal evidence suggests that there are ambiguities in the protocol that can lead to difficulties is assuring compatibility between devices (this suggestion is necessarily made without inspecting the proprietary documentation, which might preclude its description here).

29.6 GPS–IMU Integration

It has become commonplace for GPS and IMU technologies to be integrated into a single GPS-aided inertial navigation system (GPS/INS) to leverage their complementary advantages. Although GPS offers the promise of high resolution positioning information on or about the surface of the planet, it does not solve all of the problems associated with robot pose estimation. First, it does not directly obtain information about vehicle orientation – to determine the orientation of the vehicle (yaw, and for many vehicles pitch and roll) must be estimated by either differentiating the GPS signal or through integration with other sensors including compasses, gyrocompasses, and IMU's. Second, GPS receivers are generally unable to provide continuous independent estimates of position. Estimates are only available at distinct time instances with (for inexpensive receivers at least) considerable delays between measurements. A continuous

estimate of pose requires estimation of pose between GPS readings. Finally, it is not always possible to obtain a GPS fix. Local geography (e.g., mountains, buildings, trees) or an overhead cover that is opaque to radio signals (e.g., indoors, underwater) can block the signal entirely. Integration of a GPS receiver with other sensor technology (often an IMU) can be used to deal with these issues, at least for short periods of time.

The process of integrating GPS and IMU data is typically expressed as an extended Kalman filter estimation process (Sect. 35.2.3). Essentially the IMU data is used to bridge between solid GPS measurements and is combined in a least squares optimal sense with the GPS data when both are available. Given the complementary nature and true independence of the two sensors, a wide range of commercial packages have been developed to integrate GPS and IMU data [29.22].

When GPS measurements are integrated in a robotic vehicle, and in particular when they are combined with data from an IMU the orientation of the antenna becomes a consideration. The GPS position *fix* is based on the position of the GPS antenna and if it is located far from the IMU or the center of the vehicle coordinate system then this must be accounted for or instabilities in the position or orientation can occur.

29.7 Further Reading

29.7.1 Odometry

Many general robotics books including [29.2] and [29.23] provide considerable information on vehicle odometry and derive odometry equations for standard vehicle designs.

29.7.2 Gyroscopic Systems

Everett's book [29.24] provides a review of various sensor technologies including gyroscopic systems and accelerometers. Interesting historical documentation on the gyrocompass and its inventor can be found in *Hughes*' book [29.5].

29.7.3 Accelerometers

Everett's book [29.24] provides a review of various sensor technologies including gyroscopic systems and accelerometers.

29.7.4 GPS

Considerable details on the theory and implementation of GPS systems can be found in *Leick*'s book [29.25]. See also [29.26]. Details of various approaches to GPS/INS integration can be found in [29.27] and [29.28]

29.8 Currently Available Hardware

Although the specific models listed below are likely to have a short shelf life, the list of contacts is may be a good starting point for the identification of specific inertial sensing devices.

29.8.1 Gyroscopic Systems

- KVN DSP-3000 Tactical Grade
  Fiber Optic Gyro (FOG),
  KVH Industries Inc., 50 Enterprise Center,
  Meddletown RI, 02842-5279,
  USA, 1-401-847-3327
- Fiber Optic Gyroscope HOFG-1(A),
  Corporate Headquarters Hitachi Cable Ltd.
  4-14-1 Sotokanda, Chiyoda-ku,
  Tokyo 101-8971, Japan.
- Rate Gyroscope CRS03,
  Silicon Sensing Systems Japan, Ltd.
  1-10 Fusocho (Sumitomo Precision Complex),
  Amagaxaki, Hyogo 660-0891, Japan.

29.8.2 Accelerometers

- Accelerometer GSA 101,
  A-KAST Measurements and Control Ltd.,
  1054-2 Centre St. Suite #299,
  Thornhill, ON, L4J 8E5, Canada.
- ENDEVCO MODEL 22,
  Brüel and Kjaer,
  DK-2850 Naerum, Denmark.

29.8.3 IMU Packages

- μIMU,
  MEMSense, 2693D Commerce Rd.,
  Rapid City, SD 57702, USA.
- IMU400 MEMS Inertial Measurement Unit,
  Crossbow Technology Inc.,
  4145 N. First St., San Jose, CA 95134, USA.
- IntertiaCube3, (3DOF IMU),
  Intersense,
  36 Crosby Dr, #15,
  Bedford, MA 01730, USA.

29.8.4 GPS Components

- Garmin GPS 18,
  Garmin International Inc.,
  1200 East 151st St.,
  Olathe, KS 66062-3426, USA.
- Magellan Meridian Color,
  Thales Navigation
  471 El Camino Real,
  Santa Clara, CA 95050-4300, USA.
- TomTom Bluetooth GPS Receiver,
  Rembrandtplein 35,
  1017 CT Amsterdam,
  The Netherlands.

References

29.1 Vitruvius: *Ten Books on Architecture* (Harvard Univ. Press, London 1914) p. 301, English translation by M. H. Morgan

29.2 G. Dudek, M. Jenkin: *Computational Principles of Mobile Robotics*, 2nd edn. (Cambridge Univ. Press, Cambridge 2010)

29.3 E. Maloney: *Dutton's Navigation and Piloting* (US Naval Institute Press, Annapolis 1985)

29.4 J.G.F. Bohnenberger: Beschreibung einer Maschine zur Erläuterung der Geseze der Umdrehung der Erde um ihre Axe, und der Veränderung der Lage der lezteren, Tüb. Bl. Naturw. Arzneik. **3**, 72–83 (1817)

29.5 T.P. Hughes: *Elmer Sperry: Inventor and Engineer* (The Johns Hopkins Univ. Press, Baltimore 1993)

29.6 H.W. Sorg: From Serson to Draper – Two centuries of gyroscopic development, Navigation **23**, 313–324 (1976)

29.7 W. Davenport: *Gyro! The Life and Times of Lawrence Sperry* (Charles Scribner, New York 1978)

29.8 J.F. Wagner: From Bohnenberger's machine to integrated navigation systems, 200 years of inertial navigation, Photogramm. Week (2005)

29.9 R.P. Feynman, R.B. Leighton, M. Sands: *The Feynman Lectures on Physics* (Addison-Wesley, Reading 1963)

29.10 T.F.T. Submarine: *Navpers 16160 Standards and Curriculum Division Training* (Bureau of Naval Personnel United States Navy, Arlington 1946)

29.11 G. Sagnac: L'ether lumineux demontre par l'effect du vent relatif d'ether dans un interferometre en rotation uniforme, C. R. Acad. Sci. Paris **157**, 708–710 (1913)

29.12 G. Sagnac: Sur la preuve de la realitet de l'ether lumineaux par l'experience de l'interferographe tournant, C. R. Acad. Sci. Paris **157**, 1410–1413 (1913)

29.13 F. Harress: Die Geschwindigkeit des Lichtes in bewegten Körpern, Ph.D. Thesis (Richters, Jena 1912)

29.14 A.A. Michelson, H.G. Gale: The effect of the Earth's rotation on the velocity of light, J. Astrophys. **61**, 140–145 (1925)

29.15 S. Ekeziel, H.J. Arditty: *Fiber-Optic Rotation Sensors*, Springer Ser. Opt. Sci. (Springer, Berlin, Heidelberg 1982)

29.16 G.E. Stedman: Ring-laser tests of fundamental physics and geophysics, Rep. Prog. Phys. **60**, 615–688 (1997)

29.17 D. Mackenzie: From the luminiferous ether to the Boeing 757: A history of the laser gyroscope, Technol. Cult. **34**(3), 475–515 (1993)

29.18 E. Foxlin, M. Harrington, Y. Altshuler: Miniature 6-DOF inertial system for tracking HMDs, SPIE Proc. 3362 (1998) pp. 214–228

29.19 M. Mostafa: History of inertial navigation systems in survey applications, J. Am. Soc. Photogramm. Remote Sens. **67**, 1225–1227 (2001)

29.20 A.J. Van Dierendonck, P. Fenton, T. Ford: Theory and performance of narrow correlator spacing in a GPS receiver, Navigation **39**, 265–283 (1992)

29.21 A. Rietdorf, C. Daub, P. Loef: Precise positioning in real-time using navigation satellites and telecommunication, Proc. 3rd Workshop Posit. Navig. Commun. (2006) pp. 123–128

29.22 J. Rios, E. White: Low cost solid state GPS/INS package, Proc. Inst. Navig. Conf. (2000)

29.23 R. Siegwart, I.R. Nourbakhsh: *Introduction to Autonomous Mobile Robots* (MIT Press, Cambridge 2004)

29.24 H.R. Everett: *Sensors for Mobile Robots: Theory and Application* (Peters, Wellesley 1995)

29.25 A. Leick: *GPS Satellite Surveying* (Wiley, Hoboken 2004)

29.26 P. Misra, P. Enge: *Global Positioning System: Signals, Measurements, and Performance* (Ganga-Jamuna Press, Lincoln 2006)

29.27 J. Farrell, M. Barth: *The Global Positioning System and Inertial Navigation* (McGraw-Hill, New York 1989)

29.28 M.S. Grewal, L.R. Weill, A.P. Andrews: *Global Positioning Systems, Inertial Navigation, and Integration* (Wiley, Hoboken 2007)

Multimedia Contents

30. Sonar Sensing

Lindsay Kleeman, Roman Kuc

Sonar or ultrasonic sensing uses the propagation of acoustic energy at higher frequencies than normal hearing to extract information from the environment. This chapter presents the fundamentals and physics of sonar sensing for object localization, landmark measurement and classification in robotics applications. The source of sonar artifacts is explained and how they can be dealt with. Different ultrasonic transducer technologies are outlined with their main characteristics highlighted.

Sonar systems are described that range in sophistication from low-cost threshold-based ranging modules to multitransducer multipulse configurations with associated signal processing requirements capable of accurate range and bearing measurement, interference rejection, motion compensation, and target classification. Continuous-transmission frequency-modulated (CTFM) systems are introduced and their ability to improve target sensitivity in the presence of noise is discussed. Various sonar ring designs that provide rapid surrounding environmental coverage are described in conjunction with mapping results. Finally the chapter ends with a discussion of biomimetic sonar, which draws inspiration from animals such as bats and dolphins.

30.1 **Sonar Principles** 754
30.2 **Sonar Beam Pattern** 756
30.3 **Speed of Sound** 758
30.4 **Waveforms** 758
30.5 **Transducer Technologies** 759
30.5.1 Electrostatic 759
30.5.2 Piezoelectric 760
30.5.3 MEMS 760
30.6 **Reflecting Object Models** 760
30.7 **Artifacts** 761
30.8 **TOF Ranging** 762
30.9 **Echo Waveform Coding** 765
30.10 **Echo Waveform Processing** 767
30.10.1 Ranging and Wide-Bandwidth Pulses ... 767
30.10.2 Bearing Estimation 768
30.11 **CTFM Sonar** 769
30.11.1 CTFM Transmission Coding 769
30.11.2 CTFM TOF Estimation 769
30.11.3 CTFM Range Discrimination and Resolution 770
30.11.4 Comparison of CTFM and Pulse-Echo Sonar 770
30.11.5 Applications of CTFM 771
30.12 **Multipulse Sonar** 772
30.12.1 Interference Rejection 772
30.12.2 On-the-Fly Target Classification 772
30.13 **Sonar Rings and Arrays** 773
30.13.1 Simple Ranging Module Rings .. 773
30.13.2 Advanced Rings 773
30.13.3 Sonar Rings with FPGA Hardware Processing 774
30.13.4 Double Refresh Rate Sonar Ring 775
30.13.5 Sparse 3-D Arrays 775
30.14 **Motion Effects** 775
30.14.1 Moving Observation of a Plane . 776
30.14.2 Moving Observation of a Corner 776
30.14.3 Moving Observation of a Edge .. 777
30.14.4 The Effect of a Moving Observation on the Angle of Reception 777
30.14.5 Plane, Corner, and Edge Moving Observation Arrival Angles 777
30.15 **Biomimetic Sonars** 778
30.16 **Conclusions** 779
Video-References 780
References 780

30.1 Sonar Principles

Sonar is a popular sensor in robotics that employs acoustic pulses and their echoes to measure the range to an object. Since the sound speed is usually known, the object range is proportional to the echo travel time.

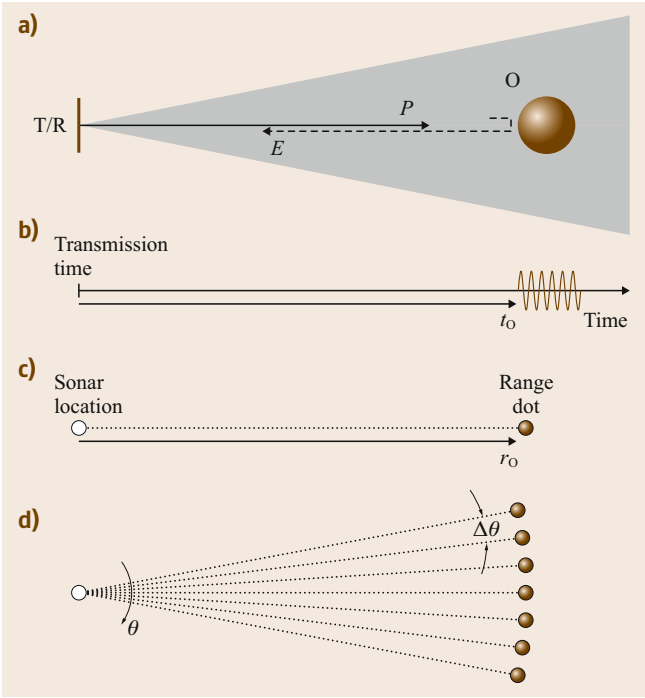


Fig.30.1a–d Sonar ranging principles: **(a)** sonar configuration, **(b)** echo waveform, **(c)** range dot placement, **(d)** sonar map

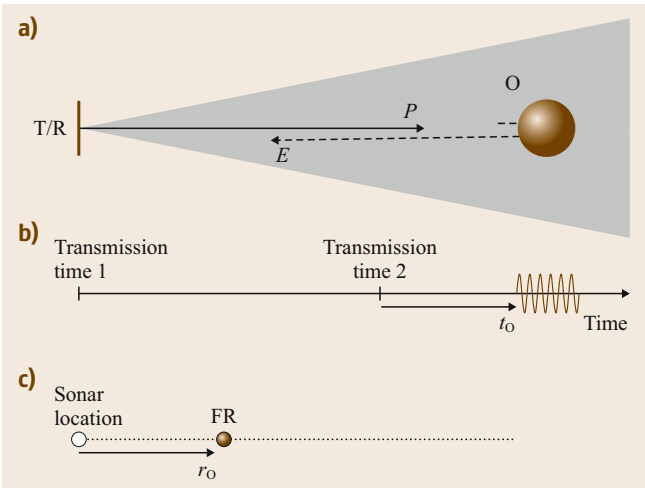


Fig.30.2a–c False range reading: **(a)** sonar configuration, **(b)** probing pulse 2 transmitted before echo from pulse 1 arrives, **(c)** false range (FR) is measured from transmission time 2

At ultrasonic frequencies the sonar energy is concentrated in a beam, providing directional information in addition to range. Its popularity is due to its low cost, light weight, low power consumption, and low computational effort, compared to other ranging sensors. In some applications, such as in underwater and low-visibility environments, sonar is often the only viable sensing modality.

Sonars in robotics have three different, but related, purposes:

1. Obstacle avoidance: the first detected echo is assumed to measure the range to the closest object. Robots use this information to plan paths around obstacles and to prevent collisions.
2. Sonar mapping: a collection of echoes acquired by performing a rotational scan or from a sonar array is used to construct a map of the environment. Similar to a radar display, a range dot is placed at the detected range along the probing pulse direction.
3. Object recognition: a sequence of echoes or sonar maps is processed to classify echo-producing structures composed of one or more physical objects. When successful, this information is useful for robot registration or landmark navigation.

Figure 30.1 shows a simplified sonar system, from its configuration to the resulting sonar map. A sonar transducer, T/R, acts as both the transmitter (T) of a probing acoustic pulse (P) and the receiver of echoes (E). An object O lying within the sonar beam, indicated by the shaded region, reflects the probing pulse. A part of the reflected signal impinges on the transducer as is detected as an echo. The echo travel time t_o, commonly called the *time-of-flight* (TOF) is measured from the probing pulse transmission time. In this case the echo waveform is a replica of the probing pulse, which usually consists of as many as 16 cycles at the resonant frequency of the transducer. The object range r_o is computed from t_O using

$$r_O = \frac{ct_O}{2}\,, \tag{30.1}$$

where c is the sound speed (343 m/s at standard temperature and pressure). The factor of 2 converts the round-trip ($P + E$) travel distance to a range measurement. Beam-spreading losses and acoustic absorption limit sonar range.

In forming a sonar map, a range dot is placed along the direction corresponding to the transducer's physical orientation. A sonar map is usually built by rotating the sensor about the vertical axis, indicated by the orientation angle θ, through a series of discrete angles

separated by $\Delta\theta$ and placing sonar dots the corresponding ranges. Since the range from the object O to the center of T/R is almost constant as T/R rotates, the range dots typically fall on a circle as long as O lies within the beam. Hence, sonar maps are made up of arcs.

The major limitations of sonar include:

1. The wide sonar beam causes poor directional resolution. Objects are located at the middle of isolated arcs, but shorter-range objects shorten the arcs of those at farther ranges, and the arcs produced by a collection of objects are often difficult to interpret. A consequence of this effect is that wide beams occlude small openings, limiting robot navigation.
2. The slow sound speed, relative to an optical sensor, reduces the sonar sensing rate. A new probing pulse should be transmitted after all detectable echoes from the previous pulse have expired, otherwise the false reading shown in Fig. 30.2 can occur. The echo from probing pulse 1 occurs after probing pulse 2 is emitted. Sonar measures the TOF from the most recent probing pulse. Many sonars transmit probing pulses every 50 ms, but encounter false readings in reverberant environments.
3. Smooth surfaces at oblique incidence do not produce detectable echoes. Figure 30.3 shows a planar surface (a wall) that acts as a mirror to the sonar beam. The important point is that the nearby wall does not itself produce a detectable echo, and a robot using sonar for obstacle avoidance may collide with the wall.
4. Artifacts caused by beam side-lobes and multiple reflections produce range readings in the environment where no objects exist. Figure 30.3 also shows the redirected beam enclosing object O. The echo is also redirected by the wall back to the transducer. From the transducer's reference, the object is at the virtual object location VO, and it would generate the same sonar map as shown in Fig. 30.1. Since there is no physical object corresponding to the sonar dot location, it is an artifact. Also, note that the acoustic energy indicated by the dot-dashed line reflected back to the transducer is not detected because it does not lie within the beam cone. Beam side-lobes often detect these echoes and produce shorter-range readings but placed along the sonar orientation.
5. Travel time and amplitude variations in the echoes caused by inhomogeneities in the sound speed. Both effects cause random fluctuations in the detected echo travel time, even in static environments. Figure 30.4 illustrates thermal fluctuations that cause speed up, retardation, and travel redirection by refraction of echoes. These cause temporal and amplitude variations in the echoes and jitter in the range readings. While these typically introduce minor changes in sonar maps, they often cause havoc with approaches using finer analysis.

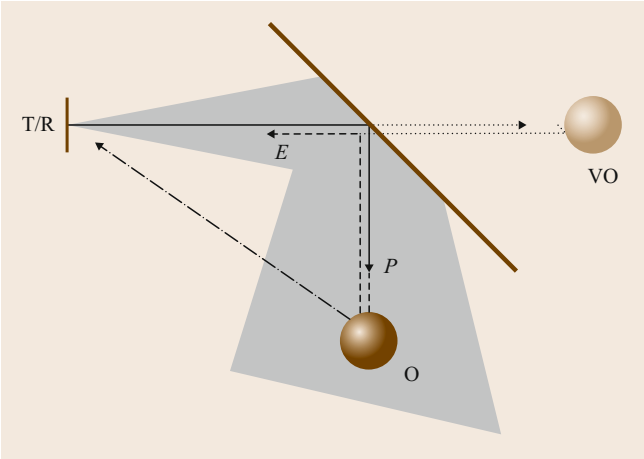


Fig. 30.3 The smooth surface redirects the beam, causing a sonar artifact at virtual object (VO) location. The *dot-dashed* echo path falls outside the sonar beam and does not produce a detectable echo

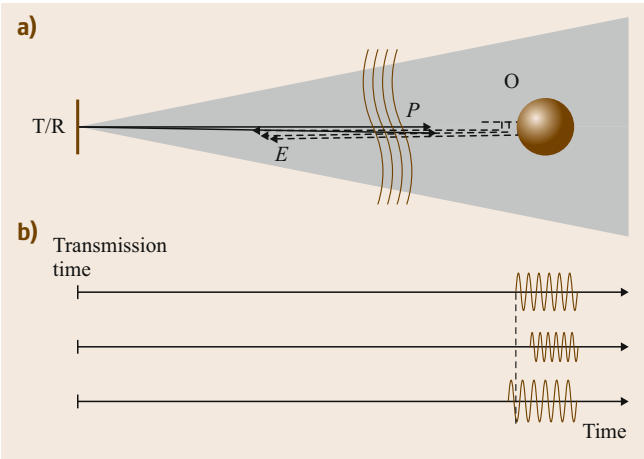


Fig.30.4a,b Random echo jitter. **(a)** Sonar configuration; thermal inhomogeneities in the acoustic transmission medium cause refraction effects. **(b)** Examples of variations in echo travel times and amplitudes in a static environment

This chapter describes the physical and mathematical details that extend this simplified sonar model to practical sonar systems.

30.2 Sonar Beam Pattern

To derive a qualitative description of the sonar transducer, we apply elementary acoustics theory to a simplified model to achieve a simple analytic form [30.1]. A sonar emitter is commonly modeled as a circular piston surface of radius a vibrating with frequency f in an infinite planar baffle. The wavelength λ is

$$\lambda = \frac{c}{f}\,, \tag{30.2}$$

where c the sound speed in air, 343 m/s at 25 °C [30.2]. When $a > \lambda$ the emitted pressure field forms a beam consisting of a main lobe surrounded by side-lobes. In the far field, or range greater than a^2/λ, the beam is described by its *directivity pattern*, which is the two-dimensional Fourier transform of the aperture function, in this case the circular aperture produces a Bessel function. The emitted pressure amplitude at range r and angle θ relative to the piston axis can be written as

$$P_{\mathrm{E}}(r,\theta) = \frac{\alpha a^2 f}{r}\left(\frac{2J_1(ka\sin\theta)}{ka\sin\theta}\right)\,, \tag{30.3}$$

where α is a proportionality constant that includes the density of air and the source strength, $k = 2\pi/\lambda$, and J_1 is the Bessel function of the first kind. The term in the brackets evaluates to 1 along the sonar axis, $\theta = 0$. The a^2 term indicates that the emitted pressure increases with the piston area. The frequency f appears in the numerator because the faster-moving piston generates higher pressures. The range r appears in the denominator because the conservation of energy requires the pressure to decrease as the beam widens with range.

The main lobe is defined by its first off-axis null, which occurs at an angle of

$$\theta_0 = \arcsin\left(\frac{0.61\lambda}{a}\right)\,. \tag{30.4}$$

For example, the popular electrostatic instrument grade transducer, formerly produced by Polaroid [30.3], had a radius of $a = 1.8$ cm and is conventionally driven at $f = 49.4$ kHz, making $\lambda = 0.7$ cm and $\theta_0 = 14.7°$.

An object that is small compared to λ and located in the emitted pressure field produces an echo with a spherical wavefront whose amplitude decays with the inverse of the distance propagated. In the common pulse-echo single transducer (*monostatic*) ranging sensor, only part of the echo wavefront impinges on the receiving aperture. The *sensitivity pattern* of the circular aperture, now acting as the receiver, has the same beam-like Bessel function form given in (30.3) by the reciprocity theorem [30.1]. If the reflecting object is located at (r,θ) relative to the transducer, the detected echo pressure amplitude, referenced to the receiver output, is given by

$$P_{\mathrm{D}}(r,\theta) = \frac{\beta f a^4}{r^2}\left(\frac{2J_1(ka\sin\theta)}{ka\sin\theta}\right)^2\,, \tag{30.5}$$

where β is a proportionality constant that includes parameters that cannot by controlled in a design, such as the density of air. The additional a^2 in the numerator occurs because larger apertures detect more of the echo wavefront.

Figure 30.5 shows the echo amplitude from a small (point-like) object located in the far field as a function

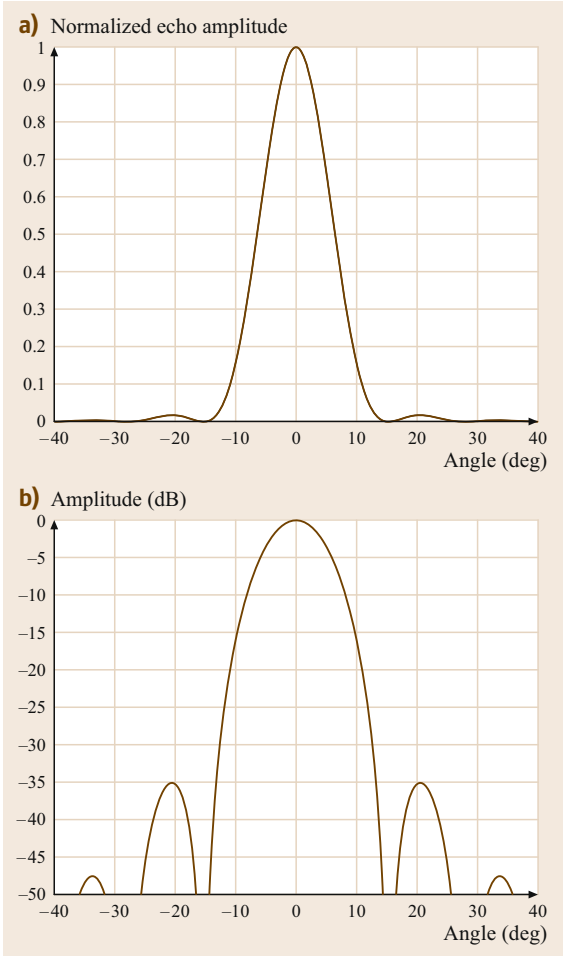


Fig.30.5a,b Normalized amplitude of echo from small object predicted by the piston model as a function of angle. **(a)** Linear scale. **(b)** Decibel scale

of angle detected by the electrostatic instrument grade transducer. The curve has been normalized by the on-axis echo amplitude.

This model is qualitative in that it provides the following practically useful insights:

- For a small reflector size relative to the wavelength, the echo amplitude decreases inversely with the square of the range because there is a $1/r$ dispersion loss from the transmitter to the object, followed by an additional $1/r$ dispersion loss in the echo back to the receiver. However, larger reflectors can be treated using a Huygens principle approach [30.4] by dividing them into smaller reflectors and coherently adding their echo contributions. When this is done in two dimensions over a normally incident extended plane reflector, the echo amplitude decreases as $1/r$ rather than $1/r^2$. A cylindrical reflector extends over one dimension and results in an amplitude variation with range between $1/r$ and $1/r^2$. A more-extreme situation can occur with a concave reflector that acts as an acoustic magnifier, resulting in an amplitude that decreases with a negative power of range of less than one.
- The transducer excited by an approximation to a sinusoid exhibits side-lobes due to null caused by phase cancellation. For example, the 16-cycle excitation employed in the conventional sonar exhibits side-lobes. The peak of the first side-lobe is −35 dB relative to the echo amplitude when a small reflector lies on the transducer axis. The specification sheet for the 600 series instrument-grade transducer shows the first off-axis null at 15° and a first side-lobe peak magnitude of −26 dB. We presume that these measurements were made using a plane as a reflector.
- This model can be used to compute approximate beam parameter values for other common transducers. For example, the SensComp 7000 Series [30.5] with $a = 1.25$ cm yields $\theta = 20°$, equal to the specified value. However, the specified first side-lobe peak magnitude equals approximately −16 dB, which is substantially different from the expected −35 dB.

The limitations of the qualitative model include:

- Actual transducers only approximate pistons vibrating in an infinite planar baffle. The infinite baffle directs all the radiated sound pressure into the half-space in front of the transducer. Actual transducers radiate in all directions, but most of the acoustic energy is concentrated within the main lobe.
- All pulse-echo ranging sonars operate with finite-duration pulses rather than infinite-duration sinusoids. Several systems described below use pulses that are quite different from a sinusoidal excitation, either in duration or in form. These are commonly analyzed by computing the spectrum of the pulse and decomposing it into several sinusoidal frequencies, each having its own beam pattern. For example, the echo amplitude predictions above are reasonably accurate, including beam width and side-lobes, for the 16-cycle pulses. However, when impulse or swept-frequency excitations are used, the net beam profile becomes the superposition (of linear amplitudes) of the beam patterns produced by each frequency component in the excitation. Such broadband excitations do not exhibit nulls because the nulls formed by one frequency are filled in by main and side-lobes of beams produced by other frequencies.
- Most sonar transducers are encased in protective housings. The electrostatic instrument-grade transducer cover forms a mechanical filter that enhances the acoustic output at 49.4 kHz. The cases of other transducers may distort the transmitted field, but most form some type of directional beam.
- The model does not include frequency-dependent acoustic absorption of the transmission medium. These reduce the echo amplitudes predicted by the model.

The analytic model above is limited to simple configurations. With current computational power, transducers can be extended to those with arbitrary, even multiple, apertures and with various excitations. Waveforms of echoes from objects having arbitrary shapes can be simulated by using Huygens principle [30.4]. The transmitter, receiver, and object surfaces are broken up into two-dimensional (2-D) surface arrays of emitting, reflecting, and detecting elements, using squares of dimension $< \lambda/5$ (the smaller the better, but taking longer). The impulse response of a given configuration is computed by assuming an impulsive emission and superimposing the travel times along all possible paths from all transmitter elements to all object elements and then to all receiver elements. The temporal resolution should be $< (20 f_{\max})^{-1}$, where $f_{\max}$ is the maximum frequency in the excitation. A 1 —s resolution is adequate for a 16-cycle 49.4 kHz excitation. A much finer resolution (< 0.1 —s) is required for an impulsive excitation. The echo waveform is then computed as the convolution of this impulse response with the actual transmitted pulse waveform [30.4].

30.3 Speed of Sound

The speed of sound c varies significantly with atmospheric temperature, pressure, and humidity and can be critical in determining the accuracy of a sonar system. This section outlines the relationship between c and these variables and is based on [30.6, 7].

The speed of sound in dry air at sea-level air density and one atmosphere pressure is given by

$$c_{\mathrm{T}} = 20.05\sqrt{T_{\mathrm{C}} + 273.16}\,\mathrm{ms}^{-1}\,, \tag{30.6}$$

where T_{C} is the temperature in degrees Celsius. Under most conditions (30.6) is accurate to within 1%. However, should the relative humidity be known, a better estimate can be made as

$$c_{\mathrm{H}} = c_{\mathrm{T}} + h_{\mathrm{r}}\left[1.0059\times10^{-3} + 1.7776\times10^{-7}(T_{\mathrm{C}} + 17.78)^3\right]\mathrm{ms}^{-1}\,. \tag{30.7}$$

Equation (30.7) is accurate to within 0.1% for temperatures in the range −30 to 43 °C for most pressures at sea level. Should atmospheric pressure, p_{s} be known then the following expression can be used

$$c_{\mathrm{P}} = 20.05\sqrt{\frac{T_{\mathrm{C}} + 273.16}{1 - 3.79\times10^{-3}\left(\frac{h_{\mathrm{r}}p_{\mathrm{sat}}}{p_{\mathrm{s}}}\right)}}\,\mathrm{ms}^{-1}\,, \tag{30.8}$$

where the saturation pressure of air, p_{sat} is dependent on temperature as follows

$$\begin{aligned}\log_{10}\left(\frac{p_{\mathrm{sat}}}{p_{\mathrm{s0}}}\right) &= 10.796\left[1 - \left(\frac{T_{01}}{T}\right)\right] - 5.0281\log_{10}\left(\frac{T_{01}}{T}\right) \\ &\quad + 1.5047\times10^{-4}\left(1 - 10^{-8.2927\left[\left(\frac{T}{T_{01}}\right)-1\right]}\right) \\ &\quad + 0.42873\times10^{-3}\left(-1 + 10^{4.7696\left[1-\left(\frac{T_{01}}{T}\right)\right]}\right) \\ &\quad - 2.2196\,,\end{aligned} \tag{30.9}$$

p_{s0} is the reference atmospheric pressure of 101.325 kPa and T_{01} is the triple-point isotherm temperature with the exact value of 273.16 K.

30.4 Waveforms

Sonars employ a variety of waveforms, the most common types of which are shown in Fig. 30.6. Each waveform can be considered the echo from a normally incident plane. Waveforms are classified as being narrow- or wide-band depending on their spectral bandwidth. Narrow-band pulses provide superior detection performance in the presence of additive noise, while wide-band pulses provide better range resolution and do not have side-lobes.

Figure 30.6a shows the waveform produced by the Murata 40 kHz piezoelectric transducer excited by an eight-cycle 40 kHz square wave 40 V_{rms}. The Murata sensor is small, lightweight, and efficient, but has an approximately 90° beam width. These transducers are used in monostatic, bistatic, and multiple transducer arrays [30.8, 9].

The next three waveforms were produced by the Polaroid 600 electrostatic transducer. Similar waveforms are generated by the smaller Polaroid 7000 transducer. Figure 30.6b shows the waveform produced by the 6500 ranging module. This ranging module with its 10 m range, low cost, and simple digital interface is a popular choice for implementing sonar arrays and rings. While the electrostatic transducer is inherently wide-band, with a usable frequency range from 10 to 120 kHz [30.10], narrow-band pulses are produced by exciting the transducer with 16 cycles at 49.4 kHz. Figure 30.6c illustrates a means to exploit the wide bandwidth of the Polaroid electrostatic transducer by exciting it with a decreasing-frequency square wave. Such frequency-sweep pulses are processed by a band of band-pass filters to extract the frequency dependence of reflecting objects. A correlation detector, also know as a matched filter, compresses swept-frequency pulses to improve range resolution. Longer-duration (100 ms) pulses are used in CTFM systems. Figure 30.6d shows a wide-band pulse when the excitation is a 10 µs-duration 300 V pulse. The metal protective mesh, which also acts as a mechanical filter resonant at 50 kHz, was removed by machining to achieve a usable bandwidth from 10 to 120 kHz, with the peak occurring at 60 kHz. Such wide-band pulses are useful for object classification [30.10, 11]. These pulses have small amplitudes, limiting their range to 1 m or less.

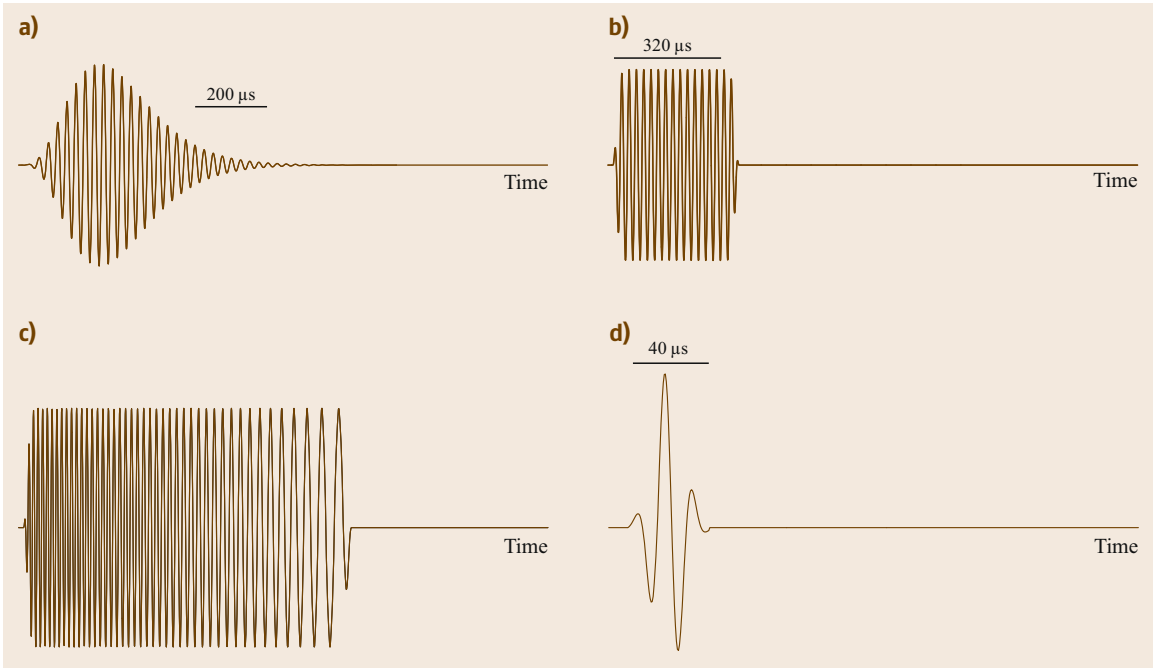


Fig.30.6a–d Common sonar pulse waveforms. **(a)** Murata 40 kHz transducer (narrow band). **(b)** Polaroid 600 electrostatic transducer excited with 16-cycle sinusoid at 49.4 kHz (6500 ranging module – narrow band). **(c)** Polaroid 600 electrostatic transducer excited with decreasing-frequency excitation signal (wide band). **(d)** Polaroid 600 electrostatic transducer excited with 10 µs 300 V pulse (wide band)

30.5 Transducer Technologies

Electrostatic and piezoelectric transducers are the two major types available that operate in air and can in principle operate both as a transmitter and receiver – some samples are shown in Fig. 30.7. In general electrostatic devices have a higher sensitivity and bandwidth but typically require a bias voltage above 100 V. Piezoelectric devices operate at lower voltages, making their electronic interfacing simpler, but have a high-Q resonant ceramic crystal and this results in a narrow frequency response compared to electrostatic transducers.

30.5.1 Electrostatic

An example of an electrostatic transducer is the Polaroid instrument-grade transducer (now available from SensComp.com) constructed from a gold-coated plastic foil membrane stretched across a round grooved aluminium back plate. The conductive foil is charged via a bias voltage of 150 V with respect to the back plate. Incoming sound waves vibrate the foil and change the average distance between the foil and back plate and thereby changing the capacitance of the foil. Assuming that the charge q is constant, the voltage $v(t)$ is generated proportional to this varying capacitance $C(t)$ as $v(t) = qC(t)$. As a transmitter, the transducer membrane is vibrated by applying 0–300 V pulses across this capacitor, typically using a pulse transformer. The

Fig. 30.7 *Left to right*: Series 9000, instrument-grade, and Series 7000 transducers; front and back views are shown (courtesy Acroname, Inc., Boulder)

charge induced by the 300 V on the capacitor causes an electrostatic attraction force between the membrane and the back plate. The grooves on the back plate allow stretching of the membrane and by creating randomness in the back plate roughness a broad resonance can be achieved in the frequency response. For example the bandwidth of the 7000 series Polaroid transducer is 20 kHz. A front grille is mounted on the transducer and removing this grille reduces losses and reverberation between the grill and the membrane. Another electrostatic transducer was designed by Kay and details of its design can be found in [30.12].

30.5.2 Piezoelectric

Piezoelectric ceramic transducers can be used as both transmitters and receivers, however some manufacturers sell transmitters and receivers separately in order to optimize the transmitted power and receiver sensitivity, respectively. A piezoelectric resonant crystal mechanically vibrates when a voltage is applied across the crystal, and in reverse generates a voltage when mechanically vibrated. Often a conical concave horn is mounted on the crystal to acoustically match the crystal acoustic impedance to that of air. An example is the Murata MA40A5R/S receiver and sender transducers, which operate at 40 kHz. This device has a diameter of 16 mm and a 60° beam angle for transmitter combined with receiver for −20 dB loss compared to the maximum sensitivity. The effective bandwidth of transmitter and receiver is only a few kHz due to the resonant nature of the crystals. This limits the envelope rise time of pulses to around 0.5 ms. An advantage is the ability to drive piezoelectric devices with low voltages, for example by connecting each terminal to complementary CMOS logic outputs. There is a wide range of resonant frequencies for piezoelectric transducers from 20 kHz to megahertz. Also available is piezoelectric film called polarized fluoropolymer, polyvinylidene fluoride (PVDF) from [30.13]. This flexible film can be cut to shape and custom ultrasonic transmitters and receivers can be formed. The sensitivities of the transmitters and receivers made from PVDF is generally lower than that of ceramic crystal transducers and most applications are short range where the broadband nature of PVDF allows short pulses to be formed, allowing pulse-echo ranging to as little as 30 mm.

30.5.3 MEMS

Microelectromechanical system (MEMS) ultrasonic transducers can be fabricated on a silicon chip and integrated with electronics. The sensors offer a low-cost mass-produced alternative to standard transducers. MEMS ultrasonic transducers operate as electrostatic capacitive transducers where the membrane can be made from thin nitride. Devices operate at frequencies up to several megahertz and offer advantages in signal-to-noise ratio over piezoelectric devices due to their better matching to air acoustic impedance [30.14]. Two-dimensional arrays of devices can be deployed on a chip that are well matched and steerable.

30.6 Reflecting Object Models

Modeling the reflection processes helps in interpreting echo information. In this section we consider three simple reflector models: planes, corners, and edges, shown in Fig. 30.8. These models apply to both single transducers and arrays.

A plane is a smooth surface that acts as an acoustic mirror. Smooth walls and door surfaces act as planar reflectors. The plane must be sufficiently wide to produce the two reflections whose path is shown in dotted line. The plane reflector is then slightly larger than the intersection area of the beam with a plane of infinite extent. Smaller planes produce weaker echoes because of a smaller reflecting surface and negative interference by echoes diffracted from the edges of the plane. An acoustic mirror allows the analysis using a virtual transducer, indicated by primes in the figure.

A corner is the concave right-angle intersection of two surfaces. Corners formed by intersecting walls, the sides of file cabinets, and door jambs are commonly observed corner reflectors in indoor environments. The novel feature of the corner, and its three-dimensional (3-D) counterpart the corner cube, is that waves reflect back in the same direction from which they originate. This is caused by planar reflections at each of the two surfaces defining the corner. The virtual transducer is then obtained by reflecting the transducer about one plane of the corner and then the other plane. This gives rise to a reflection through the intersection point of the corner as shown in Fig. 30.8b. The virtual transducer analysis indicates that, for a monostatic sonar, echoes from a plane and corner are identical and that planes and corners can generate identical sonar maps [30.4]. The difference in the virtual transducer orientation between planes and corners has been exploited using transducer arrays to differentiate these reflectors [30.11, 15].

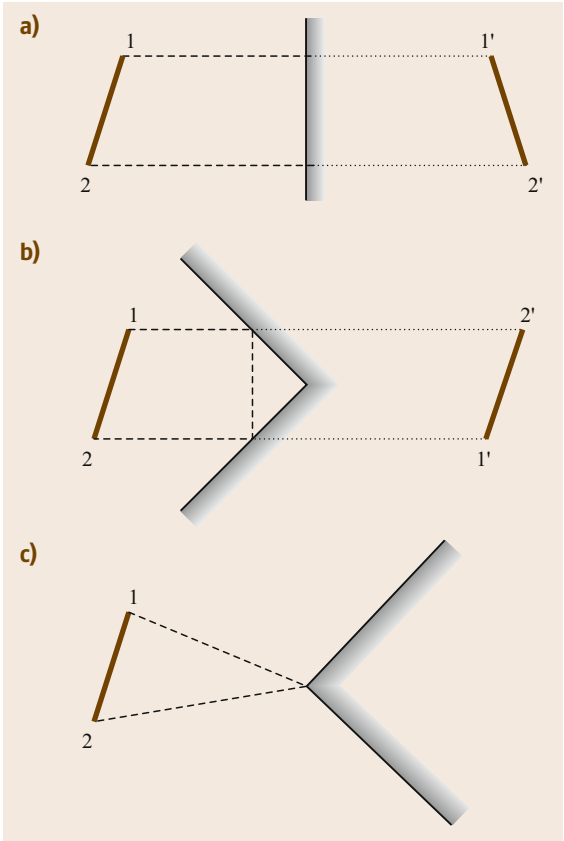


Fig.30.8a–c Reflector models: **(a)** plane, **(b)** corner, and **(c)** edge

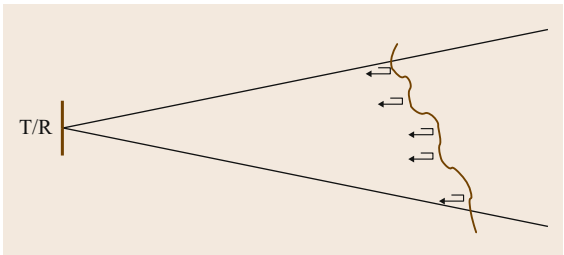


Fig. 30.9 Random reflector model. Echoes reflect back from normally incident surface sections within the beam

The edge shown in Fig. 30.8c models physical objects such as convex corners and high-curvature surfaces (posts), where the point of reflection is approximately independent of transducer position. Edges are encountered in hallways. While planes and corners generate strong echoes, edges generate weak echoes that are detected only a short range [30.4], making them difficult objects to detect. Early robot sonar researchers placed bubble wrap material on edge surfaces to make them reliably detectable.

Many environmental objects can be configured as a collection of planes, corners, and edges. Models for echo production [30.16, 17] indicate that normally incident surface patches and locations at which sharp changes in the surface function and its derivatives generate echoes. Objects with rough surfaces or a collection of many objects generate echoes from a variety of ranges and bearings, as illustrated in Fig. 30.9. If $p(t)$ represents a single echo waveform, often a replica of the probing waveform, the total echo waveform $p_T(t)$ is the sum of individual echoes $p_i(t)$ from N normally incident patches at range r_i and bearing θ_i, scaled by amplitude a_i, or

$$p_T(t) = \sum_{i=1}^{N} a_i(\theta_i) p_i\left(t - \frac{2r_i}{c}\right), \tag{30.10}$$

where $a_i(\theta_i)$ is an amplitude factor related to the surface patch size and its bearing in the beam. Wide-bandwidth echoes are more complicated because their waveform changes in a deterministic fashion due to diffraction [30.11].

Sonars that analyze $p_T(t)$ employ analog-to-digital converters to obtain waveform samples [30.11, 18]. Reflecting patches separated in range produce isolated patches [30.11], but more often the incremental travel time is less than the pulse duration, causing pulse overlap. Rough surfaces and volume scatterers, such as indoor foliage, have large N, allowing $p_T(t)$ to be treated as a random process [30.19, 20]. Conventional TOF sonars output the first time that $p_T(t)$ exceeds a threshold [30.11].

30.7 Artifacts

Sonars usually work well in simple environments, while complex environments often produce mysterious readings, artifacts, that foil attempts to build reliable sonar maps. Artifacts have given sonar a bad reputation as being a noisy, or low-quality, sensing modality. Sonar stalwarts believe sonar would open up many new applications, *if only* we understood echoes at a level that approximates that employed by bats and dolphins [30.21]. Sonar stalwarts divide into two categories in terms of how they treat artifacts. The first attempts to build *intelligent sensors* that identify and suppresses artifacts before transmitting data to a higher-level reasoning pro-

gram. Previous approaches [30.22, 23] required custom electronics, which other researchers have been reluctant to adopt because of expense or lack of experience. An alternate approach is to control conventional sonar in a novel fashion to produce a series of spikes and requires only software changes [30.24]. Sonar arrays have been used to find consistent data [30.25–27]. Echoes from specular reflectors, such as planes, corners, or posts, exhibit detectable features, which can be obscured by artifacts.

The second category of sonar users attempts to eliminate artifacts produced by conventional sensors by using higher-level postprocessing. These include proponents of occupancy (or certainty) grids [30.28, 29], including those that apply simplified physical models, such as sonar arcs [30.30, 31]. In simple environments postprocessing usually eliminates artifacts that are inconsistent with a feature [30.32] or with a physical map [30.31]. More-sophisticated methods handle artifacts by treating them as noise and applying hidden Markov models (HMM) [30.33]. However, multiple passes are needed to successfully teach the system about relatively simple environments, mostly because artifacts are not amenable to being treated as independent additive noise. Eliminating troublesome artifacts would replace HMM with simpler Markov chains [30.34, 35], and sufficient sonar data can be obtained in a single pass. What frustrates this second category, and mildly amuses the first, is that this postprocessing works well *in simple environments*, but fails in real-world environments. This second category eventually abandons sonar and joins the camera and laser ranging crowd.

There are two important classes of artifacts: axial multiple reflection (MR) artifacts and dynamic artifacts. These artifacts are important in sonar mapping when they indicate the presence of a static object at a location where none exists. Troublesome MR artifacts are caused by delayed echoes produced by a previous probing pulse exceeding the detection threshold after the current probing pulse has been transmitted. Such artifacts then appear as close-range objects and obscure actual farther-range objects in conventional sonars. Most sonars employ probing pulse emission periods longer than 50 ms to avoid MR artifacts, although some reverberant environments can still produce artifacts [30.36].

Dynamic artifacts are produced by moving objects, such as individuals passing through the sonar beam. Even though these are actual objects and echoes indicate their true range, their presence should not be part of a sonar map that describes the static environment. Such dynamic artifacts make quantitative matchings between stored and generated sonar maps error-prone.

Another common artifact is a nonaxial MR artifact [30.4] caused by an obliquely incident smooth surface that redirects the sonar beam to some other echo-producing object. The TOF produces a range reading that is positioned along the sonar axis. While the object is not at the location indicated on a sonar map, its location in the sonar map is a stable element and can be useful for navigation.

One may argue that, if the locations of all objects are known, the echoes can be determined and should not be treated as random processes. However, the presence of speed fluctuations in the medium due to thermal gradients and ever-present electronic noise cause random fluctuations in the times that thresholds are exceeded. Even a stationary sonar in a static environment exhibits random fluctuations [30.37], similar to the visual experience of fading when viewing objects beyond a heated surface.

Sonar can identify artifacts by applying three physical criteria that are met by echoes from static environmental objects. Artifact features include [30.36]:

1. Echo amplitude – echoes with amplitudes less than a specified threshold
2. Coherence – echoes forming constant-range azimuthal intervals less than a specified threshold, and
3. Coincidence – echoes detected with a sonar array at different times (lacking temporal coherence) or corresponding to different locations (lacking spatial coherence).

30.8 TOF Ranging

Most conventional sonars employ Polaroid 6500 ranging modules [30.38] connected to the Polaroid 600 series electrostatic ultrasound transducer. The module is controlled with digital signals on two input lines (*INIT* for initialization and probing pulse transmission and *BLNk* for clearing the indication and resetting the detector) and the TOF reading occurs on its output line (*ECHO*). A logic transition on *INIT* causes the transducer to emit a pulse lasting for 16 cycles at 49.4 kHz. The same transducer detects echoes after a short delay to allow transmission transients to decay. Another interrogation pulse is typically emitted only after all the echoes produced by the previous pulse have decayed below a detection threshold.

The module processes echoes by performing rectification and lossy integration. Figure 30.10 illustrates a simulation of the processed waveform applied to the threshold detector. While the echo arrives at time t_0 after the emission, *ECHO* exhibits a transition at the measured TOF time t_m, the first time the processed echo signal exceeds a detection threshold τ. By convention, the range r of the reflecting object is calculated by

$$r = \frac{ct_m}{2}\,, \tag{30.11}$$

where c is the speed of sound in air, usually taken as 343 m/s.

Figure 30.10b shows details around the threshold detection point including the residual high-frequency ripple after full-wave rectification and integration. Two effects can be noticed. First, t_m will always occur after t_0, making threshold detection a biased estimate of the true echo arrival time. Moreover, this bias is related to the echo amplitude: stronger echoes will produce an integrator output having a greater slope, which exceeds τ sooner than t_m. Second, as the echo amplitude decreases, for example, when the object moves away from the transducer axis, the threshold level occurs later in the integrator output and t_m will experience small jumps in time approximately equal to half the period [30.39].

The first step in developing a model of the detection process is to develop a model for the echo amplitude as a function of bearing. The Polaroid transducer is often modeled as a vibrating piston to yield the transmitter/receiver beam pattern shown in Fig. 30.11. To simplify the analysis, the peak of the beam profile is approximated with a Gaussian function, a parabola in logarithmic units in Fig. 30.11, to determine the echo amplitude as a function of object bearing θ, or

$$A_\theta = A_0 \exp\left(-\frac{\theta^2}{2\sigma^2}\right), \tag{30.12}$$

where A_0 is the on-axis amplitude and σ is a measure of the beam width. The value $\sigma = 5.25°$ provides a good fit around the peak of the beam pattern. The Gaussian model is reasonable only over the central section of the main lobe that produces detectable echoes.

We assume that the echo arrival time t_0 does not change significantly with transducer orientation (object bearing); this effect was investigated and found to be minor [30.4]. In contrast, measured TOFs, denoted by t_m and t'_m in Fig. 30.12, are amplitude dependent and a function of object bearing, which affects echo amplitude as shown in Fig. 30.11.

The module processes the detected echo waveform by rectification and lossy integration, as discussed above. To derive a useful analytic model, assume that the integration is lossless and the rectified echo is a unit step function with amplitude A. This approximates the processed waveform shown in Fig. 30.10b by a linear function around time t_m, shown in Fig. 30.12. The model ignores the residual ripple and the decreasing slope of the waveform as the lossy rectification approaches a constant value, shown in Fig. 30.10. The linear function with a slope proportional to the echo amplitude is given by $A_\theta(t - t_0)$, for $t \geq t_0$. This function

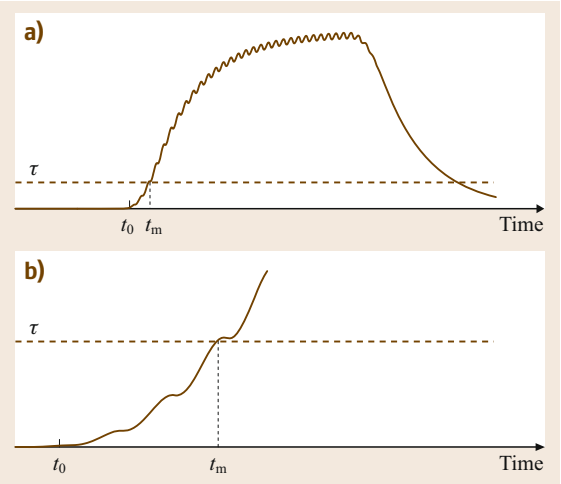


Fig.30.10a,b Simulation of Polaroid ranging module operation. **(a)** Processed echo waveform. **(b)** Expanded time and amplitude scale around threshold crossing point

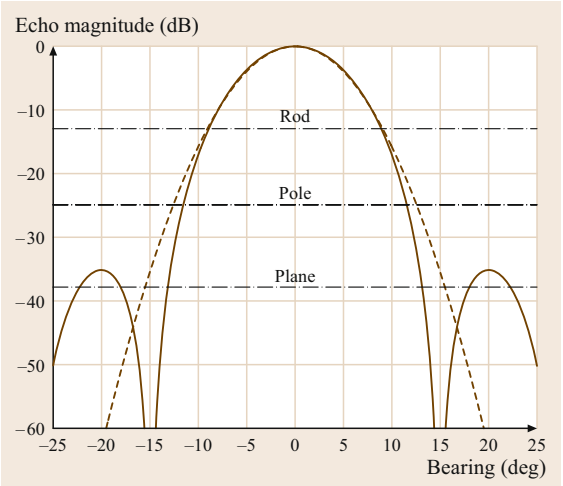


Fig. 30.11 Piston model transmitter/receiver pattern for the Polaroid 600 series transducer. A Gaussian approximation with SD = 5.25° is shown in *dashed line*. Equivalent threshold levels for plane, pole, and rod objects at 1.5 m range are shown as *dot-dashed lines*

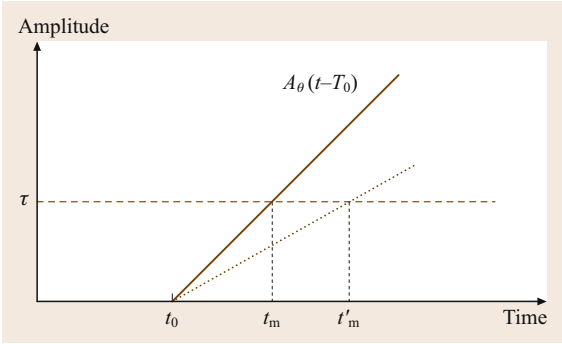


Fig. 30.12 TOF values t_m and t'_m for idealized processed echo waveforms having two amplitudes. The *solid line* indicates a larger-amplitude echo

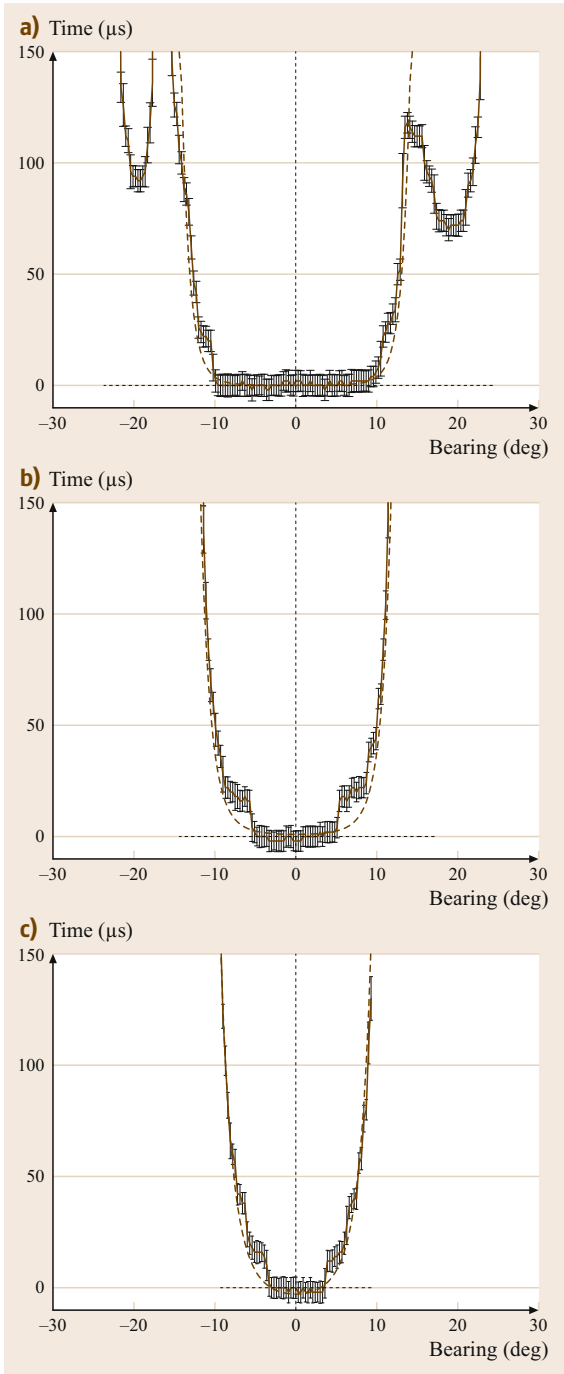


Fig.30.13a–c TOF data from an object at 1.5 m range; mean of 100 measurements with bars indicating ±1 SD. The *dashed lines* are model predictions. **(a)** 1 m wide plane ($\tau/A = 0.15\,\mu s$). **(b)** 8.9 cm diam pole ($\tau/A = 0.67\,\mu s$). **(c)** 8 mm diam rod ($\tau/A = 2.68\,\mu s$) ►

exceeds the threshold τ at

$$t_m = t_0 + \frac{\tau}{A_\theta} = t_0 + \frac{\tau}{A_0}\exp\left(\frac{\theta^2}{2\sigma^2}\right). \qquad (30.13)$$

For fixed τ, the incremental delay in t_m is a function of the bearing θ and inversely proportional to the echo amplitude. When a constant echo amplitude A (in volts) is applied to the integrator, the slope of the linear output is A V/s, with typical values on the order of $A_\theta = 10^5$ V/s. If $\tau = 0.10$ V, $\tau/A_\theta = 10^{-6}$ s $= 1\,\mu s$.

Experiments were conducted with a Polaroid 600 series transducer connected to a model 6500 ranging module [30.39]. The Polaroid module was operated conventionally to generate t_m values as a rotational scan was performed. Objects include a 1 m-wide plane, a 8.9 cm-diameter pole, and an 8 mm-diameter rod, all located at 1.5 m range. A rotational scan was performed from $-40°$ to $+40°$ in $0.3°$ steps. At each angle, 100 t_m values were recorded. The mean deviations from the t_m when the object is on the sonar axis ($\theta = 0$) were determined and the standard deviation (SD) values were computed. There were no other objects in proximity to the object being scanned. Echoes from objects beyond 2 m were eliminated by a range gate.

Figure 30.13a shows the data for the plane, Fig. 30.13b for the pole, and Fig. 30.13c for the rod. The values are shown relative to the t_m value observed at $0°$ bearing. Dashed lines indicate the values predicted by the model. The t_m values showed a variation with SD $= 5\,\mu s$ (0.9mm) at zero bearing, which is about nine times greater than that predicted by sampling jitter alone. This random time jitter is caused by dynamic thermal inhomogeneities in the air transmission medium, which change the local sound speed and cause refraction [30.37, 40]. The SD increases with deviation from zero bearing because smaller echoes exceed the threshold later in the processed waveform. The smaller

slope of the latter part of the processed echo waveform shown in Fig. 30.10 causes greater t_m differences for a given variation in echo amplitude, thus increasing the SD.

One feature in the data not described by the model is due to residual ripple in the integrator output, which causes jumps in TOF readings equal to half periods (10μs) added to the value predicted by (30.13). These jumps are clearly evident in the mean values of Fig. 30.13.

The angular extent over which echoes were detected was 45° for the plane, 22.8° for the pole, and 18.6° for the rod. Side-lobes produced by the plane are visible and have small echo amplitudes, which cause their t_m values to be retarded in time. These angular extents can be related to the echo amplitudes that would have produced the respective arcs according to the piston model; these are indicated in Fig. 30.11. For the plane, the threshold level relative to the maximum echo amplitude is −38 dB, for the pole −25 dB, and for the rod −13 dB. Since the ranging module threshold at 1.5 m range is the same for each object, the difference in levels indicates the relative echo strength from each object, i. e., the plane echo is 13 dB (a factor of 4.5) greater than the pole echo, and the pole echo is 12 dB (a factor of 4) greater than the rod echo.

30.9 Echo Waveform Coding

Systems that display echo information beyond the first echo have been investigated [30.11, 18, 25, 41–43], but typically employ custom electronics. One motivation for examining the entire echo waveform is the success of diagnostic medical ultrasound imaging systems, which adopt this method [30.44, 45].

As a less expensive alternative to analog-to-digital conversion, the Polaroid ranging module can detect echoes beyond the initial echo by repeatedly resetting the detection circuit. The 6500 module specification suggests a delay before resetting to prevent the current echo from retriggering the detection circuit [30.3]. Let us ignore this suggestion and control the Polaroid module in nonstandard way to provide information about the entire echo waveform. Since the echo amplitude is estimated from the digital output produced by the Polaroid module, this operation has been called *pseudo-amplitude scan* (PAS) sonar [30.24].

The conventional ranging module processes detected echoes by performing rectification and forming a lossy integration, as illustrated in Fig. 30.14a.

The *BLNK* input is typically kept at zero logic level, which enables the *ECHO* output. *ECHO* exhibits a transition at the time when the processed echo signal exceeds a threshold, as shown in Fig. 30.14b. By convention, the time interval between the *INIT* and *ECHO* transitions indicates the time of flight (TOF), from which the range r of the reflecting object is calculated by

$$r = \frac{c \times \mathrm{TOF}}{2} \,. \tag{30.14}$$

Echoes occurring after the initial echo can be detected by resetting *ECHO* by pulsing the *BLNK* input. The specification suggests that the *BLNK* pulse should be delayed after the *ECHO* indication by at least 440 μs to account for all 16 returning cycles in the echo and to allow it to decay below the threshold for the largest observable echo. The largest echoes typically saturate the detection circuit, providing a predetermined maximum value. This duration corresponds to the time interval over which the processed signal is above the threshold, as shown in Fig. 30.14a.

When an *ECHO* event is observed, the PAS system issues a short 3 μs (corresponding to a software query period) pulse on the *BLNK* input line, which clears the *ECHO* signal as shown in Fig. 30.14c. Upon being cleared, the Polaroid module exhibits a delay inversely related to the echo amplitude, lasting at least 140 μs for large-amplitude echoes, and then produces

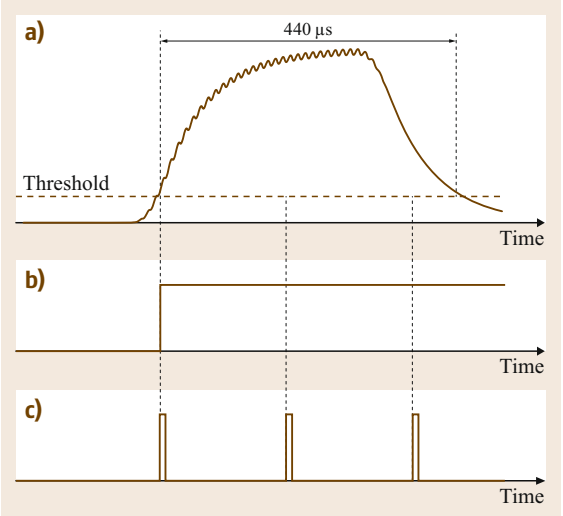


Fig.30.14a–c Polaroid ranging module operation modes. **(a)** Processed echo waveform. **(b)** *ECHO* output produced in conventional time-of-flight mode. **(c)** *ECHO* output produced in PAS mode

another *ECHO* event if the processed echo signal still exceeds the threshold. The PAS system repeatedly issues a *BLNK* pulse whenever an *ECHO* event is observed. Hence, a strong echo is represented by three pulses on the *ECHO* line, the first corresponding to the conventional TOF, followed by two more pulses. Because lower amplitude echoes spend less time above the threshold, a weaker echo produces two pulses spaced farther apart, and a very weak echo may produce only one pulse. Shorter *INIT* pulses discharge the integrator by a lesser amount and thus increase the number of pulses produced by the standard Polaroid emission [30.46, 47].

A PAS sonar map is generated by placing a range dot along the transducer axis as a rotational scan is executed. With multiple readings per interrogation pulse, a PAS sonar map contains multiple dots at each interrogation angle. Rotational scans then form arcs, with isolated arcs indicating weak echoes, arc pairs moderate echoes, and arc triplets large echoes. To illustrate, Fig. 30.15 shows arcs formed by a large plane (2.3 m width by 0.6 m height) and five cylinders with different diameters, all placed at 1 m range. Examining objects at the same range eliminates effects caused by the range-dependent gain of the module.

A conventional TOF sonar map by comparison would display only the nearest arc in the PAS map for each object. Qualitatively, the arc length increases and the number of arcs increase with the echo amplitude, which is bearing dependent. The strongest reflectors produce concave arcs [30.4, 48]. This occurs because, with echo amplitudes much greater than the threshold, the threshold is exceeded near the beginning of the echo, yielding a nearly constant range reading over a significant extent over bearing. In contrast, the weakest reflectors produce convex arcs, caused by echoes whose amplitudes are comparable to the threshold. As the echo amplitude decreases the threshold is exceeded at later points along the processed waveform, producing greater range readings. This effect also appears at the edges of the arcs produced by strong reflectors.

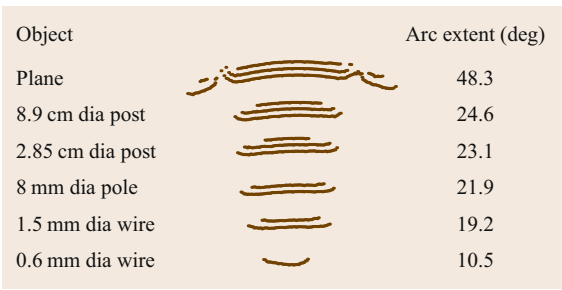


Fig. 30.15 PAS sonar maps of six objects located at 1 m range. The sonar is located below the objects in the figure

Computing the beam pattern of the vibrating piston model [30.1], which is a reasonable approximation to the Polaroid transducer, yields the curve shown in Fig. 30.16.

This figure describes the detected echo magnitude normalized to have a maximum of 0 dB, which occurs along the beam axis. The larger echoes are much greater than the threshold, such as those produced by the plane, whose maximum amplitudes can be 44 dB relative to the threshold. The −44 dB threshold agrees with the PAS map for a plane: strong echoes (three stripes) occur within 10° of normal incidence, range readings increase due to echo amplitude reduction at ±15.6°, approximating the predicted nulls at ±14.7°, and smaller-amplitude echoes from the sidelobes are present. Weaker reflectors correspond to larger thresholds when their on-axis echoes normalize to 0 dB. The beam pattern model explains how arc length varies with object reflecting strength. The indicated thresholds were found by matching the angular beam width to the arc extent.

It is apparent that PAS maps provide information useful for solving the inverse problem, that of determining the identity of the object from the echoes. Figure 30.15 shows that PAS maps contain information about the echo amplitude. While it is true that the conventional TOF sonar maps, represented by the closest arc, can determine the object location from the arc center and can infer the echo amplitude from the arc extent, it is also true that this information is presented in a more robust way in the PAS maps. For this simple case of isolated objects, the posts can be clearly differentiated from the pole, while the corresponding conventional TOF arcs are comparable. A tenfold in-

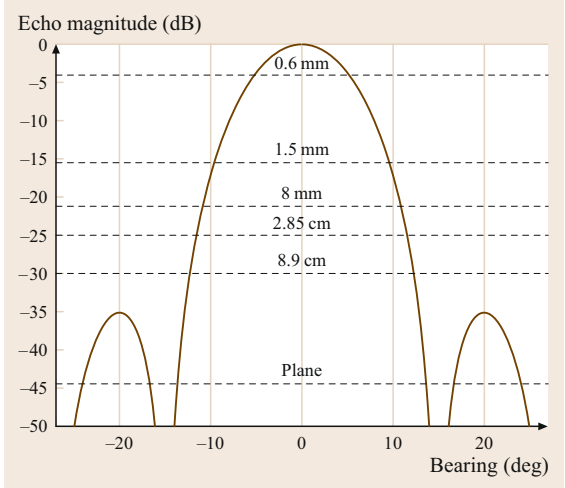


Fig. 30.16 Transmitter–receiver beam pattern. *Dashed lines* indicate the equivalent threshold level for each object

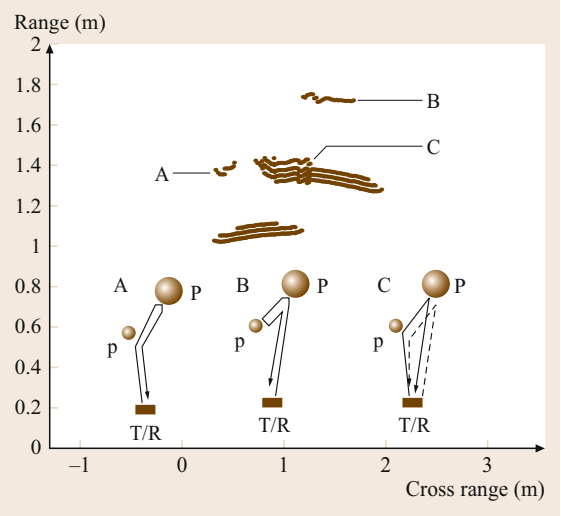


Fig. 30.17 PAS sonar map of 2.85 cm-diameter post (p) and 8.9 cm-diameter post (P). The transducer is located at (0,0). A: Artifact caused by echo originating at transmitter T, reflected by p, bouncing off P back to p, and directed toward receiver T → p → P → p → R. B: T → P → p → P → R. C: T → p → P → R and T → P → p → R

crease in post diameter yields only a modest increase in the conventional TOF arc length, while increasing the number of arcs from two to three in the PAS maps. Spike data can be processed to produce sonar images of the environment that are similar to diagnostic ultrasound B-scans [30.49]. VIDEO 315 shows a sonar B-scan of a tree.

When examining the entire echo waveform, one must account for artifacts that are produced when objects interact acoustically. Some artifacts occur after the first detected echo, so these are not a problem in TOF sonar maps [30.4], but must be addressed in interpreting PAS maps. Consider a simple environment consisting of two posts: a 2.85 cm-diameter post (p), located at $r = 1$ m and bearing 12°, and an 8.9 cm-diameter post (P), at $r = 1.3$ m and bearing −10°. The corresponding PAS map shown in Fig. 30.17 displays the echoes from the two objects plus additional echoes that illustrate two types of multiple reflection artifacts.

The first type, indicated by A and B, results when only one object is within the transducer beam. An interrogation pulse that is redirected by a reflected object must be directed back to the receiver within its beam pattern in order to be detected. The paths that do this are shown in the figure. The single-arc convex shape of A indicates that the echo has a small amplitude. This is reasonable since both reflectors are nonplanar, and hence weak.

The second type of artifact (C) shown in Fig. 30.17 occurs when both objects are within the beam pattern. This allows two distinct paths for the echoes to return to the receiver, occurring in opposite directions and doubling the artifact amplitude. With both objects lying near the beam edges, the echo amplitude is small. Since the distance traveled by these echoes is slightly greater than the range to the farther object, this artifact shows a range slightly beyond the more distant object. The superposition of these two components makes the echo from the farther object appear spread out in time. This pulse stretching explains why four arcs are observed, and at one angle five arcs. If the bearing angle between p and P was increased to exceed the beam width, this artifact would disappear.

30.10 Echo Waveform Processing

In this section pulse-echo sonar that processes sampled digitized receiver waveforms is described. These systems offer superior performance over the simple Polaroid ranging module systems described above, which report the TOF based on a threshold. Echo waveform processing does however incur the overhead of more-complex electronics and signal processing and is not readily available commercially.

30.10.1 Ranging and Wide-Bandwidth Pulses

It is shown in [30.11, 50] that the maximum-likelihood estimator (MLE) for the TOF is obtained by maximizing the correlation $\mathrm{cor}(\tau)$ between the received pulse $p(t)$ (containing Gaussian white noise) and the known pulse shape shifted by τ, $\mathrm{rec}(t-\tau)$

$$\mathrm{cor}(\tau) = \frac{\int_a^b p(t)\mathrm{rec}(t-\tau)\,\mathrm{d}t}{\sqrt{\int_a^b p^2(t)\,\mathrm{d}t \int_a^b \mathrm{rec}^2(t)\,\mathrm{d}t}}\,, \qquad (30.15)$$

where the pulse extends from time a to b. The known pulse shape at the receiver depends on the angle of transmission and reception with respect to the normals of the respective transducers. The pulse shape can be obtained by collecting a good signal to noise pulse at 1 m range at normal incidence to the receiver and transmitter and using elliptical impulse response models to obtain template pulses at angles different to normal incidence. Pulse shape also changes with range due to the dispersive properties of absorption due losses in air

transmission. These can be modeled using an estimate of the impulse response due to one meter path through air as is done in [30.11].

The correlation, $\mathrm{cor}(\tau)$ is normalized in (30.15) to be between -1 and $+1$. The correlation at the maximum thus gives a good indication of the match between the expected and actual pulse shapes and can be used to assess the quality of the TOF estimate. In practice (30.15) is used in discrete time form, where the integrals are replaced by sums of products and digital signal processors are an ideal implementation since they are highly optimized to perform this calculation [30.51, 52]. Finding the maximum correlation of (30.15) is known as template matching. To achieve an arrival time estimator with resolution smaller than the discrete time sample rate, parabolic interpolation can be used on the maximum three correlations [30.11]. Of interest is the jitter standard deviation σ_R in the TOF estimator due to receiver noise. From [30.11, 50]

$$\sigma_\mathrm{R} = \frac{\sigma_n}{B\sqrt{\sum_k \mathrm{rec}(kT_\mathrm{s})^2}}\,, \tag{30.16}$$

where the summation index k is over the entire receiver pulse sampled every T_s seconds (1 —s in [30.11, 51]), B is the bandwidth of the receiver pulse, and σ_n is the standard deviation of the receiver noise. Equation (30.16) shows that broadband high-energy pulses achieve low errors in the TOF estimator. In [30.11] this is achieved by using a 300 V pulse to excite the transmitter and achieve close to the impulse response from the device with a pulse shape similar to that shown in Fig. 30.6d.

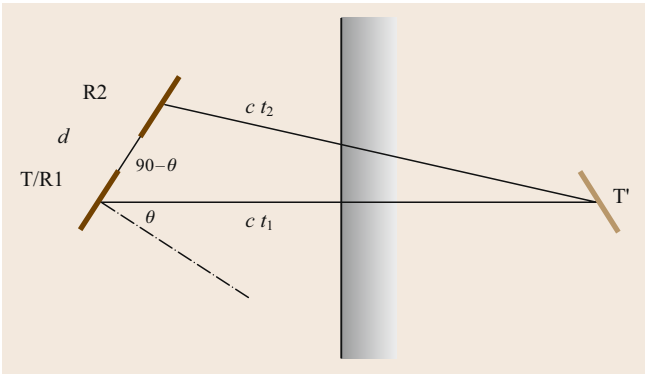


Fig. 30.18 Bearing (θ) calculation for a plane using a transceiver T/R1 and a receiver R2. T′ is the virtual image of T

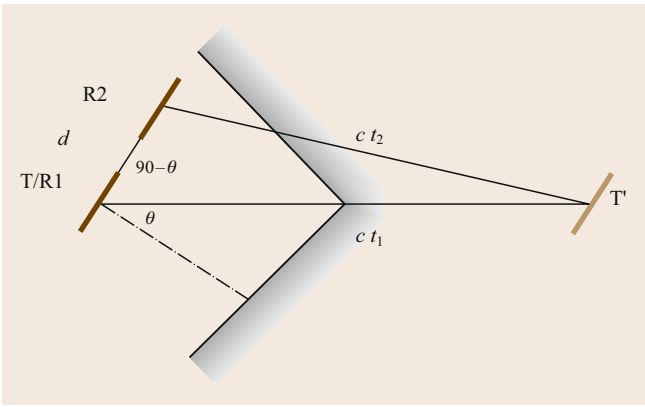


Fig. 30.19 Bearing (θ) calculation for a corner using a transceiver T/R1 and a receiver R2. T′ is the virtual image of T

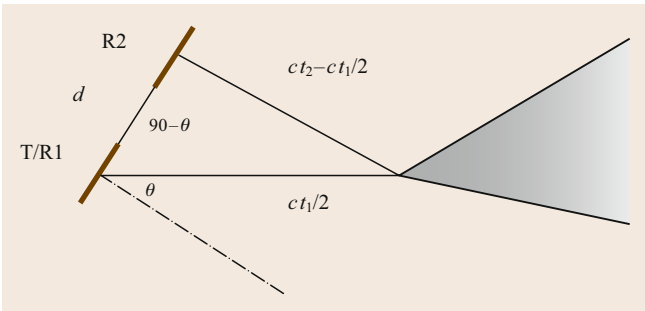


Fig. 30.20 Bearing (θ) calculation for an edge using a transceiver T/R1 and a receiver R2. No virtual image is present since the edge radiates from a point source at the edge

30.10.2 Bearing Estimation

There are many proposed methods for bearing estimation. A single transducer [30.53] can be used by exploiting the dependency of the received pulse shape on the angle of reception. This approach works for angles within one half of the beam width since the pulse shape is symmetric with respect to the transducer normal angle. Differences in zero-crossing times either side of the maximum amplitude of the pulse are used to obtain an accuracy of the order of 1°. Other single-receiver techniques rely on repeated measurements from a scan across the scene [30.54, 55] and achieve a similar level of accuracy but at much slower sensing speed since multiple readings are necessary.

Other single measurement approaches rely on two or more receivers [30.11, 12, 25]. This gives rise to a correspondence problem where data must be associated between the receivers. The closer the spacing between receivers, the simpler and more reliable the correspondence procedure. The misconception that bearing accuracy improves with larger receiver spacing ignores the correlation between measurement errors that can arise due to the measurements sharing an overlapping space of air in the propagation of the ultrasound. Due to the high accuracy of TOF estimation in [30.11] the receivers could be spaced as close as physically feasible (35 mm) and still bearing accuracies lower than any

other systems are reported. Standard deviations of bearing errors are reported to be below 0.2° for a plane at a range of 4 m within a −10° to +10° beam width.

There are two common approaches to bearing estimation – interaural amplitude difference (IAD) [30.56] and interaural time difference (ITD) [30.11, 25, 51–53, 56]. IAD uses two receivers pointing away from each other so that an echo has a different amplitude response in each receiver's beam width. In ITD both receivers usually point in the same direction and the TOF is measured on each receiver and triangulation is applied to determine the angle of arrival. The bearing calculation is dependent on the target type, such as a plane, corner or edge; these geometries are analyzed in [30.11]. A simple arrangement with a transceiver and receiver is shown in Fig. 30.18, where T/R1 is the transceiver and R2 is the second receiver, spaced a distance d from each other.

The virtual image of the transmitter is shown as T′. The two TOFs measured on the two receivers are t_1 and t_2, and these are used to estimate the bearing angle, θ, to the plane which is the angle to the plane normal. Applying the cosine rule to the triangle R2 R1 T′ in Fig. 30.18 gives

$$\cos(90-\theta) = \sin\theta = \frac{d^2 + c^2t_1^2 - c^2t_2^2}{2dct_1} . \qquad (30.17)$$

When $d \ll ct_1$, (30.17) can be approximated by

$$\sin\theta \approx \frac{c(t_1 - t_2)}{d} . \qquad (30.18)$$

Note that any common (i. e., correlated) noise in t_1 and t_2 is removed by the difference in (30.18) and hence the correlation in noise components of the TOF cannot be overlooked in bearing estimation as described above.

The situation for a corner is shown in Fig. 30.19 and the same result applies as in (30.17).

For an edge the situation is shown in Fig. 30.20, where R1 has a TOF from T to the edge and back to R1, whilst R2 has a TOF from T to the edge and back to R2.

From the geometry, we use the same approach as in (30.17) to give

$$\sin\theta = \frac{d^2 + \frac{c^2t_1^2}{4} - c^2\left(t_2 - \frac{t_1}{2}\right)^2}{2dc\frac{t_1}{2}} = \frac{d^2 + c^2t_2(t_1 - t_2)}{dct_1} . \qquad (30.19)$$

Note that (30.19) can be approximated by (30.18) when $d \ll ct_1$.

30.11 CTFM Sonar

The continuous-transmission frequency-modulated (CTFM) sonar differs from the more common pulse-echo sonar discussed in previous sections in the transmission coding and the processing required to extract information from the receiver signal.

30.11.1 CTFM Transmission Coding

The CTFM transmitter continuously emits a varying-frequency signal, usually based on a sawtooth pattern as shown in Fig. 30.21, where the frequency is often swept through an octave every sweep cycle T.

The transmitted signal with a linearly changing frequency can be expressed as

$$S(t) = \cos\left[2\pi\left(f_H t - bt^2\right)\right] \qquad (30.20)$$

for $0 \le t < T$. The sweep cycle is repeated every T seconds as shown in Fig. 30.21. Frequency is $1/2\pi$ times the time derivative of the phase in (30.20). Note that the highest frequency is f_H and the lowest transmitted frequency is $f_H - 2bT$, where b is a constant that determines the sweep rate. We can then define the swept frequency ΔF as

$$\Delta F = 2bT . \qquad (30.21)$$

30.11.2 CTFM TOF Estimation

Echoes are generated when the transmitted wavefront encounters reflectors and are an attenuated, delayed version of the transmitted signal

$$E(t) = AS\left(t - \frac{2R}{c}\right) , \qquad (30.22)$$

where R is the range to the reflector, c is the speed of sound and A is the amplitude that may in the case of curved objects depend on the frequency of the sound at reflection.

The TOF is estimated by the two step process of demodulation and spectral analysis. Demodulation is achieved by multiplying the received signal by a copy of the transmitted signal and low pass filtering. This can best be understood in the simple case of one echo. The

signal $D(t)$ is obtained using (30.20) and (30.22)

$$\begin{aligned} D(t) &= E(t)S(t) \\ &= \frac{A}{2}\big[\cos\left(2\pi f_e t - \phi\right) \\ &\quad + \cos\left(2\pi f_u t - 2bt^2 - \phi\right)\big] \end{aligned}$$

$$\text{for } f_e = \frac{4Rb}{c}\,,$$

$$f_u = \left(2f_H + \frac{4Rb}{c}\right),$$

$$\phi = f_H\frac{2R}{c} + \frac{4bR^2}{c^2}\,, \qquad (30.23)$$

where the following trigonometric identity has been used in (30.23)

$$\cos(x)\cos(y) = \frac{1}{2}[\cos(x-y) + \cos(x+y)]\,. \qquad (30.24)$$

A low-pass filter removes frequency components above f_H and this results in the base-band signal

$$D_b(t) = \frac{A}{2}\left[\cos\left(2\pi\frac{4Rb}{c}t - \phi\right)\right]\,, \qquad (30.25)$$

which has a frequency proportional to the range R. The ranges of echoes can be extracted by examining the spectrum of D_b using, for example, a discrete Fourier transform (DFT) or the fast Fourier transform (FFT). From (30.25) for a frequency peak of f_r Hz the corresponding range R is given by

$$R = f_r\frac{c}{4b}\,. \qquad (30.26)$$

Note that the above analysis relies on excluding the receiver waveform at the start of each sweep for a *blind time* (Fig. 30.21) of $R_m/2c$, where R_m is the maximum target range. During this blind time the receiver signal is dependent on the previous sweep rather than the current sweep as assumed in the analysis above. The sweep time T needs to be much larger than this blind time for the sonar to operate effectively. The blind time can be eliminated at the expense of introducing complexity in the demodulation process as described in [30.57], where an interlaced double demodulation scheme is described.

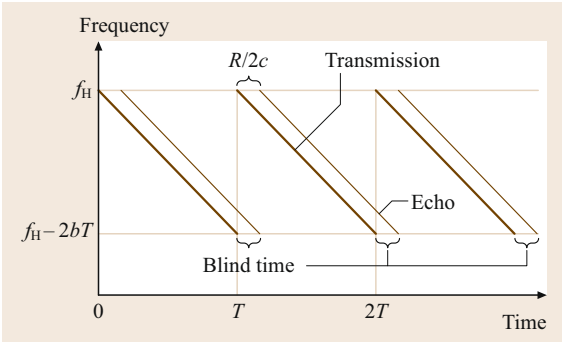


Fig. 30.21 18 Plot of frequency versus time for CTFM. The blind time applies if the shown echo corresponds to a maximum range target at R_m

30.11.3 CTFM Range Discrimination and Resolution

We define range discrimination as the separation in range of two targets that can be simultaneously detected as distinct. The range resolution is defined as the smallest increment in range that can be measured by the sonar.

Suppose that, in order to extract ranges of targets, $D_b(t)$ from (30.25) is sampled at ΔT intervals and k samples are collected before a DFT (or FFT) is performed. The frequency samples of the DFT will be $\Delta f = 1/(k\Delta T)$ apart. From (30.26), this represents a range resolution ΔR of

$$\Delta R = \frac{c\Delta f}{4b} = \frac{c}{4bk\Delta T}\,. \qquad (30.27)$$

We can relate this to the swept frequency ΔF from (30.21) as

$$\Delta R = \frac{c}{2\Delta F}\times\frac{T}{k\Delta T}\,, \qquad (30.28)$$

where the second term is the ratio of the sweep time to the spectral sample time. In order to discriminate two peaks in the DFT, they must be at least two samples apart and hence

$$\text{range discrimination} = \frac{c}{\Delta F}\times\frac{T}{k\Delta T}\,. \qquad (30.29)$$

Note that (30.28),(30.29)) show that, subject to signal-to-noise constraints, CTFM can lengthen the *data integration time* $k\Delta T$ in order to improve the range discrimination and resolution of the sonar. Also it is possible, subject to signal noise, to use interpolation techniques (e.g., parabolic interpolation) on the DFT peaks to resolve to a smaller than Δf frequency and hence improve range resolution (but not range discrimination).

30.11.4 Comparison of CTFM and Pulse-Echo Sonar

- The range resolution of pulse-echo sonar and CTFM sonar is theoretically the same given the same signal-to-noise ratios and bandwidths [30.57].

The range discrimination in pulse-echo sonar is limited by the pulse length, where shorter pulse lengths require higher bandwidth. However in CTFM, range discrimination can be improved by increasing the data integration time, allowing more design flexibility.

- CTFM also allows for the energy of the transmitted signal to be spread evenly over time, resulting in lower peak acoustic power emission compared to pulse-echo systems with the same receiver signal-to-noise ratio. CTFM can provide a greater average power in a practical context and consequently a greater sensitivity to weak reflectors is possible.
- CTFM requires more-complex transmitter circuitry and the requirement for FFT processing on the receiver side.
- Separate transmitter and receiver transducers are necessary with CTFM, whilst pulse-echo systems can use a single transducer for both transmission and reception, resulting in restriction on the minimum range of pulse-echo sonar due to the blanking of the receiver during transmission. CTFM has no inherent restriction on minimum range.
- CTFM sonar can continuously derive range information from targets every $k\Delta T$ seconds at a delay of $R/c + k\Delta T$ compared to every $2R_m/c$ with a delay of $2R/c$ in pulse-echo sonar (ignoring processing delays in both), which may be important in real-time tracking applications.
- Other benefits of CTFM are that the number of range measurements per cycle is limited only by the range discrimination constraint of (30.28) and the signal-to-noise ratio.
- In terms of bearing estimation and classification of targets from a moving platform, short pulse-echo sonar systems like [30.27, 51] do not suffer from the CTFM data integration time required to estimate accurately the frequencies corresponding to ranges (and hence bearing). During the data integration time, the target can move with respect to the sensor and blur the measurements, making bearing estimation and classification less accurate. In short pulse-echo systems, the target is effectively sampled with a pulse of less than 100 µs, resulting in a consistent snapshot of the target.

30.11.5 Applications of CTFM

Kay [30.58, 59] developed a mobility aid for blind people using a CTFM sonar system based on a sweep of $f_H = 100$ down to 50 kHz with a sweep period of $T = 102.4$ ms. After demodulation, ranges are heard as audible tones with frequencies up to 5 kHz corresponding to ranges of up to 1.75 m. The system uses one transmitter and three receivers as shown in Fig. 30.22.

Users of the system can listen to the demodulated signal in stereo headphones corresponding to the left and right receivers, each mixed with the large central oval receiver. Higher frequencies correspond to more-distant ranges. To illustrate the sensitivity, a 1.5 mm-diameter wire is easily detectable at 1 m range – the echo produced is 35 dB above the noise floor in the system.

CTFM sonar has been used to recognize isolated plants [30.41, 60]. The advantage gained from CTFM is that extensive range and echo amplitude information is obtained from the whole plant given the spectrum of the demodulated received signal, and these echoes are obtained from an excitation across an octave of frequencies from 100 down to 50 kHz with a high signal-to-noise ratio that allows weak reflections from leaves to be sensed. This information is called the *acoustic density profile* and 19 different features are found to be useful in classifying the plants, such as the number of range cells above a threshold in amplitude, the sum of all range cells, the variation about the centroid, the distance from the first to the highest amplitude cell, and the range over which reflections are detected. With a population of 100 plants, an average of 90.6% correct pairwise classification was obtained using a statistical classifier.

Fig. 30.22 Aid for blind people – the small oval transducer is the transmitter and the other three components are receivers. The large oval receiver provides high resolution, enabling fixation by users' fine neck control (after *Kay* [30.58])

Scanning CTFM with a singe transmitter and single receiver has been successfully applied to mapping of indoor environments that include smooth and rough surfaces [30.55] with bearing errors on the order of 0.5° for smooth surfaces and higher for edges. The classification uses amplitude information that is normalized with range using a fixed attenuation constant of sound. In practice this attenuation constant varies with temperature and humidity and needs to be calibrated before each experiment for consistent results. Greater robustness, speed, and accuracy has been demonstrated with TOF methods of classification that require at least two transmitter positions and two receivers as described in [30.11, 51]. CTFM could be applied to array systems to achieve higher sensitivity to weak targets than the existing pulse-echo systems.

CTFM has been employed in three binaural systems [30.12] where a rigorous theoretical and experimental comparison of these ultrasonic sensing systems based on different range and bearing estimators is made. *Stanley* [30.12] also contains detailed engineering design information of CTFM sonar systems. The conclusion is that CTFM can insonify large areas due to its higher average power transmissions and consequently good signal to noise performance. The use of autoregressive estimators for spectral lines in the demodulated signal were found to provide better resolution than the DFT. The interaural distance and power difference CTFM approaches provided state-of-the-art performance except that the pulse-echo approach in [30.11] using a high-energy short pulse was found to be a *factor of six to eight* times superior in bearing precision.

30.12 Multipulse Sonar

This section examines sonar systems that employ more than one pulse in the transmitter(s). The main motivations are interference rejection and on-the-fly classification. Multipulse sonar has also been used to generate a better signal-to-noise ratio by creating longer transmitted pulse sequence using Barker codes [30.61]. The autocorrelation of a Barker code gives a narrow peak with low autocorrelation away from the central lobe. The matched filter then gives rise to pulse compression that averages noise over a longer time period.

30.12.1 Interference Rejection

External acoustic noise, such as compressed air, is a source of sonar interference. Sonar systems attempt to reduce the effects of external interference by filtering the signal and the optimal filter is the matched filter where the impulse response is the time reversal of the pulse shape that is expected. Since a time-reversed convolution is a correlation, the matched filter then acts as a correlation with the expected pulse shape as discussed in Sect. 30.10. Approximations to matched filtering can be designed based on a bandpass filter with a frequency response that is similar to the spectrum of the expected receiver pulse. CTFM systems allow robust suppression of external interference by employing a matched filter across a broad range of frequencies contained in the continuous chirp transmission.

When more than one sonar system operates in the same environment, the transmitted signal from one sonar system can be received by another, causing crosstalk errors. This is particularly evident in classical sonar rings constructed from Polaroid ranging modules. Error-eliminating rapid ultrasonic firing strategies have been developed [30.62] and are claimed to remove most of this interference and allow faster operation of these sonar rings.

More-sophisticate coding of transmitted pulse(s) has been employed [30.23, 63–66] to allow rejection of external interference and crosstalk. One difficulty with multiple transmitted pulses over a greater time period than a single pulse is that target clutter can produce many overlapping pulses at the receivers that are difficult to unravel and interpret, and the sonar range discrimination can be compromised.

30.12.2 On-the-Fly Target Classification

Target classification into planes, cylinders, and edges has been achieved by deploying a single transmitter and three receivers [30.25] using a single measurement cycle. At least two transmitters are required to differentiate planes from concave right-angled corners [30.11] where a two transmitter arrangement is used to classify targets into planes, corners, and edges in two successive measurement cycles. The method of classification can be understood with virtual images and mirrors, since specular sonar reflections occur. Looking into a plane mirror gives an image that is left–right reversed compared to looking into a right-angled mirror. An edge is analogous to observing a high-curvature specular surface, such as a polished chair leg, where the whole image is compressed to a point. Sonar classification exploits the difference in bearing angles to a target from

two transmitters to classify as follows: a positive difference δ indicates a plane, a negative δ value indicates a corner, and zero difference an edge, where the angle δ depends on the sensor geometry and target range. More sophistication can be added by using range measurements in addition to bearing, with maximum-likelihood estimation.

This arrangement [30.11] was refined to work with just one measurement cycle of around 35 ms to 5 m range: hence the term *on-the-fly* in [30.51]. This on-the-fly approach uses pulses fired at a precise time difference ΔT and 40 mm apart from two transmitters with two further receivers completing a square. ΔT is usually around 200 µs but can vary randomly from cycle to cycle to achieve interference rejection (both crosstalk and environmental) with identical sonar systems. Classification is performed simultaneously in one measurement cycle. The sensor achieves high accuracy in range and bearing with robust classification by exploiting the tight correlation between TOF jitter in the different transmitter to receiver paths due to the close temporal and spatial arrangement. The sensor has been deployed for large-scale mapping in [30.67].

30.13 Sonar Rings and Arrays

Since sonar only detects objects lying within its beam, a common means to scan the entire environment outside the robot is to use an array of sonars, or a *ring* [30.68].

30.13.1 Simple Ranging Module Rings

The most common is the Denning ring that contains 24 sonars equally spaced around the robot periphery. This 15° spacing allows some overlap in the sonar beams so at least one of the sonars will detect a strong reflecting object. The sonars in the ring are typically employed sequentially one at a time. Using a 50 ms probing pulse period to reduce false readings a complete environmental scan is accomplished every 1.2 s. This sample time is adequate for a translate-and-stop operation in research settings, but may be too slow for a continually moving robot. A robot moving at 1 m/s may not detect an object with sufficient warning to prevent a collision. Some researchers propose simultaneously employing sonars on opposite ends of the ring to speed up acquisition times, while others also reduce the probing pulse period and attempt to identify artifacts.

30.13.2 Advanced Rings

Yata et al. [30.53] developed a 32 cm-diameter sonar ring with 30 transmitters and 30 receivers placed alternately. Murata piezoelectric MA40S4R wide-angle transducers are used to enable overlapping reception of echoes produced by firing all the transmitters simultaneously. An axial symmetrical exponential horn structure is used to narrow the beam shape of the transmitters vertically to avoid reflections from the floor. Received signals are compared with a decaying threshold to produce a 1 bit digitized sampled signal without rectification. Bearing is estimated from the leading edge of echoes and an error standard deviation of 0.4° is reported for ranges up to 1.5 m.

A sonar ring with seven digital signal processors (DSPs) [30.52, 69, 70] that uses 24 pairs of 7000 series Polaroid transducers consisting of a transceiver and receiver has been developed (Fig. 30.23). Each pair can derive range and accurate bearing information using template-matching digital signal processing (Sect. 30.10) on each of the two receiver channels, which are sampled at 250 kHz with 12 bit analog to digital converters. In total eight receiver channels are processed per DSP. All transceivers are fired simultaneous to enable full surrounding sensing of the environment approximately 11 times a second to a 6 m range with experimentally validated range and bearing accuracies to smooth targets of 0.6 mm and 0.2°, respectively. To suppress interference between neigh-

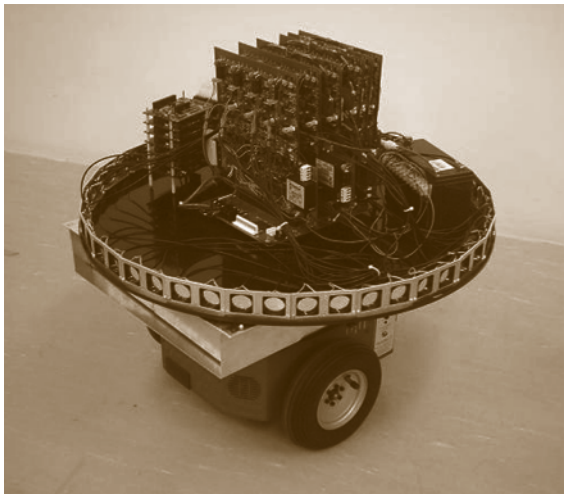


Fig. 30.23 DSP sonar ring hardware

boring pairs, two different transmitted pulse shapes are employed in an interleaved fashion around the perimeter of the ring. The pulse shapes are derived from two and three cycles of 65 kHz excitation. The DSP sonar ring allows rapid and accurate wall following, map building, and obstacle avoidance due to the high repetition and accurate range and bearing sensing. The beam width of the transducer pairs allows full 360° coverage with respect to smooth specular targets to a range of 3 m. An example of the DSP sonar ring producing a feature for a simultaneous localization and mapping (SLAM) map is shown in Fig. 30.24.

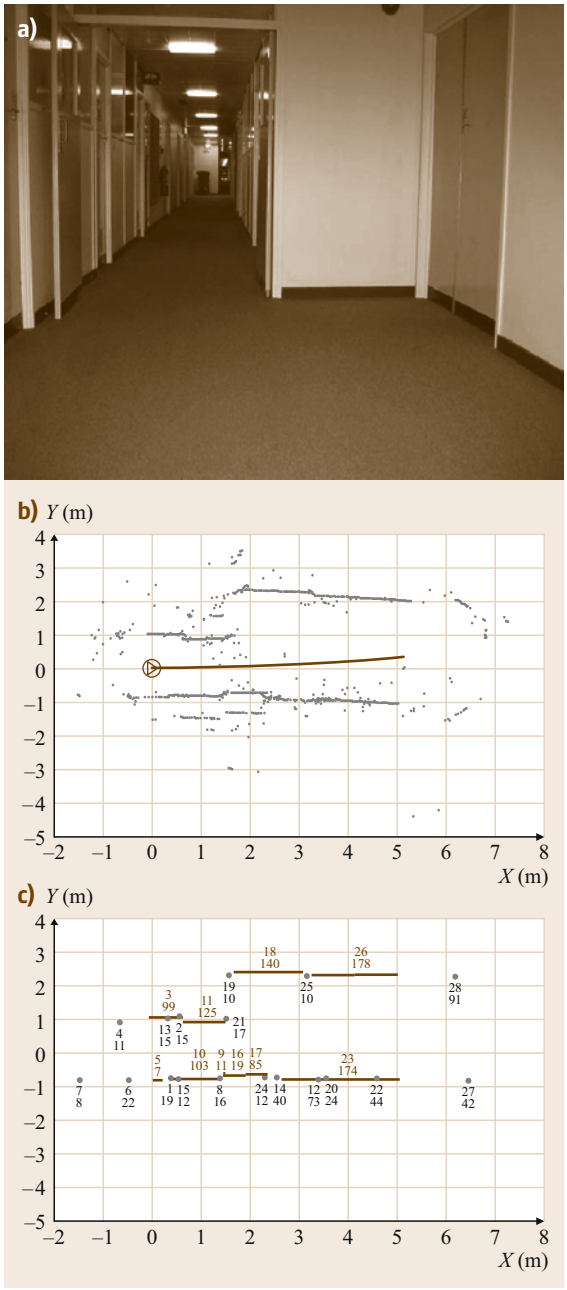


Fig.30.24a–c DSP sonar ring mapping an indoor environment (**a**), raw data (**b**) and SLAM feature map with feature number and number of associations shown as numbers (**c**). The sonar ring is moving at 10 cm/s with a 11.5 Hz sampling rate

30.13.3 Sonar Rings with FPGA Hardware Processing

One of the limitations of DSP based implementation of the signal processing arrangments such as [30.52, 69, 70] is that insufficient processing is available to perform matched filtering on the complete receiver echo signals in a multi-channel sonar ring. That is selected segments of the echo are processed where the signal exceeds a threshold in order to limit the computational load on the DSP processors. Once the full echo has arrived, matched filtering is then applied to these segments. More recent work [30.71, 72] describes a sonar ring that processes the complete echo signals from 48 receivers as the signal arrives with matched filtering. Complete processing of the entire receiver signal allows weak echoes to be detected that would be below the threshold of segmentation in [30.52, 69, 70]. This is achieved using a single field programmable gate array (FPGA) configured with a custom data path hardware design performing 4.9 Giga-arithmetic operations per

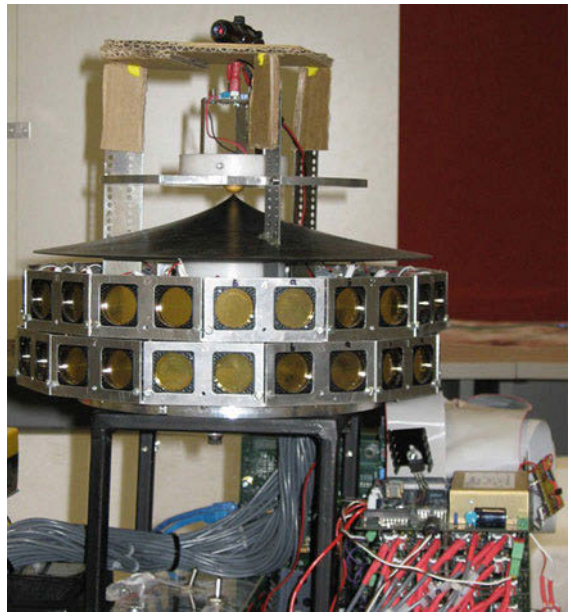


Fig. 30.25 FPGA based sonar ring

second. VIDEO 313 shows DSP sonar tracking an object. Range and bearing measurements are obtained at a rate of 30 Hz to 4 m range with full surrounding 360° angular coverage.

A photo of the FPGA sonar ring is shown in Fig. 30.25. Note that two layers of receivers are employed to reduce the radius of the 48 receiver ring. The other notable feature is the use of a single transmitter at the center of the ring. The transmitter is a hemispheric tweeter ESTD01 from Murata and a conical parabolic reflector to distribute the transmitter pulse uniformly in all directions horizontally [30.73]. Two cycles of a 48 kHz sine wave form the transmitter pulse. The transmitter can be modelled as point source and the pulse shape is approximately invariant with respect to received angle for each pair of receivers. This simplifies the matched filtering since the template for each receiver pair does not depend on the receiver angle. However there is some variation of pulse shape between different pairs in practice and as a function of range. The template match correlation processing hardware has been designed to allow dynamic switching of the template at predetermined ranges to account for the range variation of the pulse shape.

30.13.4 Double Refresh Rate Sonar Ring

Sonar rings have been limited in their firing rate by waiting for maximum range echoes to return before a consecutive firing of the transmitter. For example, the FPGA sonar ring presented in [30.71] is limited to 30 Hz refresh rate by this constraint. A double refresh rate FPGA sonar ring is presented in [30.74] that operates at 60 Hz for a 5.7 m range. To achieve a double refresh rate, the next sonar transmission is scheduled by selecting a transmit time from a random set so as to minimize interference with predicted echo arrival times. Two matched filters are applied to the receiver signal based on the two possible transmit times. Echoes are associated with a transmit epoch with the following criteria:

- The arrival time must be consistent with previous echoes since this is bounded by the sensor to obstacle relative speed
- The pulse energies should be within 50% of a previous echo from the same obstacle, and finally
- The template match correlation must be acceptable for the assumed range.

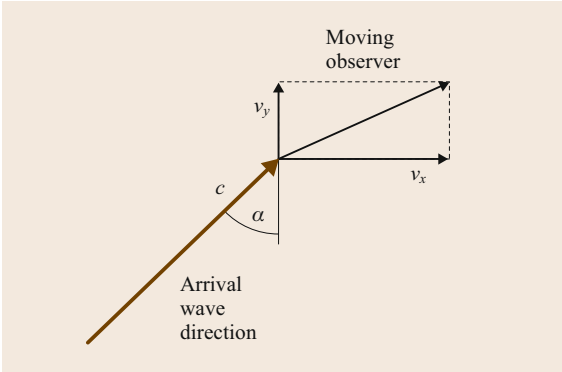
30.13.5 Sparse 3-D Arrays

The work by *Steckel* et al. [30.75] presents a sonar system that borrows beamforming theory from the design of antenna arrays. The sonar array is based on a sparse random array of 32 condensor omnidirectional ultrasonic microphones and a single Polaroid 7000 acting as a transmitter. The microphone signals are sampled at a maximum rate of 500 kHz and processed by an FPGA to produce a measurement repetition rate of 12 Hz. The transmitter waveform is a carefully controlled hyperbolic chirp of 100 down to 20 kHz in 3 ms. Matched filtering and beamforming processing are applied to the received signals to generate an energy scape – the distribution of energy from reflectors from discrete directions of around 1° resolution. Experiments show the system can discriminate 20 mm diameter poles at 800 mm with 5° separation. The minimum angular separation is related to the width of the mainlobe of the spatial array filters. Azimuth and elevation standard deviations of errors are reported as 1.1 and 0.6° respectively for a 80 mm sphere at 1.5 m. The advantage of the system is its ability to discriminate overlapping echoes in cluttered environments.

30.14 Motion Effects

When a sensor moves with respect to its targets, sonar measurements are effected. For example, a sonar sensor moving at a speed of 1% of the speed of sound (around 3.4 m/s) will experience errors of the order of 0.6° for some bearing measurements. The effects of linear velocity on the TOF and reception angle are dependent on the target type and hence for motion compensation to be meaningful a target classification sensor is needed. We consider the classical plane, edge, and corner target types in this section. Rotational motion effects are discussed in [30.27] where it is shown that very high speeds of rotation are necessary to give rise to a small bearing error (e.g., 0.1° error for approximately 1700 deg/s). Narrowing of the effective beam width is another effect of high rotation speeds of a sonar sensor.

The sensor is assumed to transmit from a point labeled T and receiver measurements are referenced to this position on the sensor. However, due to the motion of the sensor, the ground referenced position R at the time of reception of the echo moves from T over the course of the TOF. For a linear velocity, the distance between T and R is TOF$\times v$, where v is the magnitude of the sensor velocity vector relative to the ground, with components v_x and v_y parallel to their respective



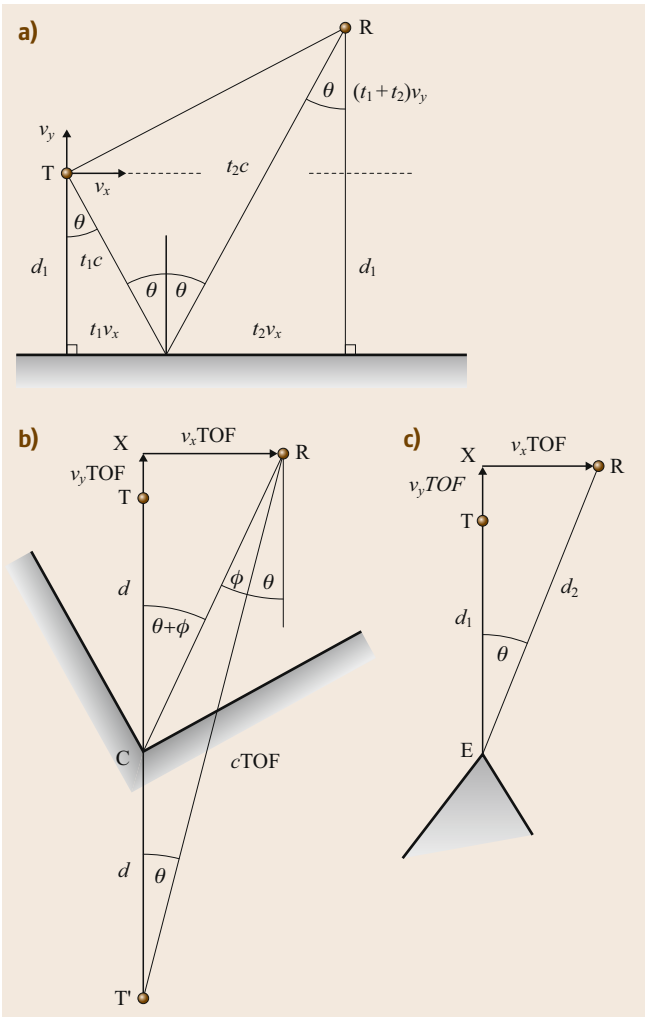


Fig.30.27a–c Observing a target from a moving sensor. T is the position of the transmitter and R is where the echo is received at the end of the TOF. The target is a plane in **(a)**, a corner in **(b)**, and an edge in **(c)**

Fig. 30.26 Observation of arriving wave from a moving observer ◀

coordinate axes. The expressions derived for linear motion apply to any sonar sensor, since only the physics of sound propagation and reflection are used. All targets are assumed to be stationary.

This section is based on [30.27], where further experimental work not included here can be found.

30.14.1 Moving Observation of a Plane

A plane target reflects the transmission from position T to R as shown in Fig. 30.26a. The TOF is broken up into two parts: t_1 is the time of propagation to the plane and t_2 from the plane to the receiver R. Here we derive the effect of linear motion on the TOF $= t_1 + t_2$ and the angle of reception θ all taken from the view of a *stationary observer*. A moving observer is discussed below.

From the right-angle triangle on the left of Fig. 30.27 a, we have

$$\sin\theta = \frac{v_x}{c} \text{ and } \cos\theta = \sqrt{1-\left(\frac{v_x}{c}\right)^2} \qquad (30.30)$$

and also

$$\cos\theta = \frac{d_1}{t_1 c} \implies t_1 = \frac{d_1}{c\cos\theta}\,. \qquad (30.31)$$

From the right-angled triangle on the right of Fig. 30.27 a, we have

$$\cos\theta = \frac{(t_1+t_2)v_y + d_1}{t_2 c}$$

$$\implies t_2 = \frac{(t_1+t_2)v_y + d_1}{c\cos\theta}\,. \qquad (30.32)$$

The TOF is obtained by adding (30.31) and (30.32) and then substituting (30.31) giving

$$\text{TOF} = \left(\frac{2d_1}{c}\right)\frac{1}{\sqrt{1-\frac{v_x{}^2}{c}-\frac{v_y}{c}}}\,. \qquad (30.33)$$

The first factor in (30.33) represents the stationary TOF. The second factor approaches unity as the velocity approaches zero.

30.14.2 Moving Observation of a Corner

Figure 30.27 b shows the situation for a corner with the virtual image of T is shown as T′. From the right-angled triangle T′XR

$$c^2\text{TOF}^2 = (2d_1 + v_y\text{TOF})^2 + v_x^2\text{TOF}^2\,, \qquad (30.34)$$

which gives

$$\mathrm{TOF} = \frac{2d_1}{c}\left(\frac{\sqrt{1-\left(\frac{v_x}{c}\right)^2}+\frac{v_y}{c}}{1-\left(\frac{v}{c}\right)^2}\right), \tag{30.35}$$

where $v^2 = v_x^2 + v_y^2$. The left-hand term of (30.35) is the stationary TOF and the right-hand term approaches unity for small velocities. The angle ϕ in Fig. 30.27 b is the angle deviation due to motion as reference by a stationary observer. From the triangles T′XR and CXR

$$\tan\theta = \frac{v_x\mathrm{TOF}}{2d_1 + v_y\mathrm{TOF}} \quad \text{and}$$
$$\tan(\theta+\phi) = \frac{v_x\mathrm{TOF}}{d_1 + v_y\mathrm{TOF}}\,. \tag{30.36}$$

From (30.36), we have

$$\tan(\theta+\phi) = \left(2 - \frac{v_y\mathrm{TOF}}{d_1 + v_y\mathrm{TOF}}\right)\tan\theta \tag{30.37}$$

and solving for $\tan\phi$ yields

$$\tan\phi = \tan\theta\left(\frac{1-\sin^2\theta}{\frac{v_y\mathrm{TOF}}{d_1}+1+\sin^2\theta}\right) = \left(\frac{v_x}{\frac{2d_1}{\mathrm{TOF}}+v_y}\right)\left(\frac{1-\sin^2\theta}{\frac{v_y\mathrm{TOF}}{d_1}+1+\sin^2\theta}\right). \tag{30.38}$$

For $v_x, v_y \ll c$, $\sin\theta \ll 1$ and $2d_1/\mathrm{TOF} \approx c$ we can approximate (30.38) as

$$\phi \approx \frac{v_x}{c}\,. \tag{30.39}$$

30.14.3 Moving Observation of a Edge

Since an edge reradiates the incoming ultrasound from an effective point source, the reception angle with respect to a stationary observer is unaffected by motion as shown in Fig. 30.27 c. The TOF is affected due to the motion moving the receiving position. From the right-angled triangle XER, $d_2^2 = (d_1+v_y)^2 + v_x^2\mathrm{TOF}^2$ and $d_1 + d_2 = c\mathrm{TOF}$ leads to

$$\mathrm{TOF} = \frac{2d_1}{c}\left(\frac{1+\frac{v_y}{c}}{1-\frac{v^2}{c^2}}\right) \approx \frac{2d_1}{c}\left(1+\frac{v_y}{c}\right), \tag{30.40}$$

where the approximate holds in (30.40) for $v \ll c$.

30.14.4 The Effect of a Moving Observation on the Angle of Reception

The expressions for the reception angle in the previous sections are based on an observer that is stationary with respect to the propagating medium air. In practice the observer is the sensor, and is moving with a velocity v. Suppose that the sonar wave arrives at an angle α relative to the air, as shown in Fig. 30.26.

The velocity components of the wavefront relative to the observer, w_x and w_y, are as follows

$$w_x = c\sin\alpha - v_x \text{ and } w_y = c\cos\alpha - v_y\,. \tag{30.41}$$

From (30.41) the observed angle of arrival, β is

$$\tan\beta = \frac{c\sin\alpha - v_x}{c\cos\alpha - v_y} = \frac{\sin\alpha - \frac{v_x}{c}}{\cos\alpha - \frac{v_y}{c}}\,. \tag{30.42}$$

30.14.5 Plane, Corner, and Edge Moving Observation Arrival Angles

In this section the arrival angles (in radians) for each target type are summarized and approximated for speeds expected of a mobile robot. The speed is assumed to be less than a few percent of the speed of sound (typically 340 m/s at room temperature). These effects have been observed experimentally at speeds of up to 1 m/s [30.27].

Equations (30.41) and (30.30) cancel exactly, and for a plane the arrival angle relative to the sensor is exactly zero

$$\beta_{\mathrm{plane}} = 0\,. \tag{30.43}$$

This can be explained by noting that the forward wave velocity component is always the same as the sensor's as reflection preserves this component.

For a corner the angle ϕ results in a wavefront that appears to be displaced in the same direction as the sensor motion from the real corner direction, as can be seen in Fig. 30.27 b. The effect of the moving observer doubles this effect as seen by (30.41) and (30.39)

$$\beta_{\mathrm{corner}} \approx -\frac{2v_x}{c}\,. \tag{30.44}$$

For an edge the result is due to the observer only

$$\beta_{\mathrm{edge}} \approx \tan^{-1}\left(\frac{0-\frac{v_x}{c}}{\cos\alpha - \frac{v_y}{c}}\right) \approx -\frac{v_x}{c}\,. \tag{30.45}$$

30.15 Biomimetic Sonars

The success of biosonars, bats and dolphins [30.76], has led researchers to implement sonars based on biosonar morphology, strategy, nonlinear processing and cleverness [30.77]. The capabilities exhibited by biosonars have caused researchers to examine biomimicking (*biomimetic*) systems.

Biosonar morphology typically has a single transmitter and a pair of receivers. Bats transmit sound pulses through the mouth or nose, while dolphins transmit through a melon. The two receivers correspond to ears that permit binaural processing. Mimicking binaural hearing has led to small arrays that localize objects [30.8] and scanning strategies [30.78]. VIDEO 311, VIDEO 316, VIDEO 317 show biomemetic sonars in action. Movable pinnae observed in bats have motivated research in receivers that rotate [30.79, 80] and deform [30.81]. VIDEO 312 shows pinna deformation. Figure 30.28 shows one such example.

Fig. 30.28 Biomimetic configuration sonar with center transmitter flanked by receivers that rotate

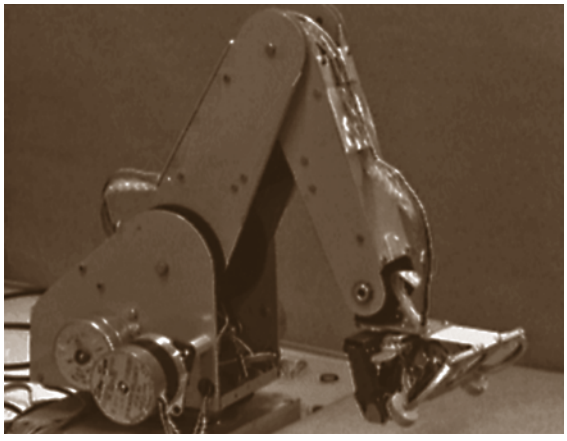


Fig. 30.29 Biomimetic sonar mounted on the end of a robot arm

Rotating the receivers so their axes fall onto the reflecting object not only increases the detected echo amplitude, but also its bandwidth, both effects improving the ability to classify an object.

Biosonar strategy provides clues for successful object localization. It is well known that the object location within the transducer beam affects the echo waveform and complicates the inverse problem of object classification [30.10, 82]. Dolphin movies show that they maneuver to position an object at a repeatable location and range, guided by binaural echo processing. This has motivated a dolphin-mimicking movable sonar positioned at the end of a robot arm for object classification [30.10, 82], as shown in Fig. 30.29.

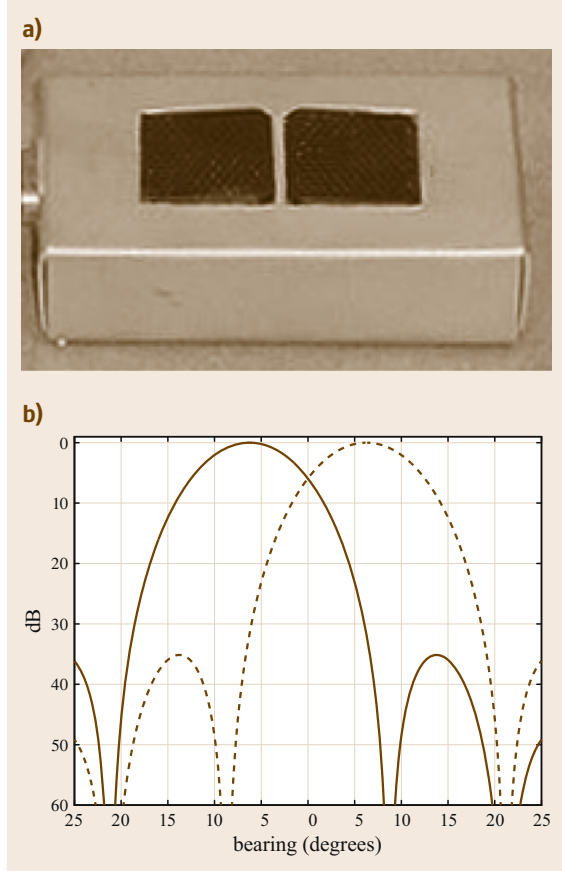


Fig.30.30a,b Binaural vergence sonar **(a)** and beam patterns **(b)** along the sonar axis with vergence angle $\alpha = 8^0$: *solid-left* transducer, *dashed-right* transducer

This system was able to differentiate reliably the head and tail side of a coin, but only after introducing a scan in elevation to accommodate for the lack of such positioning afforded by binaural hearing. The idea for a scan over elevation was motivated by the nodding motion that dolphins exhibit when searching for prey lying under the sand.

Another useful strategy, suggested by probing pulses emitted by bats, is processing echo sequences. As an extension to the conventional stop-and-scan operation of most sonars, sonar data were acquired while the sonar was moving along piecewise linear paths to reveal hyperbolic trends, similar to acoustic flow [30.83]. Matching data to hyperbolic trends permits the estimation of the passing range, which is useful for collision avoidance and passing through narrow openings [30.83].

Most sonar systems use classical estimation procedures involving correlation detection and spectrum analysis. The cochlear model has led to multiple band-pass filters to process wide-band pulses for environmental landmark classification [30.43]. The action potential spikes observed in the biological nervous system also suggest neuromorphic processing based on coincidence detection. The sparse information provided by conventional TOF measurements motivated sonar detectors that provided complete echo waveform information from multiple detections that result in spike-like data [30.24, 84]. Applying temporal and spacial coincidence to spike data has led to reverberation artifact recognition [30.36] and passing-range estimation [30.85]. VIDEO 302 shows a simulation of side-looking sonar passing two targets. VIDEO 314 shows camera view of mobile robot traveling down corridor. VIDEO 303 shows sonar spike map of corridor.

One shortcoming of sonar is its relatively wide beam that limits the accuracy of the reflector bearing estimate. Biosonars have two ears that assist in echolocation, with a strategy that is surprising for engineers. Figure 30.30 shows a biomimetic binaural sonar described in [30.86] that employs two Polaroid 6500 ranging modules connected to series-7000 ultrasound transducers (dia $d = 2.5$ cm), designated as *right* and *left*. Both transducers transmit emissions simultaneously, but each makes a TOF reading. The transducers are separated by D cm, with D minimized so that the incident echo wavefront is as similar as possible across both transducers. If both transducers are oriented parallel to the sonar axis, the TOF difference for an echo from an object at bearing θ is

$$\Delta\text{TOF} = \frac{D}{c}\sin\theta;, \tag{30.46}$$

which is small for small D.

While most sensor engineers would maximize signal strength by having both transducers point at the object, biological sonars trade off signal strength for improved localization by directing the ears away from the object with vergence angle α. This vergence angle spreads the sensitivity patterns away from the sonar axis, as shown in Fig. 30.30. As was shown in Fig. 30.12, this enhances ΔTOF by increasing the echo amplitude difference. VIDEO 301 shows vergence sonar in operation.

Such a vergence sonar has several applications for mobile robots. By noting when both TOFs are nearly coincident, the binaural vergence sonar can detect when a reflecting object is within $\pm 0.1^\circ$ of the sonar orientation. This sonar is useful for mobile robot navigation because, in addition to a range measurement, it indicates whether an obstacle is to the left or right of the intended path. VIDEO 295 shows mobile robot equipped with multiple vergence sonars. Also, the occlusion problem associated with sonar sensing is solved by placing vergence sonars on the outside limits of a robot to determine if an opening lies to the outside of the robot dimensions, thus allowing the robot to pass through narrow openings.

Such biomimetic techniques provide insights into the information content present in echoes and the type of sensing tasks for which sonar is best suited.

30.16 Conclusions

Sonar is a useful, inexpensive, low-power, light-weight and simple ranging sensor for robotics applications that can provide accurate object localisation. For sonar to be effectively employed, understanding its physical principles and implementation is important and these topics are covered in the early sections of this chapter. Various approaches to sonar sensing are highlighted from simple single-transducer ranging to more sophisticated multi-transducer and multi-pulse configurations with associated signal processing requirements. Sophisticated sonars are capable of measuring target range and angle accurately as well as classifying targets, rejecting interference, and compensating for motion. Sonar rings provide surrounding environmental coverage, and CTFM systems improve the sensitivity for detecting small reflectors. Research is ongoing in areas such as

signal and data processing, sonar map building, sonar configurations, transducer technology and biomimetic sonar that draw inspiration from biological sonar systems, such as used by bats and dolphins.

Video-References

VIDEO 295 Sonar guided chair at Yale
available from http://handbookofrobotics.org/view-chapter/30/videodetails/295

VIDEO 301 Vergence sonar
available from http://handbookofrobotics.org/view-chapter/30/videodetails/301

VIDEO 302 Side-looking TOF sonar simulation
available from http://handbookofrobotics.org/view-chapter/30/videodetails/302

VIDEO 303 Side-looking multi-pulse sonar moving down cider-block hallway
available from http://handbookofrobotics.org/view-chapter/30/videodetails/303

VIDEO 311 Antwerp biomimetic sonar tracking complex object
available from http://handbookofrobotics.org/view-chapter/30/videodetails/311

VIDEO 312 Biological bat ear deformation in sonar detection
available from http://handbookofrobotics.org/view-chapter/30/videodetails/312

VIDEO 313 Monash DSP sonar tracking a moving plane
available from http://handbookofrobotics.org/view-chapter/30/videodetails/313

VIDEO 314 Side-looking sonar system traveling down hallway (camera view)
available from http://handbookofrobotics.org/view-chapter/30/videodetails/314

VIDEO 315 B-scan image of indoor potted tree using multi-pulse sonar
available from http://handbookofrobotics.org/view-chapter/30/videodetails/315

VIDEO 316 Antwerp biomimetic sonar tracking single ball
available from http://handbookofrobotics.org/view-chapter/30/videodetails/316

VIDEO 317 Antwerp biomimetic sonar system tracking two balls
available from http://handbookofrobotics.org/view-chapter/30/videodetails/317

References

30.1 L.E. Kinsler, A.R. Frey, A.B. Coppens, J.V. Sanders: *Fundamentals of Acoustics* (Wiley, New York 1982)

30.2 R.C. Weast, M.J. Astle (Eds.): *CRC Handbook of Chemistry and Physics*, 59th edn. (CRC, Boca Raton 1978)

30.3 J. Borenstein, H.R. Everett, L. Feng: *Navigating Mobile Robots* (Peters, Wellesley 1996)

30.4 R. Kuc, M.W. Siegel: Physically-based simulation model for acoustic sensor robot navigation, IEEE Trans. Pattern Anal. Mach. Intell. **9**(6), 766–778 (1987)

30.5 SensComp: 7000, http://www.senscomp.com (2007)

30.6 H.H. Poole: *Fundamentals of Robotics Engineering* (Van Nostrand, New York 1989)

30.7 J.E. Piercy: *American National Standard: Method for Calculation of the Absorption of Sound by the Atmosphere*, Vol. ANSI S1-26-1978 (Acoust. Soc. Am., Washington 1978)

30.8 B. Barshan, R. Kuc: A bat-like sonar system for obstacle localization, IEEE Trans. Syst. Man Cybern. **22**(4), 636–646 (1992)

30.9 R. Kuc: Three dimensional docking using qualitative sonar. In: *Intelligent Autonomous Systems IAS-3*, ed. by F.C.A. Groen, S. Hirose, C.E. Thorpe (IOS, Washington 1993) pp. 480–488

30.10 R. Kuc: Biomimetic sonar locates and recognizes objects, J. Ocean Eng. **22**(4), 616–624 (1997)

30.11 L. Kleeman, R. Kuc: Mobile robot sonar for target localization and classification, Int. J. Robotics Res. **14**(4), 295–318 (1995)

30.12 B. Stanley: A Comparison of Binaural Ultrasonic Sensing Systems, Ph.D. Thesis (University of Wollongong, Wollongong 2003)

30.13 Material Systems Inc.: http://www.matsysinc.com/

30.14 F.L. Degertekin, S. Calmes, B.T. Khuri-Yakub, X. Jin, I. Ladabaum: Fabrication and characterization of surface micromachined capacitive ultrasonic immersion transducers, J. Microelectromech. Syst. **8**(1), 100–114 (1999)

30.15 B. Barshan, R. Kuc: Differentiating sonar reflections from corners and planes by employing an intelligent sensor, IEEE Trans. Pattern Anal. Mach. Intell. **12**(6), 560–569 (1990)

30.16 A. Freedman: A mechanism of acoustic echo formation, Acustica **12**, 10–21 (1962)

30.17 A. Freedman: The high frequency echo structure of somae simple body shapes, Acustica **12**, 61–70 (1962)

30.18 Ö. Bozma, R. Kuc: A physical model-based analysis of heterogeneous environments using sonar – ENDURA method, IEEE Trans. Pattern Anal. Mach. Intell. **16**(5), 497–506 (1994)

30.19 Ö. Bozma, R. Kuc: Characterizing pulses reflected from rough surfaces using ultrasound, J. Acoust. Soc. Am. **89**(6), 2519–2531 (1991)

30.20 P.J. McKerrow: Echolocation – from range to outline segments. In: *Intelligent Autonomous Systems IAS-3*, ed. by F.C.A. Groen, S. Hirose, C.E. Thorpe (IOS, Washington 1993) pp. 238–247

30.21 J. Thomas, C. Moss, M. Vater (Eds.): *Echolocation in Bats and Dolphins* (University of Chicago Press, Chicago 2004)

30.22 J. Borenstein, Y. Koren: Error eliminating rapid ultrasonic firing for mobile robot obstacle avoidance, IEEE Trans. Robotics Autom. **11**(1), 132–138 (1995)

30.23 L. Kleeman: Fast and accurate sonar trackers using double pulse coding, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1999) pp. 1185–1190

30.24 R. Kuc: Pseudo-amplitude sonar maps, IEEE Trans. Robotics Autom. **17**(5), 767–770 (2001)

30.25 H. Peremans, K. Audenaert, J.M. Van Campenhout: A high-resolution sensor based on tri-aural perception, IEEE Trans. Robotics Autom. **9**(1), 36–48 (1993)

30.26 A. Sabatini, O. Di Benedetto: Towards a robust methodology for mobile robot localization using sonar, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 3142–3147

30.27 L. Kleeman: Advanced sonar with velocity compenstation, Int. J. Robotics Res. **23**(2), 111–126 (2004)

30.28 A. Elfes: Sonar-based real world mapping and navigation, IEEE Trans. Robotics Autom. **3**, 249–265 (1987)

30.29 S. Thrun, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Haehnel, C. Rosenberg, N. Roy, J. Schulte, D. Schulz: MINERVA: A second geration mobile tour-guide robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 1999–2005

30.30 K. Konolige: Improved occupancy grids for map building, Auton. Robotics **4**, 351–367 (1997)

30.31 R. Grabowski, P. Khosla, H. Choset: An enhanced occupancy map for exploration via pose separation, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2003) pp. 705–710

30.32 J.D. Tardos, J. Neira, P.M. Newman, J.J. Leonard: Robust mapping and localization in indoor environments using sonar data, Int. J. Robotics Res. **21**(6), 311–330 (2002)

30.33 O. Aycard, P. Larouche, F. Charpillet: Mobile robot localization in dynamic environments using places recognition, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 3135–3140

30.34 B. Kuipers, P. Beeson: Bootstrap learning for place recognition, Proc. 18th Nat. Conf. Artif. Intell. (ANAI) (2002)

30.35 A. Bandera, C. Urdiales, F. Sandoval: Autonomous global localization using Markov chains and optimized sonar landmarks, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 288–293

30.36 R. Kuc: Biomimetic sonar and neuromorphic processing eliminate reverberation artifacts, IEEE Sens. J. **7**(3), 361–369 (2007)

30.37 A.M. Sabatini: A stochastic model of the time-of-flight noise in airborne sonar ranging systems, IEEE Trans. Ultrason. Ferroelectr. Freq. Control **44**(3), 606–614 (1997)

30.38 C. Biber, S. Ellin, E. Sheck, J. Stempeck: The Polaroid ultrasonic ranging system, Proc. 67th Audio Eng. Soc. Conv. (1990)

30.39 R. Kuc: Forward model for sonar maps produced with the Polaroid ranging module, IEEE Trans. Robotics Autom. **19**(2), 358–362 (2003)

30.40 M.K. Brown: Feature extraction techniques for recognizing solid objects with an ultrasonic range sensor, IEEE J. Robotics Autom. **1**(4), 191–205 (1985)

30.41 N.L. Harper, P.J. McKerrow: Classification of plant species from CTFM ultrasonic range data using a neural network, Proc. IEEE Int. Conf. Neural Netw. (1995) pp. 2348–2352

30.42 Z. Politis, P.J. Probert: Target localization and identification using CTFM sonar imaging: The AURBIT method, Proc. IEEE Int. Symp. Comput. Intell. Robotics Autom. (CIRLA) (1999) pp. 256–261

30.43 R. Mueller, R. Kuc: Foliage echoes: A probe into the ecological acoustics of bat echolocation, J. Acoust. Soc. Am. **108**(2), 836–845 (2000)

30.44 P.N.T. Wells: *Biomedical Ultrasonics* (Academic, New York 1977)

30.45 J.L. Prince, J.M. Links: *Medical Imaging Signals and Systems* (Prentice Hall, Upper Saddle River 2006)

30.46 F.J. Alvarez, R. Kuc: High resolution adaptive spiking sonar, IEEE Trans. Ultrason. Ferroelectr. Freq. Control **56**(5), 1024–1033 (2009)

30.47 F.J. Alvarez, R. Kuc, T. Aguilera: Identifying fabrics with a variable emission airborne spiking sonar, IEEE Sens. J. **11**(9), 1905–1912 (2011)

30.48 J.J. Leonard, H.F. Durrant-Whyte: Mobile robot localization by tracking geometric beacons, IEEE Trans. Robotics Autom. **7**(3), 376–382 (1991)

30.49 R. Kuc: Generating B-scans of the environmental with conventional sonar, IEEE Sens. J. **8**(2), 151–160 (2008)

30.50 P.M. Woodward: *Probability and Information Theory with Applications to Radar*, 2nd edn. (Pergamon, Oxford 1964)

30.51 A. Heale, L. Kleeman: Fast target classification using sonar, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 1446–1451

30.52 S. Fazli, L. Kleeman: A real time advanced sonar ring with simultaneous firing, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 1872–1877

30.53 T. Yata, A. Ohya, S. Yuta: A fast and accurate sonar-ring sensor for a mobile robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 630–636

30.54 L. Kleeman: Scanned monocular sonar and the doorway problem, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1996) pp. 96–103

30.55 G. Kao, P. Probert: Feature extraction from a broadband sonar sensor for mapping structured environments efficiently, Int. J. Robotics Res. **19**(10), 895–913 (2000)

30.56 B. Stanley, P. McKerrow: Measuring range and bearing with a binaural ultrasonic sensor, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 565–571

30.57 P.T. Gough, A. de Roos, M.J. Cusdin: Continuous transmission FM sonar with one octave bandwidth and no blind time. In: *Autonomous Robot Vehicles*,

ed. by I.J. Cox, G.T. Wilfong (Springer, Berlin, Heidelberg 1990) pp. 117–122
30.58 L. Kay: A CTFM acoustic spatial sensing technology: Its use by blind persons and robots, Sens. Rev. **19**(3), 195–201 (1999)
30.59 L. Kay: Auditory perception and its relation to ultrasonic blind guidance aids, J. Br. Inst. Radio Eng. **24**, 309–319 (1962)
30.60 P.J. McKerrow, N.L. Harper: Recognizing leafy plants with in-air sonar, IEEE Sens. J. **1**(4), 245–255 (2001)
30.61 K. Audenaert, H. Peremans, Y. Kawahara, J. Van Campenhout: Accurate ranging of multiple objects using ultrasonic sensors, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1992) pp. 1733–1738
30.62 J. Borenstein, Y. Koren: Noise rejection for ultrasonic sensors in mobile robot applications, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1992) pp. 1727–1732
30.63 K.W. Jorg, M. Berg: Mobile robot sonar sensing with pseudo-random codes, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 2807–2812
30.64 S. Shoval, J. Borenstein: Using coded signals to benefit from ultrasonic sensor crosstalk in mobile robot obstacle avoidance, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 2879–2884
30.65 K. Nakahira, T. Kodama, T. Furuhashi, H. Maeda: Design of digital polarity correlators in a multiple-user sonar ranging system, IEEE Trans. Instrum. Meas. **54**(1), 305–310 (2005)
30.66 A. Heale, L. Kleeman: A sonar sensor with random double pulse coding, Aust. Conf. Robotics Autom. (2000) pp. 81–86
30.67 A. Diosi, G. Taylor, L. Kleeman: Interactive SLAM using Laser and Advanced Sonar, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1115–1120
30.68 S.A. Walter: The sonar ring: obstacle detection for a mobile robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 1574–1578
30.69 S. Fazli, L. Kleeman: Wall following and obstacle avoidance results from a multi-DSP sonar ring on a mobile robot, Proc. IEEE Int. Conf. Mechatron. Autom. (2005) pp. 432–436
30.70 S. Fazli, L. Kleeman: Sensor design and signal processing for an advanced sonar ring, Robotica **24**(4), 433–446 (2006)
30.71 D. Browne, L. Kleeman: An advanced sonar ring design with 48 channels of continuous echo processing using matched filters, Proc. IEEE/RSJ Intell. Robots Syst. Conf. (IROS) (2009) pp. 4040–4046
30.72 D.C. Browne, L. Kleeman: A sonar ring with continuous matched filtering and dynamically switched templates, Robotica **30**(6), 891–912 (2012)
30.73 L. Kleeman, Akihisa Ohya: The design of a transmitter with a parabolic conical reflector for a sonar ring, Aust. Conf. Robotics Autom. (ICRA), Auckland (2006)
30.74 D.C. Browne, L. Kleeman: A double refresh rate sonar ring with FPGA-based continuous matched filtering, Robotica **30**(7), 1051–1062 (2012)
30.75 J. Steckel, A. Boen, H. Peremans: Broadband 3-D sonar system using a sparse array for indoor navigation, IEEE Trans. Robotics **91**, 1–11 (2012)
30.76 W.W.L. Au: *The Sonar of Dolphins* (Springer, Berlin, Heidelberg 1993)
30.77 R. Kuc, V. Kuc: Bat wing air pressures may deflect prey structures to provide echo cues for detecting prey in clutter, J. Acoust. Soc. Am. **132**(3), 1776–1779 (2012)
30.78 B. Barshan, R. Kuc: Bat-like sonar system strategies for mobile robots, Proc. IEEE Int. Conf. Syst. Man Cybern. (1991)
30.79 R. Kuc: Biologically motivated adaptive sonar, J. Acoust. Soc. Am. **100**(3), 1849–1854 (1996)
30.80 V.A. Walker, H. Peremans, J.C.T. Hallam: One tone, two ears, three dimensions: A robotic investigation of pinnae movements used by rhinolophid and hipposiderid bats, J. Acoust. Soc. Am. **104**, 569–579 (1998)
30.81 L. Gao, S. Balakrishnan, W. He, Z. Yan, R. Mueller: Ear deformations give bats a physical mechanism for fast adaptation of ultrasonic beam patterns, Phys. Rev. Lett. **1007**, 214–301 (2011)
30.82 R. Kuc: Biomimetic sonar system recognizes objects using binaural information, J. Acoust. Soc. Am. **102**(2), 689–696 (1997)
30.83 R. Kuc: Recognizing retro-reflectors with an obliquely-oriented multi-point sonar and acoustic flow, Int. J. Robotics Res. **22**(2), 129–145 (2003)
30.84 T. Horiuchi, T. Swindell, D. Sander, P. Abshire: A low-power CMOS neural amplifier with amplitude measurements for spike sorting, Proc. Int. Symp. Circuits Syst. (ISCAS), Vol. IV (2004) pp. 29–32
30.85 R. Kuc: Neuromorphic processing of moving sonar data for estimating passing range, IEEE Sens. J. **7**(5), 851–859 (2007)
30.86 R. Kuc: Binaural sonar electronic travel aid provides vibrotactile cues for landmark, reflector motion, and surface texture classification, IEEE Trans. Biomed. Eng. **49**(10), 1173–1180 (2002)

Multimedia Contents

31. Range Sensing

Kurt Konolige, Andreas Nüchter

Range sensors are devices that capture the three-dimensional (3-D) structure of the world from the viewpoint of the sensor, usually measuring the depth to the nearest surfaces. These measurements could be at a single point, across a scanning plane, or a full image with depth measurements at every point. The benefits of this range data is that a robot can be relatively certain where the real world is, relative to the sensor, thus allowing the robot to more reliably find navigable routes, avoid obstacles, grasp objects, act on industrial parts, etc.

This chapter introduces the main representations for range data (point sets, triangulated surfaces, voxels), the main methods for extracting usable features from the range data (planes, lines, triangulated surfaces), the main sensors for acquiring it (Sect. 31.1 – stereo and laser triangulation and ranging systems), how multiple observations of the scene, for example, as if from a moving robot, can be registered (Sect. 31.3) and several indoor and outdoor robot applications where range data greatly simplifies the task (Sect. 31.4).

31.1 **Range Sensing Basics** 783
31.1.1 Range Images and Point Sets 783
31.2 **Sensor Technologies** 785
31.2.1 Triangulation 786
31.2.2 Time of Flight 786
31.2.3 Comparison of Methods 788
31.2.4 Stereo Vision 789
31.2.5 Structured Light Stereo 793
31.3 **Registration** 794
31.3.1 The ICP Algorithm 794
31.3.2 Marker and Feature-Based Registration 795
31.3.3 Multiple-View Registration 801
31.3.4 Model Matching 802
31.3.5 Relative Pose Estimation 802
31.3.6 3-D Applications 803
31.4 **Navigation and Terrain Classification and Mapping** 804
31.4.1 Mobile Mapping 804
31.4.2 Urban Navigation 805
31.4.3 Rough Terrain 806
31.5 **Conclusions and Further Reading** 807
References 807

31.1 Range Sensing Basics

Here we present the basic representations used for range image data and discuss some issues relating to representations and robotics applications. While there are ranging devices that use sound or other waves to determine distance, this chapter concentrates on sensors that use light.

31.1.1 Range Images and Point Sets

Range data is a two-and-a-half-dimensional (2.5-D) or 3-D representation of the scene around the robot. The 3-D aspect arises because we are measuring the (X, Y, Z) coordinates of one or more points in the scene. Often only a single range image is used at each time instance. This means that we only observe the front sides of objects – the portion of the scene visible from the robot. In other words, we do not have a full 3-D observation of all sides of a scene. This is the origin of the term 2.5-D. Figure 31.1 shows a sample range image and a registered reflectance image, where each pixel records the level of reflected infrared light.

Fig.31.1a,b 3-D laser scan with a field of view of 360° × 100° acquired in an urban environment (Bremen downtown). **(a)** Registered infrared reflectance image. **(b)** Range image where closer is darker

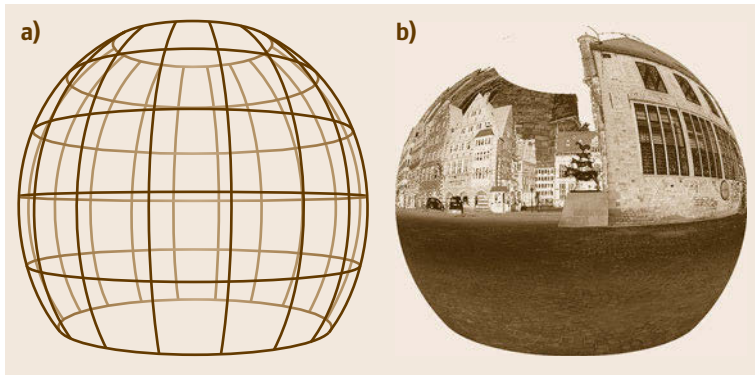


Fig. 31.2 **(a)** Mesh showing the rasterization of a 3-D range scanner using rotating mirrors and axis. **(b)** Sphere with superimposed reflectance values

There are two standard formats for representing range data. The first is an image $d(i,j)$, which records the distance d to the corresponding scene point (x,y,z) for each image pixel (i,j). There are several common mappings from $(i,j,d(i,j))$ to (X,Y,Z), usually arising from the geometry of the range sensor or from the application needs. The most common image mappings are illustrated in Fig. 31.3. Range scanners with rotating mirrors sample in spherical coordinates, that is,

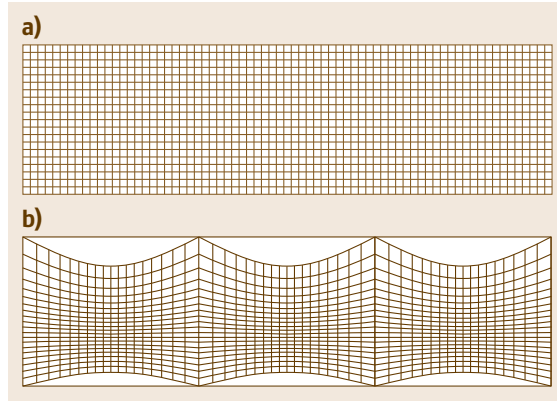


Fig. 31.3 **(a)** Equirectangular projection of a spherical range scan. Latidute and longitude mesh is shown. **(b)** Rectilinear projection with three images compined to unwrap a scanning sphere ►

Fig. 31.4 3-D range scan represented with three rectilinear projections (Fig. 31.1)

(θ, φ, d). Figure 31.2 shows a sphere as longitude and latitude mesh and superimposed with the reflectance values of Fig. 31.1. The most natural range image representation is to put (θ, φ) on the (i, j)-axis of the range image, that is,

$$i = \theta \,,$$
$$j = \varphi \,,$$

where the longitude θ and the latitude φ are the spherical coordinates. This is a projection of the samples sphere to a two-dimensional (2-D) image, which includes a distortion. The conversion between $(i, j, d) = (\theta, \varphi, d)$ and (x, y, z) proceeds as

$$d = \sqrt{x^2 + y^2 + z^2} \,,$$
$$\theta = \arccos\left(\frac{z}{r}\right) \,,$$
$$\varphi = \arctan2\,(y, x) \,.$$

Conversely, the Cartesian coordinates may be retrieved from the spherical coordinates by

$$x = d \sin\theta \cos\varphi \,,$$
$$y = d \sin\theta \sin\varphi \,,$$
$$z = d \cos\theta \,.$$

The further common projection is the perspective one, usually called rectilinear projection. It is also called *gnomonic* or *tangent-plane* projection. This include range images produced by kinetic-like devices. Here the range values are assumed to be projected in a pinhole camera like fashion. The primary advantage of the rectilinear projection is that it maps straight lines in the 3-D space and in the 2-D image. Its disadvantage is the smaller field of view (FOV). The projection is stretched toward the corners and the distortion grow with larger fields of view. Furthermore, FOVs are often not realizable by the hardware. The projection proceeds as

$$i = \frac{\cos\varphi \sin(\theta - \theta_0)}{\sin\varphi_1 \sin\varphi + \cos\varphi_1 \cos\varphi \cos(\theta - \theta_0)} \,,$$
$$j = \frac{\cos\varphi_1 \sin\varphi - \sin\varphi_1 \cos\varphi \cos(\theta - \theta_0)}{\sin\varphi_1 \sin\varphi + \cos\varphi_1 \cos\varphi \cos(\theta - \theta_0)} \,,$$

where θ_0 and φ_1 are the central longitude and central latitude, respectively. Figure 31.4 shows a 360° scan composed of three rectilinear range images.

Further common projections are the isogonic Mercator projection, the cylindrical projection, Pannini projection, and the stereographic [31.1]. All these projections are used to unwrap the scanning sphere to a 2-D array representation. In some applications, orthographic projections are processed from 3-D point cloud data. Some range sensors only record distances in a slice, so the scene (x, z) is represented by the linear image $d(i)$ for each pixel i.

The second format is as a list $\{(x_i, y_i, z_i)\}$ of 3-D data points, but this format can be used with all of the mappings listed above. Given the conversions from image data $d(i, j)$ to (x, y, z) the range data is only supplied as a list, which is usually called the point cloud.

31.2 Sensor Technologies

There are two main technologies for range sensing: triangulation and time-of-flight. There are many variations of each of these, with different strengths and weaknesses. Here we briefly describe the basic concepts, and summarize the characteristics of the main types of sensors. More detailed descriptions of stereo and structured light sensors are given in subsequent sections.

31.2.1 Triangulation

Triangulation sensors measure depth by determining the angle formed by the rays from a world point to two *sensors*. The sensors are separated by a baseline of length b, which forms the third segment of a triangle between the two sensors and the point. For simplicity, let one of the rays form a right angle with the baseline. Then the angle θ of the other sensor ray is related to the depth Z perpendicular to the baseline by

$$\tan\theta = \frac{Z}{b} \,. \tag{31.1}$$

An image sensor measures the angle θ by an offset on an image plane from the primary ray; this offset x is called the *disparity*. If we assume that the image plane is parallel to the baseline, then $\tan\theta = f/x$, and we get the basic equation of triangulation depth sensors

$$Z = \frac{fb}{x} \,. \tag{31.2}$$

Depth Precision

An important concept is the *depth resolution* of a sensor: how precisely can the sensor measure the depth? Differentiating with respect to x and substituting for x gives

$$\frac{\mathrm{d}Z}{\mathrm{d}x} = \frac{-Z^2}{fb} \,. \tag{31.3}$$

Triangulation precision falls off with the square of the distance to an object: if it is 1 mm at 1 m, then it is 4 mm at 2 m. Increasing the baseline or reducing the field of views FOV is inversely related to precision, that is, if the baseline is doubled or FOV is halved, the precision is halved (e.g., from 1 mm at 0.1 m baseline to 0.5 mm at 0.2 m baseline). This makes triangulation sensors more difficult to use at a distance; changing the FOV or baseline can help compensate, but brings in other tradeoffs (Sect. 31.2.4).

Types of Triangulation Sensors

The two main types are stereo cameras, which have two cameras separated by a baseline, and structured light sensors, which substitute a projector for one of the cameras.

Stereo cameras take images of a scene from two slightly different viewpoints, and match texture in the images to determine corresponding points and disparities. Some issues with stereo cameras are that there is often too little texture, especially indoors, to make reliable matches; and the matching uses small patches, which blurs out the spatial resolution. Many ingenious methods have been developed to deal with these issues, including painting a texture with a projector (not to be confused with structured light), using different kinds of support neighborhoods and regularization methods.

Structured light sensors project a known pattern from one end of a baseline, and view the pattern with a camera from the other. By decoding the pattern in the image, the placement of the pattern in the projector is determined, and hence the disparity. The simplest type of projectors output a point or a line, usually with a laser; the decoding problem is easy, but the amount of information is restricted to the point or line, and must be scanned to produce a full 3-D image.

A full-image structured light system projects a pattern covering the whole image, that can be decoded to get information at each pixel or small group of pixels. The PrimeSense technology, used in the Kinect, projects a known pattern of dots and decodes a small neighborhood to find the projected coordinates; thus this sensor also blurs out the spatial resolution. Other sensors project a time series of images with a vertical structure, so that each pixel on a horizontal line accumulates a unique code, which is then decoded against the projected images. Popular binary codes are Gray codes, which require N images if the image width is 2^N; and sinusoidal phase-shifted patterns, which typically use three images and are decoded by determining the phase at each point. These time-series structured light systems can produce very good spatial and depth resolution.

In general, structured light systems are not used outdoors because of interference from natural light, and for distant objects indoors. They can also have difficulty with high-contrast and specular objects, and the phase-based systems can suffer from 2π phase ambiguities. Relative motion between the sensor and the scene can also distort readings for time series and scanning sensors, although very fast projectors and cameras can alleviate this problem for the former.

31.2.2 Time of Flight

The principle behind time-of-flight (TOF) sensors is like that of radar: measure the time it takes for light to be projected out to an object and return. Because light travels about 0.3 m per nanosecond, very precise timers are required for direct TOF measurement; an alternative is indirect methods, typically phase difference with a modulated reference beam.

Because they measure time of flight, these sensors can theoretically have constant precision in depth measurement, no matter how far an object – unlike triangulation sensors, which fall off as the square of distance. But TOF sensors cannot duplicate the very fine precision of triangulation sensors for close objects,

and hence are not used in close-range metrology applications, such as small object reconstruction or parts quality measurements.

Direct Time of Flight

In direct TOF sensors, the travel time is measured by a high-speed chronometer. Laser-based direct TOF range sensors are also called LIDAR (light detection and ranging) or LADAR (laser radar) sensors. The travel time multiplied by the speed of light (in the given medium – space, air, or water and adjusted for the density and temperature of the medium) gives the distance

$$2d = ct\,, \tag{31.4}$$

where d is the distance to the object, c is the speed of light, and t is the measured travel time. Error in measuring the time t gives rise to a proportional distance error. In practice, one tries to measure the peak of the output pulse, which has a finite extent; weak reflections from distant objects make it harder to measure this peak and hence the error tends to increase with distance. Averaging multiple readings can reduce the random error in these readings.

The simplest TOF sensors transmit only a single beam, thus range measurements are only obtained from a single surface point. Robotics applications usually need more information, so the range data is usually supplied as a vector of range to surfaces lying in a plane (Fig. 31.5) or as an image (Fig. 31.1). To obtain these denser representations, the laser beam is swept across the scene. Normally, the beam is swept by a set of mirrors rather than moving the laser and detector themselves (mirrors are lighter and less prone to motion damage). The most common technologies for this are using a stepper motor (for program-based range sensing) or rotating or oscillating mirrors for automatic scanning.

Scanning or averaging requires multiple TOF pulses at perhaps high repetition rates, which can give rise to ambiguity about which pulse is actually being received. If δt is the time between pulses, then the *ambiguity interval* for the device is $1/2c\delta t$, for example, for a pulse repetition rate of 100 kHz, the ambiguity interval is 1500 m. If an object is further than this, it will be seen as within the interval, with distance $z \bmod 1/2c\delta t$. An *unwinding algorithm* can recover the true depth, if range values for the scanning system change slowly, and start within the ambiguity interval.

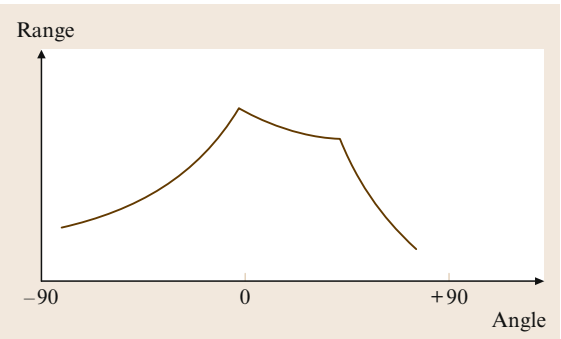


Fig. 31.5 Plot of ideal one-dimensional (1-D) range image of sample distance versus angle of measurement

Typical ground-based time of flight sensors suitable for robotics applications have a range of 10–100 m, and an accuracy of 5–10 mm. The amount of the scene scanned will depend on the sweep rate of the mirrors and the pulse rate, but 1–25k points per second are typical. Manufacturers of these sensors include Acuity, Hokuyo, Sick, Mensi, DeltaSphere, and Cyrax.

Multiple-beam scanning LIDARs can increase the amount of information available. Velodyne [31.2] makes three devices, with 16, 32, and 64 beams arranged vertically, that acquire point data at rates of up to 15 scans/s (1.3 MPixel/s), with a full 360° horizontal scan and a 27° vertical FOV from the laser array. The laser pulses are 5 ns long, and depth precision is approximately 2 cm. These devices are typically used in autonomous driving, for environment reconstruction and obstacle avoidance.

Flash LIDAR

In contrast to scanning devices, a *flash LIDAR* has a 2-D detector array (also called a *focal plane array* similar to a camera imager, but where each pixel incorporates timing circuitry to measure TOF of a laser pulse). Instead of a single or multiple laser beams, a light source pulse (LED (light-emitting diode) or laser) is shaped to cover a large area. All pixels have their timers started when the pulse is initiated, and measure the time it takes to receive the backscattered light. Typically, some tens of samples are captured and averaged, to reduce noise in the measurements – the amount of energy received is quite small, since the laser is not focused into a beam. As might be expected, the detector array pixels are quite large because of the timing electronics; a typical device from ASC [31.3] has 128×128 pixels, and can capture data at rates up to 60 Hz (≈ 1 MPixel/s, similar to the Velodyne devices). These devices are expensive, and hence not used in consumer applications. Alternatively, indirect ways of measuring TOF have simpler per-pixel electronics – see the next subsection. Table 31.1 lists the characteristics of one of ASC's devices.

Table 31.1 Characteristics of TigerEye device

Manuf./Model	Resolution	Max range	Repeatability
ASC Inc. TigerEye	128×128	60–1100 m	0.04 m @ 60 m

Indirect Time of Flight Sensors

Indirect TOF sensors measure the distance by inferring transit times from particular properties of a propagating beam. The two main methods are based on *modulation and phase differences* and *gated intensity*.

Modulation-Based TOF

Modulation-based range sensors are commonly of two types, where a continuous laser signal is either amplitude or frequency modulated. By observing the phase shift between the outgoing and return signals, the signal transit time is estimated and from this the target distance.

Amplitude modulation works by varying the intensity of a light signal $s(t) = \sin(2\pi ft)$ with frequency f. The signal reflected from an object has a phase shift φ, and the returned signal is

$$r(t) = R\sin(2\pi ft - \varphi) = R\sin\left[2\pi f\left(t - \frac{2d}{c}\right)\right],$$
$$d = \frac{c\varphi}{4\pi f}, \tag{31.5}$$

with c is the speed of light, d the distance to the object, and R the reflected amplitude. Measuring the phase shift yields the distance; note because the phase shift wraps around at 2π, the ambiguity interval is $c/2f$. For a modulation of 10 MHz, the interval is 15 m.

To measure the phase difference, the returned signal is mixed with the original reference signal and a 90° shifted version, then low-pass filtered. Figure 31.6 compares the two mixed signals that give both the phase difference and the intensity.

There are commercial devices using single-beam and planar array versions of amplitude modulation. The planar arrays take advantage of CMOS integration to implement the full signal comparison at each pixel, in a compact sensor [31.4, 5]. Because the signal-to-noise ratio (SNR) depends on the amplitude of the returned signal, these devices typically have a depth resolution that falls off as d^2, comparable to triangulation devices. Typical repeatability is about 3 cm at 0.5 m, significantly less than triangulation devices; the arrays are up to 200×200 pixels. The attraction of the planar array is that it is a single-chip ranging device, simple and potentially cheap to produce. They also have good ambient light rejection, and potentially can be used outdoors.

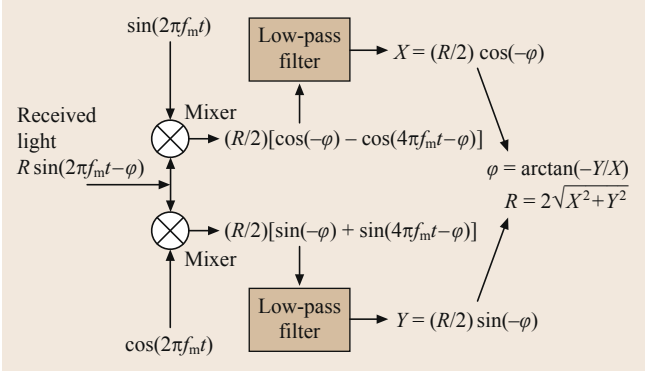


Fig. 31.6 Comparing the two mixed signals gives both the phase difference and the intensity

Frequency modulation, also called frequency modulation continuous wave (FMCW), modulates a laser frequency with a sawtooth ramp of period t_m, where the maximum frequency spread is Δf. The outgoing and incoming signals are mixed to give a signal difference f_i, which is measured by frequency counting. The distance is calculated as $d = f_i c t_m / 2\Delta f$. Since the frequency difference can be measured very accurately, this technique can yield excellent depth precision, on the order of 5 mm at 2 m [31.6]. The electronics required for mixing and frequency counting, as well as the difficulty of linear laser frequency modulation, have restricted this technique to high-end single-beam devices [31.7].

Range-Gated Intensity

Another indirect method for measuring TOF of a pulse is to measure the amount of the pulse that gets returned within a precise time period. If a pulse of width w is emitted, a detector will start to receive the pulse reflected from an object of distance d at $t_0 = 2d/c$, and finish receiving it at $t_w = (w + 2d)/c$. If the detector is *open* for a precise amount of time t, then the amount of the reflected signal it receives, relative to the full pulse reflection, is a linear measure of the distance d. Thus, by comparing two returns, one with a large open time to measure the full pulse, and one with a smaller open time, d can be determined. A planar array CMOS device can be fabricated to measure the two pulse returns sequentially, as long as each pixel can be triggered to be photosensitive for a precise amount of time. Commercial devices using this principle have precision on the order of a few cm at 1 m, with arrays up to 128×96 [31.8, 9].

31.2.3 Comparison of Methods

A useful overall summary of different types of range-sensing technology is in reference [31.6]. The two main parameters are spatial precision and depth precisions. In Fig. 31.7, available sensors are classified along these two dimensions. The direct TOF laser scanners (LMS, UT devices) cluster around 0.5–1 cm in both spatial and range precision. The Photon80 device uses FMCW

modulation with outstanding spatial and range precision; it is also a scanning device.

Triangulation devices (stereo, Kinect, and structured light in general) have less favorable characteristics. However, the target is at 2 m, and with triangulation precision falling with d^2, the comparison is biased. All these devices would be better than the TOF devices at 0.5 m in terms of range error. The stereo sensors can also exhibit much better spatial density in different configurations, that is, higher resolution sensors, larger baseline, smaller correlation window. In general, we would expect the triangulation sensors to outperform TOF sensors at shorter distances.

Finally, a single-chip flash LIDAR device, the IFM O3D200, uses amplitude modulation, and gives characteristically poor results relative to the other sensors.

31.2.4 Stereo Vision

This section discusses stereo analysis in more detail. Stereo analysis uses two or more input images to estimate the distance to points in a scene. The basic concept is *triangulation*: a scene point and the two camera points form a triangle, and knowing the baseline between the two cameras, and the angle formed by the camera rays, the distance to the object can be determined.

In practice, there are many difficulties in making a stereo imaging system that is useful for robotics applications. Most of these difficulties arise in finding reliable matches for pixels in the two images that correspond to the same point in the scene. A further consideration is that stereo analysis for robotics has a real-time constraint, and the processing power needed for some algorithms can be very high. But, in recent years much progress has been made, and the advantage of stereo imaging is that it can provide full 3-D range images, registered with visual information, potentially out to an infinite distance, at high frame rates – something which no other range sensor can match.

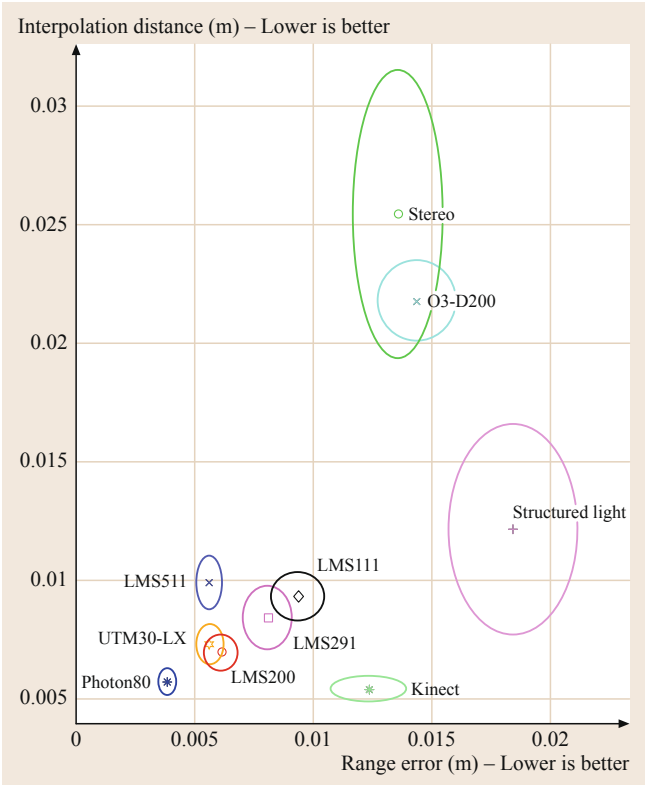


Fig. 31.7 Range device characteristics (after [31.6]). Spatial precision is on the Y-axis, depth precision on the X-axis. Measurements are taken on a target at 2 m

In this subsection, we will review the basic algorithms of stereo analysis, and highlight the problems and potential of the method. For simplicity, we use binocular stereo.

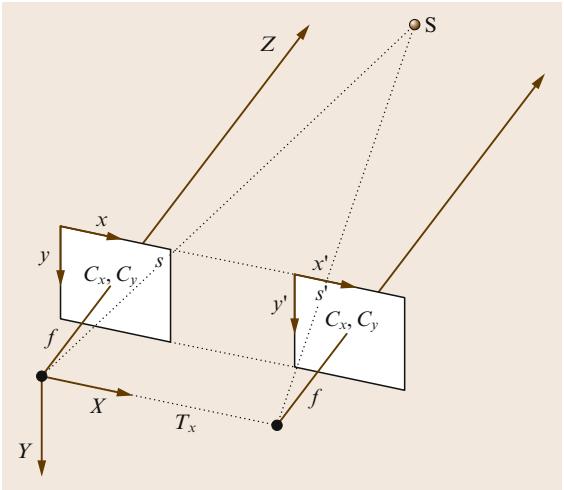


Fig. 31.8 Ideal stereo geometry. The global coordinate system is centered on the focal point (camera center) of the left camera. It is a right-handed system, with positive Z in front of the camera, and positive X to the right. The camera principal ray pierces the image plane at C_x, C_y, which is the same in both cameras (a variation for *verged* cameras allows C_x to differ between the images). The focal length is also the same. The images are lined up, with $y = y'$ for the coordinates of any scene point projected into the images. The difference between the x coordinates is called the *disparity*. The vector between the focal points is aligned with the X-axis ►

Stereo Image Geometry

This subsection gives some more detail of the fundamental geometry of stereo, and in particular the relationship of the images to the 3-D world via projection and reprojection. A more in-depth discussion of the geometry and the rectification process can be found in [31.10].

The input images are *rectified*, which means that the original images are modified to correspond to ideal pinhole cameras with a particular geometry, illustrated in Fig. 31.8. Any 3-D point S projects to a point in the images along a ray through the focal point. If the principal rays of the cameras are parallel, and the images are embedded in a common plane and have collinear scan lines, then the search geometry takes a simple form. The *epipolar line* of a point s in the left image, defined as the possible positions of s′ in the right image, is always a scan line with the same y coordinate as s. Thus, search for a stereo match is linear. The process of finding a rectification of the original images that puts them into standard form is called *calibration*, and is discussed in [31.10].

The difference in the x coordinates of s and s′ is the *disparity* of the 3-D point, which is related to its distance from the focal point, and the baseline T_x that separates the focal points.

A 3-D point can be projected into either the left or right image by a matrix multiplication in homogenous coordinates, using the *projection matrix*. The 3-D coordinates are in the frame of the left camera (Fig 31.8)

$$\mathbf{P} = \begin{pmatrix} F_x & 0 & C_x & -F_x T_x \\ 0 & F_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} . \tag{31.6}$$

This is the projection matrix for a single camera. F_x and F_y are the focal lengths of the rectified images, and C_x and C_y are the optical center; T_x is the translation of the camera relative to the left (reference) camera. For the left camera, it is 0; for the right camera, it is the baseline times the x focal length.

A point in 3-D is represented by homogeneous coordinates and the projection is performed using a matrix multiply

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathbf{P} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} , \tag{31.7}$$

where $(x/w, y/w)$ are the idealized image coordinates.

If points in the left and right images correspond to the same scene feature, the depth of the feature can be calculated from the image coordinates using the *reprojection matrix*

$$\mathbf{Q} = \begin{pmatrix} 1 & 0 & 0 & -C_x \\ 0 & 1 & 0 & -C_y \\ 0 & 0 & 0 & F_x \\ 0 & 0 & -1/T_x & \frac{(C_x - C_{x'})}{T_x} \end{pmatrix} . \tag{31.8}$$

The primed parameters are from the left projection matrix, the unprimed from the right. The last term is zero except for verged cameras. If x, y and x', y are the two matched image points, with $d = x - x'$, then

$$\begin{pmatrix} X \\ Y \\ Z \\ W \end{pmatrix} = \mathbf{Q} \begin{pmatrix} x \\ y \\ d \\ 1 \end{pmatrix} , \tag{31.9}$$

where $(X/W, Y/W, Z/W)$ are the coordinates of the scene feature, and $d = x - x'$ is the disparity. Assuming $C_x = C'_x$, the Z distance assumes the familiar inverse form of triangulation

$$Z = \frac{F_x T'_x}{d} . \tag{31.10}$$

Reprojection is valid only for rectified images – for the general case, the projected lines do not intersect. The disparity d is an *inverse depth* measure, and the vector (x, y, d) is a perspective representation range image (Sect. 31.1.1), sometimes called the *disparity space* representation. The disparity space is often used in applications instead of 3-D space, as a more efficient representation for determining obstacles or other features (Sect. 31.4.3).

Equation (31.9) is a homography between disparity space and 3-D Euclidean space. Disparity space is also useful in translating between 3-D frames. Let $p_0 = [x_0, y_0, d_0, 1]$ in frame 0, with frame 1 related by the rigid motion R, t. From the reprojection (31.9) the 3-D position is $\mathbf{Q}p_0$. Under the rigid motion, this becomes

$$\begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \mathbf{Q} p_0 ,$$

and finally applying $\mathbf{Q}^{-1}$ yields the disparity representation in frame 1. The concatenation of these operations is the homography

$$H(R, t) = \mathbf{Q}^{-1} \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \mathbf{Q} . \tag{31.11}$$

Using the homography allows the points in the reference frame to be directly projected onto another frame, without translating to 3-D points.

Stereo Matching Methods

The fundamental problem in stereo analysis is matching image elements that represent the same object or object part in the scene. Once the match is made, the range to the object can be computed using the image geometry.

Matching methods can be characterized as local or global. Local methods attempt to match small regions of one image to another based on intrinsic features of the region. Global methods supplement local methods by considering physical constraints such as surface continuity or base of support. Local methods can be further classified by whether they match discrete features among images, or correlate a small area patch [31.11]. Features are usually chosen to be lighting and viewpoint independent, for example, corners are a natural feature to use because they remain corners in almost all projections. Feature-based algorithms compensate for viewpoint changes and camera differences, and can produce rapid, robust matching. But they have the disadvantage of requiring perhaps expensive feature extraction, and yielding only sparse range results.

In the next section, we present local area correlation in more detail, since it is one of the most efficient and practical algorithms for real-time stereo. A survey and results of recent stereo matching methods are in [31.12], and the authors maintain a web page listing up-to-date information in [31.13].

Area Correlation Stereo

Area correlation compares small patches among images using correlation. The area size is a compromise, since small areas are more likely to be similar in images with different viewpoints, while larger areas increase the signal-to-noise ratio. In contrast to the feature-based method, area-based correlation produces dense results. Because area methods need not compute features, and have an extremely regular algorithmic structure, they can have optimized implementations.

The typical area correlation method has five steps (Fig. 31.9):

1. Geometry correction. In this step, distortions in the input images are corrected by warping into a *standard form*.
2. Image transform. A local operator transforms each pixel in the grayscale image into a more appropriate form, for example, normalizes it based on average local intensity.
3. Area correlation. This is the correlation step, where each small area is compared with other areas in its search window.
4. Extrema extraction. The extreme value of the correlation at each pixel is determined, yielding a disparity image: each pixel value is the disparity between the left and right image patches at the best match.
5. Post-filtering. One or more filters clean up noise in the disparity image result.

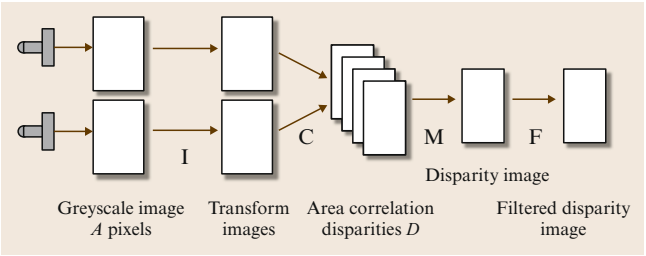


Fig. 31.9 Basic stereo processing (see text for details)

Correlation of image areas is disturbed by illumination, perspective, and imaging differences among images. Area correlation methods usually attempt to compensate by correlating not the raw intensity images, but some transform of the intensities. Let u, v be the center pixel of the correlation, d the disparity, and $I_{x,y}$, $I'_{x,y}$ the intensities of the left and right images:

1. Normalized cross-correlation

$$\frac{\sum_{x,y}[I_{x,y} - \hat{I}_{x,y}][I'_{x-d,y} - \hat{I'}_{x-d,y}]}{\sqrt{\sum_{x,y}[I_{x,y} - \hat{I}_{x,y}]^2 \sum_{x,y}[I'_{x,y} - \hat{I'}_{x-d,y}]^2}}$$

2. High-pass filter such as Laplacian of Gaussian (LOG). The Laplacian measures directed edge intensities over some area smoothed by the Gaussian. Typically, the standard deviation of the Gaussian is 1–2 pixels

$$\sum_{x,y} s(\mathrm{LOG}_{x,y} - \mathrm{LOG}_{x-d,y}) \ ,$$

where $s(x)$ is x^2 or $||x||$.

3. Nonparametric. These transforms are an attempt to deal with the problem of outliers, which tend to overwhelm the correlation measure, especially using a square difference. The census method [31.14] computes a bit vector describing the local environment of a pixel, and the correlation measure is the Hamming distance between two vectors

$$\sum_{x,y} (I_{x,y} > I_{u,v}) \oplus \left(I'_{x-d,y} > I'_{u,v}\right) \ .$$

Results on the different transforms and their error rates for some standard images are compiled in [31.13].

Another technique for increasing the signal-to-noise ratio of matches is to use more than two images [31.15].

Table 31.2 Post-filtering techniques for eliminating false matches in area correlation

Correlation surface [31.17]	Peak width: Wide peak indicates poor feature localization Peak height: Small peak indicates poor match Number of peaks: Multiple peaks indicate ambiguity
Mode filter	Lack of supporting disparities violates smoothness
Left/right check [31.18, 19]	Nonsymmetric match indicates occlusion
Texture [31.20]	Low texture energy yields poor matches

This technique can also overcome the problem of viewpoint occlusion, where the matching part of an object does not appear in the other image. The simple technique of adding the correlations between images at the same disparity seems to work well [31.16]. Obviously, the computational expenditure for multiple images is greater than that for two.

Dense range images usually contain false matches that must be filtered, although this is less of a problem with multiple-image methods. Table 31.2 lists some of the post-filters that have been discussed in the literature.

Disparity images can be processed to give subpixel accuracy, by trying to locate the correlation peak between pixels. This increases the available range resolution without much additional work. Typical accuracies are 1/10 pixel.

Stereo Range Quality

Various artifacts and problems affect stereo range images.

Smearing. Area correlation introduces expansion in foreground objects, for example, the woman's head in Fig. 31.10. The cause is the dominance of strong edges on the object. Nonparametric measures are less subject to this phenomenon. Other approaches include multiple correlation windows and shaped windows.

Dropouts. These are areas where no good matches can be found because of low texture energy. Dropouts are a problem for indoor and outdoor man-made surfaces. Projecting a random texture can help [31.21].

Range Resolution. Unlike LADAR devices, stereo range accuracy is a quadratic function of distance, found by differentiating (31.10) with respect to disparity

$$\delta Z = -\frac{F_x T'_x}{d^2} \,. \tag{31.12}$$

The degradation of the stereo range with distance can be clearly seen in the 3-D reconstruction of Fig. 31.10.

Processing. Area correlation is processor-intensive, requiring Awd operations, where A is the image area, w is the correlation window size, and d is the number of disparities. Clever optimizations take advantage of redundant calculations to reduce this to Ad (independent of window size), at the expense of some storage. Real-time implementations exist for standard personal computers (PC)s [31.22, 23], graphics accelerators [31.24, 25], digital signal processors (DSPs) [31.26], field programmable gate arrays (FPGAs) [31.22, 27], and specialized application-specific integrated circuits (ASICs) [31.28].

Other Visual Sources of Range Information

Here we briefly list the most popular, but less reliable sources of range information. These sources can potentially supplement other sensors:

- *Focus/defocus*: Knowledge of the camera parameters and the amount of blur of image features allows the estimation of how far the corresponding scene features are from the perfect focus distance [31.29]. Sensors may be passive (using a precaptured image) or active (capturing several images with different focus settings).

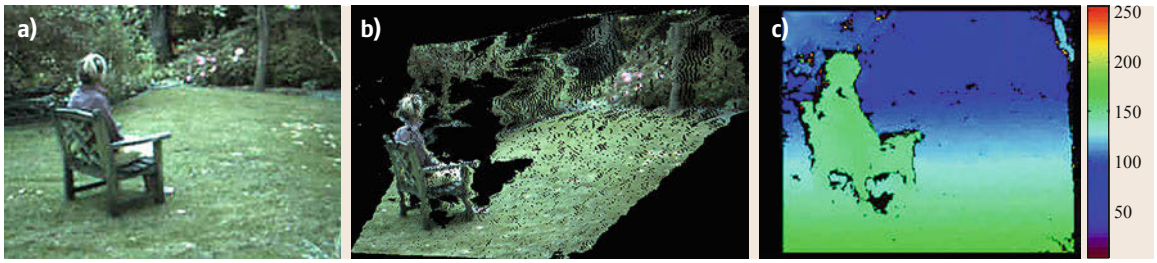


Fig.31.10a–c Sample stereo results from an outdoor garden scene; baseline is 9 cm. **(a)** original left image. **(b)** computed 3-D points from a different angle. **(c)** disparity in pseudo-color

- *Structure and motion*: Structure and motion algorithms compute 3-D scene structure and the sensor positions simultaneously [31.30]. This is essentially a binocular stereo process (see discussion before), except that only a single moving camera is used. Thus, the images needed by the stereo process are acquired by the same camera in several different positions. Video camcorders are also used, but they have lower resolution. One important advantage of this approach over the normal algorithm is that features can be tracked easily if the time between frames or the motion is small enough. This simplifies the *correspondence problem*; however, it can lead to another problem. If the pair of images used for the stereo calculation are taken close together in time, then the separation between the cameras' images will not be much – this is a *short baseline*. Triangulation calculations are then more inaccurate, as small errors in estimating the position of image features results in large errors in the estimated 3-D position (particularly the depth estimate). This problem can be partly avoided by tracking for longer periods. A second problem that can arise is that not all motions are suitable for estimate of the full 3-D scene structure. For example, if the video recorder only rotates about its optical axis or about its focus point, then no 3-D information can be recovered. A little care can avoid this problem.
- *Shading*: The pattern of shading on a surface is related to the orientation of the surface relative to the observer and light sources. This relationship can be used to estimate the surface orientation across the surface. The surface normals can then be integrated to give an estimate of the relative surface depth.
- *Photometric stereo*: Photometric stereo [31.31] is a combination of shading and stereo processes. The key concept is that the shading of an object varies with the position of the light sources. Hence, if you had several aligned pictures of an object or scene with the light source in different positions (e.g., the sun moved), then you can calculate the scene's surface normals. From these, the relative surface depth can be estimated. The restriction of a stationary observer and changing light sources makes this approach less likely to be useful to most robotics applications.
- *Texture*: The way uniform or statistical textures vary on a surface is related to the orientation of the surface relative to the observer. As with shading, the texture gradients can be used to estimate the surface orientation across the surface [31.32]. The surface normals can then be integrated to give an estimate of the relative surface depth.

31.2.5 Structured Light Stereo

Recently, the Kinect range-sensing device became the fastest-selling consumer electronics device in history [31.33]. The Kinect uses a structured-light stereo technique from PrimeSense (bought out by Apple, Inc.) to give a reasonable balance of spatial and depth precision, in a highly-integrated device, at a very low price.

The PrimeSense technology projects a structured pattern generated from an IR (infrared) laser and patented combination of diffractive elements (Fig. 31.11). The pattern consists of bright dots on a grid, with unique combinations of dots over a horizontal area. A camera with resolution 1280×960 captures the IR image at a horizontal offset of 7.5 cm from the projector, and correlates a 19×19 block against the known dot pattern for a given horizontal line. If a match is found, the position of the pattern in the known projected pattern gives the angle subtended by the reflecting object to the projector and camera, and hence the distance. A depth image is returned at a resolution of 640×480, with 11 bits of depth precision, over an FOV of approximately 50°; the depth is interpolated to a precision of 1/8 pixel. Error characteristics for the Kinect have been studied in [31.34]. The depth decision is limited by the resolution, similarly to (31.3).

There are additional sources of error. *Dropouts* occur when the surfaces are minimally reflective, or specular. *False positives*, where there are gross mismatches in the match, are very rare with PrimeSense technology. *Spatial precision* is much lower than the returned 640×480 depth image would indicate, because of the *smear-*

Fig. 31.11 The laser grid of the Kinect for calculating depth

ing effect of block correlation. The projected pattern expands and contracts based on temperature, leading to offset in disparities along the vertical axis. While the projector is temperature controlled via a Peltier unit, its temperature can vary. A temperature sensor measures this variation, and applies disparity steps across the image, which are apparent as vertical steps in the depth image.

31.3 Registration

This section introduces techniques for 3-D localization of parts for robot manipulation, self-localization of robot vehicles, and scene understanding for robot navigation. All of these are based on the ability to register 3-D shapes, for example, range images to range images, triangulated surfaces or geometric models. Registration puts two or more independently acquired 3-D shapes into one frame of reference. It can be solved as an optimization problem that uses a cost function for the quality of the alignment. The 3-D shapes are registered by determining the rigid transformation (rotation and translation) which minimizes the cost function. Feature based registration extracts distinguishing features of the range images and uses corresponding features for calculating the alignment.

31.3.1 The ICP Algorithm

The following method is used for the registration of point sets. The complete algorithm was invented at the same time in 1991 by *Besl* and *McKay* [31.35], by *Chen* and *Medioni* [31.36], and by *Zhang* [31.37]. The method is called iterative closest points (ICP) algorithm. It is the de-facto standard for registration of point sets, but is also applicable to 3-D shapes, if one samples them.

Given two independently acquired sets of 3-D points, $\hat{M}$ (model set) and $\hat{D}$ (data set) which correspond to a single shape, we want to find the transformation $(\mathbf{R}, \boldsymbol{t})$ consisting of a rotation matrix $\mathbf{R}$ and a translation vector $\boldsymbol{t}$ which minimizes the following cost function,

$$E(\mathbf{R}, \boldsymbol{t}) = \frac{1}{N} \sum_{i=1}^{N} \|\boldsymbol{m}_i - (\mathbf{R}\boldsymbol{d}_i + \boldsymbol{t})\|^2 . \qquad (31.13)$$

All the corresponding points can be represented in a tuple $(\boldsymbol{m}_i, \boldsymbol{d}_i)$ where $\boldsymbol{m}_i \in M \subset \hat{M}$ and $\boldsymbol{d}_i \in D \subset \hat{D}$. Two things have to be calculated. First, the corresponding points, and second, the transformation $(\mathbf{R}, \boldsymbol{t})$ that minimizes $E(\mathbf{R}, \boldsymbol{t})$ on the basis of the corresponding points. The ICP algorithm uses closest points as corresponding points. A sufficiently good starting guess enables the ICP algorithm to converge to the correct minimum. Figure 31.12 shows two 3-D point clouds, their initial alignment and the final registration after a few ICP iterations.

Implementations of ICP (Algorithm 31.1) use a maximal distance for closest points to handle partially overlapping point sets. In this case, the proof about ICP's monotonic convergence in [31.35] no longer holds, since the number of points as well as the value of $E(\mathbf{R}, \boldsymbol{t})$ might increase after applying a transformation.

Algorithm 31.1 The ICP algorithm

1: **for** $i = 0$ to *maxIterations* **do**
2: **for all** $\boldsymbol{d}_j \in D$ **do**
3: find the closest point within a range $d_{\max}$ in the set M for point $\boldsymbol{d}_j$
4: **end for**
5: Calculate transformation $(\mathbf{R}, \boldsymbol{t})$ that minimizes the error function (31.13)
6: Apply the transformation found in step 5 to the data set D.
7: Compute the difference of the quadratic error, that is, compute the difference of the value $\|E_{i-1}(\mathbf{R}, \boldsymbol{t}) - E_i(\mathbf{R}, \boldsymbol{t})\|$ before and after the ap-

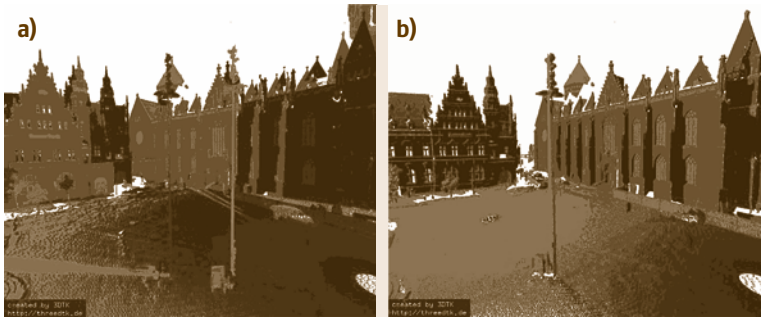


Fig.31.12a,b Initial alignment of two 3-D point clouds **(a)** and after optimization with ICP **(b)**

plication of the transformation. If this difference falls below a threshold ε, terminate.

8: **end for**

Current research in the context of ICP algorithms mainly focuses on fast variants of ICP algorithms [31.38]. If the input are 3-D meshes then a point-to-plane metric can be used instead of (31.13). Minimizing using a point-to-plane metric outperforms the standard point-to-point one, but requires the computation of normals and meshes in a preprocessing step.

The computation of closest points is the crucial step of the ICP algorithm. A naive implementation examines all points in $\hat{M}$ and the resulting computation time for ICP is $O(|\hat{D}|\,|\hat{M}|)$, that is, $O(n^2)$. Note that $\hat{N}$ can be very large; advanced high-precise 3-D laser scanners such as the Zoller + Fröhlich yield a data rate up to 1 000 000 3-D points per second. An efficient tree data structure, a k-d tree, is widely used to speed up the closest point computation. Every node of the k-d tree represents a partition of the point set into two distinct sets, the successor nodes. The root of the tree represents the whole point set. The leaves of the tree are called buckets and are a partition of the set into small, disjunctive point sets. Furthermore, every node of the tree consists of the center and the dimension of the point set. In the original k-d tree paper, these so-called split dimensions have been chosen depending on the depth of node in a robin-round fashion [31.39]. The split dimension and value define an axis-aligned hyperplane in the k-dimensional space. The data are partitioned according to their position to the hyperplane into the successor nodes. The k-d tree is constructed until the number of points in the nodes falls below a threshold b (bucket size). Only the leaves of the tree contain the data points. Searching in k-d trees is done recursively. A given 3-D point $\boldsymbol{p}_q$ needs to be compared to the separating plane (splitting dimension and splitting value) in order to decide on which side the search must continue. This procedure is executed until the leaves are reached. There, the algorithm has to evaluate all bucket points. However, the closest point may be in a different bucket, iff the distance d of the query point $\boldsymbol{p}_\mathrm{q}$ to the limits is smaller than the one to the closest point in the bucket $\boldsymbol{p}_\mathrm{b}$. In this case backtracking has to be performed. The test is known as Ball-Within-Bounds test [31.39–41]. The optimized k-d tree chooses the split dimension and split value [31.40], such that the expected amount of backtracking is minimized. Since one typically has no information about the query points k-d tree algorithms take only the distribution of the given points into account. For all possible queries, this works sufficiently, but it will not be optimal for a specific query [31.40]. This enables the recursive construction and avoids the overall optimization that is known to be NP-complete [31.42].

Improvements to k-d tree search, especially for small dimensions, have been shown in the last decade. They include approximate k-d tree search [31.41], registration using d^2-trees [31.43] and cached k-d tree search [31.44]. In addition, the spatial data structure octree might, which is discussed later in this section, can be used to search the point clouds with similar performance.

In each ICP iteration, the transformation can be calculated in $O(N)$ by any of these four methods:

1. A singular value decomposition (SVD)-based method by *Arun* et al. [31.45]
2. A quaternion-based method by *Horn* et al. [31.46]
3. An algorithm using orthonormal matrices by *Horn* [31.47]
4. A calculation based on dual quaternions by *Walker* et al. [31.48]. Besides these closed-form solutions, there are several linearized, approximative version [31.49].

The challenge is to ensure thar $\mathbf{R}$ is orthonormal. Most often, the first method is implemented, due to it's simplicity and the availability of numerical SVDs in various libraries. The rotation $\mathbf{R}$ is represented as an orthonormal 3×3 matrix. The optimal rotation is calculated by $\mathbf{R}=\mathbf{V}\mathbf{U}^\mathrm{T}$. Here the matrices $\mathbf{V}$ and $\mathbf{U}$ are derived by the SVD $\mathbf{H}=\mathbf{U}\mathbf{\Lambda}\mathbf{V}^\mathrm{T}$ of a cross-correlation matrix $\mathbf{H}$. This 3×3 matrix $\mathbf{H}$ is given by

$$\mathbf{H}=\sum_{i=1}^{N}\boldsymbol{m}_i'^\mathrm{T}\boldsymbol{d}_i'=\begin{pmatrix}S_{xx}&S_{xy}&S_{xz}\\S_{yx}&S_{yy}&S_{yz}\\S_{zx}&S_{zy}&S_{zz}\end{pmatrix}, \tag{31.14}$$

where

$$S_{xx}=\sum_{i=1}^{N}m'_{x,i}d'_{x,i},\ S_{xy}=\sum_{i=1}^{N}m'_{x,i}d'_{y,i},\ \ldots\ .$$

31.3.2 Marker and Feature-Based Registration

To avoid issues with starting guess in the ICP framework, marker-based registration uses defined artificial or natural landmarks as corresponding points. This manual data association ensures that by minimizing (31.13) the scans are registered at the correct location. Iterations are no longer required, but possible as verification with the RANSAC algorithm. The RANSAC algorithm is a general, randomized procedure that iter-

atively finds an accurate model for observed data that may contain a large number of outliers [31.50].

3-D Feature Representations and Extraction

There are many representations available for encoding 3-D scene structure and model representations, but the following representations are the ones most commonly encountered in robotics applications. Some scene models or descriptions may use more than one of these simultaneously to describe different aspects of the scene or object models.

Normals. Normals describe the surface orientation in a points. There are a number of approaches that have been developed for computing normals in point clouds and range images. Most of these approaches involves some form of eigenvalue decomposition resembling total linear least squares. The normals are usually computed when taking into account the nearby neighbor points which are computed using a variety of methods like nearest neighbor search. These methods include k nearest neighbors (k-NN) and radius search among others. The method of total least squares is robust to noise as it inherently includes low-pass filtering, but it is sensitive to the distribution and density of the point samples as well as the curvature of the underlying manifold. Improvements to the above method by using higher order surfaces are discussed in [31.51] and it is noted that such methods can fail even with arbitrarily dense samples.

The total least-squares problem finds the plane parameters that optimally fit a small surface area in a point set given by the the nearest neighbors. A plane is defined by

$$n_x x + n_y y + n_z z - d = 0 \,,$$

where $\boldsymbol{p} = (x, y, z)^{\mathrm{T}}$ lies on the plane and (n_x, n_y, n_z, d) are the parameters to compute. Given a subset of k 3-D points $p_i, i = 1, 2, \ldots, k$ of the surface, least squares finds the optimal normal vector $n = (n_x, n_y, n_z)^{\mathrm{T}}$ and scalar d that minimizes the following error equation

$$e = \sum_{i=1}^{k} (\boldsymbol{p}_i \boldsymbol{n} - d) \,. \tag{31.15}$$

The basic method for normal estimation from the neighboring points using a fitting plane is the principal component analysis (PCA). PCA is the eigenvalue decomposition of the data covariance (or correlation) matrix or singular value decomposition of a data matrix, usually after mean centering (and normalizing) the data matrix for each attribute (3-D query point). PCA can be interpreted as fitting a Gaussian distribution and computing the main axes. For each query point

$$\boldsymbol{\mu} = \frac{1}{k} \sum_{i=1}^{k} \boldsymbol{p}_i \,,$$

$$\boldsymbol{\Sigma} = \frac{1}{k} \sum_{i=1}^{k} (\boldsymbol{p}_i - \boldsymbol{\mu})^{\mathrm{T}} (\boldsymbol{p}_i - \boldsymbol{\mu}) \,.$$

The vector corresponding to the smallest eigenvalue of the above matrix $\boldsymbol{\Sigma}$ is the normal direction if the neighboring points belong to a plane and this is also the closed form solution of the total least-squares problem.

For equirectangular range images, that is, spherical coordinates, a fast algorithm has been developed in [31.52] that avoids computing eigenvalues. Dividing (31.15) by d^2 gives a simplified function and further division by the squared range ρ^2 yields

$$e = \sum_{i=1}^{k} \left((\rho^{-1} \boldsymbol{p}_i)^{\mathrm{T}} \boldsymbol{n} - \rho_i^{-1} \right)^2 \,,$$

$$\boldsymbol{p}_i = \begin{pmatrix} \cos\theta_i \sin\varphi_i \\ \sin\theta_i \sin\varphi_i \\ \cos\varphi_i \end{pmatrix} .$$

Therefore, a solution for $\boldsymbol{n}$ is given as

$$\boldsymbol{n} = \mathbf{M}^{-1} \boldsymbol{b} \,,$$

where

$$\mathbf{M} = \sum_{i=1}^{k} \boldsymbol{p}_i \boldsymbol{p}_i^{\mathrm{T}} \,, \boldsymbol{b} = \sum_{i=1}^{k} \frac{\boldsymbol{p}_i}{\rho_i} \,.$$

This way the computation of eigenvalues is avoided and the matrix $\mathbf{M}$ can be precomputed for the desired image coordinates as it does not depend on the range. The tangential surface and therefore its normal vector are obtained by simply taking the derivative of the surface function in the point of interest

$$\begin{aligned} \boldsymbol{n} &= \nabla\rho = \nabla\rho(\theta, \varphi) \\ &= \begin{pmatrix} \cos\theta \sin\theta - \frac{\sin\theta}{\rho \sin\varphi} \frac{\partial\rho}{\partial\theta} + \frac{\cos\theta \cos\varphi}{\rho} \frac{\partial\rho}{\partial\varphi} \\ \sin\theta \sin\theta - \frac{\cos\theta}{\rho \sin\varphi} \frac{\partial\rho}{\partial\theta} + \frac{\sin\theta \cos\varphi}{\rho} \frac{\partial\rho}{\partial\varphi} \\ \cos\varphi - \frac{\sin\varphi}{\rho} \frac{\partial\rho}{\partial\varphi} \end{pmatrix} . \end{aligned}$$

3-D Point Features. This is a set $\{\boldsymbol{p}_i = (x_i, y_i, z_i)\}$ of 3-D points that describe some salient and identifiable points in the scene. They might be the centers of spheres (often used as markers), corners where three planes

intersect, or the extrema of some protrusion or indentation on a surface. They may be a subset of an initially acquired 3-D full scene point set, or they might be extracted from a range image, or they might be computed theoretical points based on extracted data features.

An early example of 3-D point features and descriptors are spin images, which are used for surface matching between point clouds and meshes. Scanned 3-D points become the vertices of the mesh and connectivity is established by the 3-D scanning geometry. A fundamental component of the surface matching representation is an oriented point, a three-dimensional point with an associated direction. *Huber* et al. define an oriented point $\boldsymbol{O}$ at a surface mesh vertex using the 3-D position of the vertex $\boldsymbol{p}$ and the surface normal at the vertex $\boldsymbol{n}$ [31.53]. Two coordinates can be calculated given an oriented point: α the radial distance to the surface normal line L and β the axial distance above the tangent plane P (Fig 31.13).

$$(\boldsymbol{x}) \mapsto (\alpha, \beta)$$
$$= \left(\sqrt{||\boldsymbol{x}-\boldsymbol{p}||^2 - [\boldsymbol{n}\cdot(\boldsymbol{x}-\boldsymbol{p})]^2}\ .\boldsymbol{n}\cdot(\boldsymbol{x}-\boldsymbol{p})\right)$$

The term spin map originates from the cylindrical symmetry of the oriented point basis; the basis can spin about its axis with no effect on the coordinates of points with respect to the basis [31.53]. A consequence of the cylindrical symmetry is that points that lie on a circle that is parallel to P and centered on L will have the same coordinates (α, β) with respect to the basis.

Seminal work on 3-D point features are the point feature histogram (PFH) descriptors. They encode a point's k-neighborhood geometrical properties by generalizing the mean curvature around the point using a multidimensional histogram of values [31.54]. This highly dimensional hyperspace aims at providing an informative signature for the feature representation, being invariant to the six-dimensional (6-D) pose of the underlying surface, and coping very well with different sampling densities or noise levels present in the neighborhood [31.55].

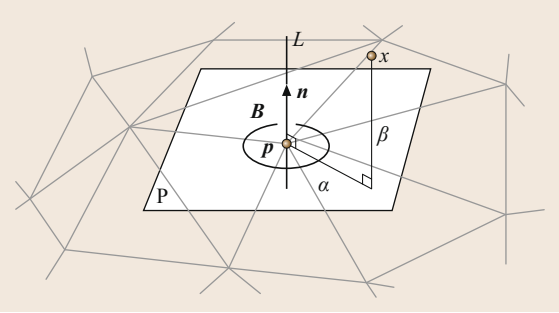


Fig. 31.13 Definition of a spin image

To formulate the new feature space, the concept of a dual-ring neighborhood is first introduced. Following the notation and the text of [31.54], let P be a set of 3-D points with x_i, y_i, z_i being the geometric coordinates. A point $\boldsymbol{p}_i$ of P is said to have a dual-ring neighborhood if

$$r_1, r_2 \in R r_1 < r_2\ ,\ \text{such that}\ \begin{cases} r_1 \to P^{k_1} \\ r_2 \to P^{k_2} \end{cases}\ ,$$

with $0 < k_1 < k_2$. The two radii r_1 and r_2 are used to determine two distinct layers of feature representations for $\boldsymbol{p}_i$. The first layer represents the surface normal at the query point from the neighborhood patch P^{k_1}. The second layer comprises the PFH as a set of angular features (Fig. 31.14)

$$\alpha = \boldsymbol{v}\cdot\boldsymbol{n}_\mathrm{t}\ ,$$
$$\varphi = \boldsymbol{u}\cdot\frac{(\boldsymbol{p}_\mathrm{t}-\boldsymbol{p}_\mathrm{s})}{d}\ ,$$
$$\theta = \arctan(\boldsymbol{w}\cdot\boldsymbol{n}_\mathrm{t}, \boldsymbol{u}\cdot\boldsymbol{n}_\mathrm{t})\ .$$

In addition to 3-D structural features, 3-D features derived from texture, for example, coregistered color images or scan reflectivities, are widely used for registration [31.56, 57]. Figure 31.15 shows scale invariant feature transform (SIFT) features extracted from a 3-D scanner with calibrated reflectivity values.

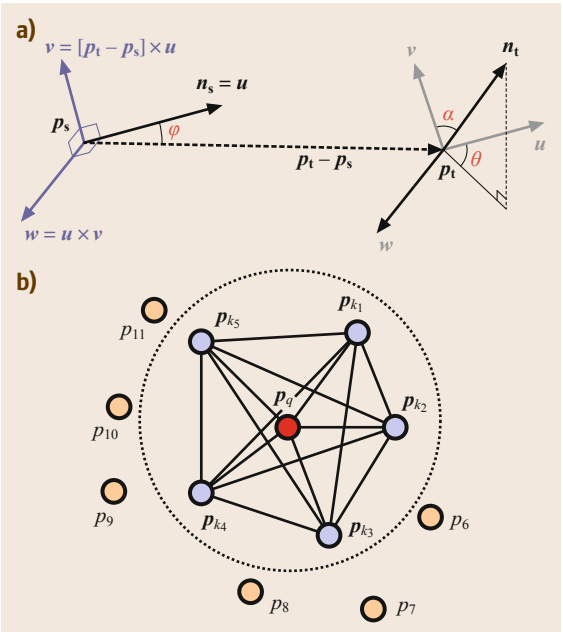


Fig.31.14a,b Angular features in PFHs (**a**) and the definition of the two regions (**b**)

Fig. 31.15 Sift features extracted from the reflectivity values of a 3-D laser scan

Planes. A planar surface may only be described by the infinite surface as given by the equation, but it may also include a description of the boundary of the surface patch. Convenient representations for robotics applications are lists of the 3-D points $\{(x_i, y_i, z_i)\}$ that form the patch boundary, or polylines, which represent the boundary by a set of connected line segments. A polyline is represented by the sequence of 3-D points $\{(x_i, y_i, z_i)\}$ that form the vertices that join the line segments.

Plane extraction, or plane fitting, is the problem of modeling a given 3-D point cloud as a set of planes that ideally explain every data point. The RANSAC algorithm is one possible algorithm. When adapted for plane finding, this algorithm selects three 3-D points at random (although exploiting some locality to the point selection algorithm can improve the efficiency of the algorithm). These three points determine a plane with a parameter vector $\boldsymbol{a}$. Test all points $\{\boldsymbol{p}\}$ in the set for belonging to the plane ($|\,\boldsymbol{p}_i \cdot \boldsymbol{a}\,| < \tau$). If enough points are close to the plane, then potentially a plane has been found. These points should also be processed to find a connected set, from which a more accurate set of plane parameters can be estimated using the least-squares algorithm given above. If a planar patch is successfully found, the points that lie in that plane are removed from the dataset. The random selection of three points then continues until no more planes are found (a bound on how many tries to make can be estimated). For example, *Schnabel* et al. [31.58] have adapted RANSAC for plane extraction and found that the algorithm performs precise and fast plane extraction, but only if the parameters have been fine-tuned properly. For their optimization, they use information , which is not readily available in point cloud data, such as normals, neighboring relations, and outlier ratios.

A further standard method for plane detection is region growing, based on a seed patch. When the scene largely consists of planes, a particularly simple approach is based on selecting a previously unused point and the set of points $\{\boldsymbol{p}_i = (x_i, y_i, z_i)\}$ in its neighborhood. A plane is fit to these points using the least-squares method (cf. normal computation). This hypothesized plane then needs to be tested for reasonableness by 1) examining the smallest eigenvalue – it should be small and of the order of the square of the expected noise level and 2) ensuring that most of the 3-D points in the fitted set lie on the plane ($|\,\boldsymbol{p}_i \cdot \boldsymbol{a}\,| < \tau$).

Larger planar regions are *grown* by locating new adjacent points $\boldsymbol{p}_i$ that lie on the plane ($|\,\boldsymbol{p}_i \cdot \boldsymbol{a}\,| < \tau$). When enough of these are found, the parameters $\boldsymbol{a}$ of the plane are re-estimated. Points on the detected plane are removed and the process is repeated with a new seed patch. This process continues until no more points can be added. Complete descriptions of planar feature extraction with region growing are given in [31.59].

Bauer and *Polthier* use the radon transform to detect planes in volume data [31.60]. The idea and the speed of the algorithm are similar to that of the standard Hough transform. The Hough transform [31.61] is a method for detecting parametrized objects. For the Hough transform, planes are represented in the Hesse normal form, using normal vectors. A plane is thereby given by a point $\boldsymbol{p}$ on the plane, the normal vector $\boldsymbol{n}$ that is perpendicular to the plane and the distance ρ to the origin

$$\rho = \boldsymbol{p} \cdot \boldsymbol{n} = p_x n_x + p_y n_y + p_z n_z = \rho\,.$$

Considering the angles between the normal vector and the coordinate system, the coordinates of $\boldsymbol{n}$ are factorized to

$$p_x \cdot \cos\theta \cdot \sin\varphi + p_y \cdot \sin\varphi \cdot \sin\theta + p_z \cdot \cos\varphi = \rho\,, \tag{31.16}$$

with θ is the angle of the normal vector on the xy-plane and φ the angle between the xy-plane and the normal vector in the z direction. φ, θ, and ρ define the 3-dimensional Hough space (θ, φ, ρ) such that each point in the Hough space corresponds to *one plane* in $\mathbb{R}^3$. To find planes in a point set, one calculates the Hough transform for each point. Given a point $\boldsymbol{p}$ in Cartesian coordinates, one finds all planes the point lies on, that is, find all the θ, φ, and ρ that satisfy (31.16). Marking these points in the Hough space, that is, leads to a 3-D sinusoid curve as shown in Fig. 31.16. The intersections of two curves in the Hough space denote the planes that are rotated around the line built by the two points. Consequently, the intersection of three curves

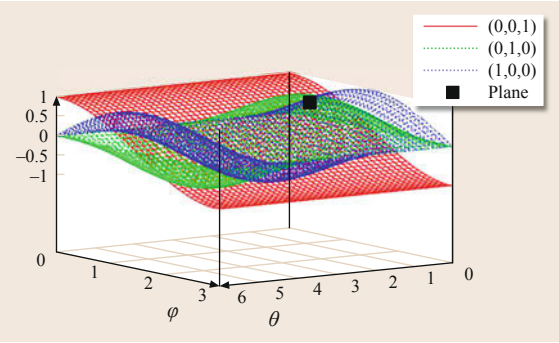


Fig. 31.16 Transformation of three points from $\mathbb{R}$ to the Hough space (θ, φ, ρ). The intersection of the curves (*marked in black*) depicts the plane spanned by the three points

in the Hough space corresponds to the polar coordinates defining the plane spanned by the three points. In Fig. 31.16, the intersection is marked in black. Given a set P of points in Cartesian coordinates, one transforms all points $\boldsymbol{p}_i \in P$ into the Hough space. The more curves intersect in $\boldsymbol{h}_j \in (\theta, \varphi, \rho)$, the more points lie on the plane represented by $\boldsymbol{h}_j$ and the higher is the probability that $\boldsymbol{h}_j$ is actually extracted from P.

The standard Hough transform is far too slow for plane detection in real-world data. A variant called randomized Hough transform is the method of choice when dealing with 3-D data due to its exceptional performance as far as runtime and quality are concerned [31.62].

Other plane extraction algorithms are highly specialized for a specific application and are not in widespread use for miscellaneous reasons. *Lakaemper* and *Latecki* [31.63] used an expectation maximization (EM) algorithm to fit planes that are initially randomly generated, *Wulf* et al. [31.64] detected planes relying on the specific properties of a sweeping laser scanner and *Yu* et al. [31.65] developed a clustering approach to solve the problem.

Triangulated Surfaces. Most commonly, surfaces are approximated by polygonal meshes, particularly triangle meshes, a standard data structure in computer graphics to represent 3-D objects. This representation describes an object or scene by a set of triangular patches. More general polygonal surface patches or even various smooth surface representations are also used, but triangles are most commonly used because they are simpler and there are inexpensive PC graphics cards that display triangles at high speed.

The triangles can be large (e.g., when representing planar surfaces) or small (e.g., when representing curved surfaces). The size chosen for the triangles reflects the accuracy desired for representing the object or scene surfaces. The triangulated surface might be complete in the sense that all observable scene or objects surfaces are represented by triangles, or there might be disconnected surface patches with or without internal holes. For grasping or navigation, you do not want any unrepresented scene surface to lie in front of the represented portion of the surface where a gripper or vehicle might collide with it. Hence, we assume that the triangulation algorithms produce patch sets that, if completely connected at the edges, implicitly bound all real scene surfaces. Figure 31.17 shows an example of a triangulated surface and the original point cloud.

The de-facto standard is the marching cubes method introduced by *Lorensen* nad *Cline* [31.66]. This algorithm subdivides the scanned volume into cubic cells or voxels. For each cell, the intersections between the cell edges and the surface are calculated. Precalculated surface patterns are then used to generate a local triangle mesh approximation. An example of such patterns are given in Fig. 31.18. To interpolate the intersections, implicit continuous surface representations like planes or splines are fitted to the local data using least-squares fits [31.67, 68]. A feature of the marching cubes algorithm is that it produces more triangles than are needed to represent an object. Hence, several mesh simplification algorithms have been introduced over the past years. Most of them define error metrics that indicate the error that a certain operation causes to the model, that is, the removal of an edge [31.69, 70]. To optimize the model, the edges causing the minimal error to the

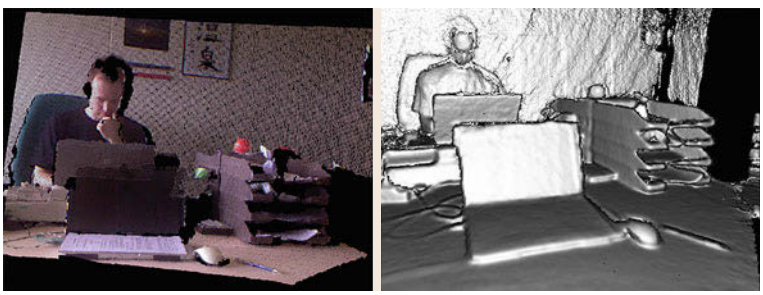


Fig. 31.17 **(a)** Image acquired with a kinect-like sensor. **(b)** Reconstruction with kinect fusion

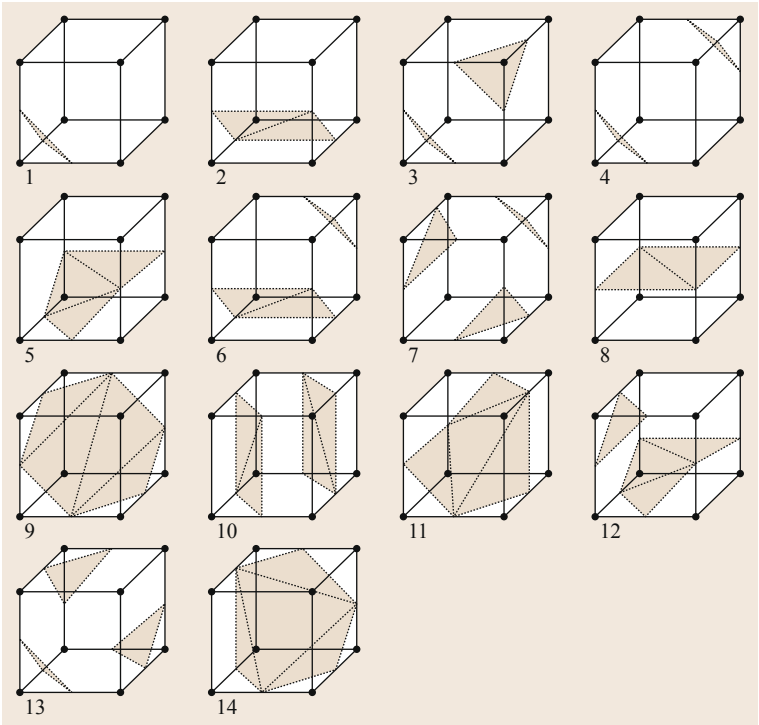


Fig. 31.18 Pattern for meshing 3-D voxels. 256 combinations are possible, but it is sufficient to model 16, iff symmetries are considered

topology are removed iteratively. Since after each edge removal new vertices have to be inserted into the mesh, the initial topology can be altered.

The kinect fusion approach modifies Hoppe's distance function [31.71]. It exploits the properties of the depth image (rectangular) to calculate normals and the associated planes. The method is massively parallelized by a GPU (graphics processing unit) implementation and can reconstruct and register meshes in real time.

3-D Lines. The 3-D lines where planar surfaces meet are features that can be easily detected by both stereo and range sensors. These features occur commonly in built environments (e.g., where walls, floors, ceilings, and doorways meet, around the edges of wall structures like notice boards, at the edges of office and warehouse furniture, etc.). They are also common on manmade objects. In the case of stereo, changes of surface shape or coloring are detected as edges, which can be matched in the stereo process to directly produce the 3-D edge. In the case of a range sensor, planar surfaces can be easily extracted from the range data (see the next section) and adjacent planar surfaces can be intersected to give the edges.

The most straightforward representation for 3-D lines is the set of points $\boldsymbol{x} = \boldsymbol{p} + \lambda \boldsymbol{v}$ for all λ, where $\boldsymbol{v}$ is a unit vector. This has 5 degrees of freedom; more complex representations for example, with 4 degrees of freedom exist [31.10].

Voxels. The voxel (volume pixel) approach represents the 3-D world by 3-D boxes/cells that indicate where there is a scene structure and where there is free space. The simplest representation is a 3-D binary array, encoded as 1 for having a structure and 0 for free space. This can be quite memory intensive, and also requires a lot of computation to check many voxels for content. A more complex but more compact representation is the hierarchical representation called the octree [31.72]. This divides the entire (bounded) rectangular space into eight rectangular subspaces called octants (Fig. 31.19). A tree data structure encodes the content of each octant as empty, full or mixed. Mixed octants are then subdivided into eight smaller rectangular octants, encoded as subtrees of the larger tree. Subdivision continues until some minimum octant size is reached. Determining whether a voxel is empty, full, or mixed depends on the sensor used, however, if no 3-D data points are located in the volume of a voxel, then it is likely to be empty. Similarly, if many 3-D points are present, then the voxel is likely to be full. Currently, many implementations using octrees for range data are available [31.73, 74]. Furthermore, these voxel representations are the basis of surface/mesh reconstruction algorithms as described earlier.

For the purpose of robot navigation, localization or grasping, only the surface and free-space voxels need to be marked accurately. The interior of objects and scene structure are largely irrelevant.

Straight Lines. While straight lines are common in man-made scenes, direct extraction from 3-D datasets is not easy. The main source of the difficulty is that the 3-D sensors often do not acquire good responses at edges of surfaces. For this reason, most 3-D line detection algorithms are indirect, whereby planes are first detected, for example, using the method of the previous section, and then adjacent planes are intersected. Adjacency can be tested by finding paths of connected pixels that lead from one plane to the other. If planes 1 and 2 contains points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ and have surface normals $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$, respectively, then the resulting intersection line has equation $\boldsymbol{x} = \boldsymbol{a} + \lambda \boldsymbol{d}$ where $\boldsymbol{a}$ is a point on the line and $\boldsymbol{d} = \frac{\boldsymbol{n}_1 \times \boldsymbol{n}_2}{||\boldsymbol{n}_1 \times \boldsymbol{n}_2||}$ is the line direction.

There are an infinite number of possible points $\boldsymbol{a}$, which can be found by solving the equations $\boldsymbol{a}'\boldsymbol{n}_1 = \boldsymbol{p}_1'\boldsymbol{n}_1$ and $\boldsymbol{a}'\boldsymbol{n}_2 = \boldsymbol{p}_2'\boldsymbol{n}_2$. A reasonable third constraint that obtains a point near $\boldsymbol{p}_2$ is the equation $\boldsymbol{a}'\boldsymbol{d} = \boldsymbol{p}_2'\boldsymbol{d}$. This gives us an infinite line. Most practical applications require a finite segment. The endpoints can be estimated by (1) finding the points on the line that lie close to observed points in both planes and then (2) finding the two extremes of those points. On the other hand, finding straight 3-D lines can be easier with a stereo sensor, as these result from matching two straight 2-D image lines.

31.3.3 Multiple-View Registration

A globally consistent representation of a robot's environment is crucial for many robotic applications. Equipped with a 3-D depth-perceiving sensor, many mobile systems gather spatial information about their local 3-D environments. Any iterative application of matching algorithms leads to inconsistencies due to sensing errors and due to the inaccuracies in the matching procedures itself. To avoid these problems, global matching algorithms are needed, taking global correspondences between range sensor data into account. Simultaneous localization and mapping (SLAM) algorithms as discussed in Chap. 46 solve a very similar problem. In addition to registering multiple views, they estimate a map. Multiple-view registration also relates to bundle adjustment in the photogrammetry community and structure from motion (SFM).

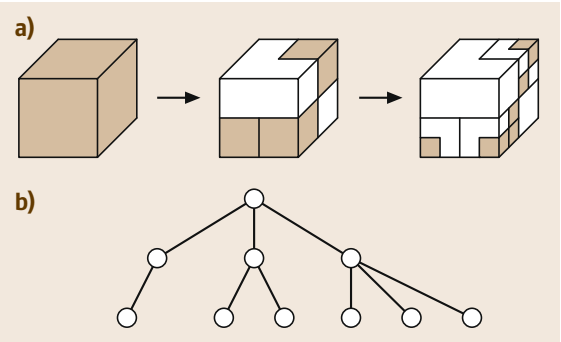


Fig. 31.19 (a) Spatial subdivisions of an octree up to level 3. Occupied leaf nodes are shaded grey. (b) The corresponding tree structure of the sparse data structure

If n-views have to be registered, any sequential application of a two point-set registration method will accumulate errors, and therefore the registration algorithm (Sect. 31.3.1) has to be extended. The global error function becomes

$$E = \sum_{l \to k} \sum_{i} \left\| (\mathbf{R}_l \boldsymbol{m}_{l,i} + \boldsymbol{t}_l) - (\mathbf{R}_k \boldsymbol{d}_{k,i} + \boldsymbol{t}_k) \right\|^2 , \tag{31.17}$$

where all views have their unique pose $(\mathbf{R}, \boldsymbol{t})$. After the point pairs for all overlapping views (l, k) have been found, (31.17) is minimized. Unfortunately, a closed-form solution for minimizing (31.17) is not know, but a small angle approximation or the helix transform yield a system of linear equations that can be solved by Choleskey decomposition [31.49]. In an ICP-like fashion after every transformation new point pairs have to be found (Algorithm 31.2).

Algorithm 31.2 The globally consistent ICP algorithm

1: **for** $i = 0$ to *maxIterations* **do**
2: find the closest point within a range $d_{\max}$ of every pair of overlapping 3-D point clouds (l, k).
3: Calculate n transformations $(\mathbf{R}, \boldsymbol{t})$ simultaneously that minimize the error function (31.17)
4: Apply the n transformations found in step 4 to all data sets.
5: Compute the difference of the quadratic error, that is, compute the difference of the value $\|E_{i-1} - E_i\|$ before and after the application of the transformation. If this difference falls below a threshold ε, terminate.
6: **end for**

A probabilistic SLAM-like notation of (31.17) was formulated in [31.75] for 2-D range scans. For each pose $\boldsymbol{X}$, the term $\bar{\boldsymbol{X}}$ denotes a pose estimate, and $\Delta\boldsymbol{X}$ is the pose error. The positional error of two poses $\boldsymbol{X}_j$

and X_k is described by

$$E_{j,k} = \sum_{i=1}^{m} \left\| X_j \oplus d_i - X_k \oplus m_i \right\|^2 .$$

Here, $\oplus$ is the compounding operation that transforms a point into the global coordinate system. For small pose differences, $E_{j,k}$ can be linearized by using a Taylor expansion. With the linearized error metric $E'_{j,k}$ and the Gaussian distribution $(\bar{E}_{j,k}, C_{j,k})$ a Mahalanobis distance that describes the global error of all the poses is constructed

$$\begin{aligned} W &= \sum_{j \to k} \left(\bar{E}_{j,k} - E'_{j,k}\right)^{\mathrm{T}} C_{j,k}^{-1} \left(\bar{E}'_{j,k} - E'_{j,k}\right) \\ &= \sum_{j \to k} \left[\bar{E}_{j,k} - \left(X'_j - X'_k\right)\right] C_{j,k}^{-1} \left[\bar{E}'_{j,k} - \left(X'_j - X'_k\right)\right], \end{aligned} \tag{31.18}$$

which can be solved efficiently using iterative least squares techniques, for example, Levenberg–Marquadt or conjugate gradient methods [31.76, 77]. The covariances are computed from point pairs. In the presence of correct covariances, (31.18) is minimized once. In case of scan matching, new pose estimates yield new closest point pairs and in turn new covariances. Iterating the process of calculating point pairs and minimization yields a stable algorithm that converges rapidly. The probabilisty notation and the global ICP notation are very similar [31.49]. The solution for 2-D range scans has been extended to to 6-DOF in [31.78].

31.3.4 Model Matching

Model matching is the process of matching some stored representation to some observed data. In the case discussed here, we assume that both are 3-D representations. Furthermore, we assume that the representations being matched are both of the same type, for example, 3-D model and scene lines. (While different types of data can also be matched, we ignore these more specialized algorithms here.)

A special case of matching is when the two structures being matched are both scene or model surfaces. The algorithm used for matching depends on the complexity the structures being matched. If the structures being matched are extended geometric entities such as planes or 3-D lines, then a discrete matching algorithm like the Interpretation Tree algorithm [31.79] can be used. It is suitable for matching small numbers (e.g., less than about 20–30) discrete objects, such as vertical edges seen in 2-D or 3-D. If there are M model and D data objects, then potentially there are M^D different matches. The key to efficient matching is to identify pairwise constraints that eliminate unsuitable matches. Constraints between pairs of model features and pairs of data features also greatly reduce the matching space. If the constraints eliminate enough features, a polynomial time algorithm results. The core of the algorithm is defined as follows. Let $\{m_i\}$ and $\{d_j\}$ be the sets po model and data features to be matched, $u(m_i, d_j)$ is true if m_i and d_j are compatible features, $b(m_i, m_j, d_k, d_l)$ is true if the four model and data features are compatible and T is the minimum number of matched features before a successful match is declared. `Pairs` is the set of successfully matched features. The function `truesizeof` counts the number of actual matches in the set, disregarding matches with the wildcard * which matches anything.

Algorithm 31.3

```
pairs=it(0,{})
if truesizeof(pairs) >= T, then success

function pairs=it(level,inpairs)
  if level >= T, then return inpairs
  if M-level+truesizeof(inpairs) < T
    then return {} % can never succeed
  for each d_i % loopD start
    if not u(m_level,d_i), then
                              continue loopD
    for each (m_k,d_l) in inpairs
      if not b(m_level,m_k,d_i,d_l)
        then continue loopD
    endfor
    % have found a~successful new pair
                                    % to add
    pairs = it(level+1,
               union(inpairs,(m_level,d_i)))
    if truesizeof(pairs) >= T, then return
  endfor % loopD end

  % no success, so try wildcard
  it(level+1,union(inpairs,(m_level,*)))
```

31.3.5 Relative Pose Estimation

Central to many tasks is the estimation of the coordinate system relative position or pose transformation between two coordinate systems. For example, this might be the pose of a scanner mounted on a mobile vehicle relative to scene landmarks. Or, it might be the relative pose of some scene features as observed in two views taken from different positions.

We present here three algorithms that cover most instances of the pose estimation process, which differ slightly based on the type of feature being matched.

Point Set Relative Pose Estimation

The ICP algorithm can be used relative pose estimation as well (Sect. 31.3.1).

Straight Line Relative Pose Estimation

If 3-D lines are the features that are extracted, then the relative pose transformation can be estimated as follows. Assume N paired lines. The first set of lines is described by direction vectors $\{\boldsymbol{e}_i\}$ and a point on each line $\{\boldsymbol{a}_i\}$. The second set of lines is described by direction vectors $\{\boldsymbol{f}_i\}$ and a point on each line $\{\boldsymbol{b}_i\}$. In this algorithm, we assume that the direction vectors on the matched segments always point the same direction (i. e., are not inverted). This can be achieved by exploiting some scene constraints, or trying all combinations and eliminating inconsistent solutions. The points $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$ need not correspond to the same point after alignment. The desired rotation matrix $\mathbf{R}$ minimizes $\sum_i || \mathbf{R}\boldsymbol{e}_i - \boldsymbol{f}_i ||^2$. Construct the $3 \times N$ matrices $\mathbf{E}$ that consists of the vectors $\{\boldsymbol{e}_i\}$ stacked up. Construct the $3 \times N$ matrices $\mathbf{F}$ in a similar way from the vectors $\{\boldsymbol{f}_i\}$. Compute the singular value decomposition $\text{svd}(\mathbf{F}\mathbf{E}') = \mathbf{U}'\mathbf{D}\mathbf{V}'$. Compute the rotation matrix $\mathbf{R} = \mathbf{V}\mathbf{U}'$. The translation estimate $\boldsymbol{t}$ minimizes the sum of the square of the distances λ_i between the rotated points $\boldsymbol{a}_i$ and corresponding line $(\boldsymbol{f}_i, \boldsymbol{b}_i)$. Define matrix $\mathbf{L} = \sum_i (\mathbf{I} - \boldsymbol{f}_i \boldsymbol{f}_i')'(\mathbf{I} - \boldsymbol{f}_i')$. Define the vector $\boldsymbol{n} = \sum_i (\mathbf{I} - \boldsymbol{f}_i \boldsymbol{f}_i')'(\mathbf{I} - \boldsymbol{f}_i \boldsymbol{f}_i')(\mathbf{R}\boldsymbol{a}_i - \boldsymbol{b}_i)$. Then the translation is $\boldsymbol{t} = -\mathbf{L}^{-1}\boldsymbol{n}$.

Plane Relative Pose Estimation

Finally, if planes are the 3-D features extracted for matching, then the relative pose transformation can be estimated as follows. Assume N paired planes. The first set of planes is described by surface normals $\{\boldsymbol{e}_i\}$ and a point on each plane $\{\boldsymbol{a}_i\}$. The second set of planes is described by surface normals $\{\boldsymbol{f}_i\}$ and a point on each plane $\{\boldsymbol{b}_i\}$. Here we assume that the surface normals always point outward from the surface. The points $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$ need not correspond to the same point after alignment. The desired rotation matrix $\mathbf{R}$ minimizes $\sum_i || \mathbf{R}\boldsymbol{e}_i - \boldsymbol{f}_i ||^2$. Construct the $3 \times N$ matrices $\mathbf{E}$ that consists of the vectors $\{\boldsymbol{e}_i\}$ stacked up. Construct the $3 \times N$ matrices $\mathbf{F}$ in a similar way from the vectors $\{\boldsymbol{f}_i\}$. Compute the singular value decomposition $\text{svd}(\mathbf{F}\mathbf{E}') = \mathbf{U}'\mathbf{D}\mathbf{V}'$ [31.45]. Compute the rotation matrix $\mathbf{R} = \mathbf{V}\mathbf{U}'$. The translation estimate $\boldsymbol{t}$ minimizes the sum of the square of the distances λ_i between the rotated point $\boldsymbol{a}_i$ and the corresponding plane $(\boldsymbol{f}_i, \boldsymbol{b}_i)$. Define matrix $\mathbf{L} = \sum_i \boldsymbol{f}_i \boldsymbol{f}_i'$. Define the vector $\boldsymbol{n} = \sum_i \boldsymbol{f}_i \boldsymbol{f}_i' (\mathbf{R}\boldsymbol{a}_i - \boldsymbol{b}_i)$. Then the translation is $\boldsymbol{t} = -\mathbf{L}^{-1}\boldsymbol{n}$.

In all of the calculations described above, we assumed normally distributed errors. For techniques to robustify these sorts of calculations, see *Zhang* [31.80].

31.3.6 3-D Applications

This section links the techniques presented above to the robotics applications of 3-D localization of parts for robot manipulation, self-localization of robot vehicles and scene understanding for robot navigation. The robotics tasks mentioned here are discussed in more detail in other chapters in the series. While this chapter focusses on robotics applications, there are many other 3-D sensing applications. An area of much current research is that of acquiring 3-D models, particularly for reverse engineering of mechanical parts [31.81], historical artifacts [31.82], buildings [31.83] and people for computer games and movies (Cyberware Whole Body X 3-D Scanner).

The key tasks in robot manipulation are:

1. Identification of grasping points (Chaps. 37 and 38),
2. Identification of a collision free grasp (Chaps. 37 and 38),
3. Recognition of parts to be manipulated (Chap. 32) and
4. Position estimation of parts for manipulation (Chaps. 32 and 42).

The key tasks in robot navigation and self-localization are:

4. Identification of a navigable groundplane (Sect. 31.4),
5. Identification of a collision free path (Chap. 47),
6. Identification of landmarks (Chap. 45) and
7. Estimation of vehicle location (Chap. 53).

The mobile and assembly robotics tasks link together rather naturally. Tasks 1 and 5 have a connection, when we consider these tasks in the context of unknown parts or paths. Part grasping requires finding regions on a part that are graspable, which usually means locally planar patches that are large enough that a gripper can make good contact with them. Similarly, navigation usually requires smooth ground regions that are large enough for the vehicle – again locally planar patches. Both tasks are commonly based on triangulated scene methods to represent the data, from which connected regions of nearly coplanar patches can be extracted. The main difference between these two tasks is the groundplane detection task is looking for a larger patch, that must be on the *ground* and upward facing.

Tasks 2 and 6 require a method of representing empty space along the proposed trajectory of the gripper contacts or the vehicle. The voxel representation is good for this task.

Tasks 3 and 7 are model matching tasks and can use the methods of Sect. 31.3.3 to match observed scene features to prestored models of known parts or scene locations. Commonly used features are large planar surfaces, 3-D edges and 3-D feature points.

Tasks 4 and 8 are pose estimation tasks and can use the methods of Sect. 31.3.5 to estimate the pose of the object relative to the sensor or vehicle (i. e., sensor) relative to the scene. Again, commonly used features are large planar surfaces, 3-D edges and 3-D feature points.

31.4 Navigation and Terrain Classification and Mapping

One of the more compelling uses for range data is for navigation of mobile robot vehicles. Range data provides information about obstacles and free space for the vehicle, in a direct geometric form. Because of the real-time constraints of navigation, it is often impractical to reconstruct a full 3-D model of the terrain using the techniques presented in this chapter. Instead, most systems use an *elevation model*. An elevation model is a tesselated 2-D representation of space, where at each cell there is information about the distribution of 3-D points in the cell. In its simplest incarnation, the elevation map just contains the mean height of range points above the nominal ground plane (Fig. 31.20). This representation is sufficient for some indoor and urban environments; more sophisticated versions that determine a local plane, scatter of points in the cell, etc., are useful for more complicated off-road driving. Elevation maps marked with obstacles have obvious utility for planning a collision-free path for the vehicle.

31.4.1 Mobile Mapping

Laser range scanning provides an efficient way to actively acquire accurate and dense 3-D point clouds of object surfaces or environments. Mobile scanning is currently used for modeling in architecture and agriculture as well as urban and regional planning. Modern systems like the Riegl VMX-450 and the Lynx mobile mapper as produced by Optech work along the same basic principal. They combine a highly accurate global positioning system (GPS), a high precision inertial measurement unit (IMU) and the odometry of the vehicle to compute the fully timestamped trajectory. Using a process called motion compensation this trajectory is then used to *unwind* the laser range measurements that were acquired by the 2-D laser scanner also mounted on the vehicle. The quality of the resulting point cloud depends on several factors:

- The calibration of the entire system, that is, the accuracy to which the position and orientation of each individual sensor in relation to the vehicle has been determined.
- The accuracy of the external positioning sensors, that is, the GPS, IMU and odometry.
- The availability of the GPS, as it may suffer temporary blackouts under bridges, in tunnels and between high-rises.
- The accuracy of the laser scanner itself.

The movement of the mobile laser scanner between time t_0 and t_n creates a trajectory $T = \{\boldsymbol{V}_0, \ldots, \boldsymbol{V}_n\}$, where $\boldsymbol{V}_i = (t_{x,i}, t_{y,i}, t_{z,i}, \theta_{x,i}, \theta_{y,i}, \theta_{z,i})$ is the 6-DOF (degree of freedom) pose of the vehicle at time t_i with $t_0 \leq t_i \leq t_n$. Using the trajectory of the vehicle a 3-D representation of the environment can be obtained by *unwinding* the laser measurements M to create the final map P. However, sensor errors in odometry, IMU and GPS as well as systematic calibration errors and the accumulation of pose errors during temporary GPS outages degrade the accuracy of the trajectory and therefore the point cloud quality. Furthermore, please note that modern systems easily create thouthands of 3-D scan slices.

These mobile mapping systems extend the early work on 3-D reconstruction with two 2-D lasers under exploitation of the robot motion [31.84]. One laser scanner is scanning horizontal, one is mounted vertically. The resultant point cloud is typically registered using the pose of the robot as corrected by the 2-D SLAM algorithm, rather than any of the 3-D registration techniques covered in this chapter. The current state of the art developed by *Bosse* et al. [31.85] for improving overall map quality of mobile mappers in the robotics

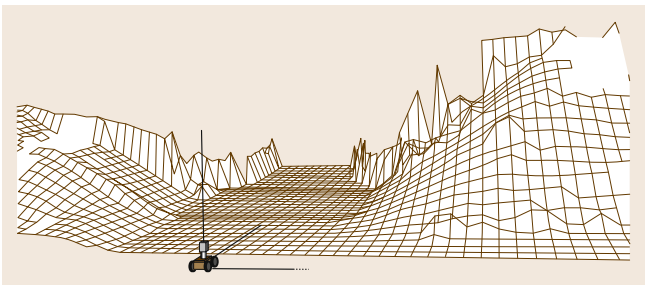


Fig. 31.20 Elevation map in urban terrain. Each cell holds the height of the terrain at that point. More extensive features can also be incorporated: slope, point variance, etc.

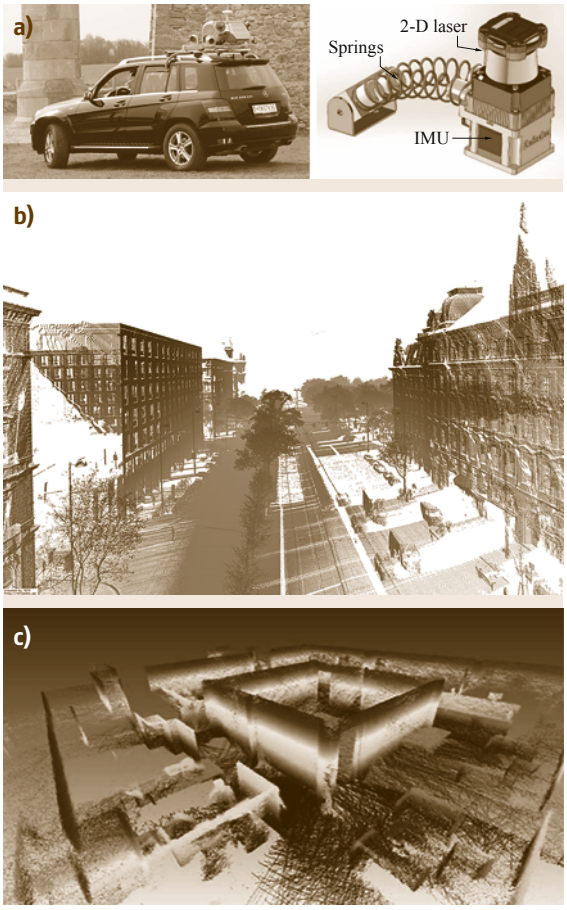


Fig. 31.21 **(a)** Mobile mapping car and **(b)** Zedebee handheld mapper. **(c)** 3-D point cloud acquired with the Riegl VMX-250 system. **(d)** 3-D point cloud from Zedebee

community is to coarsely discretize the time. This results in a partition of the trajectory into subscans that are treated rigidly. Then rigid registration algorithms like the ICP and other solutions to the SLAM problem are employed.

Figure 31.21 shows a car equipped with the Riegl VMX-450 mobile mapping system of the company Riegl laser measurement systems and the Zebedee handhald mapper from CSIRO. Figure 31.21 shows typical resulting 3-D point clouds as well.

31.4.2 Urban Navigation

In urban navigation, the environment is structured, with roads, buildings, sidewalks, and also moving objects – people and other vehicles. There are two main challenges: how to register laser scans from a fast-moving vehicle for consistent mapping, and how to detect moving objects using range scans (of course, other methods

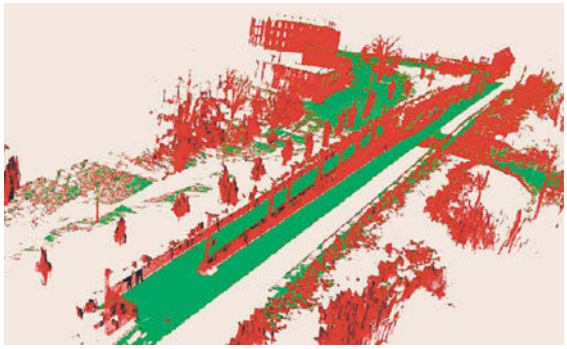


Fig. 31.22 Elevation map of an urban scene, using 10×10 cm cells. Obstacles in *red*, ground plane in *green* (after [31.88])

are also used for detecting moving objects, for example, appearance-based vision).

Outdoor vehicles can use precision GPS, inertial measurement units, and wheel odometry to keep track of their position and orientation, typically with an extended Kalman filter. This method is good enough to obviate the need for precise registration matching among scans, as long as the motion model of the vehicle, and timing from the range scanner, is used to place each scan reading in its proper position in the world model. This method also works in relatively easy offroad terrain such as in the DARPA (Defense Advanced Research Projects Agency) Grand Challenge [31.86]. In all cases, the reduction of pose estimation error is critical for good performance [31.87].

Once scan readings are registered using the vehicle pose estimation, they can be put into an elevation map, and obstacles detected using the slope and vertical extent of the range readings in the cells of the map. A complication is that there may be multiple levels of elevation in an urban setting, for example, an overpass would not be an obstacle if it were high enough. One proposal is to use multiple elevation clusters within each cell; this technique is called a *multilevel surface map* (MLS, [31.88]). Each cell in the map stores a set of surfaces represented by a mean height and variance. Figure 31.22 shows an MLS with a cell size of 10 cm^2, with ground plane and obstacles marked.

For dynamic objects, real-time stereo at 15–30 Hz can capture the motion of the objects. When the stereo rig is fixed, range background subtraction isolates just the moving objects [31.89]. When the rig is on a moving vehicle, the problem is more difficult, since the whole scene is moving with respect to the rig. It can be solved by estimating the motion of the rig with respect to the dominant rigid background of the scene. Let R, t be the motion of the rig between two frames, estimated by extracting features and matching them across the two

Fig.31.23a–c Independent motion detection from a moving platform. Reference image on panel **(a)** is forward-projected using the motion homography to the image on panel **(b)**; image on panel **(c)** is the difference with the actual image

temporal frames, using the techniques of Chap. 46. The homography $H(R,t)$ of (31.11) provides a direct projection of the disparity vectors $p_0 = [x_0, y_0, d_0, 1]$ of the first frame to their correspondences $H(R,t)p_0$ under R,t in the second frame. Using the homography allows the points in the reference frame to be directly projected onto the next frame, without translating to 3-D points. Figure 31.23 shows the projected pixels under rigid motion from a reference scene. The difference between the projected and actual pixels gives the independently moving objects (from [31.90]).

31.4.3 Rough Terrain

Rough outdoor terrain presents two challenges:

- There may be no extensive ground plane to characterize driveability and obstacles.
- Vegetation that is pliable and driveable may appear as an obstacle in range images.

Figure 31.24 shows a typical outdoor scene, with a small (1 m) robot driving through vegetation and rough ground [31.91]. Range data from stereo vision on the robot will see the top of the vegetation and some ground points below. The elevation model can be extended to look at *point statistics* within each cell, to capture the notion of a local ground plane and penetrability related to vegetation. In [31.92], for example, the set of proposed features includes:

- Major plane slope using a robust fit (Sect. 31.3.2, *3-D Feature Representations and Extraction*)
- Height difference of max and min heights
- Points above the major plane
- Density: ratio of points in the cell to rays that pass through the cell.

The density feature is interesting (and expensive to compute), and attempts to characterize vegetation such a grass or bushes, by looking at whether range readings penetrate an elevation cell. The idea of using vegetation permeability to range readings has been discussed in several other projects on off-road driving [31.94–96].

Elevation map cells can be characterized as obstacles or driveable through learning or hand-built classifiers. Among the learning techniques are neural nets [31.92] and Gaussian mixture models with

Fig. 31.24 Rough terrain, no ground plane, driveable vegatation

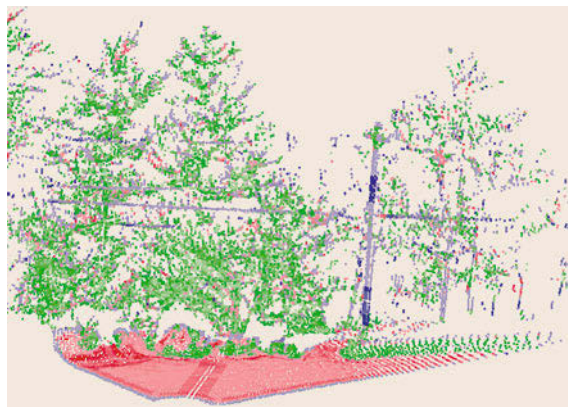


Fig. 31.25 Classification using point statistics. *Red* is planar surface, *blue* is thin linear surface, *green* is scattered penetrable surface (after [31.93])

expectation-maximization learning [31.93]. The latter work also includes a lower level of interpretation, classifying surfaces into planar patches (ground plane, solid obstacles), linear features (telephone wires), and scattered features (vegetation). Figure 31.25 shows some results from a laser-scanned outdoor scene. Linear features such as telephone wires and the telephone pole are accurately determined, as well as vegetation with high penetrability.

Some additional problems occur in rough-terrain navigation. For planar laser rangefinders that are swept over the terrain by vehicle motion, the precision of vehicle pose estimation is important for accurate reconstruction. Attitude errors of less than 0.5° can cause false positives in obstacle detection, especially for sweeps far ahead of the vehicle. In [31.87], this problem is solved by looking at the time of each laser reading, and noting a correlation between height errors and time difference in the readings.

Negative obstacles (ditches and cliffs) are difficult to detect with range information, because the sensor may not see the bottom of the obstacle. This is especially true for vehicle-mounted sensors that are not very high off the ground, and that are looking far ahead. Negative obstacles can be inferred when there is a gap in the ground plane, and a plane slanted upward at the back edge of the gap. Such artifacts can be efficiently found using column search on the disparity image [31.97].

31.5 Conclusions and Further Reading

Range sensing is an active and expanding field of research in robotics. The presence of new types of devices – flash ladars, multibeam ladars, on-camera stereo processing – and the continuing development of robust algorithms for object reconstruction, localization and mapping has helped to bring applications out of the laboratory and into the real world. Indoor navigation with ladars is already being exploited in commercial products (for example, [31.98]). As the basic capabilities become more robust, researchers are looking to perform useful tasks, such as fetching items or doing dishes [31.99].

Another set of challenges are found in less benign environments, such as urban and off-road driving (DARPA Grand Challenge and Urban Challenge [31.86]). Stereo vision and laser rangefinding also will play a role in helping to provide autonomy for a new generation of more-capable robotic platforms that rely on walking for locomotion [31.100]. The challenges are dealing with motion that is less smooth than wheeled platforms, environments that contain dynamic obstacles, and task-oriented recognition of objects.

References

31.1 H. Houshiar, J. Elseberg, D. Borrmann, A. Nüchter: A study of projections for key point based registration of panoramic terrestrial 3D laser scans, J. Geo-Spat. Inf. Sci. **18**(1), 11–31 (2015)

31.2 Velodyne: High definition lidar, http://velodynelidar.com/ (2015)

31.3 R. Stettner, H. Bailey, S. Silverman: Three-dimensional flash Ladar focal planes and time-dependent imaging, Int. J. High Speed Electron. Syst. **18**(2), 401–406 (2008)

31.4 S.B. Gokturk, H. Yalcin, C. Bamji: A time-of-flight depth sensor – system description, issues and solutions, Computer Vis. Pattern Recognit. Workshop (CVPRW) (2004)

31.5 T. Oggier, M. Lehmann, R. Kaufmannn, M. Schweizer, M. Richter, P. Metzler, G. Lang, F. Lustenberger, N. Blanc: An all-solid-state optical range camera for 3D-real-time imaging with sub-centimeter depth-resolution (SwissRanger), Proc. SPIE **5249**, 534–545 (2003)

31.6 U. Wong, A. Morris, C. Lea, J. Lee, C. Whittaker, B. Garney, R. Whittaker: Red: Comparative evaluation of range sensing technologies for underground void modeling, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 3816–3823

31.7 D.D. Lichti: A review of geometric models and self-calibration methods for terrestrial laser scanner, Bol. Cienc. Géod. **16**(1), 3–19 (2010)

31.8 G. Iddan, G. Yahav: 3D imaging in the studio (and elsewhere...), Proc. SPIE 4298 (2003) pp. 48–55

31.9 TriDiCam GmbH: http://www.tridicam.de/en.html (2015)

31.10 R. Hartley, A. Zisserman: *Multiple View Geometry in Computer Vision* (Cambridge Univ. Press, Cambridge 2000)

31.11 S. Barnard, M. Fischler: Computational stereo, ACM Comput. Surv. **14**(4), 553–572 (1982)

31.12 D. Scharstein, R. Szeliski, R. Zabih: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, Int. J. Computer Vis. **47**(1–3), 7–42 (2002)

31.13 D. Scharstein, R. Szeliski: Middlebury College Stereo Vision Research Page, http://vision.middlebury.edu/stereo (2007)

31.14 R. Zabih, J. Woodfill: Non-parametric local transforms for computing visual correspondence, Proc. Eur. Conf. Comput. Vis., Vol. 2 (1994) pp. 151–158

31.15 O. Faugeras, B. Hotz, H. Mathieu, T. Viéville, Z. Zhang, P. Fua, E. Théron, L. Moll, G. Berry, J. Vuillemin, P. Bertin, C. Proy: Real time correlation based stereo: algorithm implementations and applications, Int. J. Computer Vis. **47**(1–3), 229–246 (2002)

31.16 M. Okutomi, T. Kanade: A multiple-baseline stereo, IEEE Trans. Pattern Anal. Mach. Intell. **15**(4), 353–363 (1993)

31.17 L. Matthies: Stereo vision for planetary rovers: stochastic modeling to near realtime implementation, Int. J. Comput. Vis **8**(1), 71–91 (1993)

31.18 R. Bolles, J. Woodfill: Spatiotemporal consistency checking of passive range data, Proc. Int. Symp. Robotics Res. (1993)

31.19 P. Fua: A parallel stereo algorithm that produces dense depth maps and preserves image features, Mach. Vis. Appl. **6**(1), 35–49 (1993)

31.20 H. Moravec: Visual mapping by a robot rover, Proc. Int. Jt. Conf. Artif. Intell. (IJCAI) (1979) pp. 598–600

31.21 A. Adan, F. Molina, L. Morena: Disordered patterns projection for 3D motion recovering, Proc. Int. Conf. 3D Data Process. Vis. Transm. (2004) pp. 262–269

31.22 Videre Design LLC: http://www.videredesign.com (2007)

31.23 Point Grey Research Inc.: http://www.ptgrey.com (2015)

31.24 C. Zach, A. Klaus, M. Hadwiger, K. Karner: Accurate dense stereo reconstruction using graphics hardware, Proc. EUROGRAPHICS (2003) pp. 227–234

31.25 R. Yang, M. Pollefeys: Multi-resolution real-time stereo on commodity graphics hardware, Int. Conf. Comput. Vis Pattern Recognit., Vol. 1 (2003) pp. 211–217

31.26 K. Konolige: Small vision system. Hardware and implementation, Proc. Int. Symp. Robotics Res. (1997) pp. 111–116

31.27 Focus Robotics Inc.: http://www.focusrobotics.com (2015)

31.28 TYZX Inc.: http://www.tyzx.com (2015)

31.29 S.K. Nayar, Y. Nakagawa: Shape from Focus, IEEE Trans. Pattern Anal. Mach. Intell. **16**(8), 824–831 (1994)

31.30 M. Pollefeys, R. Koch, L. Van Gool: Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters, Int. J. Computer Vis. **32**(1), 7–25 (1999)

31.31 A. Hertzmann, S.M. Seitz: Example-based photometric stereo: Shape reconstruction with general, Varying BRDFs, IEEE Trans. Pattern Anal. Mach. Intell. **27**(8), 1254–1264 (2005)

31.32 A. Lobay, D.A. Forsyth: Shape from texture without boundaries, Int. J. Comput. Vis. **67**(1), 71–91 (2006)

31.33 Wikipedia: http://en.wikipedia.org/wiki/List_of_fastest-selling_products (2015)

31.34 K. Khoshelham, S.O. Elberink: Accuracy and resolution of kinect depth data for indoor mapping applications, Sensors **12**(5), 1437–1454 (2012)

31.35 P.J. Besl, N.D. McKay: A method for registration of 3D shapes, IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 239–256 (1992)

31.36 Y. Chen, G. Medioni: Object modeling by registration of multiple range images, Image Vis. Comput. **10**(3), 145–155 (1992)

31.37 Z. Zhang: *Iterative Point Matching for Registration of Free-Form Curves, Tech. Rep. Ser., Vol. RR-1658* (INRIA–Sophia Antipolis, Valbonne Cedex 1992)

31.38 S. Rusinkiewicz, M. Levoy: Efficient variants of the ICP algorithm, Proc. 3rd Int. Conf. 3D Digital Imaging Model. (2001) pp. 145–152

31.39 J.L. Bentley: Multidimensional binary search trees used for associative searching, Commun. ACM **18**(9), 509–517 (1975)

31.40 J.H. Friedman, J.L. Bentley, R.A. Finkel: An algorithm for finding best matches in logarithmic expected time, ACM Trans. on Math. Software **3**(3), 209–226 (1977)

31.41 M. Greenspan, M. Yurick: Approximate K-D tree search for efficient ICP, Proc. 4th IEEE Int. Conf. Recent Adv. 3D Digital Imaging Model. (2003) pp. 442–448

31.42 L. Hyafil, R.L. Rivest: Constructing optimal binary decision trees is NP-complete, Inf. Proc. Lett. **5**, 15–17 (1976)

31.43 N.J. Mitra, N. Gelfand, H. Pottmann, L. Guibas: Registration of point cloud data from a geometric optimization perspective, Proc. Eurographics/ACM SIGGRAPH Symp. Geom. Process. (2004) pp. 22–31

31.44 A. Nüchter, K. Lingemann, J. Hertzberg: Cached *k*-d tree search for ICP Algorithms, Proc. 6th IEEE Int. Conf. Recent Adv. 3D Digital Imaging Model. (2007) pp. 419–426

31.45 K.S. Arun, T.S. Huang, S.D. Blostein: Least-squares fitting of two 3-D point sets, IEEE Trans. Pattern Anal. Mach. Intell. **9**(5), 698–700 (1987)

31.46 B.K.P. Horn, H.M. Hilden, S. Negahdaripour: Closed-form solution of absolute orientation using orthonormal matrices, J. Opt. Soc. Am. A **5**(7), 1127–1135 (1988)

31.47 B.K.P. Horn: Closed-form solution of absolute orientation using unit quaternions, J. Opt. Soc. Am. A **4**(4), 629–642 (1987)

31.48 M.W. Walker, L. Shao, R.A. Volz: Estimating 3-d location parameters using dual number quaternions, J. Comput. Vis. Image Underst. **54**, 358–367 (1991)

31.49 A. Nüchter, J. Elseberg, P. Schneider, D. Paulus: Study of parameterizations for the rigid body transformations of the scan registration problem, J. Comput. Vis. Image Underst. **114**(8), 963–980 (2010)

31.50 M.A. Fischler, R.C. Bolles: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Comm. ACM **24**(6), 381–395 (1981)

31.51 N.J. Mitra, A. Nguyen: Estimating surface normals in noisy point cloud data, Proc. Symp. Comput. Geom. (SCG) (2003) pp. 322–328

31.52 H. Badino, D. Huber, Y. Park, T. Kanade: Fast and accurate computation of surface normals from

range images, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 3084–3091
31.53 D. Huber: Automatic Three-Dimensional Modeling from Reality, Ph.D. Thesis (Robotics Institute, Carnegie Mellon University, Pittsburg 2002)
31.54 R.B. Rusu: Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments, Dissertation (TU Munich, Munich 2009)
31.55 Point Cloud Library (PCL): http://www.pointclouds.org (2015)
31.56 J. Böhm, S. Becker: Automatic marker-free registration of terrestrial laser scans using reflectance features, Proc. 8th Conf. Opt. 3D Meas. Tech. (2007) pp. 338–344
31.57 N. Engelhard, F. Endres, J. Hess, J. Sturm, W. Burgard: Real-time 3D visual SLAM with a hand-held camera, Proc. RGB-D Workshop 3D Percept. Robotics at Eur. Robotics Forum (2011)
31.58 R. Schnabel, R. Wahl, R. Klein: Efficient RANSAC for point-cloud shape detection, Computer Graph. Forum (2007)
31.59 A. Hoover, G. Jean-Baptiste, X. Jiang, P.J. Flynn, H. Bunke, D. Goldgof, K. Bowyer, D. Eggert, A. Fitzgibbon, R. Fisher: An experimental comparison of range segmentation algorithms, IEEE Trans. Pattern Anal. Mach. Intell. **18**(7), 673–689 (1996)
31.60 U. Bauer, K. Polthier: Detection of planar regions in volume data for topology optimization, Proc. 5th Int. Conf. Adv. Geom. Model. Process. (2008)
31.61 P.V.C. Hough: Method and means for recognizing complex patterns, Patent US 306 9654 (1962)
31.62 D. Borrmann, J. Elseberg, A. Nüchter, K. Lingemann: The 3D Hough transform for plane detection in point clouds – A review and a new accumulator design, J. 3D Res. **2**(2), 1–13 (2011)
31.63 R. Lakaemper, L.J. Latecki: Extended EM for planar approximation of 3D data, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)
31.64 O. Wulf, K.O. Arras, H.I. Christensen, B.A. Wagner: 2D Mapping of cluttered indoor environments by means of 3D perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4204–4209
31.65 G. Yu, M. Grossberg, G. Wolberg, I. Stamos: Think globally, cluster locally:A unified framework for range segmentation, Proc. 4th Int. Symp. 3D Data Process. Vis. Transm. (2008)
31.66 W.E. Lorensen, H.E. Cline: Marching Cubes: A high resolution 3D surface construction algorithm, Computer Graph. **21**(4), 163–169 (1987)
31.67 M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C.T. Silva: Computing and rendering point set surfaces, IEEE Trans. Vis. Comput. Graph. **9**(1), 3–15 (2003)
31.68 H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle: Surface reconstruction from unorganized points, Comput. Graph. **26**(2), 71–78 (1992)
31.69 S. Melax: A Simple, fast and effective polygon reduction algorithm, Game Dev. **5**(11), 44–49 (1998)
31.70 M. Garland, P. Heckbert: Surface simplification using quadric error metrics, Proc. SIGGRAPH (1997)
31.71 S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, A. Fitzgibbon: KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera, ACM Symp. User Interface Softw. Technol. (2011)
31.72 J.D. Foley, A. van Dam, S.K. Feiner, J.F. Hughes: *Computer Graphics: Principles and Practice*, 2nd edn. (Addison-Wesley, Reading 1996)
31.73 J. Elseberg, D. Borrmann, A. Nüchter: One billion points in the cloud – An octree for efficient processing of 3D laser scans, ISPRS J. Photogramm. Remote Sens. **76**, 76–88 (2013)
31.74 A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard: OctoMap: An efficient probabilistic 3D mapping framework based on octrees, Auton. Robots **34**(3), 189–206 (2013)
31.75 F. Lu, E. Milios: Globally consistent range scan alignment for environment mapping, Auton. Robots **4**, 333–349 (1997)
31.76 K. Konolige: Large-scale map-making, Proc. Natl. Conf. Artif. Intell. (AAAI) (2004) pp. 457–463
31.77 A. Kelly, R. Unnikrishnan: Efficient construction of globally consistent ladar maps using pose network topology and nonlinear programming, Proc. Int. Symp. Robotics Res. (2003)
31.78 D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter, J. Hertzberg: Globally consistent 3d mapping with scan matching, J. Robotics Auton. Syst. **56**(2), 130–142 (2008)
31.79 E. Grimson, T. Lozano-Pérez, D.P. Huttenlocher: *Object Recognition by Computer: The Role of Geometric Constraints* (MIT Press, Cambridge 1990)
31.80 Z. Zhang: Parameter estimation techniques: a tutorial with application to conic fitting, Image Vis. Comput., Vol. 15 (1997) pp. 59–76
31.81 P. Benko, G. Kos, T. Varady, L. Andor, R.R. Martin: Constrained fitting in reverse engineering, Computer Aided Geom. Des. **19**, 173–205 (2002)
31.82 M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, D. Fulk: The digital Michelangelo project: 3D scanning of large statues, Proc. 27th Conf. Computer Graph. Interact. Tech. (SIGGRAPH) (2000) pp. 131–144
31.83 I. Stamos, P. Allen: 3-D model construction using range and image data, Proc. IEEE Conf. Computer Vis. Pattern Recognit., Vol. 1 (2000) pp. 531–536
31.84 S. Thrun, W. Burgard, D. Fox: A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping, Proc. IEEE Inf. Conf. Robotics Autom. (2000) pp. 321–328
31.85 M. Bosse, R. Zlot, P. Flick: Zebedee: Design of a spring-mounted 3-D range sensor with application to mobile mapping, IEEE Trans. Robotics **28**(5), 1104–1119 (2012)
31.86 The DARPA Grand Challenge: http://archive.darpa.mil/grandchallenge05/gcorg/index.html (2015)
31.87 S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband,

C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, P. Mahoney: Stanley: The robot that won the DARPA grand challenge, J. Field Robot. **23**(9), 661–692 (2006)

31.88 R. Triebel, P. Pfaff, W. Burgard: Multi-level surface maps for outdoor terrain mapping and loop closing, Proc. IEEE Int. Conf. Intel. Robots Syst. (IROS) (2006)

31.89 C. Eveland, K. Konolige, R. Bolles: Background modeling for segmentation of video-rate stereo sequences, Proc. Int. Conf. Computer Vis. Pattern Recog. (1998) pp. 266–271

31.90 M. Agrawal, K. Konolige, L. Iocchi: Real-time detection of independent motion using stereo, IEEE Workshop Motion (2005) pp. 207–214

31.91 K. Konolige, M. Agrawal, R.C. Bolles, C. Cowan, M. Fischler, B. Gerkey: Outdoor mapping and Navigation using stereo vision, Intl. Symp. Exp. Robotics (ISER) (2006)

31.92 M. Happold, M. Ollis, N. Johnson: Enhancing supervised terrain classification with predictive unsupervised learning, Robotics: Sci. Syst. Phila. (2006)

31.93 J. Lalonde, N. Vandapel, D. Huber, M. Hebert: Natural terrain classification using three-dimensional ladar data for ground robot mobility, J. Field Robotics **23**(10), 839–862 (2006)

31.94 R. Manduchi, A. Castano, A. Talukder, L. Matthies: Obstacle detection and terrain classification for autonomous off-road navigation, Auton. Robots **18**, 81–102 (2005)

31.95 J.-F. Lalonde, N. Vandapel, M. Hebert: Data structure for efficient processing in 3-D, Robotics: Sci. Syst. (2005)

31.96 A. Kelly, A. Stentz, O. Amidi, M. Bode, D. Bradley, A. Diaz-Calderon, M. Happold, H. Herman, R. Mandelbaum, T. Pilarski, P. Rander, S. Thayer, N. Vallidis, R. Warner: Toward reliable off road autonomous vehicles operating in challenging environments, Int. J. Robotics Res. **25**(5/6), 449–483 (2006)

31.97 P. Bellutta, R. Manduchi, L. Matthies, K. Owens, A. Rankin: Terrain perception for Demo III, Proc. IEEE Intell. Veh. Conf. (2000) pp. 326–331

31.98 KARTO: Software for robots on the move, http://www.kartorobotics.com (2015)

31.99 The Stanford Artificial Intelligence Robot: http://www.cs.stanford.edu/group/stair (2015)

31.100 Perception for Humanoid Robots: https://www.ri.cmu.edu/research_project_detail.html?project_id=595 (2015)

Multimedia Contents

32. 3-D Vision for Navigation and Grasping

Danica Kragic, Kostas Daniilidis

In this chapter, we describe algorithms for three-dimensional (3-D) vision that help robots accomplish navigation and grasping. To model cameras, we start with the basics of perspective projection and distortion due to lenses. This projection from a 3-D world to a two-dimensional (2-D) image can be inverted only by using information from the world or multiple 2-D views. If we know the 3-D model of an object or the location of 3-D landmarks, we can solve the pose estimation problem from one view. When two views are available, we can compute the 3-D motion and triangulate to reconstruct the world up to a scale factor. When multiple views are given either as sparse viewpoints or a continuous incoming video, then the robot path can be computer and point tracks can yield a sparse 3-D representation of the world. In order to grasp objects, we can estimate 3-D pose of the end effector or 3-D coordinates of the graspable points on the object.

32.1 Geometric Vision 812
32.1.1 Calibration 812
32.1.2 Pose Estimation or PnP 813
32.1.3 Triangulation 814
32.1.4 Moving Stereo 815
32.1.5 Structure from Motion 815
32.1.6 Multiple Views SfM 818
32.2 3-D Vision for Grasping 820
32.3 Conclusion and Further Reading 822
Video-References 822
References 822

With the rapid progress and cost reduction in digital imaging, cameras became the standard and probably the cheapest sensor on a robot. Unlike positioning (global positioning system – GPS), inertial (IMU), and distance sensors (sonar, laser, infrared) cameras produce the highest bandwidth of data. Exploiting information useful for a robot from such a bit stream is less explicit than in case of GPS or a laser scanner but semantically richer. In the years since the first edition of the handbook, we had significant advances in hardware and algorithms. RGB-D sensors like the Primesense Kinect enabled a new generation of full model reconstruction systems [32.1] with an arbitrary camera motion. Google's project Tango [32.2] established the state of the art in visual odometry using the latest fusion methods between visual and inertial data ([32.3] and VIDEO 120). 3-D modeling became a commodity software (see, for example, 123D Catch App from Autodesk) and the widely used open source Bundler ([32.4] VIDEO 121) has been possible by advances in wide baseline matching and bundle adjustment. Methods for wide baseline matching have been proposed for several variations of pose and structure from motion [32.5]. Last, the problem of local minima for non-minimal overconstrained solvers has been addressed by a group of method using Branch and Bound global optimization of a sum of fractions subject to convex constraints [32.6] or an L_∞-norm of the error function [32.7].

Let us consider the two main robot perception domains: navigation and grasping. Assume for example the scenario that a robot vehicle is given the task of going from place A to place B given as instruction only intermediate visual landmarks and/or GPS waypoints. The robot starts at A and has to decide where is a drivable path. Such a decision can be accomplished through

the detection of obstacles from at least two images by estimating a depth or occupancy map with a *stereo* algorithm. While driving, the robot wants to estimate its trajectory which can be accomplished with a *matching* and *structure from motion* algorithm. The result of the trajectory can be used to build a lay out of the environment through dens matching and *triangulation* which in turn can be used as a reference for a subsequent *pose estimation*. At each time instance the robot has to parse the surrounding environment for risks like pedestrians, or for objects it is searching for like a trash-can. It has to become aware of *loop closing* or a reentry if the robot has been kidnaped or blind for a while. This can be accomplished through *object and scene recognition* yielding the *what* and *where* of objects around the robot. In an extreme scenario, a vehicle can be left to explore a city and build a semantic 3-D map as well as a trajectory of all places it visited, the ultimate *visual simultaneous localization and semantic mapping* problem. In the case of grasping, the robot detects an object given a learnt representation, and subsequently, it has to estimate the *pose* of the object and in some cases its shape by *triangulation*. When a camera is not mounted on an end-effector, the *absolute orientation* between the hand the object has to be found.

In the next section we will present the geometric foundations for 3-D vision and in the last section we describe approaches for grasping.

32.1 Geometric Vision

Let us start by introducing the projection of the world to an image plane. Assume that a point in the world (X, Y, Z) has coordinates (X_{ci}, Y_{ci}, Z_{ci}) with respect to the coordinate system of a camera c_i related to each other by the following transformation

$$\begin{pmatrix} X_{ci} \\ Y_{ci} \\ Z_{ci} \end{pmatrix} = \mathbf{R}_i \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \boldsymbol{T}_i \,, \tag{32.1}$$

where $\mathbf{R}_i$ is a rotation matrix whose columns are the world axes with respect to the camera. The translation vector $\boldsymbol{T}_i$ is starting from the origin of the camera and ending at the origin of the world coordinate system. The rotation matrix is orthogonal $\mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{1}$ with determinant one. We assume that the center of projection is the origin of the coordinate system and that the optical axis is the Z_{ci} axis of the camera. If we assume that the image plane is the plane $Z_{ci} = 1$ then the image coordinates (x_i, y_i) read

$$x_i = \frac{X_{ci}}{Z_{ci}} \,, \quad y_i = \frac{Y_{ci}}{Z_{ci}} \,. \tag{32.2}$$

In practice, what we measure are the pixel coordinates (u_i, v_i) in the image which are related to image coordinates (x_i, y_i) with the affine transformation

$$u_i = f\alpha x_i + \beta y_i + c_u \,, \quad v_i = f y_i + c_v \,, \tag{32.3}$$

where f is the distance of the image plane to the projection center measured in pixels. It is also called focal length, because they are considered approximately equal. The aspect ratio α is a scaling induced by nonsquare sensor cells or different sampling rates horizontally and vertically. The skew factor β accounts for a shearing induced by a nonperfectly frontal image plane. The image center c_u, c_v is the point of intersection of the image plane with the optical axis called the image center. These five parameters are called intrinsic parameters and the process of recovering them is called intrinsic calibration. Upon recovering them we can talk about a calibrated system and we can work with the image coordinates (x_i, y_i) instead of the pixel coordinates (u_i, v_i). In many vision systems in particular on mobile robots, wide-angle lenses introduce a radial distortion around the image center which can be modelled polynomially

$$\begin{aligned} x_i^{\mathrm{dist}} &= x_i(1 + k_1 r + k_2 r^2 + k_3 r^3 + \ldots) \\ y_i^{\mathrm{dist}} &= y_i(1 + k_1 r + k_2 r^2 + k_3 r^3 + \ldots) \\ \text{where} \quad r^2 &= x_i^2 + y_i^2 \,, \end{aligned}$$

where we temporarily assumed that the image center is at (0,0). The image coordinates (x_i, y_i) in ((32.3)) have to be replaced with the distorted coordinates $(x^{\mathrm{dist}}, y^{\mathrm{dist}})$.

32.1.1 Calibration

Recovering the intrinsic parameters when we can make multiple views of a reference pattern like a checker board without variation of the intrinsic parameters has become a standard procedure using tools like the MATLAB calibration toolbox or *Zhang*'s OpenCV calibration function [32.8]. When intrinsics like the focal length vary during operation and viewing reference patterns is not practically feasible, we rely on the state of the art method by *Pollefeys* et al. [32.9, 10]. When all intrinsic are unknown on the Kruppa equations and

several stratified self-calibration approaches [32.11, 12] which require at least three views. Apart radial distortion, the projection relations shown above can be summarized in matrix form. By denoting

$$\boldsymbol{u}_i = (u_i, v_i, 1)$$

and

$$\boldsymbol{X} = (X, Y, Z, 1)$$

we obtain

$$\lambda_i \boldsymbol{u}_i = \mathbf{K}_i \begin{pmatrix} \mathbf{R}_i & \boldsymbol{T}_i \end{pmatrix} \boldsymbol{X} = \mathbf{P}\boldsymbol{X} \,, \tag{32.4}$$

where $\lambda_i = Z_{ci}$ is the depth of point $\boldsymbol{X}$ in camera coordinates and $\mathbf{P}$ is the 3×4 projection matrix. The depth λ_i which can be eliminated to obtain two equations relating the world to the pixel coordinates.

32.1.2 Pose Estimation or PnP

When we have landmarks in the world with known positions $\boldsymbol{X}$ and we can measure their projections, the problem of recovering the unknown rotation and translation in the calibrated case is called pose estimation or the Prespective-n-Point problem (PnP). Of course, it presumes the identification of the world points in the image. In robotics, the pose estimation is a variant of the localization problem in a known environment. When grasping objects of known shape PnP yields the target pose for an end-effector module the grasping point positions. We assume that a camera is calibrated and that measurements of N points are given in world coordinates $\boldsymbol{X}_{j=1..N}$ and calibrated image coordinates $\boldsymbol{x}_{j=1..N}$. Let us assume two scene points and denote the known angle between their projections $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ as δ_{12} (Fig. 32.1). Let us denote the squared distance $\|\boldsymbol{X}_i - \boldsymbol{X}_j\|^2$ with d_{ij}^2 and the lengths of $\boldsymbol{X}_j$ with d_j^2. Then cosine law reads

$$d_i^2 + d_j^2 - 2 d_i d_j \cos\delta_{ij} = d_{ij}^2 \,. \tag{32.5}$$

If we can recover d_i and d_j the rest will be an absolute orientation problem

$$\mathrm{d}_j \boldsymbol{x}_j = \mathbf{R}\boldsymbol{X}_j + \boldsymbol{T} \tag{32.6}$$

to recover translation and rotation between camera and world coordinate system.

Minimal Solution

The cosine law has two unknowns d_1 and d_2 so with three points we should be able to solve for the pose estimation problem. Indeed, three points yield a system of three quadratic equations in three unknowns, so it will have a maximum of eight solutions.

We follow here the analysis of the classic solution in [32.13] and set $d_2 = u d_1$ and $d_3 = v d_1$ and solve all three equations for d_1

$$d_1^2 = \frac{d_{23}^2}{u^2 + v^2 - 2uv\cos\delta_{23}} \,,$$
$$d_1^2 = \frac{d_{13}^2}{1 + v^2 - 2v\cos\delta_{13}} \,,$$
$$d_1^2 = \frac{d_{12}^2}{u^2 + 1 - 2u\cos\delta_{12}} \,,$$

which is equivalent to two quadratic equations in u and v

$$d_{12}^2(1 + v^2 - 2v\cos\delta_{13}) = d_{13}^2(u^2 + 1 - 2u\cos\delta_{12}) \,, \tag{32.7}$$
$$d_{13}^2(u^2 + v^2 - 2uv\cos\delta_{23}) = d_{23}^2(1 + v^2 - 2v\cos\delta_{13}) \,. \tag{32.8}$$

Solving (32.8) for u^2 and substituting in (32.7) allows solving E1 for u because u appears linearly. Substituting u back in (32.8) yields a quartic in v which can have as many as four real roots. For each v we obtain two roots for u through any of the quadratic equations yielding a maximum of eight solutions [32.13, 14]. Popular pose estimation algorithms are based either on an iterative method [32.15, 16] or linear versions using auxiliary unknowns of higher dimension [32.17, 18].

A more recent method [32.19] for n world points expresses 3-D points as the barycentric coordinates with

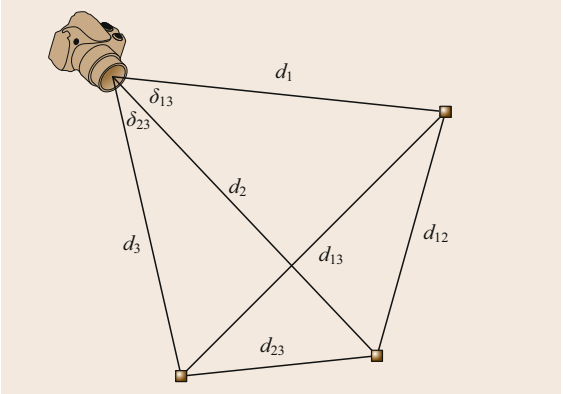


Fig. 32.1 Pose estimation problem: A camera seeing 3 points at unknown distances d_1, d_2, and d_3 with known angles between the rays and known point distances d_{12}, d_{13}, d_{23}

respect to four virtual control points

$$\boldsymbol{X}_i = \sum_{j=1}^{4} \alpha_{ij} \boldsymbol{C}_j \,, \quad \text{where} \quad \sum_{j=1}^{4} \alpha_{ij} = 1 \,.$$

A rigid transformation to the camera coordinate system leaves the barycentric coordinates invariant and a perspective projection yields

$$\lambda_i \boldsymbol{x}_i = \sum_{j=1}^{4} \alpha_{ij} \left(X_{\mathrm{c}i}, Y_{\mathrm{c}i}, Z_{\mathrm{c}i}\right)^{\mathrm{T}} \,.$$

Eliminating λ_i yields two linear equations for each point

$$\sum_{j=1}^{4} \alpha_{ij} C_{x_{\mathrm{c}j}} = \alpha_{ij} x_i C_{z_{\mathrm{c}j}}$$

$$\sum_{j=1}^{4} \alpha_{ij} C_{y_{\mathrm{c}j}} = \alpha_{ij} y_i C_{z_{\mathrm{c}j}}$$

with the coordinate triples of the control points in the camera frame being the 12 unknowns. This is a linear homogeneous system with the solution being the nullspace of a $2n \times 12$ matrix. The unknown control points are found up to a scale factor which is easily fixed because we know the inter point distances. The pose is found from absolute orientation between control points in the camera and the world frame. This yields a very efficient solution for $n \geq 6$ points but leaves you with the initial choice of the control points as a factor affecting the solution.

In case that $n \geq 4$ points lie on a plane we can compute the homography H between the world and the camera plane [32.8]. Assuming $Z = 0$ is the world plane the homography reads

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} \approx \underbrace{K \begin{pmatrix} r_1 & r_2 & T \end{pmatrix}}_{H} \begin{pmatrix} X \\ Y \\ W \end{pmatrix} \,,$$

where $r_{1,2}$ are the first two columns of the rotation matrix and $\approx$ denotes the projective equivalence, namely, for any two points $\boldsymbol{p}$ and $\boldsymbol{p}'$ in the projective plane $\boldsymbol{p} \approx \boldsymbol{p}'$ iff $\boldsymbol{p} = \lambda \boldsymbol{p}'$ for real $\lambda \neq 0$. Hence the first two columns of $K^{-1}H$

$$K^{-1}H = \begin{pmatrix} h'_1 & h'_2 & h'_3 \end{pmatrix}$$

have to be orthogonal. We seek thus an orthogonal matrix $\mathbf{R}$ that is the closest to $\begin{pmatrix} h'_1 & h'_2 & h'_1 \times h'_2 \end{pmatrix}$

$$\arg\min_{R \in \mathrm{SO}(3)} \| \mathbf{R} - \begin{pmatrix} h'_1 & h'_2 & h'_1 \times h'_2 \end{pmatrix} \|_F^2 \,.$$

If the singular value decomposition (SVD) of

$$\begin{pmatrix} h'_1 & h'_2 & h'_1 \times h'_2 \end{pmatrix} = USV^{\mathrm{T}} \,,$$

then the solution is [32.20]

$$\mathbf{R} = U \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(UV^{\mathrm{T}}) \end{pmatrix} V^{\mathrm{T}} \,. \tag{32.9}$$

The diagonal matrix is a projection from the orthogonal group O(3) to the special orthogonal group SO(3).

Last, we present a method [32.21] for n points that computes all local minima of the over constrained PnP problem. This involves solving the first derivatives explicitly with respect to the pose unknowns. To achieve this, following observation allows the elimination of the depths λ and the translation. Rigid transformation $\lambda \boldsymbol{x} = \mathbf{R}\boldsymbol{X} + \boldsymbol{T}$ can be written for n points as a linear system for $\lambda_{j=1..n}$ and the translation $\boldsymbol{T}$

$$\begin{pmatrix} \boldsymbol{x}_1 & & & -I \\ & \ddots & & \vdots \\ & & \boldsymbol{x}_n & -I \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_n \\ \boldsymbol{T} \end{pmatrix} = \begin{pmatrix} \mathbf{R}\boldsymbol{X}_1 \\ \vdots \\ \mathbf{R}\boldsymbol{X}_n \end{pmatrix} \,.$$

We can solve for the unknown depths-translation vector and back substitute it into a least squares minimization problem with respect to rotation parameters. It turns out that if we use the three Rodriguez parameters as rotation parametrization the necessary conditions for an extremum (vanishing derivatives) turn out to be three cubic equations [32.21]. Last we would like to point out to the reader that a nonlinear function of the rotation matrix can also be solved as an optimization problem on the Lie-group SO(3) [32.22–24] for the case of line correspondences.

32.1.3 Triangulation

When we know both the intrinsics and extrinsics or their summarization in matrix $\mathbf{P}$ and we measure a point we cannot recover its depth from just one camera position. Assuming that we have the projection of the same point $\boldsymbol{X}$ in two cameras

$$\lambda_1 \boldsymbol{u}_1 = \mathbf{P}_1 \begin{pmatrix} \boldsymbol{X} \\ 1 \end{pmatrix} \,,$$

$$\lambda_2 \boldsymbol{u}_2 = \mathbf{P}_2 \begin{pmatrix} \boldsymbol{X} \\ 1 \end{pmatrix} \,, \tag{32.10}$$

with known projection matrices $\mathbf{P}_1$ and $\mathbf{P}_2$ we can recover the position $\boldsymbol{X}$ in space, a process well known

as triangulation. Observe that we can achieve triangulation without decomposing the projection matrices into intrinsic and extrinsic parameters, we neeed though to remove the distortion in order to write them as above.

Having correspondences of the same point in two cameras with known projection matrices $\mathbf{P}_\mathrm{l}$ and $\mathbf{P}_\mathrm{r}$ we can solve the two projection equations for the world point $\boldsymbol{X}$. It is worth noting that each point provides two independent equations so that triangulation becomes an overconstrained problem for two views. This is not a contradiction since two rays do not intersect in general in space unless they satisfy the epipolar constraint as presented in the next paragraph. The following matrix in the left hand side has in general rank 4 unless the epipolar constraint is satisfied in which case it has rank 3.

$$\begin{pmatrix} x\mathbf{P}_\mathrm{l}(3,:)-\mathbf{P}_\mathrm{l}(1,:) \\ y\mathbf{P}_\mathrm{l}(3,:)-\mathbf{P}_\mathrm{l}(2,:) \\ x\mathbf{P}_\mathrm{r}(3,:)-\mathbf{P}_\mathrm{r}(1,:) \\ y\mathbf{P}_\mathrm{r}(3,:)-\mathbf{P}_\mathrm{r}(2,:) \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \mathbf{0} \,, \tag{32.11}$$

where $\mathbf{P}(i,:)$ means the i-th row of matrix $\mathbf{P}$.

Obviously, the homogeneous system above can be transformed into an inhomogeneous linear system with unknowns (X, Y, Z). Otherwise it can be solved by finding the vector closest to the null-space of the 4×4 matrix above using SVD. A thorough treatment of triangulation is the classic [32.25].

32.1.4 Moving Stereo

Imagine now that a rigid stereo system consisting of cameras c_l (left) and c_r (right)

$$\boldsymbol{u}_{\mathrm{l}i} \approx \mathbf{P}_\mathrm{l}\boldsymbol{X}_i \,, \tag{32.12}$$

$$\boldsymbol{u}_{\mathrm{r}i} \approx \mathbf{P}_\mathrm{r}\boldsymbol{X}_i \,, \tag{32.13}$$

is attached to a moving robot and observe this system at two time instances

$$\boldsymbol{X}_0 = \mathbf{R}_1\boldsymbol{X}_1 + \boldsymbol{T}_1 \,, \tag{32.14}$$

where $\boldsymbol{X}_0$ are point coordinates with respect to the world coordinate system, usually assumed aligned with one of the camera instances, and $\boldsymbol{X}_1$ are the coordinates of the same point with respect to the camera rig, after a motion $(\mathbf{R}_1, \boldsymbol{T}_1)$. To estimate the motion of the rig, we have to solve two correspondence problems, first, between left and right image, and second, between left (or right) at the first time instance and left (or right, respectively) at the second time instance. The left to right correspondence enable the solution of the triangulation problem at each time instance. Motion can be obtained then by solving equations (32.14) for $(\mathbf{R}_1, \boldsymbol{T}_1)$, a problem called absolute orientation. Alternatively one can avoid the second triangulation and solve the pose estimation problem between triangulated points in 3-D and points in the left image only. The most popular visual odometry system today is `libviso` [32.26] and is based on a moving stereo rig (VIDEO 122).

Absolute Orientation

The treatment for moving stereo will be short and the reader is referred to a similar treatment in the chapter about range sensing. We assume that correspondences between two time instances have been established based on tracking in the images so that we can formulate equations of the form

$$\boldsymbol{X}_2 = \mathbf{R}\boldsymbol{X}_1 + \boldsymbol{T} \,.$$

The standard way [32.20, 27] to solve this problem is to eliminate the translation by subtracting the centroids yielding

$$\boldsymbol{X}_2 - \overline{\boldsymbol{X}_2} = \mathbf{R}(\boldsymbol{X}_1 - \overline{\boldsymbol{X}_1}) \,.$$

We need at least three points in total to obtain at least two noncollinear mean-free $\boldsymbol{X}-\bar{\boldsymbol{X}}$ vectors. If we concatenate the mean free vectors for n points into an $n\times3$ matrix $A_{1,2}$ we can formulate the following minimization of the Frobenius norm

$$\min_{R\in\mathrm{SO}(3)} \|A_2 - \mathbf{R}A_1\|_F \,,$$

which is known as the Procrustes problem. It can be shown [32.20] that the solution is obtained through SVD as in (32.9) where U, V are obtained from the singular value decomposition

$$A_2A_1^\mathrm{T} = USV^\mathrm{T} \,.$$

Solutions are usually obtained with RANSAC by sampling triples of points and verification with the Procrustes method.

32.1.5 Structure from Motion

Relax now the assumption that projection matrices are known and remain with measuring and matching corresponding points $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$. This is the well known structure from motion problem or more precisely structure and 3-D-motion from 2-D motion. In photogrammetry, it is well known as relative orientation problem. Even after eliminating the λ's from equations (32.12)

or by writing them in projective equivalence form

$$\boldsymbol{u}_1 \approx \mathbf{P}_1 \begin{pmatrix} \boldsymbol{X} \\ 1 \end{pmatrix} ,$$
$$\boldsymbol{u}_2 \approx \mathbf{P}_2 \begin{pmatrix} \boldsymbol{X} \\ 1 \end{pmatrix} , \tag{32.15}$$

we realize that we if $(\boldsymbol{X}, \mathbf{P}_1, \mathbf{P}_2)$ is a solution than $(\mathbf{H}\boldsymbol{X}, \mathbf{P}_1\mathbf{H}^{-1}, \mathbf{P}_2\mathbf{H}^{-1})$ is a solution, too, where $\mathbf{H}$ an invertible 4×4 real matrix or in other words a collineation in $\mathbb{P}^3$. Even if we align the world coordinate system with the coordinate system of the first camera, which practice is common

$$\boldsymbol{u}_1 \approx \begin{pmatrix} \boldsymbol{I} & \mathbf{0} \end{pmatrix} \boldsymbol{X} ,$$
$$\boldsymbol{u}_2 \approx \mathbf{P}_2 \boldsymbol{X} , \tag{32.16}$$

we remain with the same ambiguity where $\mathbf{H}$ is of the form

$$\mathbf{H} \approx \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ h_{41} & h_{42} & h_{43} & h_{44} \end{pmatrix} , \tag{32.17}$$

with $h_{44} \neq 0$. This ambiguity is possible when the projection matrices are arbitrary rank 3 real matrices without any constraint on their elements. If we assume that we have calibrated our cameras then the projection matrices depend only on displacements

$$\boldsymbol{u}_1 \approx \begin{pmatrix} \mathbf{I} & \mathbf{0} \end{pmatrix} \boldsymbol{X} ,$$
$$\boldsymbol{u}_2 \approx \begin{pmatrix} \mathbf{R} & \boldsymbol{T} \end{pmatrix} \boldsymbol{X} , \tag{32.18}$$

and the only remaining ambiguity is the scale ambiguity where $\mathbf{H}$ looks like an identity matrix except $h_{44} = s \neq 1$ being the scale factor. In other words if $(\mathbf{R}, \boldsymbol{T}, \boldsymbol{X})$ is a solution then $(\mathbf{R}, s\boldsymbol{T}, 1/s\boldsymbol{X})$ is a solution, too. These remarks generalize in multiple views. Because, in robotics the $(\mathbf{R}, \boldsymbol{T})$ matrices correspond to location and $\boldsymbol{X}$ to mapping of the environment, the problem has the more proper term SLAM: Simultaneous localization and mapping. However, because the term SLAM has been used with a variety of sensors like sonar and laser range scanners, the term monocular SLAM is better suited to describe structure from motion from multiple views [32.28].

Epipolar Geometry

This is probably one of the most studied problems in computer vision. We constrain ourselves to the calibrated case which is most relevant to robotics applications. The necessary and sufficient condition for the intersection of the two rays $\mathbf{R}\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is that the two rays are coplanar with the baseline $\boldsymbol{T}$

$$\boldsymbol{x}_2^{\mathrm{T}}(\boldsymbol{T} \times \mathbf{R}\boldsymbol{x}_1) = 0 , \tag{32.19}$$

which is the epipolar constraint (Fig. 32.2). To avoid the scale ambiguity we assume that $\boldsymbol{T}$ is a unit vector. We proceed by summarizing the unknowns into one matrix

$$\mathbf{E} = \hat{\mathbf{T}}\mathbf{R} \tag{32.20}$$

where $\hat{\boldsymbol{T}}$ is the 3×3 skew-symmetric matrix to the vector $\boldsymbol{T}$. The $\mathbf{E}$ matrix is called the essential matrix. The epipolar constraint reads then

$$\boldsymbol{x}_2^{\mathrm{T}}\mathbf{E}\boldsymbol{x}_1 = 0 , \tag{32.21}$$

which is the equation of a line in the $\boldsymbol{x}_2$ plane with coefficients $E\boldsymbol{x}_1$ or a coefficient of a line in the $\boldsymbol{x}_1$ plane with coefficients $E^{\mathrm{T}}\boldsymbol{x}_2$. These lines are called epipolar and form pencils whose centers are the epipoles $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$, in the first and second image plane respectively. The epipoles are the intersections of the baseline with the two image planes, hence $\boldsymbol{e}_2 \approx \boldsymbol{T}$ and $\boldsymbol{e}_1 \approx -\mathbf{R}^{\mathrm{T}}\boldsymbol{T}$. Looking at the equations of the epipolar lines we can immediately infer that $E^{\mathrm{T}}\boldsymbol{e}_1 = 0$ and $E\boldsymbol{e}_2 = 0$.

The set of all essential matrices

$$\mathcal{E} = \left\{ \mathbf{E} \in \mathbb{R}^{3\times3} \mid \mathbf{E} = \widehat{\boldsymbol{T}}\mathbf{R}, \right.$$
$$\left. \text{where } \boldsymbol{T} \in \mathbb{S}^2 \text{ and } \mathbf{R} \in \mathrm{SO}(3) \right\}$$

has been characterized as a manifold of dimension 5 [32.29]. It has been proven [32.30] that

Proposition 32.1
A matrix $\mathbf{E} \in \mathbb{R}^{3\times3}$ is essential if and only if it has two singular values equal to each other and third singular value equal zero.

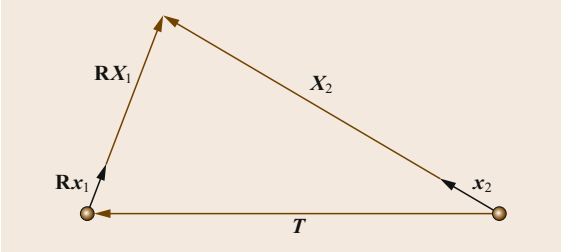


Fig. 32.2 A point is perspectively projected to calibrated image vectors $\mathbf{R}\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ which are coplanar with baseline $\boldsymbol{T}$

We present here Nister's method [32.31] for recovering an essential matrix from five point correspondences and which gained in popularity because of its suitability for RANSAC methods.

Minimal Case

We expand the epipolar constraint in terms of homogeneous coordinates $\boldsymbol{x}_1 = (x_1, y_1, z_1)$ and $\boldsymbol{x}_2 = (x_2, y_2, z_2)$ (when the points are not at infinity $z_i = 1$) and obtain

$$\begin{pmatrix} x_1\boldsymbol{x}_2^{\mathrm{T}} & y_1\boldsymbol{x}_2^{\mathrm{T}} & z_1\boldsymbol{x}_3^{\mathrm{T}} \end{pmatrix} E_{\mathrm{s}} = 0\,, \tag{32.22}$$

where E_{s} is the raw by raw stacked version of matrix $\mathbf{E}$. When we use only five point correspondences the resulting linear homogeneous system will have as a solution any vector in the four dimensional kernel of the data matrix

$$E_{\mathrm{s}} = \lambda_1\boldsymbol{u}_1 + \lambda_2\boldsymbol{u}_2 + \lambda_3\boldsymbol{u}_3 + \lambda_4\boldsymbol{u}_4\,. \tag{32.23}$$

At this point we want the matrix $\mathbf{E}$ resulting from E_{s} to be an essential matrix satisfying Proposition 32.1. It has been proven [32.30] that

Proposition 32.2

A matrix $\mathbf{E} \in \mathbb{R}^{3\times3}$ is essential if and only if

$$\mathbf{E}\mathbf{E}^{\mathrm{T}}\mathbf{E} = \frac{1}{2}\mathrm{trace}(\mathbf{E}\mathbf{E}^{\mathrm{T}})\mathbf{E}\,. \tag{32.24}$$

Though the $\det(\mathbf{E}) = 0$ constraint can be inferred from (32.24) we are still going to use it together with (32.24) to obtain ten cubic equations in the elements of $\mathbf{E}$. As described in [32.31], one can obtain a tenth degree polynomial in λ_4. The number of real roots of this polynomial are computed with a Sturm sequence. There is no proof beyond physical plausibility of the existence of at least one solution that a real root will exist at all. Several alternative 5-point solvers have been proposed since *Nister*'s paper [32.32–35] and an extensive list including code has been established by Pajdla's group [32.36].

Assuming that we have recovered an essential matrix from point correspondences, the next task is to recover an orthogonal matrix $\mathbf{R}$ and a unit vector translation $\boldsymbol{T}$ from the essential matrix. If $E = U\mathrm{diag}(\sigma, \sigma, 0)V^{\mathrm{T}}$, there are four solutions for the pair $(\widehat{T}, R)$

$$\begin{aligned}
(\widehat{\boldsymbol{T}}_1, R_1) &= (UR_{z,+\pi/2}\Sigma U^{\mathrm{T}}, UR_{z,+\pi/2}^{\mathrm{T}}V^{\mathrm{T}})\,,\\
(\widehat{\boldsymbol{T}}_2, R_2) &= (UR_{z,-\pi/2}\Sigma U^{\mathrm{T}}, UR_{z,-\pi/2}^{\mathrm{T}}V^{\mathrm{T}})\,,\\
(\widehat{\boldsymbol{T}}_1, R_2) &= (UR_{z,+\pi/2}\Sigma U^{\mathrm{T}}, UR_{z,-\pi/2}^{\mathrm{T}}V^{\mathrm{T}})\,,\\
(\widehat{\boldsymbol{T}}_2, R_1) &= (UR_{z,-\pi/2}\Sigma U^{\mathrm{T}}, UR_{z,+\pi/2}^{\mathrm{T}}V^{\mathrm{T}})\,,
\end{aligned}$$

where R_z denotes rotation around the z-axis. The four solutions can be split into two two-fold ambiguities:

- *Mirror ambiguity*: If T is a solution, then $-T$ is a solution, too. There is no way to disambiguate from the epipolar constraint: $\boldsymbol{x}_2^{\mathrm{T}}((-\boldsymbol{T}) \times R\boldsymbol{x}_1) = 0$.
- *Twisted pair ambiguity*: If R is a solution, then also $R_{T,\pi}R$ is a solution. The first image is *twisted* around the baseline 180 degrees.

These ambiguities are resolved by checking if depths of triangulated points are positive.

Critical Ambiguities

The approach with five point correspondences has a finite number of feasible (feasible means that they may produce multiple interpretations of structures in front of the camera) solutions when the points in the scene lie on a plane (a two fold ambiguity) [32.37] or when the points on the scene and the camera centers lie on a double sheet hyperboloid with the additional constraint that the camera centers lie symmetrically to the main generator of the hyperboloid [32.38]. These are inherent ambiguities which hold for any number of point correspondences when one seeks a solution for an exact essential matrix.

When someone is solving the linear least squares system for the essential matrix, a planar scene as well as the case of all points and the camera centers lying on a quadric causes a rank deficiency of the system and thus infinite solutions for $\mathbf{E}$.

Beyond the ambiguous situations, there is a considerable amount of literature regarding instabilities in the two view problem. In particularly, it has been shown [32.37, 39, 40] that a small field of view and insufficient depth variation can cause an indeterminacy in the estimation of the angle between translation and optical axis. An additional small rotation can cause a confounding between translation and rotation [32.41]. Moreover, it has been shown, that there exist local minima close to the global minimum that can fool any iterative scheme [32.42, 43].

3-Point SfM

Minimal solutions based on 5 points are still too slow to be used on mobile platforms where additional information like a reference gravity vector might be obtained from an IMU. We present here a recent solution using a reference direction and only 3 points [32.44].

We are given three image correspondences from calibrated cameras, and a single directional correspondence like the gravity vector or a vanishing point. This problem is equivalent to finding the translation vector $\boldsymbol{t}$ and a rotation angle θ around an arbitrary rotation axis.

Let us choose the arbitrary rotation axis to be $\mathbf{e}_2 = [0,1,0]^{\mathrm{T}}$. After taking the directional constraint into account, from the initial five parameters in the essential matrix, we now only have to estimate three. We can use the axis-angle parameterization of a rotation matrix to rewrite the essential matrix constraint as follows

$$\mathbf{p}_{2i}{}^{\mathrm{T}}\tilde{E}\mathbf{p}_1 = 0\,, \tag{32.25}$$

where

$$\tilde{E} = \widehat{\tilde{\boldsymbol{t}}}(I + \sin\theta\widehat{\mathbf{e}}_2 + (1-\cos\theta)\widehat{\mathbf{e}}_2^2)\,,$$

and $\tilde{\boldsymbol{t}} = (x, y, 1)$.

Each image point correspondence gives us one such equation, for a total of three equations in three unknowns (elements of $\boldsymbol{t}$ and θ). To create a polynomial system, we set $s = \sin\theta$ and $c = \cos\theta$, and add the trigonometric constraint $s^2 + c^2 - 1 = 0$, for a total of four equations in four unknowns. In order to reduce the number of unknowns, we choose the direction of the epipole by assuming that the translation vector $\tilde{\boldsymbol{t}}$ has the form $[x, y, 1]^{\top}$. This means that for each $\tilde{\boldsymbol{t}}$ that we recover, $-\tilde{\boldsymbol{t}}$ will also need to be considered as a possible solution.

Once we substitute for $\tilde{E}$ in (32.25), the resulting system of polynomial equations has the following form

$$\begin{aligned} &a_{i1}xs + a_{i2}xc + a_{i3}ys + a_{i4}yc \\ &+ a_{i5}x - a_{i2}s + a_{i1}c + a_{i6} = 0 \end{aligned} \tag{32.26}$$

for $i = 1, .., 3$, and the equation

$$s^2 + c^2 - 1 = 0\,. \tag{32.27}$$

This polynomial system can be solved in closed form and has up to four solutions. The total number of possible pose matrices arising from our formulation is therefore at most 8, when we take into account the fact that we have to consider the sign ambiguity in translation.

32.1.6 Multiple Views SfM

When we talk about simultaneous localization and mapping we obviously mean over a longer period of time. The question is how do we integrate additional frames in our 3-D motion estimation (localization) process.

To exploit multiple frames we introduce rank constraints [32.45]. We assume that the world coordinate system coincides with the coordinate system of the first frame and that a scene point is projected to $\boldsymbol{x}_i$ in the i-th frame and that its depth with respect to the 1st frame is λ_1

$$\lambda_i\boldsymbol{x}_i = \mathbf{R}_i(\lambda_1\boldsymbol{x}_1) + \boldsymbol{T}_i\,. \tag{32.28}$$

Taking the cross product with $\boldsymbol{x}_i$ and writing it for n frames yields a homogeneous system

$$\begin{pmatrix} \widehat{\boldsymbol{x}_2}\mathbf{R}_2\boldsymbol{x}_1 & \widehat{\boldsymbol{x}_2}\boldsymbol{T}_2 \\ \vdots & \vdots \\ \widehat{\boldsymbol{x}_n}\mathbf{R}_n\boldsymbol{x}_1 & \widehat{\boldsymbol{x}_2}\boldsymbol{T}_n \end{pmatrix} \begin{pmatrix} \lambda_1 \\ 1 \end{pmatrix} = 0\,, \tag{32.29}$$

that has the depth of a point in the first frame as an unknown. The $3n \times 2$ multiple view matrix has to have rank one [32.46], a constraint that infers both the epipolar and the trifocal equations. The least squares solution for the depth can easily be derived as

$$\lambda_1 = -\frac{\sum_{i=1}^{n}(\boldsymbol{x}_i \times \boldsymbol{T}_i)^{\mathrm{T}}(\boldsymbol{x}_i \times \mathbf{R}_i\boldsymbol{x}_1)}{\|\boldsymbol{x}_i \times \mathbf{R}_i\boldsymbol{x}_1\|^2}\,. \tag{32.30}$$

Given a depth for each point we can solve for motion by rearranging the multiple views constraint (32.29) as

$$\begin{pmatrix} \lambda_1^1\boldsymbol{x}_1^{1T} \otimes \hat{\boldsymbol{x}}_i^1 & \hat{\boldsymbol{x}}_i^1 \\ \vdots & \vdots \\ \lambda_1^n\boldsymbol{x}_1^{nT} \otimes \hat{\boldsymbol{x}}_i^n & \hat{\boldsymbol{x}}_i^n \end{pmatrix} \begin{pmatrix} \boldsymbol{R}_i^{\text{stacked}} \\ \boldsymbol{T}_i \end{pmatrix} = 0\,, \tag{32.31}$$

where $\boldsymbol{x}_i^n$ is the n-th image point in the i-th frame and $\mathbf{R}_i, \boldsymbol{T}_i$ is the motion from 1st to the i-th frame and $\boldsymbol{R}_i^{\text{stacked}}$ is the 12×1 vector of stacked elements of the rotation matrix $\mathbf{R}_i$. Suppose that $\boldsymbol{k}$ is the 12×1 kernel (or closest kernel in a least squares sense) of the $3n \times 12$ matrix in the left hand side obtained through singular value decomposition and let us call $\mathbf{A}$ the 3×3 matrix obtained from the first 9 elements of $\boldsymbol{k}$ and $\boldsymbol{a}$ the vector of elements 10–12. To obtain a rotation matrix we follow the SVD steps in the solution of absolute orientation (32.14) to find the closest orthogonal matrix to an arbitrary invertible matrix.

Bundle Adjustment

On top of such an approach, a bundle adjustment [32.47] minimizes the sum of all deviations between image coordinates and the backprojections of the points to be reconstructed.

$$\arg\min_{R^f, T^f, X_p} \epsilon^{\mathrm{T}} C^{-1} \epsilon\,,$$

minimized with respect to all $6(F-1)$ motions and $3N-1$ structure unknowns, where ϵ is the vector con-

taining all errors

$$\epsilon_p^f = \left(x_p^f - \frac{R_{11}^f X_p + R_{12}^f Y_p + R_{13}^f Z_p + T_x}{R_{31}^f X_p + R_{32}^f Y_p + R_{33}^f Z_p + T_z} \right.$$
$$\left. y_p^f - \frac{R_{21}^f X_p + R_{22}^f Y_p + R_{23}^f Z_p + T_y}{R_{31}^f X_p + R_{32}^f Y_p + R_{33}^f Z_p + T_z} \right)$$

and C is the error covariance matrix. We will continue with the assumption that $C = I$.

Call the objective function $\Phi(u) = \epsilon(u)^{\mathrm{T}}\epsilon(u)$ with u the vector of unknowns. Given a starting value for the vector of unknowns u we iterate with steps Δu by locally fitting a quadratic function to $\Phi(u)$

$$\Phi(u+\Delta u) = \Phi(u) + \Delta u^{\mathrm{T}} \nabla\Phi(u) + \frac{1}{2}\Delta u^{\mathrm{T}} \mathbf{H}(u)\Delta u \,,$$

where $\nabla\Phi$ is the gradient and $\mathbf{H}$ is the Hessian of Φ. The minimum of this local quadratic is at Δu satisfying

$$\mathbf{H}\delta u = -\nabla\Phi(u) \,.$$

If $\Phi(u) = \epsilon(u)^{\mathrm{T}}\epsilon(u)$ then

$$\nabla\Phi = 2\sum_i \epsilon_i(u)\nabla\epsilon_i(u)^{\mathrm{T}} = \mathbf{J}(u)^{\mathrm{T}}\epsilon \,,$$

where the Jacobian $\mathbf{J}$ consists of elements

$$\mathbf{J}_{ij} = \frac{\partial\epsilon_i}{\partial u_j} \,,$$

and the Hessian reads

$$\mathbf{H} = 2\sum_i \left(\nabla\epsilon_i(u)\nabla\epsilon_i(u)^{\mathrm{T}} + \epsilon_i(u)\frac{\partial^2\epsilon_i}{\partial u^2} \right)$$
$$= 2\left(\mathbf{J}(u)^{\mathrm{T}}\mathbf{J}(u) + \sum_i \epsilon_i(u)\frac{\partial^2\epsilon_i}{\partial u^2} \right) \approx 2\mathbf{J}(u)^{\mathrm{T}}\mathbf{J}(u)$$

by omitting quadratic terms inside the Hessian. This yields the Gauss–Newton iteration

$$(\mathbf{J}^{\mathrm{T}}\mathbf{J})\Delta u = \mathbf{J}^{\mathrm{T}}\epsilon \,,$$

involving the inversion of a $(6F+3N-7)\times(6F+3N-7)$ matrix. Bundle adjustment is about the *art* of inverting efficiently $(\mathbf{J}^{\mathrm{T}}\mathbf{J})$.

Let us split the unknown vector u into $u = (a, b)$ following [32.48] obtaining

- $6F-6$ motion unknowns a,
- $3P-1$ structure unknonws b,

and we will explain this case better if we assume two motion unknowns a_1 and a_2 corresponding to 2 frames, and 3 unknown points b_1, b_2, b_3.

For keeping symmetry in writing we do not deal here with the global reference and the global scale ambiguity.

The Jacobian for 2 frames and 3 points has 6 pairs of rows (one pair for each image projection) and 15 columns/unknowns

$$\mathbf{J} = \frac{\partial\epsilon}{\partial(a,b)} = \begin{pmatrix} \mathbf{A}_1^1 & 0 & \mathbf{B}_1^1 & 0 & 0 \\ 0 & \mathbf{A}_1^2 & \mathbf{B}_1^2 & 0 & 0 \\ \mathbf{A}_2^1 & 0 & 0 & \mathbf{B}_2^1 & 0 \\ 0 & \mathbf{A}_2^2 & 0 & \mathbf{B}_2^2 & 0 \\ \mathbf{A}_3^1 & 0 & 0 & 0 & \mathbf{B}_3^1 \\ 0 & \mathbf{A}_3^2 & 0 & 0 & \mathbf{B}_3^2 \end{pmatrix},$$

with motion and structure underbraces.

with $\mathbf{A}$ matrices being 2×6 and $\mathbf{B}$ matrices being 2×3 being Jacobians of the error ϵ_i^f of the projection of the i-th point in the f-th frame. We observe now a pattern emerging in $\mathbf{J}^{\mathrm{T}}\mathbf{J}$

$$\mathbf{J}^{\mathrm{T}}\mathbf{J} = \begin{pmatrix} \mathbf{U}^1 & 0 & \mathbf{W}_1^1 & \mathbf{W}_2^1 & \mathbf{W}_3^1 \\ 0 & \mathbf{U}^2 & \mathbf{W}_1^2 & \mathbf{W}_2^2 & \mathbf{W}_3^3 \\ .. & .. & \mathbf{V}_1 & 0 & 0 \\ .. & .. & 0 & \mathbf{V}_2 & 0 \\ .. & .. & 0 & 0 & \mathbf{V}_3 \end{pmatrix},$$

with the block diagonals for motion and structure separated. Let us rewrite the basic iteration $(\mathbf{J}^{\mathrm{T}}\mathbf{J})\Delta u = \mathbf{J}^{\mathrm{T}}\epsilon$ as

$$\begin{pmatrix} \mathbf{U} & \mathbf{W} \\ \mathbf{W}^{\mathrm{T}} & \mathbf{V} \end{pmatrix} \begin{pmatrix} \Delta a \\ \Delta b \end{pmatrix} = \begin{pmatrix} \epsilon'_a \\ \epsilon'_b \end{pmatrix},$$

and premultiply with

$$\begin{pmatrix} I & \mathbf{W}\mathbf{V}^{-1} \\ 0 & I \end{pmatrix} \begin{pmatrix} \mathbf{U} & \mathbf{W} \\ \mathbf{W}^{\mathrm{T}} & \mathbf{V} \end{pmatrix} \begin{pmatrix} \Delta a \\ \Delta b \end{pmatrix}$$
$$= \begin{pmatrix} I & \mathbf{W}\mathbf{V}^{-1} \\ 0 & I \end{pmatrix} \begin{pmatrix} \epsilon'_a \\ \epsilon'_b \end{pmatrix}$$

We find out that motion parameters can be updated separately by inverting a $6F\times 6F$ matrix

$$(\mathbf{U} - \mathbf{W}\mathbf{V}^{-1}\mathbf{W}^{\mathrm{T}})\Delta a = \epsilon'_a - \mathbf{W}\mathbf{V}^{-1}\epsilon'_b \,.$$

Each 3-D point can be updated separately by inverting a 3×3 matrix $\mathbf{V}$

$$\mathbf{V}\Delta b = \epsilon'_b - \mathbf{W}^{\mathrm{T}}\Delta a$$

It is worth mentioning that bundle adjustment though extremely slow captures the correlation between motion estimates and structure (3-D points) estimates which is artificially hidden in the iterative scheme in (32.29).

The largest scale motion estimation and registration of views has been performed by *Teller* et al. [32.49] with a decoupled computation first of relative rotations and finally of relative translations. The above multiple view SfM techniques can also be applied in a sliding window mode in time. *Davison* et al. [32.28] showed the first real-time recursive approach by decoupling the direction of the viewing rays from the depth unknowns. For other recursive approaches the reader is referred to the corresponding SLAM chapter.

32.2 3-D Vision for Grasping

In this section we will move from the basic geometry required for grasping to the main 3-D vision challenges associated with the limited knowledge we might have about the shape of the object as well as the actual selection of 3-D grasping poses.

Naturally, object grasping and manipulation is closely related to general scene understanding and problems such as object detection, recognition, categorization and pose estimation. Taking all the above, there are very few approaches that address all the problems in a single system. One example, reported in [32.50], addresses the problem of enabling transfer of grasp knowledge between object categories, defined using both their physical properties and functionality. This is a challenging problem given that a number of objects with similar physical properties afford different tasks. An example can be a screwdriver and a carrot that are structurally alike, but only the former can be used as a tool, or a ball and an orange where only the latter affords eating (Fig. 32.3).

In relation to object grasping in particular, there are methods that assume that full 3-D model of the object is available and concentrate on grasp synthesis solely. In addition, many of the approaches conduct experiments in a simulated environment without working with real sensory data. However, the knowledge generated in simulation can also be applied later onto sensory data. Another group of approaches considers grasp synthesis on real sensory data directly, dealing with problems such as noise, occlusions and missing data.

If the object to be grasped is known, there are approaches that store a database of grasp hypotheses, generated either in simulation or through experiments in a real setting. Most of the approaches assume that a 3-D mesh of the object is available and the challenge is then to automatically generate a set of feasible grasp hypotheses. This involves sampling the infinite space of possible hand configurations and ranking the resulting grasps according to some quality metric.

To simplify the process, a common approach is to approximate object's shape with a constellation of primitives such as spheres, cones, cylinders, boxes or superquadrics [32.51–55]. The purpose of using shape primitives is to reduce the number of candidate grasps and thus prune the search space for finding the optimal set of grasp hypotheses.

One example, shown in Fig. 32.4 and reported in [32.52], decomposes a point cloud from a stereo camera into a constellation of boxes. Grasp planning is performed directly on the boxes which reduces the number of potential grasps. *El-Khoury* and *Sahbani* [32.56] distinguish between graspable and nongraspable parts of an object where each part is represented by fitting a superquadric to the point cloud data. *Pelossof* et al. [32.57] approximate an object with a single superquadric and use a Support Vector Machines based approach to search for the grasp that maximizes the grasp quality. *Boularias* et al. [32.58] model an object as a Markov random field (MRF) in which the nodes are points from the point cloud and edges are spanned between the six nearest neighbors of a point. A node in the MRF carries either one of the two labels: a good or a bad grasp location. *Detry* et al.[32.59] model the object as a constellation of local multimodal contour descriptors. The set of associated grasp hypotheses is modeled as a nonparametric density function in the space of six-dimensional (6-D) gripper poses, referred to as a *bootstrap* density. *Papazov* et al. [32.60] demon-

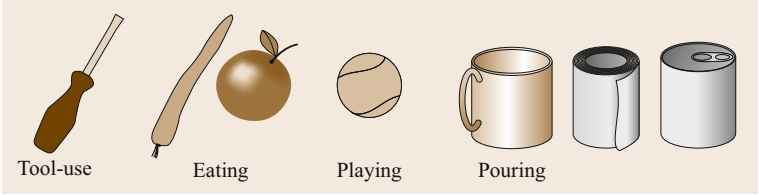


Fig. 32.3 Examples of physically similar objects that afford different tasks

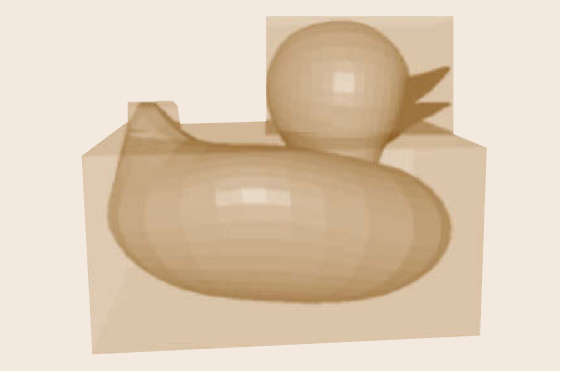


Fig. 32.4 Generation of grasp candidates through object shape approximation and decomposition from (after [32.52])

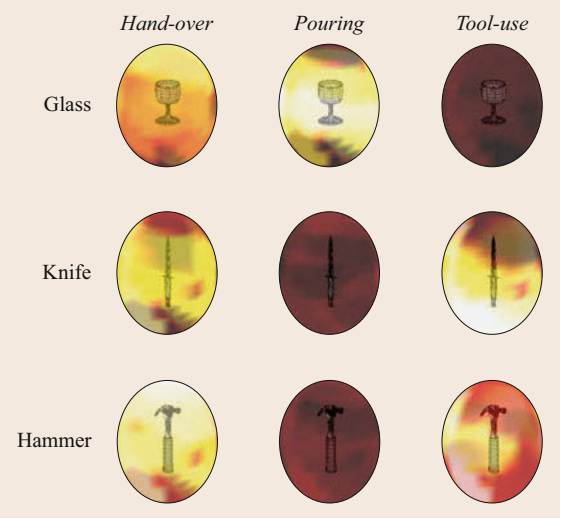


Fig. 32.5 Ranking of approach vectors on different objects given a specific task. The brighter an area the higher the rank. The darker an area, the lower the rank (after [32.62])

strates 3-D object recognition and pose estimation in a grasping scenario considering cluttered scenes. *Weisz* and *Allen* [32.61] proposes a metric suitable for predicting grasp stability under pose uncertainty.

There are several approaches that deal specifically with incomplete point clouds. *Marton* et al. [32.63] exploit symmetry by fitting a curve to a cross section of the point cloud. *Rao* et al. [32.64] concentrates of depth segmentation and sample grasp points from the surface of a segmented object using surface normals. *Bohg* et al. [32.65] presents a related approach that reconstructs full object shape assuming planar symmetry and generates grasps based on the global shape of the object. *Bone* et al. [32.66] makes no prior assumption about the shape of the object and apply shape carving for generating a parallel-jaw gripper grasps. *Hsiao* et al. [32.67] employs heuristics for generating grasp hypotheses dependent on the shape of the point cloud. Recent work in [32.68] identifies regions that afford force closure grasps by evaluating local curvature of the objects to create an initial opposing grasp with two or three fingers, dependent on the relative size of the object with respect to the hand. *Richtsfeld* and *Vincze* [32.69] uses a stereo-camera setup to generate a 3-D representation of a scene with several objects and then generates various top grasps on object candidates. *Maldonado* et al. [32.70] use time-of-flight range data, model objects using 3-D Gaussians and rely on finger torque information during grasping to monitor the grasp execution. *Stückler* et al. [32.71] generate grasp hypotheses based on eigenvectors of the object's *footprints* that are generated by projecting the 3-D object point cloud onto the supporting surface. The work of [32.72] presents a system for general scene understanding used for grasp planning and execution. The system uses a bottom-up grouping approach where contour and surface structures are used as the basis for grasp planning. The work builds upon previous work presented in [32.73].

Most of the recent work concentrates on grasp generalization either by observing human grasping or through off- and on-line learning directly on the robot. *Kroemer* et al. [32.74] demonstrates generalization capabilities using a pouring task scenario. The goal of the approach is to find a part of the object that is most likely to afford the demonstrated action. The learning method is based on the kernel logistic regression. *Herzog* et al. [32.75] stores a set of local templates of object that a human is interacting with. If a local part of an object segmented online is similar to a template in the database, the associated grasp hypothesis is executed. *Song* et al. [32.62] approach the problem of inferring a full grasp configuration in relation to a specific task the object is intended for. As in [32.76], the joint distribution over various grasping variables is modeled as a Bayesian network. Additional variables like task, object category and task constraints are introduced. The structure of this model is learned given a large number of grasp examples generated in a simulator and annotated with grasp quality metrics as well as suitability for a specific task. The learned quality of grasps on specific objects given a task is visualized in Fig. 32.5.

32.3 Conclusion and Further Reading

As main additional sources of reading, we recommend the textbooks by *Hartley* and *Zisserman* [32.12], *Ma* et al. [32.46], *Faugeras* [32.77], and *Faugeras* and *Luong* [32.11]. The reader is referred to Chap. 5 for fundamentals of estimation, to Chap. 35 for sensor fusion, to Chap. 34 for visual servoing, to Chap. 31 for range sensing, to Chap. 45 for 3-D models of the world, and to Chap. 46 for SLAM.

3-D vision is a rapidly advancing field and in this chapter we have covered only geometric approaches based on RGB cameras. Although depth sensors will become ubiquitous indoors and might be outdoors as well, RGB cameras remain formidable because of the higher number and larger diversity of features that can be matched and used for pose estimation and 3-D-modelling. Long range sensing can still be covered from motion with large translation while active sensors are constrained in terms of energy reflected from the environment.

Video-References

VIDEO 120 Google's project Tango
available from http://handbookofrobotics.org/view-chapter/32/videodetails/120

VIDEO 121 Finding paths through the world's photos
available from http://handbookofrobotics.org/view-chapter/32/videodetails/121

VIDEO 122 LIBVISO: Visual odometry for intelligent vehicles
available from http://handbookofrobotics.org/view-chapter/32/videodetails/122

VIDEO 123 Parallel tracking and mapping for small AR workspaces (PTAM)
available from http://handbookofrobotics.org/view-chapter/32/videodetails/123

VIDEO 124 DTAM: Dense tracking and mapping in real-time
available from http://handbookofrobotics.org/view-chapter/32/videodetails/124

VIDEO 125 3-D models from 2-D video – automatically
available from http://handbookofrobotics.org/view-chapter/32/videodetails/125

References

32.1 S. Izadi, R.A. Newcombe, D. Kim, O. Hilliges, D. Molyneaux, S. Hodges, P. Kohli, J. Shotton, A.J. Davison, A. Fitzgibbon: Kinectfusion: Real-time dynamic 3D surface reconstruction and interaction, ACM SIGGRAPH 2011 Talks (2011) p. 23

32.2 Google: Atap project tango, https://www.google.com/atap/projecttango (2014)

32.3 J.A. Hesch, D.G. Kottas, S.L. Bowman, S.I. Roumeliotis: Camera-IMU-based localization: Observability analysis and consistency improvement, Int. J. Robotics Res. **33**(1), 182–201 (2014)

32.4 N. Snavely, S.M. Seitz, R. Szeliski: Modeling the world from internet photo collections, Int. J. Comput. Vis. **80**(2), 189–210 (2008)

32.5 Z. Kukelova, M. Bujnak, T. Pajdla: Polynomial eigenvalue solutions to minimal problems in computer vision, IEEE Trans. Pattern Anal. Mach. Intell. **34**(7), 1381–1393 (2012)

32.6 F. Kahl, S. Agarwal, M.K. Chandraker, D. Kriegman, S. Belongie: Practical global optimization for multiview geometry, Int. J. Comput. Vis. **79**(3), 271–284 (2008)

32.7 R.I. Hartley, F. Kahl: Global optimization through rotation space search, Int. J. Comput. Vis. **82**(1), 64–79 (2009)

32.8 Z. Zhang: A flexible new technique for camera calibration, IEEE Trans. Pattern Anal. Mach. Intell. **22**, 1330–1334 (2000)

32.9 M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, R. Koch: Visual modeling with a hand-held camera, Int. J. Comput. Vis. **59**, 207–232 (2004)

32.10 M. Pollefeys, L. Van Gool: Stratified self-calibration with the modulus constraint, IEEE Trans. Pattern Anal. Mach. Intell. **21**, 707–724 (1999)

32.11 O. Faugeras, Q.-T. Luong, T. Papadopoulo: *The Geometry of Multiple Images: The Laws That Govern the Formation of Multiple Images of a Scene and Some of Their Applications* (MIT Press, Cambridge 2001)

32.12 R. Hartley, A. Zisserman: *Multiple View Geometry* (Cambridge Univ. Press, Cambridge 2000)

32.13 K. Ottenberg, R.M. Haralick, C.-N. Lee, M. Nolle: Review and analysis of solutions of the three-point perspective problem, Int. J. Comput. Vis. **13**, 331–356 (1994)

32.14 M.A. Fischler, R.C. Bolles: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, ACM Commun. **24**, 381–395 (1981)

32.15 R. Kumar, A.R. Hanson: Robust methods for estimaging pose and a sensitivity analysis, Comput. Vis. Image Underst. **60**, 313–342 (1994)
32.16 C.-P. Lu, G. Hager, E. Mjolsness: Fast and globally convergent pose estimation from video images, IEEE Trans. Pattern Anal. Mach. Intell. **22**, 610–622 (2000)
32.17 L. Quan, Z. Lan: Linear n-point camera pose determination, IEEE Trans. Pattern Anal. Mach. Intell. **21**, 774–780 (1999)
32.18 A. Ansar, K. Daniilidis: Linear pose estimation from points and lines, IEEE Trans. Pattern Anal. Mach. Intell. **25**, 578–589 (2003)
32.19 V. Lepetit, F. Moreno-Noguer, P. Fua: EPNP: An accurate $o(n)$ solution to the PNP problem, Int. J. Comput. Vis. **81**(2), 155–166 (2009)
32.20 G.H. Golub, C.F. van Loan: *Matrix Computations* (Johns Hopkins Univ. Press, Baltimore 1983)
32.21 J.A. Hesch, S.I. Roumeliotis: A direct least-squares (dls) method for pnp, IEEE Int. Conf. Comput. Vis. (ICCV) (2011) pp. 383–390
32.22 C.J. Taylor, D.J. Kriegman: *Minimization on the Lie Group SO(3) and Related Manifolds* (Yale University, New Haven 1994)
32.23 P.-A. Absil, R. Mahony, R. Sepulchre: *Optimization Algorithms on Matrix Manifolds* (Princeton Univ. Press, Princeton 2009)
32.24 Y. Ma, J. Košecká, S. Sastry: Optimization criteria and geometric algorithms for motion and structure estimation, Int. J. Comput. Vis. **44**(3), 219–249 (2001)
32.25 R.I. Hartley, P. Sturm: Triangulation, Comput. Vis. Image Underst. **68**(2), 146–157 (1997)
32.26 B. Kitt, A. Geiger, H. Lategahn: Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme, IEEE Intell. Veh. Symp. (IV) (2010)
32.27 B.K.P. Horn, H.M. Hilden, S. Negahdaripour: Closed-form solution of absolute orientation using orthonormal matrices, J. Opt. Soc. Am. A **5**, 1127–1135 (1988)
32.28 A.J. Davison, I.D. Reid, N.D. Molton, O. Stasse: Monoslam: Real-time single camera SLAM, IEEE Trans. Pattern Anal. Mach. Intell. **29**(6), 1052–1067 (2007)
32.29 R. Tron, K. Daniilidis: On the quotient representation for the essential manifold, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2014) pp. 1574–1581
32.30 T.S. Huang, O.D. Faugeras: Some properties of the E matrix in two-view motion estimation, IEEE Trans. Pattern Anal. Mach. Intell. **11**, 1310–1312 (1989)
32.31 D. Nister: An efficient solution for the five-point relative pose problem, IEEE Trans. Pattern Anal. Mach. Intell. **26**, 756–777 (2004)
32.32 H. Li, R. Hartley: Five-point motion estimation made easy, IEEE 18th Int. Conf. Pattern Recognit. (ICPR), Vol. 1 (2006) pp. 630–633
32.33 Z. Kukelova, M. Bujnak, T. Pajdla: Polynomial eigenvalue solutions to the 5-pt and 6-pt relative pose problems, BMVC (2008) pp. 1–10
32.34 H. Stewenius, C. Engels, D. Nistér: Recent developments on direct relative orientation, ISPRS J. Photogramm. Remote Sens. **60**(4), 284–294 (2006)
32.35 D. Batra, B. Nabbe, M. Hebert: An alternative formulation for five point relative pose problem, IEEE Workshop Motion Video Comput. (2007) pp. 21–21
32.36 Center for Machine Perception, Minimal problems in computer vision; http://cmp.felk.cvut.cz/minimal/5_pt_relative.php
32.37 S. Maybank: *Theory of Reconstruction from Image Motion* (Springer, Berlin, Heidelberg 1993)
32.38 S.J. Maybank: The projective geometry of ambiguous surfaces, Phil. Trans. Royal Soc. Lond. A **332**(1623), 1–47 (1990)
32.39 A. Jepson, D.J. Heeger: A fast subspace algorithm for recovering rigid motion, Proc. IEEE Workshop Vis. Motion, Princeton (1991) pp. 124–131
32.40 C. Fermüller, Y. Aloimonos: Algorithmic independent instability of structure from motion, Proc. 5th Eur. Conf. Comput. Vision, Freiburg (1998)
32.41 K. Daniilidis, M. Spetsakis: Understanding noise sensitivity in structure from motion. In: *Visual Navigation*, ed. by Y. Aloimonos (Lawrence Erlbaum, Mahwah 1996) pp. 61–88
32.42 S. Soatto, R. Brockett: Optimal structure from motion: Local ambiguities and global estimates, IEEE Conf. Comput. Vis. Pattern Recognit., Santa Barbara (1998)
32.43 J. Oliensis: A new structure-from-motion ambiguity, IEEE Trans. Pattern Anal. Mach. Intell. **22**, 685–700 (1999)
32.44 O. Naroditsky, X.S. Zhou, J. Gallier, S. Roumeliotis, K. Daniilidis: Two efficient solutions for visual odometry using directional correspondence, IEEE Trans. Patterns Anal. Mach. Intell. (2012)
32.45 Y. Ma, K. Huang, R. Vidal, J. Kosecka, S. Sastry: Rank conditions of the multiple view matrix, Int. J. Comput. Vis. **59**(2), 115–139 (2004)
32.46 Y. Ma, S. Soatto, J. Kosecka, S. Sastry: *An Invitation to 3-D Vision: From Images to Geometric Models* (Springer, Berlin, Heidelberg 2003)
32.47 W. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon: Bundle adjustment – A modern synthesis, Lect. Notes Comput. Sci **1883**, 298–372 (2000)
32.48 M. Lourakis, A. Argyros: *The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg–Marquard Method*, Tech. Rep, Vol. 340 (ICS/FORTH, Heraklion 2004)
32.49 S. Teller, M. Antone, Z. Bodnar, M. Bosse, S. Coorg: Calibrated, registered images of an extended urban area, Int. Conf. Comput. Vis. Pattern Recognit., Kauai, Vol. 1 (2001) pp. 813–820
32.50 D. Kragic, M. Madry, D. Song: From object categories to grasp transfer using probabilistic reasoning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1716–1723
32.51 A.T. Miller, S. Knoop, H.I. Christensen, P.K. Allen: Automatic grasp planning using shape primitives, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 1824–1829
32.52 K. Hübner, D. Kragic: Selection of robot pre-grasps using box-based shape approximation, IEEE/RSJ

Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 1765–1770
32.53 C. Dunes, E. Marchand, C. Collowet, C. Leroux: Active rough shape estimation of unknown objects, IEEE Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 3622–3627
32.54 M. Przybylski, T. Asfour: Unions of balls for shape approximation in robot grasping, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Taipei (2010) pp. 1592–1599
32.55 C. Goldfeder, P.K. Allen, C. Lackner, R. Pelossof: Grasp Planning Via Decomposition Trees, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 4679–4684
32.56 S. El-Khoury, A. Sahbani: Handling objects by their handles, IEEE/RSJ Int. Conf. Intell. Robots Syst. Workshop Grasp Task Learn. Imitation (2008)
32.57 R. Pelossof, A. Miller, P. Allen, T. Jebera: An SVM learning approach to robotic grasping, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 3512–3518
32.58 A. Boularias, O. Kroemer, J. Peters: Learning robot grasping from 3-d images with markov random fields, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 1548–1553
32.59 R. Detry, E. Başeski, N. Krüger, M. Popović, Y. Touati, O. Kroemer, J. Peters, J. Piater: Learning object-specific grasp affordance densities, IEEE Int. Conf. Dev. Learn. (2009) pp. 1–7
32.60 C. Papazov, S. Haddadin, S. Parusel, K. Krieger, D. Burschka: Rigid 3D geometry matching for grasping of known objects in cluttered scenes, Int. J. Robotics Res. **31**(4), 538–553 (2012)
32.61 J. Weisz, P.K. Allen: Pose error robust grasping from contact wrench space metrics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 557–562
32.62 D. Song, C.H. Ek, K. Hübner, D. Kragic: Multivariate discretization for bayesian network structure learning in robot grasping, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 1944–1950
32.63 Z.C. Marton, D. Pangercic, N. Blodow, J. Kleinehellefort, M. Beetz: General 3D modelling of novel objects from a single view, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 3700–3705
32.64 D. Rao, V. Le Quoc, T. Phoka, M. Quigley, A. Sudsang, A.Y. Ng: Grasping novel objects with depth segmentation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Taipei (2010) pp. 2578–2585
32.65 J. Bohg, M. Johnson-Roberson, B. León, J. Felip, X. Gratal, N. Bergström, D. Kragic, A. Morales: Mind the gap – Robotic grasping under incomplete observation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)
32.66 G.M. Bone, A. Lambert, M. Edwards: Automated Modelling and Robotic Grasping of Unknown Three-Dimensional Objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 292–298
32.67 K. Hsiao, S. Chitta, M. Ciocarlie, E.G. Jones: Contact-reactive grasping of objects with partial shape information, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 1228–1235
32.68 M.A. Roa, M.J. Argus, D. Leidner, C. Borst, G. Hirzinger: Power grasp planning for anthropomorphic robot hands, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)
32.69 M. Richtsfeld, M. Vincze: Grasping of Unknown Objects from a Table Top, ECCV Workshop Vis. Action: Effic. Strateg. Cogn. Agents Complex Environ. (2008)
32.70 A. Maldonado, U. Klank, M. Beetz: Robotic grasping of unmodeled objects using time-of-flight range data and finger torque information, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 2586–2591
32.71 J. Stückler, R. Steffens, D. Holz, S. Behnke: Real-time 3d perception and efficient grasp planning for everyday manipulation tasks, Eur. Conf. Mob. Robots (ECMR) (2011)
32.72 G. Kootstra, M. Popovic, J.A. Jørgensen, K. Kuklinski, K. Miatliuk, D. Kragic, N. Kruger: Enabling grasping of unknown objects through a synergistic use of edge and surface information, Int. J. Robotics Res. **31**(10), 1190–1213 (2012)
32.73 D. Kraft, N. Pugeault, E. Baseski, M. Popovic, D. Kragic, S. Kalkan, F. Wörgötter, N. Krueger: Birth of the object: Detection of objectness and extraction of object shape through object action complexes, Int. J. Humanoid Robotics **pp**, 247–265 (2009)
32.74 O. Kroemer, E. Ugur, E. Oztop, J. Peters: A Kernel-based Approach to Direct Action Perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)
32.75 A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, T. Asfour, S. Schaal: Template-based learning of grasp selection, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)
32.76 L. Montesano, M. Lopes, A. Bernardino, J. Santos-Victor: Learning object affordances: From sensory–motor coordination to imitation, IEEE Trans. Robotics **24**(1), 15–26 (2008)
32.77 O. Faugeras: *Three-Dimensional Computer Vision* (MIT Press, Cambridge 1993)

Multimedia Contents

33. Visual Object Class Recognition

Michael Stark, Bernt Schiele, Aleš Leonardis

Object class recognition is among the most fundamental problems in computer vision and thus has been researched intensively over the years. This chapter is mostly concerned with the recognition and detection of basic level object classes such as cars, persons, chairs, or dogs. We will review the state of the art and in particular discuss the most promising methods available today.

33.1 **Object Classes** 825
33.2 **Review of the State of the Art** 826
33.2.1 Bag-of-Words Models 826
33.2.2 Spatial Pyramid Matching 828
33.2.3 Flexible Part Models 829
33.2.4 Histogram of Oriented Gradients 831
33.2.5 The Deformable Part Model 832
33.2.6 Convolutional Neural Networks 834
33.2.7 Fine-Grained Categorization 836
33.2.8 Datasets 836
33.3 **Discussion and Conclusions** 837
33.3.1 Task 837
33.3.2 Data 838
33.3.3 Efficiency 838
References 838

33.1 Object Classes

Generic object class recognition has been one of the goals of computer vision since its beginnings [33.1, 2]. In this context, it is important to emphasize that the notion and the abstraction level of object classes is far from being uniquely and clearly defined. Notably, the question of how humans organize knowledge at different levels has received much attention in cognitive psychology [33.3, 4]. Taking an example from Brown's work, a dog can not only be thought of as a *dog*, but also as a *boxer*, a *quadruped*, or in general an *animate being* [33.3]. Yet, *dog* is the term that comes to mind most easily, which is by no means accidental. Experiments show that there is a *basic level* in human categorization at which most knowledge is organized [33.5]. According to *Rosch* et al. and *Lakoff* [33.5, 6], this basic level is also:

- The highest level at which category members have similar perceived shape
- The highest level at which a single mental image can reflect the entire category
- The highest level at which a person uses similar motor actions for interacting with category members
- The level at which human subjects are fastest at identifying category members
- the first level named and understood by children.

Given that the basic level of categorization is easiest for humans, it should be a good starting point for machine vision, too [33.7]. The next lower level, *subordinate categories*, corresponds to the individual level used in object identification, e.g., the next higher level, *super-ordinate categories*, requires a higher degree of abstraction and world knowledge. The following will thus discuss state-of-the-art approaches first for basic-level categorization but then also briefly touch upon subordinate categorization for fine-grained classification.

33.2 Review of the State of the Art

One can think of all current methods in object class recognition as being part-based methods, in which object class instances are described by a set of parts (such as human body parts, for instance), and the topology between the parts (such as the kinematic tree defined by the joints that connect the human body parts). The following sections give a systematic treatment of different instantiations of state of the art, part-based methods, ranging from the bag-of-words model (Sect. 33.2.1) and spatial pyramid matching (Sect. 33.2.2), over flexible part models (Sect. 33.2.3) and rigid histogram of oriented gradients templates (Sect. 33.2.4) to deformable part models (Sect. 33.2.5) and convolutional neural networks (Sect. 33.2.6). Table 33.1 summarizes the various part-based methods presented in this section. Later sections highlight recent trends in fine-grained categorization (Sect. 33.2.7) and datasets for visual recognition (Sect. 33.2.8).

Table 33.1 Classification of some prominent approaches as part-based models

Model	Parts	Topology
Bag-of-words	Feature clusters	No spatial information coded
Spatial pyramid	Feature clusters	Hierarchical but rigid spatial grid
Constellation model	Feature clusters	Full covariance matrix between clusters
Implicit shape model	Feature clusters	Star-shaped model
HOG	HOG-cells	Rigid spatial grid
DPM	HOG-parts	Star-model
Pictorial structures model	Semantic parts	Tree structure
Convolutional neural networks	Convolution filters	Spatial spooling and hierarchy

33.2.1 Bag-of-Words Models

Apart from allowing to establish pointwise correspondences between multiple images of the same scene, local image patches also provide the foundation for a variety of object class representations for visual recognition. Inspired by the natural language processing literature, the *bag-of-words* (BOw) or *bag of features* model [33.8] represents an image as an unordered collection (a bag) of local patches. This representation assumes that most of the important information about the image content is captured purely by which patches occur (and how often), irrespective of where they occur. While this is clearly an approximation, it seems reasonable from the text document analogy: even if the words in a text document are shuffled randomly, the mere occurrence of certain key terms provides a strong cue about the document's topic. Likewise, a collection of local image patches, such as two eye patches, mouth, and nose, provide strong cues for the presence of a face in an image.

The bag-of-words model is typically based on a codebook of visual words: a fixed, discrete set of visual features relative to which all others are described (Fig. 33.1c). An input image (Fig. 33.1a) can then be described relative to that codebook, by counting how often each codeword appears (Fig 33.1b). In the following paragraphs, we describe in detail the extraction of visual features, the generation of codebooks of visual words, and their use in image classification using probabilistic generative and discriminative models.

Feature Extraction

Locations and scales for feature extraction are typically either picked sparsely, based on an interest operator, such as the difference of Gaussian (DOG)

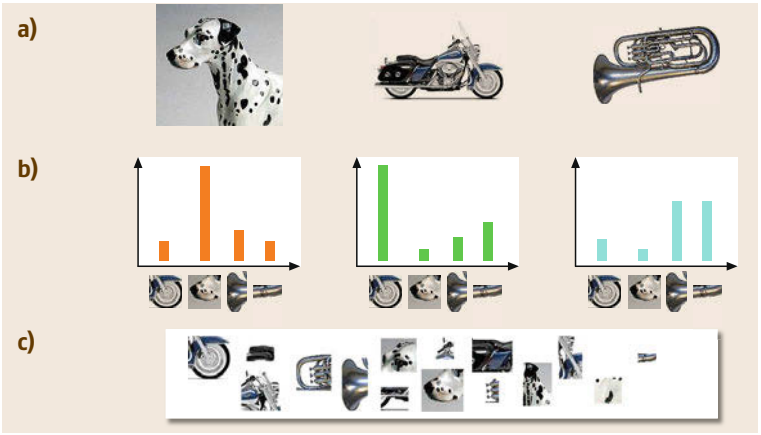


Fig.33.1a–c The bag-of-words representation. **(a)** Image; **(b)** BOw histogram; **(c)** codebook of words

interest point detector for scale invariant feature transform (SIFT) [33.9], Harris corners, scale- and affine-invariant Hessian–Laplace interest points [33.10], maximally stable extremal regions (MSER) [33.11], or based on dense feature sampling using a fixed spatial grid [33.12]. While interest operators generally provide well-localized, repeatable features with sufficient recall, the increasing availability of compute power today has made very dense feature sampling feasible (e.g., with a stride of only one pixel, resulting in many thousands of features in an image) – in the limit, dense sampling typically outperforms sparse interest operators in image classification as interest point detectors are designed to sample particular types of image structure whereas dense sampling allows to sample any type of image structure (note that this might be different for other applications, such as structure from motion, in which it is more important to find smaller numbers of features with high precision).

Codebook Generation

Codebooks of visual features are typically constructed by applying an unsupervised clustering algorithm, such as *k*-means or hierarchical agglomerative clustering [33.13], to a training set of image feature vectors. The result is a vector-quantized representation of the visual feature space that groups similar feature descriptors into clusters. Each cluster thus effectively generalizes over its members, such that small visual differences are suppressed – this is often seen as a crucial trait for categorization even though larger codebook sizes typically result in better performance. Each cluster is identified with a representative member, i. e., its centroid (for *k*-means) or medoid (for agglomerative clustering). In order to represent a new image with respect to the generated codebook, image features are extracted, and matched to the representative of each cluster (e.g., by computing pairwise distances and defining a matching threshold). The result of this matching is then recorded (e.g., as a histogram over the number of times each cluster has been matched) to yield the image or object representation. Note that in the context of BOw models the terms object and image representation are synonymous.

For codebook generation, the number of clusters is an important free parameter: it has been found that large codebook sizes typically improve recognition performance, since they represent the original feature space more faithfully, with lower reconstruction error. On the other hand, large codebooks generalize less, and come at the cost of increased computational cost when building bag-of-words representations and classifying them with nonlinear classifiers. Typically, codebooks with several and even 10 000s (multiples of ten thousand) of clusters are chosen for optimal performance [33.14].

Generative Models

Inspired by their success in the text domain, generative probabilistic models have also been adopted for image classification in connection with BOw representations. The arguably simplest one is the naive Bayes model [33.8, 15], in which the posterior probability of an image having a certain class factors into independent contributions from each histogram bin (i. e., for a given class, knowing about the presence of one feature does not make the presence of another feature any more or less likely). While this model is agnostic of correlations that often exist between features (for instance, having already observed two wheel features in a sideview of a car makes it unlikely to observe one more, assuming that only a single car is present in the image), it often works well in practice and is appealing due to its simplicity.

In line with the text domain, more powerful models have been proposed for image classification that explicitly capture the co-occurrence of features through a generative process. Here, the features contained in an image are assumed to be generated in a sequence of steps, where each step determines the choices available for the next step. More precisely, each step is implemented by sampling from a probability distribution, conditioned on the outcome of the previous step.

A popular class of generative models is the *topic model* [33.16, 17]: an image is considered to be an orderless document of visual words, composed of multiple topics (in an image depicting a beach scene, there could be a water topic, a sand topic, and a sky topic, for instance). In order to generate a particular feature in the image, a topic is first sampled from a discrete distribution over topics (a beach scene could be a roughly uniform distribution over water, sand, and sky). Then, a visual feature is sampled from a discrete distribution over features, conditioned on the previously sampled topic (e.g., for a water topic, the sampled feature is likely to be a blue image patch with a ripple pattern). In practice, the assignment of visual features to topics is represented through latent variables, which means that it has to be inferred from other quantities that can be observed (such that the likelihood of the features is maximized). This is typically implemented by means of the expectation maximization (EM) [33.18] algorithm, which alternates (re-)estimation of topic distributions (E-step) and imputation of latent variables (M-step) in an iterative procedure.

In the literature, there exists a large variety of topic models, ranging from the basic probabilistic latent semantic analysis (PLSA) [33.16] over its Bayesian extension, latent Dirichlet allocation (LDA) [33.17], to

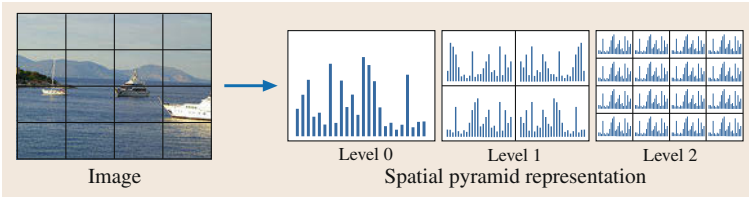


Fig. 33.2 Spatial pyramid matching as a multilevel BOw representation (after [33.20])

infinite hierarchical models of images and their features [33.19].

Discriminative Models

Once an image is represented as a histogram of matched features with respect to a codebook, it can be classified by means of any discriminative classifier that is applicable to histograms. Popular choices include support vector machines [33.21] with histogram intersection kernels or their approximate counterpart [33.22], boosting [33.23], or random forest classifiers [33.24].

Extensions

While the bag-of-words representation inherently ignores spatial information, it can be re-introduced by restricting the support of a bag-of-words representation to a local image region, and processing multiple regions independently. Specifically, instances of an object class can be localized by sliding a rectangular window over the image under consideration, extracting a bag-of-words representation at each location, and feeding the result into a classifier that distinguishes the object class of interest from a background. Objects of different sizes can be localized by applying this scheme to multiple re-scaled versions of the input image [33.25] (called an image pyramid).

Rather than applying the sliding window in a brute force fashion, it has been realized that the classifier score that an image window can attain can be bounded from above by the scores of the subwindows that it contains. This observation gives rise to an efficient subwindow search [33.26] that greatly reduces the number of classifier evaluations needed in connection with sliding window bag of words representations, making powerful but slow nonlinear kernel SVMs applicable.

33.2.2 Spatial Pyramid Matching

Instead of completely ignoring the spatial layout of local features in an image, it seems reasonable to include at least a rough notion of locality in order to increase the discriminative power of an object class representation. Spatial pyramid matching implements this notion as an extension of the bag-of-words model (Sect. 33.2.1), by extracting multiple bag-of-words representations for a single input image at different spatial locations and scales, and aggregating these local representations into a single global representation for the entire image. Specifically, spatial pyramid matching (SPM) [33.27] extracts bag-of-words histograms for each cell in a rectangular grid overlaid onto the input image (typically, coarse grids of 2^l for $l \in \{0, 1, 2\}$). This process is repeated multiple times with different numbers of cells (e.g., four by four plus two by two plus a single cell), forming an overcomplete pyramid structure of decreasing spatial resolution (Fig. 33.2). The resulting bag-of-words histograms for each cell are concatenated (and re-normalized). The appeal of SPM lies in representing the spatial layout at multiple levels of resolution without committing to any one in particular – instead, the representations of all resolutions are kept, and have the chance to contribute to image classification in case they prove useful.

The basic SPM methodology appears in numerous variations in the literature, but often follows the same pipeline of four steps (feature extraction, feature coding, spatial pooling, and classification), each of which we review in the following sections.

Feature Extraction

Like the basic bag-of-words model, the SPM model has been used in connection with various feature types, ranging from SIFT [33.9] appearance features to local edge configurations [33.28] and self-similarity features [33.29].

Feature Coding

Representing visual features relative to a pretrained codebook obtained by vector quantization is only one special case of feature coding (performing a change of representation), and several improvements have been proposed in the literature [33.30–32]. Most importantly, it has proven beneficial to explicitly formulate the codebook training and feature coding as a joint optimization problem, such that the original features can be reconstructed with small error from their respective codes. In addition, this formulation allows us to relax the assumption that each feature is matched to exactly one codebook entry by replacing the cardinality constraint with an L1 regularization. The resulting sparse coding (SC) formulation [33.31] has been shown to outperform the original SPM [33.27] by significant margins, even

when combined with a linear SVM (while SPM works best in connection with nonlinear histogram intersection or chi-square kernels, which are computationally expensive). An improved version of SC is given by locality constrained linear coding (LLC) [33.32]. It replaces sparsity by locality constraints that encourage features to be reconstructed by nearby features in feature space, such that similar features yield similar codes (which is obviously desirable in order to facilitate classifier learning).

Instead of relying solely on counting the number of occurrences of visual features in a spatial region as promoted by SPM, SC, and LLC, the Fisher vector [33.30] method characterizes that region by how well a pretrained generative probabilistic model (a Gaussian mixture model, GMM) explains the contained features. Specifically, it builds a vector from the derivatives of the log-likelihood of the image features, given the mixture coefficients, means, and diagonal covariances of a GMM. Note that this vector has much more entries than a histogram of occurrences using the same GMM as a codebook, allowing to build highly discriminative representations already for small numbers of GMM mixture components (typically only a few hundred).

Both sparse coding and Fisher vector representations haven proven to be very effective for image classification in connection with spatial pyramids, and have been successfully applied also in large-scale settings, such as classification on ImageNet [33.33] (Sect. 33.2.8).

Spatial Pooling

Apart from the classic pyramid scheme [33.27] in which histograms of different pyramid levels are concatenated, retaining the maximum value of a particular bin over a spatial region or even multiple pyramid levels (max-pooling) has gained increasing attention [33.31], in particular in connection with CNNs (Sect. 33.2.6). Max-pooling has the advantage that it promotes invariance to local translations and scale changes, and has been found to be present in the human visual cortex (V1) [33.34].

In addition to pooling features in the two-dimensional (2-D) image plane, more object-centric pooling techniques have been developed in the context of fine-grained categorization [33.35, 36] (Sect. 33.2.7).

Classification

The last stage in an SPM pipeline is typically given by a discriminative classifier that is trained on a training set of fixed length vectors created from the SPM representations. Depending on the sophistication of the feature coding step, support vector machines with nonlinear histogram intersection or chi-square [33.27] or linear kernels [33.30–32] achieve state-of-the-art performance. Further extensions include the use of multiple kernels with learned contribution weights [33.37]. When evaluated as part of a classification cascade, multiple kernel learning (MKL) can also be used in a sliding window object class detector [33.38].

33.2.3 Flexible Part Models

While orderless bags of visual features (Sect. 33.2.1) or spatial pyramids with a fixed grid layout (Sect. 33.2.2) have proven to be effective object class representations for classification, they both fail to explicitly capture the flexible nature of physical objects that are composed of multiple parts – as an example, consider the human face with eyes, eyebrows, mouth, nose, and ears: obviously, the precise shape and spatial arrangement of these parts will vary from person to person, but variations in shape and arrangement are constrained to lie within a plausible range (i. e., the parts are not unordered as in a bag-of-words representation). It also seems intuitive that the constraints that govern the range of plausible variations should be soft, and not bound to a fixed binning or a fixed spatial grid (as in spatial pyramid matching). In this section, we review a family of flexible part models that implement this intuition.

Constellation Model

The constellation model [33.39–41] represents an object class as a flexible arrangement of a fixed number of parts. It consists of two components: first, a model of the appearance of each part, and second, a model of the relative spatial layout of the parts. It is formulated in a fully probabilistic way, such that each part is characterized by a probabilistic, generative model of its appearance, and the spatial layout of the parts is governed by a joint Gaussian distribution that captures how the location (and scale) of each part is expected to vary with the location of each other part. Interestingly, the definition of what constitutes a part is driven entirely by the training data: parts are chosen and placed such that the likelihood of the training data, given the model, is maximized during training. This can be seen as an instance of a missing data problem (the association between features and parts is unknown), and solved by means of the expectation maximization (EM) algorithm.

Unfortunately, the representative power of the joint Gaussian spatial distribution comes at a cost: the fully connected dependency structure demands examining each of the exponentially many combinations of placing all parts in an image in order to find the global optimum – the exact inference problem is known to be intractable. As a consequence, the original constellation

model [33.40, 41] is limited to a few dozen candidates per part, per image, limiting its generality. An alternative approach to solving the exact inference problem is to resort to an approximation, e.g., by sampling likely constellations from part detection proposals [33.42]. In this way, the number of part candidates can be scaled up to several thousands.

Due to its probabilistic nature, the constellation model has also been extended to a full Bayesian model [33.43], in which the object class model itself is governed by a prior distribution over models. This abstraction allows to separate characteristics of one particular object class (e.g., the particular shape of a horse's head) from properties that are shared by classes (e.g., all quadrupeds are four legged). These shared properties can be re-used in multiple object class detectors (e.g., for horses, cows, dogs, etc.), facilitating learning from limited training data. The underlying idea of re-use of information is known as transfer learning in the machine learning literature, and has been successfully applied for image classification [33.43] as well as object class detection.

Implicit Shape Model

The implicit shape model (ISM) [33.44] relaxes the assumption of the constellation model that the locations and scales of all parts are mutually dependent. Instead, it limits the dependency structure to a star topology, in which part locations and scales are conditionally independent, given the object center (i. e., given the object center, revealing information about the location of one part does not alter the expectation of where to observe any other part). While this model clearly constitutes an approximation of the physical world, it greatly simplifies the inference problem of finding the configuration with maximum score (there exists an algorithm that is exact and efficient).

Specifically, the ISM represents an object class as a collection of local feature matches – the supply of matchable features is given by a codebook of visual words, as it is also used in bag-of-words (Sect. 33.2.1) or spatial pyramid representations (Sect. 33.2.2). In contrast to those, however, each ISM codebook entry also maintains a list of feature matches from a training set of images: each match specifies where the match occurred relative to the object center. As a result, this list of occurrences constitutes a nonparametric probability density estimate of the location of the object center, given the matched feature – in other words, each feature can cast votes for likely object center positions (e.g., a wheel feature can cast two votes: one to the left of it, assuming that it is the front wheel of a car, and one to the right, assuming it's the back wheel). This voting procedure can directly be translated into a recognition algorithm: at recognition time, all test image features cast their votes into a three-dimensional (3-D) voting space (x, y, and scale). All votes are accumulated, and since multiple features are likely to agree on the true object center, the maxima of this voting space can be considered candidate object detections (note that this voting can be considered a generalized Hough transform). In practice, continuous voting space maxima are found using a mean-shift algorithm.

In order to increase its discriminative power, the basic ISM model is extended by a verification step [33.44], in which object detection candidates are verified based on how consistent they are with respect to foreground–background segmentation. To that end, feature occurrences stored as part of the codebook are enriched with segmentation mask fragments (which have to be provided as additional supervision during training). At recognition time, the consistency of the cast segmentation mask votes is quantified based on minimum description length (MDL). In addition, it has been suggested to learn weights that regulate the contributions of feature matches to the final detection hypotheses in a discriminative way (i. e., optimized to distinguish object and background firings). This *discriminative Hough transform* [33.45] further improves the performance of the ISM and constitutes the basis for the *poselet* framework [33.46], in which a multitude of small HOG [33.47] detectors (Sect. 33.2.4) are combined in order to detect people.

Since, in theory, the ISM voting space can accommodate arbitrary quantities in addition to object center and scale, there have been extensions to predict the walking poses of pedestrians [33.48] as well as predicting the viewpoint from which the detected object is imaged [33.49, 50].

Pictorial Structures

While being the first to conceptualize the idea of objects being composed of flexible parts, the pictorial structures model [33.39, 51] is typically understood as a simplified version of the constellation model. The full mutual dependency structure between parts is relaxed to a tree (or a mixture of trees in more advanced incarnations) – note that the star-shaped ISM (Sect. 33.2.3) is hence a special case of a pictorial structures model with tree-depth one. In contrast to the constellation model, where parts are defined in a data-driven way, the pictorial structures model is typically associated with a semantic notion of parts. An intuitive example is the set of human body parts (head, torso, upper arm, lower arm, etc.) in connection with the natural kinematic tree structure (Fig. 33.3).

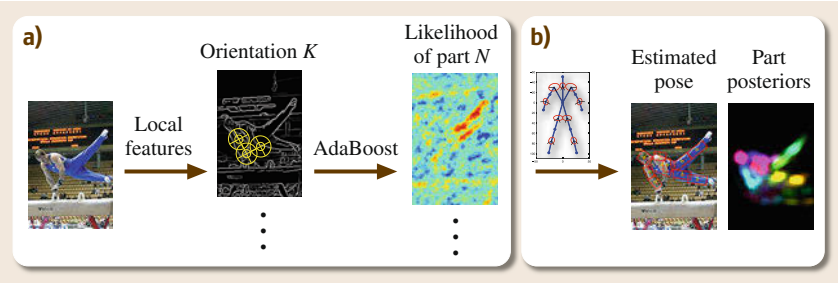


Fig.33.3a,b Human body pose estimation with the pictorial structures model. **(a)** Image likelihoods for individual parts; **(b)** final pose estimate using spatial layout prior

In fact, current state-of-the-art systems for people detection and human body pose estimation are typically built upon the pictorial structures model [33.52]. Here, pairwise relations between body parts express the angular constraints of the human body as well as the prior probability of observing people in various poses (e.g., it is unlikely to observe extreme angles at certain joints, Fig. 33.3b). In order to compensate for the weaker spatial model in comparison to the constellation model, the pictorial structures model typically relies on a strong model for the appearance of each part that is trained discriminatively (distinguishing the part from background, Fig. 33.3a) rather than in a generate way (explaining the part's appearance). Examples of discrimative part appearance models include AdaBoost [33.23] classifiers trained on dense shape context features [33.52]. Also note that part appearance and spatial layout models are trained independently (joint training has been realized to bear large potential for improvement in the context of the deformable part model [33.53] which we will review in Sect. 33.2.5).

The use of semantic parts naturally demands to provide additional supervision in the form of part annotations during training; human body parts are typically annotated with a fixed size relative to the size of the person. For recognition, inference in the tree-structured probabilistic graphical model can be carried out exactly and efficiently by max-product or sum-product belief propagation, depending on whether the maximum a posteriori (MAP) or marginal solution is preferred.

Extensions to the pictorial structures model include learning the tree structure instead of using the natural kinematic tree [33.54], introducing additional lateral edges in addition to the tree edges [33.55], and combining pictorial structures models with nonparametric representations of partial human poses (*poselets* [33.46]).

33.2.4 Histogram of Oriented Gradients

At the opposite end of the spectrum from the bag-of-words (Sect. 33.2.1) or flexible part models (Sect. 33.2.3) are approaches that represent an object class as a rigid template of local features with a fixed spatial layout. Conceptually, this corresponds to the finest level of a spatial pyramid in which each feature gets assigned its own exclusive set of histogram bins (with values equal to the feature itself). The by far most successful rigid template detector is the histogram of oriented gradients (HOG) [33.47] detector, and we will discuss its properties in detail in this section. Originally developed for people detection, the HOG feature has evolved into one of the most frequently used features in computer vision to date, and also constitutes the basis for the successful deformable part model (DPM) [33.53] described in Sect. 33.2.5.

Feature Extraction

The HOG feature can be seen as an extension of SIFT [33.9] from a single circular interest region to a rectangular detection window – at its core, it bins gradient directions into histograms, weighting their contributions by their respective magnitudes. Like SIFT, it has been carefully engineered and its parameters empirically tuned for performance (more recently, learning features end-to-end has received renewed attention, Sect. 33.2.6).

The first step for the computation of a HOG feature is to divide the detection window into a fixed grid of cells: a cell is the basic spatial unit of measurement for the HOG feature. The second step is to gamma-normalize the pixel content of each cell, in order to increase robustness to lighting changes. Third, the image is smoothed and image gradients computed using discrete derivative masks. Fourth, gradients are binned, such that each gradient casts weighted votes (by magnitude) according to nine different gradient orientation bins. Votes are interpolated bilinearly in both orientation and position. And fifth, the per-cell histograms are contrast normalized with respect to a larger block of surrounding cells in order to compensate for local variations in lighting and foreground–background contrast. Crucially, blocks are formed such that they overlap, which means that each cell contributes multiple times to the final feature descriptor, normalized in different ways, resulting in an over-complete represen-

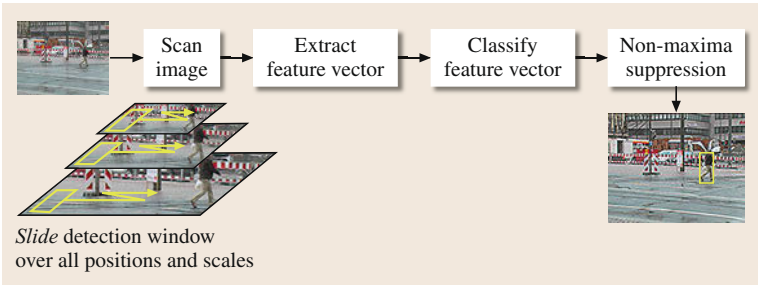


Fig. 33.4 Sliding window object class detection

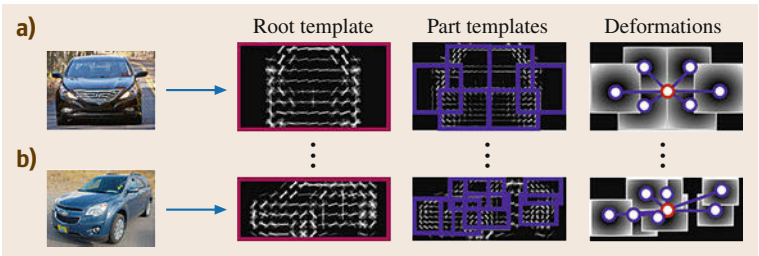


Fig.33.5a,b The deformable part model with multiple mixture components. **(a)** Mixture component for a frontal view of object class car. **(b)** Mixture component for a side view of object class car (after [33.56])

tation. This step has been found to be essential for good performance.

Sliding Window Object Detection

Similar to pure bag-of-words and spatial pyramid-based detectors, the HOG detector is typically run in a sliding window fashion over a test image, over different locations and scales (Fig. 33.4). That is, HOG features are computed for a detection window of fixed size, resulting in a feature vector that can then be fed into a discriminative classifier, such as a linear support vector machine (SVM).

Bootstrapping

One of the key ingredients for good detection performance of the HOG detector [33.47] is a technique termed as *bootstrapping*(not to be confused with a technique of the same name known from statistics, in which a dataset is repeatedly subsampled). An initially trained model is refined by re-training it on an extended set of negative training images, similar in spirit to boosting [33.23]. Specifically, the initial model is run on a validation set, and the false positive detections (wrongly detected objects) are added to the negative training set. As a result, the decision boundary of the retrained classifier is pushed further away from these *hard negatives*. Note that this procedure is not guaranteed to always lead to an improvement, but often improves performance substantially in practice.

Extensions

Due to its simple grid structure, the HOG feature lends itself to being combined with other feature channels extracted from a similar spatial grid. Notably, it has been successfully combined with optical flow features for people detection in video [33.57].

33.2.5 The Deformable Part Model

The section provides an introduction to one of the most successful computer vision algorithms to date, the DPM [33.53]. Its success is partly due to its consistently high performance for detecting a wide variety of object classes, but also due to the fact that a high quality implementation has been made publicly available from the start. The original implementation has constantly been improved and updated, and even found its way into OpenCV [33.58]. The HOG [33.47] implementation at its core has found wide spread use throughout the computer vision community.

Model Structure and Inference

The DPM can be seen as a proper generalization of the fully rigid HOG [33.47] template to a flexible, part-based object class representation (Fig. 33.5). It generalizes HOG in two different ways that both increase its ability to handle intraclass variation: first, it divides the single HOG template into a collection of multiple templates that can move relative to one another and capture small variations in object shape (hence the name *Deformable Part Model*). Specifically, it includes a low-resolution *root* template for the entire object plus a number of high-resolution *parts* that are constrained to occur not too far from their expected positions relative to the root – this constitutes a star-shaped probabilistic graphical model (PGM) in full analogy

to the implicit shape model [33.44] (Sect. 33.2.3) or a special case of a tree-structured pictorial structures model [33.39] (Sect. 33.2.3). While this is less powerful than a full constellation model [33.41] in terms of representation, it has proven very effective in connection with a discriminative learning regime.

Second, the DPM is formulated as a mixture model with multiple components, where each component takes care of another aspect of the object class of interest, such as a particular viewpoint or a particular context in which the object occurs (e.g., a person sitting on a chair). Each component consists of a star-model with root and multiple part templates as described earlier, trained to respond the strongest to its dedicated aspect – this separation of concerns leads to a simpler learning problem than trying to handle all aspects with a single component.

For recognition, each of the DPM components produces a score as follows: first, all of the component's templates (root and parts) are convolved with the image in a sliding window fashion, as in the original HOG [33.47] work (Sect. 33.2.4), resulting in dense score maps over image locations and scales. Second, root and part scores are combined with spatial deformation terms that reflect the star-shaped spatial dependency structure, resulting in new score maps over locations and scales – note that this inference step can be carried out by max-product belief propagation, which is exact and efficient due to the star topology, and implemented as an instance of a generalized distance transform [33.59]. And third, for each image location and scale, the maximum response over all components is kept as the final detection score (which is then further processed by a standard, greedy nonmaxima suppression).

Learning

It has been argued that much of the DPM's excellent detection performance is due to the specific way in which parameters are learned from training data, in addition to the added expressive power in comparison to a single, rigid HOG template. This learning also constitutes the major difference between the DPM and earlier flexible part models such as the ISM [33.44] or the pictorial structures model [33.39] (Sect. 33.2.3) that are structurally equivalent, but less effective in terms of performance.

In contrast to these earlier models, where local part appearance and spatial layout models are trained separately, all DPM parameters are learned jointly, such that a single binary classification objective is optimized (distinguish between the object class of interest and background). Crucially, this includes the placement of parts relative to the root template as well as the selection of the mixture component that is deemed responsible; both are treated as latent variables at training (and test) time. As a consequence, all aspects of the DPM are optimized towards the single goal of object class detection, without being convoluted by potentially suboptimal choices of part placement based on appearance similarity (ISM) or semantic meaning (pictorial structures).

The joint optimization problem is phrased as a generalization of the linear SVM [33.21] formulation of a single HOG template, termed latent support vector machine (LSVM) [33.60]. It is based on the following intuition: assuming that the latent variables (part placements and mixture component selection) are observed, all model parameters (HOG template and part displacement weights) can be learned using standard linear SVM procedures (the resulting objective function is convex, assuming the use of a hinge loss). Conversely, assuming known model parameters, the optimal values of all hidden variables can be found exactly and efficiently (using the distance transform for part placement and exhaustive search over components). This intuition gives rise to an EM-like alternation of two steps, which can be shown to converge to a local optimum of the original, nonconvex objective function.

Naturally, a good initialization is needed in order to reach good local optima during learning. The DPM suggests a clustering of training examples based on aspect ratio, which has proven to work surprisingly well in practice (clustering based on appearance typically results in small improvements but is much more costly in terms of computation time). As for the HOG detector [33.47], bootstrapping (here called *hard negative mining*) allows to escape bad local optima.

While the concept of jointly optimizing part appearance and placement has already existed in the constellation model [33.41], the DPM achieves this goal by largely different technical means, resulting in better performance: (1) it uses sliding-window HOG templates with linear SVM classifiers instead of a generative model over local interest point descriptors to represent local appearance, and (2) it relaxes the fully connected probabilistic graphical model topology to a star, making exact inference tractable and efficient for learning. The DPM has achieved excellent performance on a wide variety of benchmark datasets, including PASCAL VOC (visual object class) [33.61] (Sect. 33.2.8).

Multiview and 3-D DPMs

While the original DPM [33.53] achieves remarkable performance on standard detection benchmarks, it is inherently limited to predicting 2-D bounding boxes. While this might be sufficient for some applications (such as counting the number of objects in an image), it

is clearly not enough for others: suppose an autonomous driving scenario in which vehicles are to be tracked across multiple frames, and their spatial extent to be estimated in order to avoid collisions – here, a 3-D representation of each vehicle is required.

To that end, the DPM has been extended [33.56, 62, 63] to output richer object hypotheses than 2-D bounding boxes, in the form of additional viewpoint estimates (providing a better notion of 3-D object extent than a 2-D bounding box alone) and 3-D parts that are consistent across viewpoints (providing a basis for establishing correspondences across multiple frames in a sequence). Providing these richer outputs can be achieved by extending the original DPM in two different ways: first, the classification problem solved during learning can be reformulated as a structured output prediction problem [33.62]. In this way, the model can be optimized for the prediction of other quantities than a binary object-background label, such as 2-D bounding box overlap or angular viewpoint estimates, through the use of different loss functions. In its simplest variant, viewpoint estimation can be implemented by a binning of angles, and dedicating a single DPM mixture component to each angular bin. While this model is still inherently view-based, it has been shown to deliver excellent performance in both 2-D bounding box localization and viewpoint estimation (note that viewpoint labels have to be available during training in addition to bounding box annotations).

Second, instead of parameterizing the DPM parts in the image plane, an object-centric 3-D coordinate system can be used, such that parts from different mixture components refer to the same 3-D object portion (Fig. 33.6a). Part deformations are then governed by 3-D displacement distributions (Fig. 33.6b), and object as well as part appearance can be captured in a continuous appearance model (Fig. 33.7a). The result is a 3-D object class representation [33.63] that can synthesize previously unseen viewpoints at test time (by interpolating between a number of supporting viewpoints for which training data is available) and thus estimate viewpoints up to arbitrary granularity. The learning of this 3-D object class representation is facilitated by using 3-D computer-aided design (CAD) models of the object class of interest in addition to real-world images (Fig. 33.7b). During training, the CAD models serve both to establish a 3-D coordinate system and to generate artificial training data for different viewpoints through rendering. Figure 33.8 shows example detections obtained by the 3-D^{2}PM.

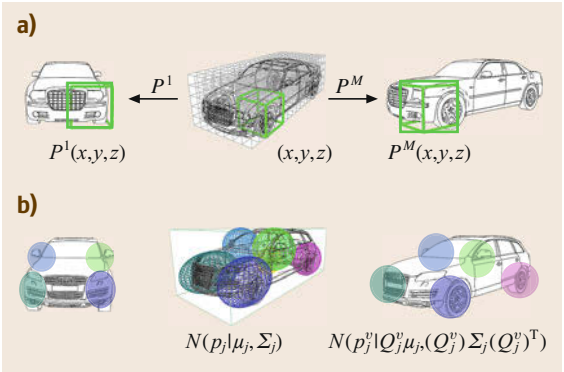


Fig.33.6a,b Part representation in the 3-D deformable part model (3-D DPM). **(a)** 3-D/2-D part coordinate systems; (after [33.62]) **(b)** 3-D/2-D part spatial distributions (after [33.63])

Computational Considerations

Interestingly, it has been realized that the major computational bottleneck of the original DPM [33.53] at recognition time is the convolution of HOG templates with image features (and not probabilistic graphical model inference as one might think). As a consequence, there have been attempts to minimize the number of HOG convolutions by various means, e.g., by part sharing parts across components and object classes [33.64], re-using parts of templates by decomposing them into an overcomplete set of dictionary elements [33.65] or cascaded detector evaluation [33.66]. Likewise, there have been attempts to reduce the training time for the DPM by exploiting HOG feature statistics to re-use computation across models [33.67, 68].

33.2.6 Convolutional Neural Networks

While artificial neural networks (ANNs) [33.69] had largely been neglected in favor of support vector machines, random forests, and other learners until around 2010, they have recently returned to the scene, with quite remarkable success. Among a number of artificial

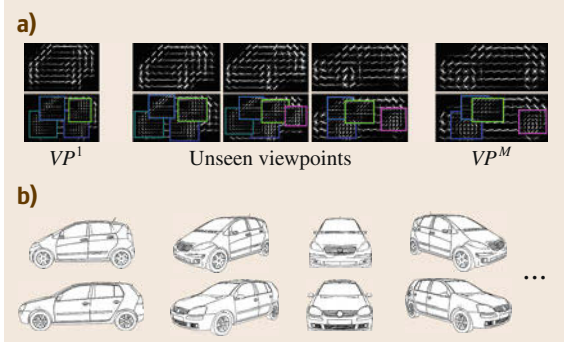


Fig.33.7a,b Continuous appearance model **(a)** in the 3-D^{2}PM, trained from rendered 3-D CAD models (after [33.63]) **(b)** and real images (not shown)

neural network architectures that had been suggested already in the early days of computer vision, the convolutional neural network (CNN) [33.70] has emerged as a particularly promising candidate for becoming a standard tool for visual recognition tasks.

One of the key properties of the CNN is its specific structure (Fig. 33.9), which is both amenable to dealing with multidimensional input (such as a color image) and effective for preventing overfitting when learned from limited training data. As an ANN, a CNN defines an artificial neural network with multiple layers on an input image, such that image pixels provide the input to the neurons of the first layer, and the outputs of each layer of neurons provide the inputs to the next layer (the final layer can provide a task specific output, such as an encoding of different image classes). Now, the topology of the CNN is restricted in two specific ways: first, neurons in one layer can only be connected to a fixed spatial region of neurons in the preceding layer (the *receptive field*) – this dramatically reduces the total number of connections in the network and hence parameters of the model that have to be learned. And second, most layers of neurons (typically the lower layers) are formed by replicating local groups of neurons multiple times, including the weights associated with their connections to the preceding layer (*weight sharing*) – this further reduces the number of parameters of the model, and encourages the network to learn a representation for the input that is invariant to translation. As a result of their specific structure, CNNs are less prone to overfitting than fully connected ANNs, and can be successfully trained using gradient-based methods and back-propagation of errors.

CNNs for Image Classification

CNNs have recently achieved remarkable performance in large-scale image classification [33.71], fueled by the availability of large-scale datasets like ImageNet [33.33] that provide thousands of images per object class. The success of CNNs in image classification is mostly attributed to their ability to learn useful image representations end-to-end (*deep learning*) rather than relying on engineered features, such as SIFT [33.9] or HOG [33.47]. While the latter have been carefully designed and empirically tuned over many years, they are *general purpose* and not specifically adapted to the classification task at hand, and hence suboptimal. A recent study provides further insights into the operation of individual CNN components (hidden layers and classifiers) through visualization [33.74], resulting in a further boost in performance.

Region Proposals for Detection

Since the evaluation of a CNN is computationally expensive due to convolutions, a CNN is typically applied to preselected candidate regions rather than in a dense sliding window fashion for detection. Candidate regions can be generated either by a specialized detector trained for the object class of interest (such as HOG [33.47], Sect. 33.2.4) that is weaker but also much faster to evaluate than the CNN, or by a more general attention mechanism that generates bounding boxes that are likely to contain an object instance of *any* class. Among various methods that generate proposal regions with a high degree of *objectness* [33.75] ([33.76] for a recent discussion and analysis of region proposal methods), selective search [33.77] has been successfully used in

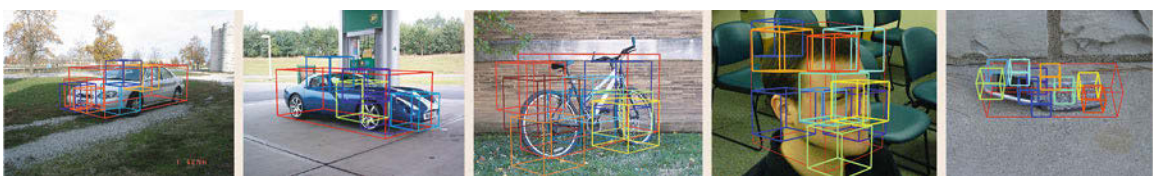


Fig. 33.8 3-D^{2}PM example detections (car, bicycle, human head, cell phone)

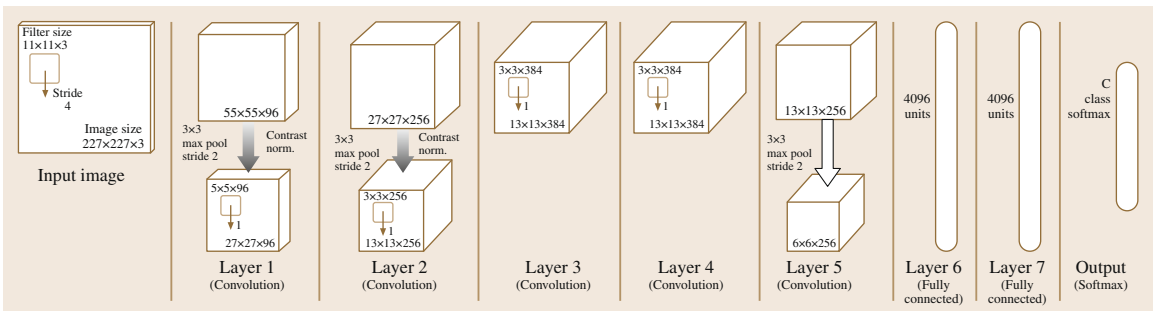


Fig. 33.9 The CNN model of (after [33.71]) in the implementation of Caffe (after [33.72, 73])

connection with CNN-based, object class specific classifiers for detection [33.78]. Selective search is built upon a superpixel-based oversegmentation of an image, and suggests a greedy procedure to group adjacent superpixels into regions of high objectness. The resulting detector [33.79] has been shown to outperform previous results in generic object class detection by a large margin (notably outperforming the previous best results of the DPM [33.53], Sect. 33.2.5, on PASCAL VOC [33.61], Sect. 33.2.8). Curiously, it is not necessary to train this detector end-to-end for the task of object class detection for good performance – instead, the lower layers of the CNN can be pretrained for classification on ImageNet as in [33.71], and later fine-tuned for detection. Both the CNN pretrained on ImageNet and the resulting state-of-the-art object class detector are publicly available under the name *Caffe* [33.72] and likely to find their way into the toolboxes of many computer vision practitioners.

Structured Outputs

While artificial neural networks can represent any function in theory, effective learning of ANNs with more complex, structured outputs than single class labels or continuous values remains a challenge. Instead of learning a full network end-to-end, structured outputs are typically obtained by feeding the outputs of the last layer of neurons into an existing structured model, such as multiple body joint regressors for human body pose estimation [33.80].

33.2.7 Fine-Grained Categorization

In recent years, there has been a growing interest in distinguishing objects not only with respect to basic-level categories [33.5] (such as car, bicycle, and person), but also on a finer level of granularity (such as different car models or plant and animal species). This *fine-grained categorization* is deemed particularly challenging: the distinction between two categories is often determined by small visual differences (such as the shape of a car's headlight): these differences are easily outweighed by appearance differences due to change in lighting or viewpoint (a car looks largely different from the front and from the side), bearing the risk of overfitting to irrelevant details.

Feature Selection

A first strategy to identifying subtle discriminative features while at the same time preventing overfitting is to combine classical feature selection techniques with randomization, e.g., in the form of random forest classifiers [33.35, 81]. Here, randomization acts as a regularizer that prevents multiple trees to latch onto the same potentially irrelevant feature, hence reducing the risk of overfitting. Feature selection can also be performed with humans in the loop, using crowd-sourcing and gamification [33.82].

Pose Normalization

A second, orthogonal strategy in fin-grained categorization is to devise factored object class representations that explicitly capture different modes of variation in object appearance, thereby separating variations caused by category membership from variations caused by other factors, such as viewpoint changes. The main idea is to establish a common frame of reference for the extraction of visual features (e.g., a local coordinate system for the dense extraction of SIFT [33.9] features), such that the same object instance will always result in (roughly) the same visual features being extracted, irrespective of viewpoint, lighting, etc.

Specifically, such a frame of reference can be established by first detecting the object of interest in a test image and also localizing its semantic parts. Visual features are then extracted and described relative to the estimated object position and detected parts, ensuring that a feature on the beak of a bird will only ever be compared to a corresponding feature on the beak of another bird, and not on its tail (assuming perfect detection, viewpoint estimation, and part localization). This kind of part-based *pose normalization* has been demonstrated to be effective for bird species categorization [33.83, 84].

Instead of localizing object parts in 2-D, some works establish a common reference frame by estimating a rough 3-D geometry of the object of interest prior to feature extraction, in addition to estimating its viewpoint. This allows not only to describe feature locations relative to the object surface rather than the 2-D image plane (effectively performing the pooling in 3-D [33.36]), but also to rectify image patches with respect to the estimated surface normals, improving the robustness of the resulting feature descriptors with respect to small viewpoint changes. This strategy has proven effective in connection with a prebuilt bird model [33.35] as well as in connection with predicted 3-D CAD model matches for fine-grained car categorization [33.36].

33.2.8 Datasets

Since visual object class recognition approaches rely on machine learning algorithms at their core, they are very much dependent on the availability of sufficient training data – it has in fact been argued that major advances

in the field might have occurred in response of obtaining more data rather than by developing more powerful algorithms [33.85]. At the same time, datasets are of utmost importance in order to quantify and compare the performance of different methods under controlled conditions, and checking the sanity of newly developed algorithms. To that end, benchmark datasets for visual object class recognition typically consist of a collection of images split into training, validation, and test sets, plus a set of ground truth annotations, and often also evaluation scripts that measure the performance of a recognition algorithm relative to the provided annotations.

Since each dataset provides only a small sample of the visual world, there is an inherent danger of being biased [33.85] with respect to which images are selected to be part of a dataset (and which ones are not), and by what means images are captured. As a result, new recognition algorithms should always be tested and cross-trained on multiple datasets in order to prevent overfitting to the characteristics of a particular dataset rather than learning a concept of interest. The following paragraphs give a small, by no means complete, selection of some of today's influential datasets for visual object class recognition.

PASCAL VOC

The PASCAL visual object classes challenge [33.61] stands for a succession of datasets that have been released on a yearly basis from 2005 to 2012. Each dataset consists of multiple challenges (image classification, object class detection, segmentation, human body pose estimation) that are defined on a given set of images. Notably, the set of images is steadily increased, such that images of year n are re-used in year $n+1$. PASCAL VOC counteracts dataset overfitting by not releasing test set ground truth annotations to the public, but instead maintaining an evaluation server that runs a limited number of automated tests for a particular recognition algorithm. PASCAL VOC images have been equipped with additional annotations, e.g., in the form of aligned 3-D CAD models (3-D-PASCAL [33.86]).

SUN and COCO

The scene understanding (SUN) [33.87] and common objects in context (COCO) [33.88] datasets have been built in an effort to capture entire visual scenes rather than objects in isolation. They are meant as benchmarks for methods that reason jointly about multiple objects in a scene rather than performing independent classification or detection.

ImageNet

ImageNet [33.33] is one of the largest image datasets to date, consisting of over 14 million internet images that have been annotated with object labels from the WordNet [33.89] hierarchy. Like PASCAL VOC, it constitutes the basis for multiple challenges that are defined on the dataset, ranging from classification (1000 classes) over object class detection to fine-grained categorization. Due to its massive size, the collection of the ImageNet dataset has been performed mostly by means of crowd sourcing. ImageNet constitutes the basis for pretraining some of the most successful CNN-based image classifiers to date [33.71] (Sect. 33.2.6).

33.3 Discussion and Conclusions

In this chapter, we have discussed visual object class recognition as one of the most fundamental problems in computer vision, with a focus on presenting the most promising models available to practitioners today. The selection of methods that we have reviewed is by no means complete – in favor of going into some depth for the most common techniques, we have skipped over object representations based on manifold learning [33.90] as well as representations based on grammars [33.91, 92], to mention just a few.

As we have seen, practically all available models are inherently based on some notion of object parts that are governed by certain topological constraints. Naturally, the choice of which part-based model to pick is not an easy one, as it depends on many different factors that should be carefully considered.

33.3.1 Task

First of all, the appropriate choice of model is dependent on the application scenario and the resulting desired recognition output: different methods can generate outputs of different granularity, ranging from image-level class labels (basic bag-of-words, Sect. 33.2.1, spatial pyramid matching, Sect. 33.2.2) over 2-D bounding box locations (bag-of-words and spatial pyramid matching run in sliding window fashion, histogram of oriented gradient detectors, Sect. 33.2.4, flexible part models, Sect. 33.2.3, basic deformable part models, Sect. 33.2.5) to multiview detections and viewpoint estimates (multiview ISM, Sect. 33.2.3, 3-D DPM, Sect. 33.2.5).

Secondly, an application scenario typically comes with a certain set of object classes of interest, which

are in turn characterized by rather different visual properties. Rigid objects (like cars or mugs) tend to be handled well by rigid templates (like histogram of oriented gradients, Sect. 33.2.4) or deformable part models (Sect. 33.2.5), while articulate objects (like cats and dogs) can be more reliably represented using looser spatial models like bag of words (Sect. 33.2.1), or by explicitly modeling their kinematic tree structure (pictorial structures, Sect. 33.2.3).

Finally, it should be noted that the distinction between object classes itself is quite arbitrary when drawing strict borders between any two classes. In reality, some classes are inherently more similar than others (e.g., dogs and horses are more similar than dogs and cars, since dogs and horses are both quadrupeds). Looking at multiple levels of object categorization rather than individual classes, it becomes a desired property that objects from the same super-ordinate category, such as quadrupeds, be classified as more similar than objects from different super-ordinate categories. In other words, an object categorization system should degrade gracefully: if the object itself is not correctly recognized, then we want it to be assigned at least to a *similar* category. We thus need a way of representing the relationship and grading similarities between object categories [33.93].

33.3.2 Data

A particular choice of model implies a certain amount of parameters that have to be learned and additional latent parameters that have to be inferred at test time. A model has to be chosen not only appropriate for the task (above) but also in accordance with the available data at hand in order to prevent over or under fitting.

33.3.3 Efficiency

While simple *feedforward* models like histograms of oriented gradients (Sect. 33.2.4) and bag-of-words (Sect. 33.2.1) are very efficient, modeling deformations and interdependence of parts (flexible part models, Sect. 33.2.3, deformable part models, Sect. 33.2.5) comes at the cost of inferring latent quantities in the model. Notably, this inference often has to be performed repeatedly during learning, underlining the importance of efficient inference procedures.

References

33.1 D. Marr: *Vision* (Freeman, San Francisco 1982)
33.2 D.H. Ballard, C.M. Brown: *Computer Vision* (Prentice Hall, Englewood Cliffs 1982)
33.3 R. Brown: How shall a thing be called?, Psychol. Rev. **65**, 14–21 (1958)
33.4 R. Brown: *Social Psychology* (Free, New York 1965)
33.5 E. Rosch, C. Mervis, W. Gray, D. Johnson, P. Boyes-Braem: Basic objects in natural categories, Cogn. Psychol. **8**, 382–439 (1976)
33.6 G. Lakoff: *Women, Fire, and Dangerous Things – What Categories Reveal About the Mind* (Univ. Chicago Press, Chicago 1987)
33.7 S. Dickinson, A. Leonardis, B. Schiele, M. Tarr: *Object Categorization: Computer and Human Vision Perspectives* (Cambridge Univ. Press, Cambridge 2009)
33.8 G. Csurka, C.R. Dance, L. Fan, J. Willarnowski, C. Bray: Visual categorization with bags of keypoints, Eur. Conf. Comput. Vis. (ECCV) (2004)
33.9 D. Lowe: Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. **60**(2), 91–110 (2004)
33.10 K. Mikolajczyk, C. Schmid: A performance evaluation of local descriptors, IEEE Trans. Pattern Anal. Mach. Intell. **27**(10), 1615–1630 (2004)
33.11 J. Matas, O. Chum, M. Urban, T. Pajdla: Robust wide baseline stereo from maximally stable extremal regions, Image Vis. Comput. **22**(10), 761–767 (2004)
33.12 T. Tuytelaars: Dense interest points, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2010)
33.13 R.O. Duda, P.E. Hart, D.G. Stork: *Pattern Classification* (Wiley, New York 2000)
33.14 F. Jurie, B. Triggs: Creating efficient codebooks for visual recognition, 10th IEEE Int. Conf. Comput. Vis. (ICCV) (2005)
33.15 B. Schiele, J.L. Crowley: Recognition without correspondence using multidimensional receptive field histograms, Int. J. Comput. Vis. **36**(1), 31–52 (2000)
33.16 T. Hofmann: Probabilistic latent semantic indexing, Proc. 22nd Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr. (1999)
33.17 D.M. Blei, A.Y. Ng, M.I. Jordan: Latent dirichlet allocation, J. Mach. Learn. Res. **3**, 983–1022 (2003)
33.18 A.P. Dempster, N.M. Laird, D.B. Rubin: Maximum likelihood from incomplete data via the EM algorithm, J. R. Statist. Soc. B **39**, 1–38 (1977)
33.19 Z. Shi, Y. Yang, T.M. Hospedales, T. Xiang: Weakly supervised learning of objects, attributes and their associations, Eur. Conf. Comput. Vis. (ECCV) (2014) pp. 472–487
33.20 S. Lazebnik, C. Schmid, J. Ponce: Spatial pyramid matching. In: *Object Categorization*, ed. by S. Dickinson, A. Leonardis, B. Schiele, M. Tarr (Cambridge Univ. Press, Cambridge 2009) pp. 401–415
33.21 C. Cortes, V. Vapnik: Support-vector networks, Mach. Learn. **20**(3), 273–297 (1995)
33.22 S. Maji, A.C. Berg, J. Malik: Classification using intersection kernel support vector machines is efficient, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2008)
33.23 Y. Freund, R.E. Schapire: A decision-theoretic generalization of on-line learning and an application

to boosting, J. Comput. Syst. Sci. **55**(1), 119–139 (1997)
33.24 L. Breiman: Random forests, Mach. Learn. **45**(1), 5–32 (2001)
33.25 T. Lindeberg: Feature detection with automatic scale selection, Int. J. Comput. Vis. **30**(2), 79–116 (1998)
33.26 C.H. Lampert, M.B. Blaschko, T. Hofmann: Efficient subwindow search: A branch and bound framework for object localization, IEEE Trans. Pattern Anal. Mach. Intell. **31**(12), 2129–2142 (2009)
33.27 S. Lazebnik, C. Schmid, J. Ponce: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2006)
33.28 V. Ferrari, L. Fevrier, F. Jurie, C. Schmid: Groups of adjacent contour segments for object detection, IEEE Trans. Pattern Anal. Mach. Intell. **30**(1), 36–51 (2008)
33.29 E. Shechtman, M. Irani: Matching local self-similarities across images and videos, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2007)
33.30 F. Perronnin, C. Dance: Fisher kernels on visual vocabularies for image categorization, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2007)
33.31 J. Yang, K. Yu, Y. Gong, T. Huang: Linear spatial pyramid matching using sparse coding for image classification, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2009)
33.32 J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong: Locality-constrained linear coding for image classification, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2010)
33.33 J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei: ImageNet: A large-scale hierarchical image database, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2009)
33.34 N. Kruger, P. Janssen, S. Kalkan, M. Lappe, A. Leonardis, J. Piater, A.J. Rodriguez-Sanchez, L. Wiskott: Deep hierarchies in the primate visual cortex: What can we learn for computer vision?, IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1847–1871 (2013)
33.35 R. Farrell, O. Oza, N. Zhang, V.I. Morariu, T. Darrell, L.S. Davis: Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance, IEEE Int. Conf. Comput. Vis. (ICCV) (2011)
33.36 J. Krause, M. Stark, J. Deng, L. Fei-Fei: 3d object representations for fine-grained categorization, 4th Int. IEEE Workshop 3D Represent. Recognit. (3dRR-13), Sydney (2013)
33.37 M. Varma, D. Ray: Learning the discriminative power-invariance trade-off, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2007)
33.38 A. Vedaldi, V. Gulshan, M. Varma, A. Zisserman: Multiple kernels for object detection, IEEE Int. Conf. Comput. Vis. (ICCV) (2009)
33.39 M.A. Fischler, R.A. Elschlager: The representation and matching of pictorial structures, IEEE Trans. Comput. **22**(1), 67–92 (1973)
33.40 M. Weber, M. Welling, P. Perona: Unsupervised learning of models for recognition, Eur. Conf. Comput. Vis. (ECCV) (2000)
33.41 R. Fergus, P. Perona, A. Zisserman: Object class recognition by unsupervised scale-invariant learning, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2003)
33.42 M. Stark, M. Goesele, B. Schiele: A shape-based object class model for knowledge transfer, IEEE Int. Conf. Comput. Vis. (ICCV) (2009)
33.43 L. Fei-Fei, R. Fergus, P. Perona: Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2004) pp. 178–186
33.44 B. Leibe, A. Leonardis, B. Schiele: Robust object detection by interleaving categorization and segmentation, Int. J. Comput. Vis. **77**(1–3), 259–289 (2008)
33.45 S. Maji, J. Malik: Object detection using a max-margin Hough transform, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2009)
33.46 L. Bourdev, J. Malik: Poselets: Body part detectors trained using 3D human pose annotations, IEEE Int. Conf. Comput. Vis. (ICCV) (2009)
33.47 N. Dalal, B. Triggs: Histograms of oriented gradients for human detection, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2005) pp. 886–893
33.48 B. Leibe, E. Seemann, B. Schiele: Pedestrian detection in crowded scenes, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Washington (2005) pp. 878–885
33.49 A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, L. Van Gool: Towards multi-view object class detection, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2006)
33.50 M. Arie-Nachimson, R. Basri: Constructing implicit 3D shape models for pose estimation, IEEE Int. Conf. Comput. Vis. (ICCV) (2009)
33.51 P.F. Felzenszwalb, D.P. Huttenlocher: Efficient matching of pictorial structures, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2000)
33.52 M. Andriluka, S. Roth, B. Schiele: Pictorial structures revisited: People detection and articulated pose estimation, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2009)
33.53 P.F. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan: Object detection with discriminatively trained part based models, IEEE Trans. Pattern Anal. Machin. Intell. **32**(9), 1627–1645 (2010)
33.54 F. Wang, Y. Li: Beyond physical connections: Tree models in human pose estimation, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2013) pp. 596–603
33.55 M. Sun, M. Telaprolu, H. Lee, S. Savarese: An efficient branch-and-bound algorithm for optimal human pose estimation, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2012)
33.56 P. Bojan, M. Stark: Multi-view and 3D deformable part models, IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **37**(11), 2232–2245 (2015)
33.57 N. Dalal, B. Triggs, C. Schmid: Human detection using oriented histograms of flow and appearance, Eur. Conf. Comput. Vis. (ECCV) (2006)
33.58 G. Bradski: Opencv. http://opencv.org/ (July 09, 2015)

33.59 P.F. Felzenszwalb, D.P. Huttenlocher: *Distance transforms of sampled functions, Technical Report 1963* (Cornell Univ., Ithaca 2004)

33.60 C.-N.J. Yu, T. Joachims: Learning structural SVMs with latent variables, ACM Proc. 26th Annu. Int. Conf. Mach. Learn., New York (2009) pp. 1169–1176

33.61 M. Everingham, L. Gool, C.K. Williams, J. Winn, A. Zisserman: The PASCAL visual object classes (VOC) challenge, Int. J. Comput. Vis. **88**(2), 303–338 (2010)

33.62 B. Pepik, M. Stark, P. Gehler, B. Schiele: Teaching 3d geometry to deformable part models, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2012)

33.63 B. Pepik, P. Gehler, M. Stark, B. Schiele: $3D^2PM$–3D deformable part models, Eur. Conf. Comput. Vis. (ECCV) (2012)

33.64 P. Ott, M. Everingham: Shared parts for deformable part-based models, IEEE Comput. Vis. Pattern Recognit. (CVPR) (2011)

33.65 H.O. Song, S. Zickler, T. Althoff, R. Girshick, M. Fritz, C. Geyer, P. Felzenszwalb, T. Darrell: Sparselet models for efficient multiclass object detection, Eur. Conf. Comput. Vis. (ECCV) (2012)

33.66 P. Felzenszwalb, R. Girshick, D. McAllester: Cascade object detection with deformable part models, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2010)

33.67 T. Gao, M. Stark, D. Koller: What makes a good detector? – structured priors for learning from few examples, Eur. Conf. Comput. Vis. (ECCV) (2012)

33.68 B. Hariharan, J. Malik, D. Ramanan: Discriminative decorrelation for clustering and classification, Eur. Conf. Comput. Vis. (ECCV) (2012)

33.69 S. Haykin: *Neural Networks: A Comprehensive Foundation*, 2nd edn. (Prentice Hall, Upper Saddle River 1998)

33.70 Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel: Backpropagation applied to handwritten zip code recognition, Neural Comput. **1**(4), 541–551 (1989)

33.71 A. Krizhevsky, I. Sutskever, G.E. Hinton: Imagenet classification with deep convolutional neural networks, Adv. Neural Inform. Process. Syst. **25**, 1097–1105 (2012)

33.72 Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell: Caffe: Convolutional architecture for fast feature embedding, http://caffe.berkeleyvision.org/ (arXiv preprint arXiv:1408.5093) (2014)

33.73 J. Hosang, M. Omran, R. Benenson, B. Schiele: Taking a deeper look at pedestrians, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2015)

33.74 M.D. Zeiler, R. Fergus: Visualizing and understanding convolutional networks, Eur. Conf. Comput Vis. (ECCV) (2014)

33.75 B. Alexe, T. Deselares, V. Ferrari: Measuring the objectness of image windows, IEEE Trans. Pattern Anal. Mach. Intell. **34**(11), 2189–2202 (2012)

33.76 J. Hosang, R. Benenson, B. Schiele: How good are detection proposals, really?, 25th Br. Mach. Vis. Conf. (BMVC) (2014)

33.77 K.E.A. van de Sande, J.R.R. Uijlings, T. Gevers, A.W.M. Smeulders: Segmentation as selective search for object recognition, IEEE Int. Conf. Comput. Vis. (ICCV) (2013)

33.78 B. Hariharan, P. Arbeláez, R. Girshick, J. Malik: Simultaneous detection and segmentation, Eur. Conf. Comput. Vis. (ECCV) (2014)

33.79 R. Girshick, J. Donahue, T. Darrell, J. Malik: Rich feature hierarchies for accurate object detection and semantic segmentation, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2014)

33.80 A. Toshev, C. Szegedy: Deeppose: Human pose estimation via deep neural networks, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2014)

33.81 J. Deng, J. Krause, M. Stark, L. Fei-Fei: Leveraging the wisdom of the crowd for fine-grained recognition, IEEE Trans. Pattern Anal. Mach. Intel. (2015)

33.82 J. Deng, J. Krause, L. Fei-Fei: Fine-grained crowdsourcing for fine-grained recognition, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2013)

33.83 N. Zhang, R. Farrell, T. Darrell: Pose pooling kernels for sub-category recognition, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2012)

33.84 T. Berg, P.N. Belhumeur: Poof: Part-based one-vs-one features for fine-grained categorization, face verification, and attribute estimation, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2013)

33.85 A. Torralba, A.A. Efros: Unbiased look at dataset bias, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (2011)

33.86 Y. Xiang, R. Mottaghi, S. Savarese: Beyond pascal: A benchmark for 3d object detection in the wild, IEEE Winter Conf. Appl. Comput. Vis. (WACV) (2014)

33.87 J. Xiao, J. Hays, K. Ehinger, A. Oliva, A. Torralba: Sun database: Large-scale scene recognition from abbey to zoo, IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR) (2010)

33.88 T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick: Microsoft coco: Common objects in context, Lect. Notes Comput. Sci. **8693**, 740–755 (2014)

33.89 G.A. Miller: Wordnet: A lexical database for english, ACM Communication **38**(11), 39–41 (1995)

33.90 M.A. Turk, A.P. Pentland: Face recognition using eigenfaces, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) (1991) pp. 586–591

33.91 Z.W. Tu, X.R. Chen, A.L. Yuille, S.C. Zhu: Image parsing: Unifying segmentation, detection and recognition, Int. J. Comput. Vis. **63**(2), 113–140 (2005)

33.92 L. Zhu, Y. Chen, A.L. Yuille: Unsupervised learning of probabilistic Grammar–Markov models for object categories, IEEE Trans. Pattern Anal. Machin. Intell. **31**(1), 114–128 (2009)

33.93 J. Deng, J. Krause, A. Berg, L. Fei-Fei: Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition, IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Providence (2012)

Multimedia Contents

34. Visual Servoing

François Chaumette, Seth Hutchinson, Peter Corke

This chapter introduces visual servo control, using computer vision data in the servo loop to control the motion of a robot. We first describe the basic techniques that are by now well established in the field. We give a general overview of the formulation of the visual servo control problem, and describe the two archetypal visual servo control schemes: image-based and pose-based visual servo control. We then discuss performance and stability issues that pertain to these two schemes, motivating advanced techniques. Of the many advanced techniques that have been developed, we discuss two-and-a-half-dimensional (2.5-D), hybrid, partitioned, and switched approaches. Having covered a variety of control schemes, we deal with target tracking and controlling motion directly in the joint space and extensions to underactuated ground and aerial robots. We conclude by describing applications of visual servoing in robotics.

34.1 **The Basic Components of Visual Servoing** 842
34.2 **Image-Based Visual Servo** 843
34.2.1 The Interaction Matrix 843
34.2.2 Approximating the Interaction Matrix 844
34.2.3 A Geometrical Interpretation of IBVS 846
34.2.4 Stability Analysis 847
34.2.5 IBVS with a Stereo Vision System 848
34.2.6 IBVS with Cylindrical Coordinates of Image Points 849
34.2.7 IBVS with Other Geometrical Features 849
34.2.8 Non-Perspective Cameras 850
34.2.9 Direct Estimation 850
34.3 **Pose-Based Visual Servo** 851
34.4 **Advanced Approaches** 854
34.4.1 Hybrid VS 854
34.4.2 Partitioned VS 855
34.5 **Performance Optimization and Planning** 856
34.5.1 Optimal Control and Redundancy Framework 856
34.5.2 Switching Schemes 856
34.5.3 Feature Trajectory Planning 857
34.6 **Estimation of 3-D Parameters** 858
34.7 **Determining s^* and Matching Issues** 859
34.8 **Target Tracking** 859
34.9 **Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space** 860
34.10 **Under Actuated Robots** 861
34.11 **Applications** 863
34.12 **Conclusions** 863
Video-References 863
References 863

Visual servo (VS) *control* refers to the use of computer vision data to control the motion of a robot. The vision data may be acquired from a camera that is mounted directly on a robot manipulator or on a mobile robot, in which case motion of the robot induces camera motion, or the camera can be fixed in the workspace so that it can observe the robot motion from a stationary configuration. Other configurations can be considered, such as

for instance several cameras mounted on pan–tilt heads observing the robot motion. The mathematical development of all these cases is similar, and in this chapter we will focus primarily on the former, so-called *eye-in-hand*, case.

Visual servo control relies on techniques from image processing, computer vision, and control theory. In the present chapter, we will deal primarily with the issues from control theory, making connections to previous chapters when appropriate.

34.1 The Basic Components of Visual Servoing

The aim of all vision-based control schemes is to minimize an error $\boldsymbol{e}(t)$, which is typically defined by

$$\boldsymbol{e}(t) = \boldsymbol{s}(\boldsymbol{m}(t), \boldsymbol{a}) - \boldsymbol{s}^* \,. \tag{34.1}$$

This formulation is quite general, and it encompasses a wide variety approaches, as we will see below. The parameters in (34.1) are defined as follows. The vector $\boldsymbol{m}(t)$ is a set of image measurements (e.g., the image coordinates of interest points, or the parameters of a set of image lines or segments). These image measurements are used to compute a vector of k visual features, $\boldsymbol{s}(\boldsymbol{m}(t), \boldsymbol{a})$, in which $\boldsymbol{a}$ is a set of parameters that represent potential additional knowledge about the system (e.g., true or approximate camera intrinsic parameters or a model of the object to be tracked). The vector $\boldsymbol{s}^*$ contains the desired values of the features. Note that the order of the desired and actual values in (34.1) is reversed with respect to the common convention for feedback control systems.

For now, we consider the case of a fixed goal pose and a motionless target, i. e., $\boldsymbol{s}^*$ is constant, and changes in $\boldsymbol{s}$ depend only on camera motion. Further, we consider here the case of controlling the motion of a camera with six degrees of freedom (e.g., a camera attached to the end effector of a six-degree-of-freedom arm). We will treat more general cases in later sections.

Visual servoing schemes mainly differ in the way that $\boldsymbol{s}$ is designed. In Sects. 34.2 and 34.3, we describe classical approaches, including image-based visual servo control (IBVS), in which $\boldsymbol{s}$ consists of a set of features that are immediately available in the image, and pose-based visual servo control (PBVS), in which $\boldsymbol{s}$ consists of a pose, which must be estimated from image measurements. Note that in the older visual servoing literature PBVS is named position-based, rather than pose-based, visual servoing [34.1, 2]. We also present in Sect. 34.4 several more-advanced methods.

Once $\boldsymbol{s}$ is selected, the design of the control scheme can be quite simple. Perhaps the most straightforward approach is to design a velocity controller. To do this, we require the relationship between the time variation of $\boldsymbol{s}$ and the camera velocity. Let the spatial velocity of the camera be denoted by $\boldsymbol{v}_c = (\boldsymbol{v}_c, \boldsymbol{\omega}_c)$ where $\boldsymbol{v}_c$ is the instantaneous linear velocity of the origin of the camera frame and $\boldsymbol{\omega}_c$ is the instantaneous angular velocity of the camera frame. The relationship between $\dot{\boldsymbol{s}}$ and $\boldsymbol{v}_c$ is given by

$$\dot{\boldsymbol{s}} = \mathbf{L}_{\boldsymbol{s}} \boldsymbol{v}_c \,, \tag{34.2}$$

in which $\mathbf{L}_{\boldsymbol{s}} \in \mathbb{R}^{k\times 6}$ is called the *interaction matrix* related to $\boldsymbol{s}$ [34.3]. The term *feature Jacobian* is also used somewhat interchangeably in the visual servo literature [34.2], but in the present chapter we will use this latter term to relate the time variation of the features to the robot's joint velocity (Sect. 34.8).

Using (34.1) and (34.2) we immediately obtain the relationship between camera velocity and the time variation of the error

$$\dot{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{e}} \boldsymbol{v}_c \,, \tag{34.3}$$

where $\mathbf{L}_{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{s}}$. Considering $\boldsymbol{v}_c$ as the input to the robot controller, and if we would like, for instance, to design for an exponential and decoupled decrease of the error (i. e., $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$) then using (34.3) we obtain as controller

$$\boldsymbol{v}_c = -\lambda \mathbf{L}_{\boldsymbol{e}}^{+} \boldsymbol{e} \,, \tag{34.4}$$

where $\mathbf{L}_{\boldsymbol{e}}^{+} \in \mathbb{R}^{6\times k}$ is the Moore–Penrose pseudo-inverse of $\mathbf{L}_{\boldsymbol{e}}$, that is, $\mathbf{L}_{\boldsymbol{e}}^{+} = (\mathbf{L}_{\boldsymbol{e}}^{\mathrm{T}} \mathbf{L}_{\boldsymbol{e}})^{-1} \mathbf{L}_{\boldsymbol{e}}^{\mathrm{T}}$ when $k \geq 6$ and $\mathbf{L}_{\boldsymbol{e}}$ is of full rank 6. When $k = 6$, if $\det \mathbf{L}_{\boldsymbol{e}} \neq 0$ it is possible to invert $\mathbf{L}_{\boldsymbol{e}}$, giving the control $\boldsymbol{v}_c = -\lambda \mathbf{L}_{\boldsymbol{e}}^{-1} \boldsymbol{e}$. When $k \leq 6$ and $\mathbf{L}_{\boldsymbol{e}}$ is of full rank k, $\mathbf{L}_{\boldsymbol{e}}^{+}$ is given by

$$\mathbf{L}_{\boldsymbol{e}}^{+} = \mathbf{L}_{\boldsymbol{e}}^{\mathrm{T}} \left(\mathbf{L}_{\boldsymbol{e}} \mathbf{L}_{\boldsymbol{e}}^{\mathrm{T}} \right)^{-1} \,.$$

When $\mathbf{L}_{\boldsymbol{e}}$ is not full rank, the numerical value of $\mathbf{L}_{\boldsymbol{e}}^{+}$ can be obtained from the singular value decomposition of $\mathbf{L}_{\boldsymbol{e}}$. In all cases, control scheme (34.4) allows $\| \dot{\boldsymbol{e}} - \lambda \mathbf{L}_{\boldsymbol{e}} \mathbf{L}_{\boldsymbol{e}}^{+} \boldsymbol{e} \|$ and $\| \boldsymbol{v}_c \|$ to be minimal. Note that the desired behavior $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$ is obtained only when $\mathbf{L}_{\boldsymbol{e}} \mathbf{L}_{\boldsymbol{e}}^{+} = \mathbf{I}_k$, where $\mathbf{I}_k$ is the $k \times k$ identity matrix, that is, only when $\mathbf{L}_{\boldsymbol{e}}$ if of full rank k, $k \leq 6$.

In real visual servo systems, it is impossible to know perfectly in practice either $\mathbf{L}_{\boldsymbol{e}}$ or $\mathbf{L}_{\boldsymbol{e}}^{+}$. So an approximation or an estimation of one of these two matrices must

be realized. In the sequel, we denote both the pseudo-inverse of the approximation of the interaction matrix and the approximation of the pseudo-inverse of the interaction matrix by the symbol $\widehat{\mathbf{L}_e^+}$. Using this notation, the control law is in fact

$$\boldsymbol{v}_c = -\lambda \widehat{\mathbf{L}_e^+} \boldsymbol{e} = -\lambda \widehat{\mathbf{L}_s^+} (\boldsymbol{s} - \boldsymbol{s}^*) \,. \tag{34.5}$$

Closing the loop and assuming that the robot controller is able to realize perfectly $\boldsymbol{v}_c$, that is inserting (34.5) into (34.3), we obtain

$$\dot{\boldsymbol{e}} = -\lambda \mathbf{L}_e \widehat{\mathbf{L}_e^+} \boldsymbol{e} \,. \tag{34.6}$$

This equation characterizes the actual behavior of the closed-loop system, which is different from the desired one ($\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$) when ever

$$\mathbf{L}_e \widehat{\mathbf{L}_e^+} \neq \mathbf{I}_k \,.$$

It is also the basis of the stability analysis of the system using Lyapunov theory.

What we have presented above is the basic design implemented by most visual servo controllers. All that remains is to fill in the details. How should $\boldsymbol{s}$ be chosen? What then is the form of $\mathbf{L}_s$? How should we estimate $\widehat{\mathbf{L}_e^+}$? What are the performance characteristics of the resulting closed-loop system? These questions are addressed in the remainder of the chapter. We first describe the two basic approaches, IBVS and PBVS, whose principles were proposed more than 20 years ago [34.1]. We then present more-recent approaches that have improved their performance.

34.2 Image-Based Visual Servo

Traditional image-based control schemes [34.1, 4] use the image-plane normalised coordinates of a set of points to define the vector $\boldsymbol{s}$. The image measurements $\boldsymbol{m}$ are usually the pixel coordinates of the set of image points (although this is not the only possible choice), and the parameters $\boldsymbol{a}$ in the definition of $\boldsymbol{s} = \boldsymbol{s}(\boldsymbol{m}, \boldsymbol{a})$ in (34.1) are nothing but the camera intrinsic parameters to go from image measurements expressed in pixels to the features.

34.2.1 The Interaction Matrix

A three-dimensional world point with coordinates $\boldsymbol{X} = (X, Y, Z)$ in the camera frame projects into the image plane of a conventional perspective camera as a two-dimensional point with normalised coordinates $\boldsymbol{x} = (x, y)$. More precisely we have

$$\begin{cases} x &= \dfrac{X}{Z} = \dfrac{u - c_u}{f\alpha} \\ y &= \dfrac{Y}{Z} = \dfrac{v - c_v}{f} \end{cases} , \tag{34.7}$$

where $\boldsymbol{m} = (u, v)$ gives the coordinates of the image point expressed in pixel units, and $\boldsymbol{a} = (c_u, c_v, f, \alpha)$ is the set of camera intrinsic parameters as defined in Chap. 32: c_u and c_v are the coordinates of the principal point, f is the focal length, and α is the ratio of the pixel dimensions. The intrinsic parameter β defined in Chap. 32 has been assumed to be 0 here. In this case, we take $\boldsymbol{s} = \boldsymbol{x} = (x, y)$, the image plane coordinates of the point. The details of the imaging geometry and perspective projection can be found in many computer vision texts, including [34.5–7].

Taking the time derivative of the projection equations (34.7), we obtain

$$\begin{cases} \dot{x} &= \dfrac{\dot{X}}{Z} - \dfrac{X\dot{Z}}{Z^2} = \dfrac{\dot{X} - x\dot{Z}}{Z} \\ \dot{y} &= \dfrac{\dot{Y}}{Z} - \dfrac{Y\dot{Z}}{Z^2} = \dfrac{\dot{Y} - y\dot{Z}}{Z} \end{cases} . \tag{34.8}$$

We can relate the velocity of the 3-D point to the camera spatial velocity using the well-known equation

$$\dot{\boldsymbol{X}} = -\boldsymbol{v}_c - \boldsymbol{\omega}_c \times \boldsymbol{X} \Leftrightarrow \begin{cases} \dot{X} = -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} = -v_y - \omega_z X + \omega_x Z \\ \dot{Z} = -v_z - \omega_x Y + \omega_y X \,, \end{cases} \tag{34.9}$$

where $\boldsymbol{v}_c = (v_x, v_y, v_z)$ and $\boldsymbol{\omega}_c = (\omega_x, \omega_y, \omega_z)$. Inserting (34.9) into (34.8), grouping terms, and using (34.7) we obtain

$$\begin{cases} \dot{x} = \dfrac{-v_x}{Z} + \dfrac{x v_z}{Z} + xy\omega_x - (1 + x^2)\omega_y + y\omega_z \\ \dot{y} = \dfrac{-v_y}{Z} + \dfrac{y v_z}{Z} + (1 + y^2)\omega_x - xy\omega_y - x\omega_z \,, \end{cases} \tag{34.10}$$

which can be written

$$\dot{\boldsymbol{x}} = \mathbf{L}_{\boldsymbol{x}} \boldsymbol{v}_c \,, \tag{34.11}$$

where the interaction matrix $\mathbf{L}_x$ is given by

$$\mathbf{L}_x = \begin{pmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{pmatrix} . \tag{34.12}$$

In the matrix $\mathbf{L}_x$, the value Z is the depth of the point relative to the camera frame. Therefore, any control scheme that uses this form of the interaction matrix must estimate or approximate the value of Z. Similarly, the camera intrinsic parameters are involved in the computation of x and y. Thus $\mathbf{L}_x^+$ cannot be *directly* used in (34.4), and an estimation or an approximation $\widehat{\mathbf{L}_x^+}$ must be used, as in (34.5). We discuss this in more detail below.

To control the six degrees of freedom, at least three points are necessary (i. e., we require $k \geq 6$). If we use the feature vector $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3)$, by merely stacking interaction matrices for three points we obtain

$$\mathbf{L}_x = \begin{pmatrix} \mathbf{L}_{\boldsymbol{x}_1} \\ \mathbf{L}_{\boldsymbol{x}_2} \\ \mathbf{L}_{\boldsymbol{x}_3} \end{pmatrix} .$$

In this case, there will exist some configurations for which $\mathbf{L}_x$ is singular [34.8]. Furthermore, there exist four distinct camera poses for which $\boldsymbol{e} = \mathbf{0}$, i. e., four global minima exist for the error function $\|\boldsymbol{e}\|$, and it is impossible to differentiate them [34.9]. For these reasons, more than three points are usually considered.

34.2.2 Approximating the Interaction Matrix

There are several choices available for constructing the estimate $\widehat{\mathbf{L}_e^+}$ to be used in the control law. One popular scheme is of course to choose

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_e^+$$

if $\mathbf{L}_e = \mathbf{L}_x$ is known, that is if the current depth Z of each point is available [34.2]. In practice, these parameters must be estimated at each iteration of the control scheme. The basic IBVS methods use classical pose-estimation methods (Chap. 32 and the beginning of Sect. 34.3). Another popular approach is to choose

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$$

where $\mathbf{L}_{e^*}$ is the value of $\mathbf{L}_e$ for the desired position $\boldsymbol{e} = \boldsymbol{e}^* = 0$ [34.3]. In this case, $\widehat{\mathbf{L}_e^+}$ is constant, and only the desired depth of each point has to be set, which means no varying 3-D parameters have to be estimated during the visual servo. Finally, the choice

$$\widehat{\mathbf{L}_e^+} = \left(\frac{\mathbf{L}_e}{2} + \frac{\mathbf{L}_{e^*}}{2} \right)^+$$

has been proposed [34.10]. Since $\mathbf{L}_e$ is involved in this method, the current depth of each point also has to be available.

We illustrate the behavior of these control schemes with an example. The goal is to position the camera so that it observes a square centered in the image (Fig. 34.1). We define $\boldsymbol{s}$ to include the x and y coordinates of the four points forming the square. Note

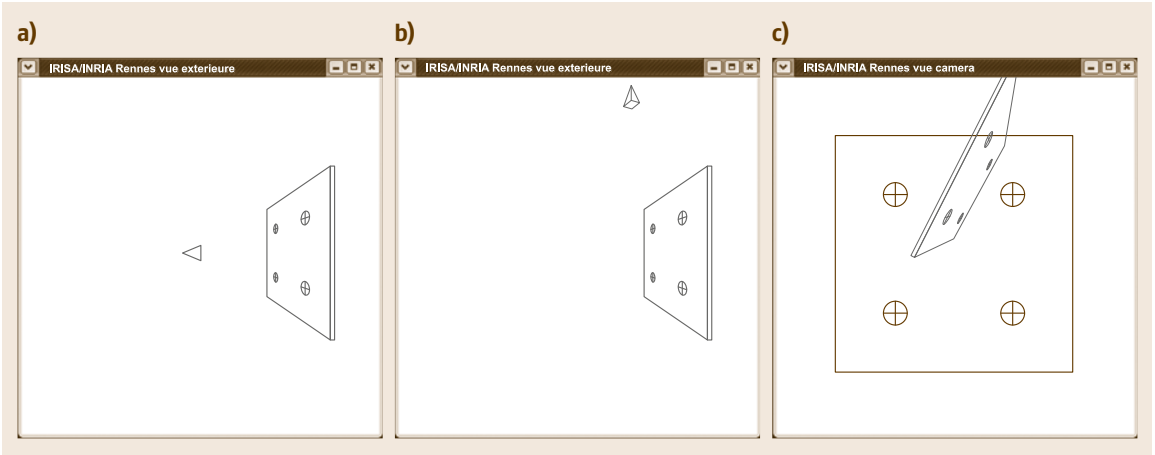


Fig.34.1a–c Example of positioning task: **(a)** the desired camera pose with respect to a simple target, **(b)** the initial camera pose, and **(c)** the corresponding initial and desired image of the target

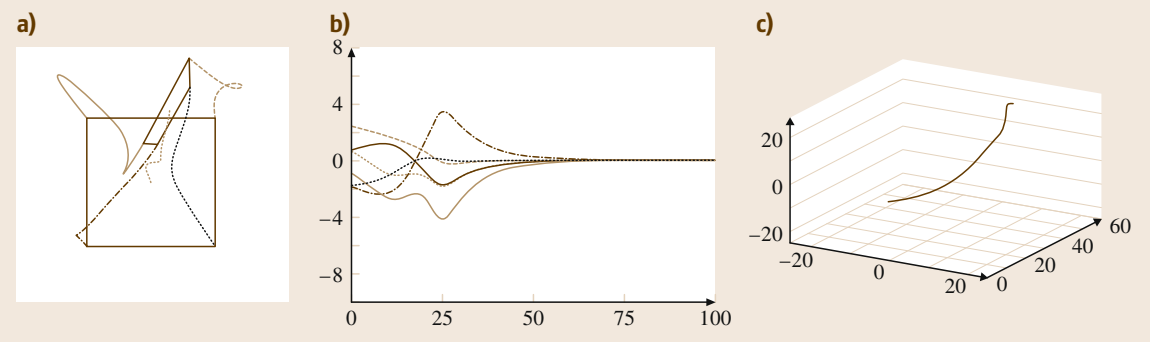


Fig.34.2a–c IBVS system behavior using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$: **(a)** image point trajectories including the trajectory of the center of the square, which is not used in the control scheme, **(b)** $\boldsymbol{v}_c$ components (cm/s and deg/s) computed at each iteration of the control scheme, and **(c)** the 3-D trajectory of the camera optical center expressed in the desired camera frame $\mathcal{R}_{c^*}$ (cm)

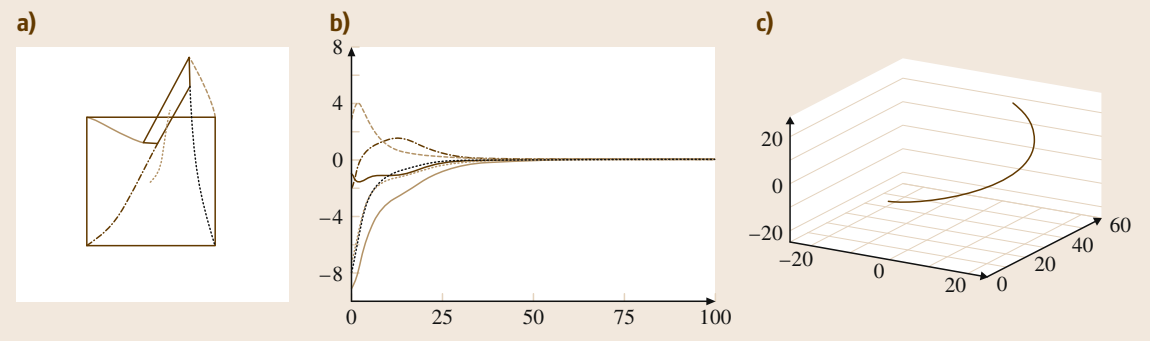


Fig.34.3a–c IBVS system behavior using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_e^+} = \mathbf{L}_e^+$ (refer to Fig. 34.2a-c for description)

that the initial camera pose has been selected far away from the desired pose, particularly with regard to the rotational motions, which are known to be the most problematic for IBVS. In the simulations presented in the following, no noise or modeling errors have been introduced in order to allow comparison of different behaviors in perfect conditions. The accompanying videos show experimental results obtained using an Adept Viper robot arm and the ViSP library [34.11]. The videos all show the same task being performed from the same initial conditions, and only the control approach varies.

The results obtained by using

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$$

are given in Fig. 34.2 and VIDEO 59. Note that despite the large displacement that is required the system converges. However, neither the behavior in the image, nor the computed camera velocity components, nor the 3-D trajectory of the camera present desirable properties far from the convergence (i. e., for the first 30 or so iterations).

The results obtained using

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_e^+$$

are given in Fig. 34.3 and VIDEO 60. In this case, the trajectories of the points in the image are almost straight lines, which means that the points never leave the camera's field of view. However the behavior induced in the camera frame is even less satisfactory than for the case of

$$\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+ .$$

The large camera velocities at the beginning of the servo indicate that the condition number of $\widehat{\mathbf{L}_e^+}$ is high at the start of the trajectory, and the camera trajectory is far from a straight line.

The choice

$$\widehat{\mathbf{L}_e^+} = \left(\frac{\mathbf{L}_e}{2} + \frac{\mathbf{L}_{e^*}}{2}\right)^+$$

provides good performance in practice. Indeed, as can be seen in Fig. 34.4, the camera velocity components do not include large oscillations, which provides

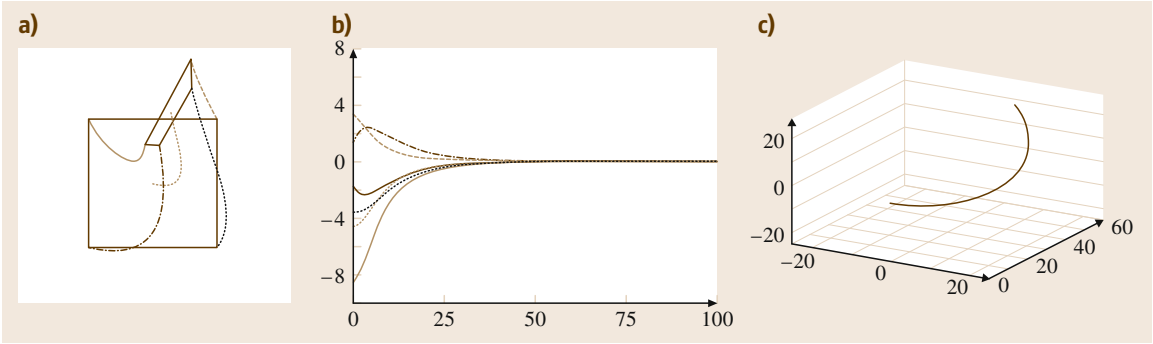


Fig. 34.4 IBVS system behavior using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_e^+} = \left(\mathbf{L}_e/2 + \mathbf{L}_e^*/2\right)^+$ (refer to Fig. 34.2a-c for description)

a smooth trajectory in both the image and in 3-D space (VIDEO 61).

34.2.3 A Geometrical Interpretation of IBVS

It is quite easy to provide a geometric interpretation of the behavior of the control schemes defined above. The example illustrated in Fig. 34.5 corresponds to a pure rotation around the optical axis from the initial configuration (shown in blue) to the desired configuration of four coplanar points parallel to the image plane (shown in red).

As explained above, using $\mathbf{L}_e^+$ in the control scheme attempts to ensure an exponential decrease of the error $\boldsymbol{e}$. This means that, when x and y image point coordinates compose this error, the points' trajectories in the image follow straight lines from their initial to their desired positions, when this is possible. This leads to the image motion plotted in green in the figure. The camera motion to realize this image motion can be easily deduced and is indeed composed of a rotational motion around the optical axis, but combined with a retreating translational motion along the optical axis [34.12]. This unexpected motion is due to the choice of the features and the form of the third and sixth columns in the interaction matrix, which induces a coupling between the features and the two degrees of freedom involved (v_z and ω_z). If the rotation between the initial and desired configurations is very large, this phenomenon is amplified, and leads to a particular case for a rotation of π radians where no rotational motion at all will be induced by the control scheme [34.13]. On the other hand, when the rotation is small, this phenomenon almost disappears. To conclude, the behavior is locally satisfactory (i. e., when the error is small), but it can be unsatisfactory when the error is large. As we will see below, these results are consistent with the local asymptotic stability results that can be obtained for IBVS.

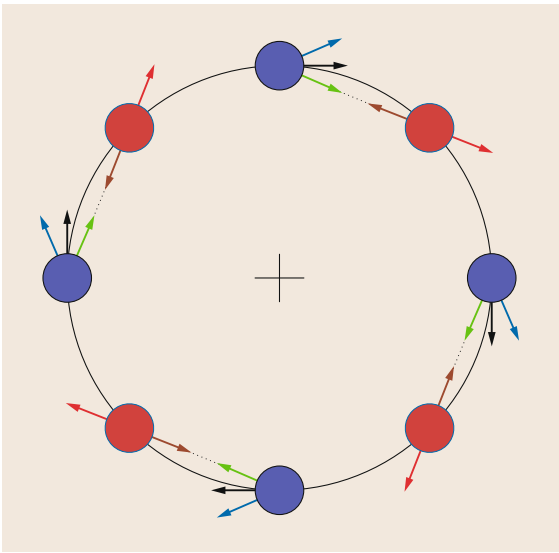


Fig. 34.5 Geometrical interpretation of IBVS: going from the *blue position* to the *red one*. In *green*, image motion when $\mathbf{L}_e^+$ is used in the control scheme; in *blue*, when $\mathbf{L}_{e^*}^+$ is used; and in *black* when $(\mathbf{L}_e/2 + \mathbf{L}_{e^*}/2)^+$ is used (see the text for more details)

If instead we use $\mathbf{L}_{e^*}^+$ in the control scheme, the image motion generated can easily be shown to be the blue one plotted in Fig. 34.5. Indeed, if we consider the same control scheme as before but starting from $\boldsymbol{s}^*$ to reach $\boldsymbol{s}$, we obtain

$$\boldsymbol{v}_c = -\lambda \mathbf{L}_{e^*}^+ (\boldsymbol{s}^* - \boldsymbol{s}) \,,$$

which again induces straight-line trajectories from the red points to the blue ones, causing the image motion plotted in brown. Going back to our problem, the control scheme computes a camera velocity that is exactly the opposite one

$$\boldsymbol{v}_c = -\lambda \mathbf{L}_{e^*}^+ (\boldsymbol{s} - \boldsymbol{s}^*) \,,$$

and thus generates the image motion plotted in red at the red points. Transformed to the blue points, the camera velocity generates the blue image motion and corresponds once again to a rotational motion around the optical axis, combined now with an unexpected forward motion along the optical axis. The same analysis can be done as before for the case of large or small errors. We can add that, as soon as the error decreases significantly, both control schemes get closer, and tend to the same one (since $\mathbf{L}_e = \mathbf{L}_{e^*}$ when $\boldsymbol{e} = \boldsymbol{e}^*$) with a nice behavior characterized with the image motion plotted in black and a camera motion composed of only a rotation around the optical axis when the error tends towards zero.

If we instead use

$$\widehat{\mathbf{L}_e^+} = \left(\frac{\mathbf{L}_e}{2} + \frac{\mathbf{L}_{e^*}}{2}\right)^+ ,$$

it is intuitively clear that considering the mean of $\mathbf{L}_e$ and $\mathbf{L}_{e^*}$ generates the image motion plotted in black, even when the error is large. In all cases but the rotation of π radians, the camera motion is now a pure rotation around the optical axis, without any unexpected translational motion.

34.2.4 Stability Analysis

We now consider the fundamental issues related to the stability of IBVS. To assess the stability of the closed-loop visual servo systems, we will use Lyapunov analysis. In particular, consider the candidate Lyapunov function defined by the squared error norm $\mathcal{L} = \frac{1}{2}\|\boldsymbol{e}(t)\|^2$, whose derivative is given by

$$\dot{\mathcal{L}} = \boldsymbol{e}^{\mathrm{T}}\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}^{\mathrm{T}}\mathbf{L}_e\widehat{\mathbf{L}_e^+}\boldsymbol{e}$$

since $\dot{\boldsymbol{e}}$ is given by (34.6). The global asymptotic stability of the system is thus obtained when the following sufficient condition is satisfied

$$\mathbf{L}_e\widehat{\mathbf{L}_e^+} > 0 . \tag{34.13}$$

If the number of features is equal to the number of camera degrees of freedom (i. e., $k = 6$), and if the features are chosen and the control scheme designed so that $\mathbf{L}_e$ and $\widehat{\mathbf{L}_e^+}$ are of full rank 6, then the condition (34.13) is satisfied if the approximations involved in $\widehat{\mathbf{L}_e^+}$ are not too coarse.

As discussed above, for most IBVS approaches we have $k > 6$. Therefore the condition (34.13) can never be ensured since

$$\mathbf{L}_e\widehat{\mathbf{L}_e^+} \in \mathbb{R}^{k\times k}$$

is at most of rank 6, and thus $\mathbf{L}_e\widehat{\mathbf{L}_e^+}$ has a nontrivial null space. In this case, configurations such that $\boldsymbol{e} \in \ker \widehat{\mathbf{L}_e^+}$ correspond to local minima. Reaching such a local minimum is illustrated in Fig. 34.6. As can be seen in Fig. 34.6d, each component of $\boldsymbol{e}$ has a nice exponential decrease with the same convergence speed, causing straight-line trajectories to be realized in the image, but the error reached is not exactly zero, and it is clear from Fig. 34.6c that the system has been attracted to a local minimum far away from the desired configuration. Thus, only local asymptotic stability can be obtained for IBVS.

To study local asymptotic stability when $k > 6$, let us first define a new error $\boldsymbol{e}'$ with $\boldsymbol{e}' = \widehat{\mathbf{L}_e^+}\boldsymbol{e}$. The time derivative of this error is given by

$$\dot{\boldsymbol{e}}' = \widehat{\mathbf{L}_e^+}\dot{\boldsymbol{e}} + \dot{\widehat{\mathbf{L}_e^+}}\boldsymbol{e} = \left(\widehat{\mathbf{L}_e^+}\mathbf{L}_e + \boldsymbol{O}\right)\boldsymbol{v}_c ,$$

where $\boldsymbol{O} \in \mathbb{R}^{6\times 6}$ is equal to $\mathbf{0}$ when $\boldsymbol{e} = 0$, whatever the choice of $\widehat{\mathbf{L}_e^+}$ [34.14]. Using the control scheme (34.5),w e obtain

$$\dot{\boldsymbol{e}}' = -\lambda\left(\widehat{\mathbf{L}_e^+}\mathbf{L}_e + \boldsymbol{O}\right)\boldsymbol{e}' ,$$

which is known to be locally asymptotically stable in a neighborhood of $\boldsymbol{e} = \boldsymbol{e}^* = \mathbf{0}$ if

$$\widehat{\mathbf{L}_e^+}\mathbf{L}_e > 0 , \tag{34.14}$$

where $\widehat{\mathbf{L}_e^+}\mathbf{L}_e \in \mathbb{R}^{6\times 6}$. Indeed, only the linearized system

$$\dot{\boldsymbol{e}}' = -\lambda\widehat{\mathbf{L}_e^+}\mathbf{L}_e\boldsymbol{e}'$$

has to be considered if we are interested in the local asymptotic stability [34.15].

Once again, if the features are chosen and the control scheme designed so that $\mathbf{L}_e$ and $\widehat{\mathbf{L}_e^+}$ are of full rank 6, then condition (34.14) is ensured if the approximations involved in $\widehat{\mathbf{L}_e^+}$ are not too coarse.

To end the demonstration of local asymptotic stability, we must show that there does not exist any configuration $\boldsymbol{e} \neq \boldsymbol{e}^*$ such that $\boldsymbol{e} \in \ker \widehat{\mathbf{L}_e^+}$ in a small neighborhood of $\boldsymbol{e}^*$ and in a small neighborhood of the corresponding pose $\boldsymbol{p}^*$. Such configurations correspond to local minima where $\boldsymbol{v}_c = 0$ and $\boldsymbol{e} \neq \boldsymbol{e}^*$. If such a pose $\boldsymbol{p}$ would exist, it is possible to restrict the neighborhood around $\boldsymbol{p}^*$ so that there exists a camera velocity $\boldsymbol{v}$ to reach $\boldsymbol{p}^*$ from $\boldsymbol{p}$. This camera velocity would imply a variation of the error $\dot{\boldsymbol{e}} = \mathbf{L}_e\boldsymbol{v}$. However, such a variation cannot belong to $\ker \widehat{\mathbf{L}_e^+}$ since $\widehat{\mathbf{L}_e^+}\mathbf{L}_e > 0$. Therefore, we have $\boldsymbol{v}_c = 0$ if and only if $\dot{\boldsymbol{e}} = \mathbf{0}$, i. e., $\boldsymbol{e} = \boldsymbol{e}^*$, in a neighborhood of $\boldsymbol{p}^*$.

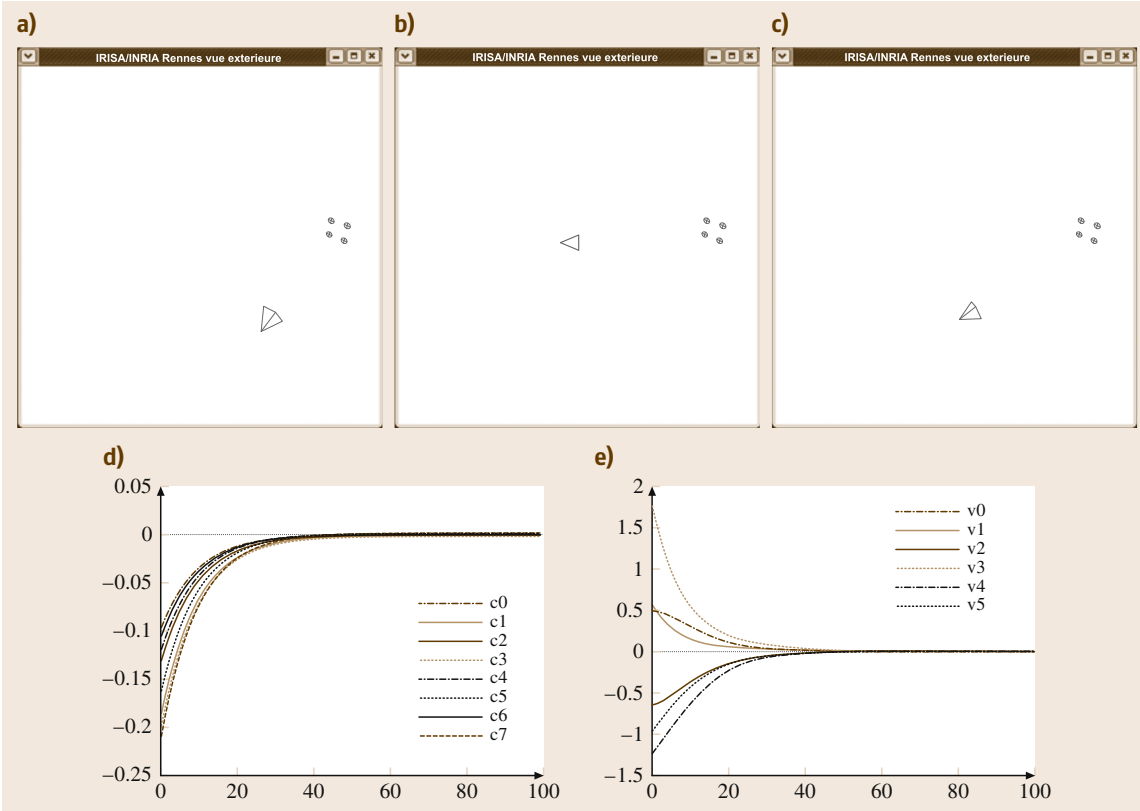


Fig.34.6a–e IBVS reaching a local minimum using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}}^{+}$: **(a)** the initial configuration, **(b)** the desired one, **(c)** the configuration reached after the convergence of the control scheme, **(d)** the evolution of the error $\boldsymbol{e}$ at each iteration of the control scheme, and **(e)** the evolution of the six components of the camera velocity $\boldsymbol{v}_c$

Even though local asymptotic stability can be ensured when $k > 6$, we recall that global asymptotic stability cannot be ensured. For instance, as illustrated in Fig. 34.6, there may exist local minima corresponding to configurations where $\boldsymbol{e} \in \ker \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$, which are outside of the neighborhood considered above. Determining the size of the neighborhood in which stability and the convergence are ensured is still an open issue, even if this neighborhood is surprisingly quite large in practice.

34.2.5 IBVS with a Stereo Vision System

It is straightforward to extend the IBVS approach to a multicamera system. If a stereo vision system is used, and a world point is visible in both left and right images (Fig. 34.7), it is possible to use as visual features

$$\boldsymbol{s} = \boldsymbol{x}_s = (\boldsymbol{x}_l, \boldsymbol{x}_r) = (x_l, y_l, x_r, y_r)$$

i. e., to represent the point by just stacking in $\boldsymbol{s}$ the x and y coordinates of the observed point in the left and right images [34.16]. However, care must be taken when constructing the corresponding interaction matrix since the form given in (34.11) is expressed in either the left or right camera frame. More precisely, we have

$$\begin{cases} \dot{\boldsymbol{x}}_l &= \mathbf{L}_{\boldsymbol{x}_l}\, \boldsymbol{v}_l \,, \\ \dot{\boldsymbol{x}}_r &= \mathbf{L}_{\boldsymbol{x}_r}\, \boldsymbol{v}_r \,, \end{cases}$$

where $\boldsymbol{v}_l$ and $\boldsymbol{v}_r$ are the spatial velocity of the left and right camera, respectively, and where the analytical form of $\mathbf{L}_{\boldsymbol{x}_l}$ and $\mathbf{L}_{\boldsymbol{x}_r}$ are given by (34.12).

By choosing a sensor frame rigidly linked to the stereo vision system, we obtain

$$\dot{\boldsymbol{x}}_s = \begin{pmatrix} \dot{\boldsymbol{x}}_l \\ \dot{\boldsymbol{x}}_r \end{pmatrix} = \mathbf{L}_{\boldsymbol{x}_s}\, \boldsymbol{v}_s \,,$$

where the interaction matrix related to $\boldsymbol{x}_s$ can be determined using the spatial motion transform matrix $\mathbf{V}$ defined in Chap. 02 to transform velocities expressed in the left or right cameras frames to the sensor frame. We

recall that $\mathbf{V}$ is given by

$$\mathbf{V} = \begin{pmatrix} \mathbf{R} & [\boldsymbol{t}]_\times \mathbf{R} \\ \mathbf{0} & \mathbf{R} \end{pmatrix} , \tag{34.15}$$

where $[\boldsymbol{t}]_\times$ is the skew-symmetric matrix associated to the vector $\boldsymbol{t}$ and where $(\mathbf{R}, \boldsymbol{t}) \in SE(3)$ is the rigid-body transformation from the camera to the sensor frame. The numerical values for these matrices are directly obtained from the calibration step of the stereo vision system. Using this equation, we obtain

$$\mathbf{L}_{x_s} = \begin{pmatrix} \mathbf{L}_{x_l}\, {}^{l}\mathbf{V}_s \\ \mathbf{L}_{x_r}\, {}^{r}\mathbf{V}_s \end{pmatrix} .$$

Note that $\mathbf{L}_{x_s} \in \mathbb{R}^{4\times 6}$ is always of rank 3 because of the epipolar constraint that links the perspective projection of a 3-D point in a stereo vision system (Fig. 34.7). Another simple interpretation is that a 3-D point is represented by three independent parameters, which makes it impossible to find more than three independent parameters using any sensor observing that point.

To control the six degrees of freedom of the system, it is necessary to consider at least three points, as the rank of the interaction matrix considering only two points is 5.

Using a stereo vision system, since the 3-D coordinates of any point observed in both images can be easily estimated by a simple triangulation process it is possible and quite natural to use these 3-D coordinates in the features set $\boldsymbol{s}$. Such an approach would be, strictly speaking, a position-based approach, since it would require 3-D parameters in $\boldsymbol{s}$.

34.2.6 IBVS with Cylindrical Coordinates of Image Points

In the previous sections, we have considered the Cartesian coordinates of image points. As proposed in [34.17] it may be useful to consider instead the cylindrical coordinates $\boldsymbol{\gamma} = (\rho, \theta)$ of the image points instead of their Cartesian coordinates $\boldsymbol{x} = (x, y)$. They are given by

$$\rho = \sqrt{x^2 + y^2} \ , \ \theta = \arctan \frac{y}{x}$$

from which we deduce

$$\dot{\rho} = \frac{x\dot{x} + y\dot{y}}{\rho} \ , \ \dot{\theta} = \frac{x\dot{y} - y\dot{x}}{\rho^2} \ .$$

Using (34.11) and then substituting x by $\rho\cos\theta$ and y by $\rho\sin\theta$, we obtain immediately

$$\mathbf{L}_{\boldsymbol{\gamma}} = \begin{pmatrix} \frac{-c}{Z} & \frac{-s}{Z} & \frac{\rho}{Z} & (1+\rho^2)s & -(1+\rho^2)c & 0 \\ \frac{s}{\rho Z} & \frac{-c}{\rho Z} & 0 & \frac{c}{\rho} & \frac{s}{\rho} & -1 \end{pmatrix} , \tag{34.16}$$

where $c = \cos\theta$ and $s = \sin\theta$. Note that θ is not defined when the image point lies at the principal point (where $x = y = \rho = 0$). It is thus not surprising that the interaction matrix $\mathbf{L}_{\boldsymbol{\gamma}}$ is singular in that case.

If we go back to the example depicted in Fig. 34.5, the behavior obtained using cylindrical coordinates will be the expected one, that is a pure rotation around the optical axis, by using either $\mathbf{L}_{e}^{+}$, $\mathbf{L}_{e^*}^{+}$ or $(\mathbf{L}_e/2 + \mathbf{L}_{e^*}/2)^{+}$ in the control scheme. This is due to the form of the third and sixth columns of the interaction matrix (34.16), which leads to a decoupled system.

34.2.7 IBVS with Other Geometrical Features

In the previous sections, we have only considered image point coordinates in $\boldsymbol{s}$. Other geometrical primitives can of course be used. There are several reasons to do so. Firstly, the scene observed by the camera cannot always be described merely by a collection of points, in which case the image processing provides other types of measurements, such as a set of straight lines or the contours of an object. Secondly, richer geometric primitives may ameliorate the decoupling and linearizing issues that motivate the design of partitioned systems (Sect. 34.4). Finally, the robotic task to be achieved may be expressed in terms of virtual linkages (or fixtures) between the camera and the observed objects [34.18, 19], sometimes expressed directly by constraints between primitives, such as point-to-line [34.20] (which means that an observed point must lie on a specified line).

It is possible to determine the interaction matrix related to the perspective projection of a large class of geometrical primitives, such as segments, straight lines, spheres, circles, and cylinders. The results are given in [34.3, 18]. Recently, the analytical form of the interaction matrix related to any image moments corresponding to planar objects has been computed. This

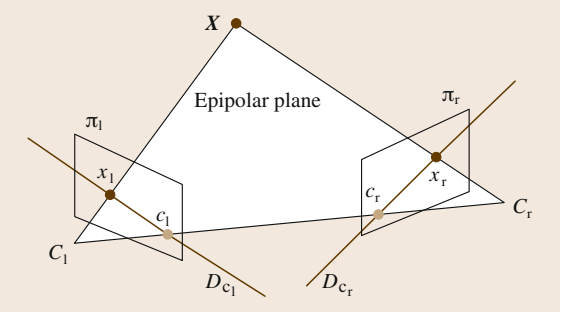


Fig. 34.7 A stereo vision system

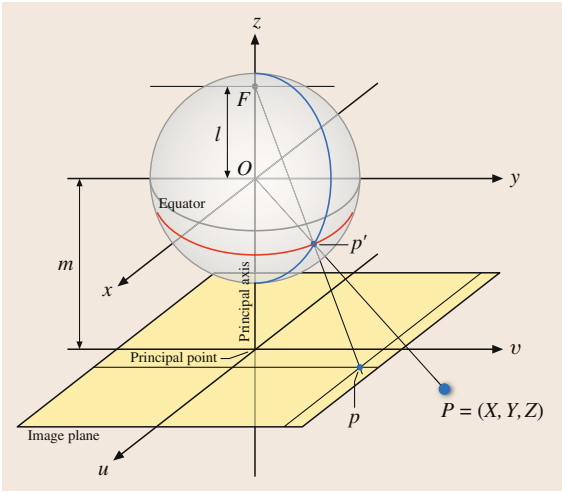


Fig. 34.8 Unified imaging model of Geyer and Daniilidis (after [34.7])

makes it possible to consider planar objects of any shape [34.21]. If a collection of points is measured in the image, moments can also be used [34.22]. In both cases, moments allow the use of intuitive geometrical features, such as the center of gravity or the orientation of an object. By selecting an adequate combination of moments, it is then possible to determine partitioned systems with good decoupling and linearizing properties [34.21, 22].

Note that, for all these features (geometrical primitives, moments), the depth of the primitive or of the object considered appears in the coefficients of the interaction matrix related to the translational degrees of freedom, as was the case for the image points. An estimation of this depth is thus generally necessary (Sect. 34.6). In a few situations, for instance with a suitable normalization of moments [34.22], only the constant desired depth appears in the interaction matrix, which makes estimating depth unnecessary.

34.2.8 Non-Perspective Cameras

The vast majority of cameras we use, and our own eyes, are characterized by a perspective projection model which closely approximate the ideal pinhole imaging model. Such cameras have a narrow field of view, typically less than one half hemisphere. For robotics it is often advantageous to have a large field of view and this can be achieved using a fisheye lens camera or a catadioptric (lens and mirror system) camera (often referred to as a panoramic camera). For these non-perspective sensors the interaction matrix of any visual feature has a different form to those discussed above, such as (34.12) and (34.16) for an image point.

Rather than determine the interaction matrix of visual features expressed in the image plane of non-perspective cameras, we can transform the images from these cameras to the view that would be seen by an ideal spherical camera. The spherical model projects world points onto a unit sphere, the intersection of the unit sphere with the ray from the world point to the center of the sphere. Such an ideal camera has the largest possible field of view. The unified imaging model [34.23] shown in Fig 34.8b provides a general mechanism to project a world point to the image plane of a large class of cameras. To be precise it includes all central projection cameras and this includes perspective and some catadioptric cameras with particular mirror shapes, but in practice it is a very good approximation to non-central cameras including fisheye and general catadioptric systems. The mechanism of this unified model can also be used to reproject points from these varied image planes to a spherical camera.

Referring to Fig. 34.8 the world point **P** is represented by the vector $\boldsymbol{X} = (X, Y, Z)$ in the camera frame, and is projected onto the surface of the unit sphere at the point $\boldsymbol{x}_s = (x_s, y_s, z_s)$ with

$$x_s = \frac{X}{R}\,, \quad y_s = \frac{Y}{R}\,, \quad \text{and} \quad z_s = \frac{Z}{R}\,,$$

where $R = \sqrt{X^2 + Y^2 + Z^2}$ is the distance from the sphere center to the world point.

The interaction matrix of $\boldsymbol{x}_s$ is derived in essentially the same manner as for the perspective camera, and it can be shown to be [34.24]

$$\mathbf{L}_{\boldsymbol{x}_s} = \left[\frac{1}{R}\left(\boldsymbol{x}_s \boldsymbol{x}_s^{\mathrm{T}} - \mathbf{I}_3\right)[\boldsymbol{x}_s]_\times\right]\,. \tag{34.17}$$

Note that R can be expressed as a function of the point depth Z by using $R = Z\sqrt{1 + x_s^2 + y_s^2}$. Therefore, the general spherical model does not add any supplementary unknown in the interaction matrix.

A particular advantage of the spherical model is that for pure camera rotation the shape of an object is invariant, which eases determining visual features that are only linked to translational motions.

34.2.9 Direct Estimation

In the previous sections, we have focused on the analytical form of the interaction matrix. It is also possible to estimate its numerical value directly using either an offline learning step, or an online estimation scheme.

All the methods proposed to estimate the interaction matrix numerically rely on the observation of a variation of the features due to a known or measured camera motion. More precisely, if we measure a feature's variation $\Delta \boldsymbol{s}$ due to a camera motion $\Delta \boldsymbol{v}_c$, we

have from (34.2)

$$\mathbf{L}_s \Delta \boldsymbol{v}_c = \Delta \boldsymbol{s} \,,$$

which provides k equations while we have $k \times 6$ unknown values in $\mathbf{L}_s$. Using a set of N independent camera motions with $N > 6$, it is thus possible to estimate $\mathbf{L}_s$ by solving

$$\mathbf{L}_s \boldsymbol{A} = \boldsymbol{B} \,,$$

where the columns of $\boldsymbol{A} \in \mathbb{R}^{6\times N}$ and $\boldsymbol{B} \in \mathbb{R}^{k\times N}$ are, respectively, formed from the set of camera motions and the set of corresponding features variations. The least-square solution is of course given by

$$\widehat{\mathbf{L}_s} = \boldsymbol{B}\boldsymbol{A}^{+} \,. \tag{34.18}$$

Methods based on neural networks have also been developed to estimate $\mathbf{L}_s$ [34.25, 26]. It is also possible to estimate the numerical value of $\mathbf{L}_s^{+}$ directly, which in practice provides a better behavior [34.27]. In this case, the basic relation is

$$\mathbf{L}_s^{+} \Delta \boldsymbol{s} = \Delta \boldsymbol{v}_c \,,$$

which provides six equations. Using a set of N measurements, with $N > k$, we now obtain

$$\widehat{\mathbf{L}_s^{+}} = \boldsymbol{A}\boldsymbol{B}^{+} \,. \tag{34.19}$$

In the first case (34.18), the six columns of $\mathbf{L}_s$ are estimated by solving six linear systems, while in the second case (34.19), the k columns of $\mathbf{L}_s^{+}$ are estimated by solving k linear systems, which explains the difference in the results.

Estimating the interaction matrix online can be viewed as an optimization problem, and consequently a number of researchers have investigated approaches that derive from optimization methods. These methods typically discretize the system equation (34.2), and use an iterative updating scheme to refine the estimate of $\widehat{\mathbf{L}_s}$ at each stage. One such online and iterative formulation uses the Broyden update rule given by [34.28, 29]

$$\widehat{\mathbf{L}_s}(t+1) = \widehat{\mathbf{L}_s}(t) + \frac{\alpha}{\Delta \boldsymbol{v}_c^{\mathrm{T}} \Delta \boldsymbol{v}_c} \left[\Delta \boldsymbol{x} - \widehat{\mathbf{L}_s}(t) \Delta \boldsymbol{v}_c \right] \Delta \boldsymbol{v}_c^{\mathrm{T}} \,,$$

where α defines the update speed. This method has been generalized to the case of moving objects in [34.30].

The main interest of using such numerical estimations in the control scheme is that it avoids all the modeling and calibration steps. It is particularly useful when using features whose interaction matrix is not available in analytical form. For instance, in [34.31], the main eigenvalues of the principal component analysis of an image have been considered in a visual servoing scheme. The drawback of these methods is that no theoretical stability and robustness analysis can be made.

34.3 Pose-Based Visual Servo

Pose-based control schemes (PBVS) [34.1, 32, 33] use the pose of the camera with respect to some reference coordinate frame to define $\boldsymbol{s}$. Computing this pose from a set of measurements in one image necessitates the camera intrinsic parameters and the 3-D model of the object observed to be known. This classic computer vision problem is called the 3-D localization problem. While this problem is beyond the scope of the present chapter, many solutions have been presented in the literature [34.34, 35] and its basic principles are recalled in Chap. 32.

It is then typical to define $\boldsymbol{s}$ in terms of the parameterization used to represent the camera pose. Note that the parameters $\boldsymbol{a}$ involved in the definition (34.1) of $\boldsymbol{s}$ are now the camera intrinsic parameters and the 3-D model of the object.

It is convenient to consider three coordinate frames: the current camera frame $\mathcal{F}_c$, the desired camera frame $\mathcal{F}_{c^*}$, and a reference frame $\mathcal{F}_o$ attached to the object. We adopt here the standard notation of using a leading superscript to denote the frame with respect to which a set of coordinates is defined. Thus, the coordinate vectors ${}^{c}\boldsymbol{t}_o$ and ${}^{c^*}\boldsymbol{t}_o$ give the coordinates of the origin of the object frame expressed relative to the current camera frame, and relative to the desired camera frame, respectively. Furthermore, let $\mathbf{R} = {}^{c^*}\mathbf{R}_c$ be the rotation matrix that gives the orientation of the current camera frame relative to the desired frame.

We can define $\boldsymbol{s}$ to be $(\boldsymbol{t}, \theta\boldsymbol{u})$, in which $\boldsymbol{t}$ is a translation vector, and $\theta\boldsymbol{u}$ gives the angle/axis parameterization for the rotation. We now discuss two choices for $\boldsymbol{t}$, and give the corresponding control laws.

If $\boldsymbol{t}$ is defined relative to the object frame $\mathcal{F}_o$, we obtain

$$\boldsymbol{s} = ({}^{c}\boldsymbol{t}_o, \theta\boldsymbol{u}) \,, \boldsymbol{s}^* = ({}^{c^*}\boldsymbol{t}_o, 0) \,,$$

and

$$\boldsymbol{e} = ({}^{c}\boldsymbol{t}_o - {}^{c^*}\boldsymbol{t}_o, \theta\boldsymbol{u}) \,.$$

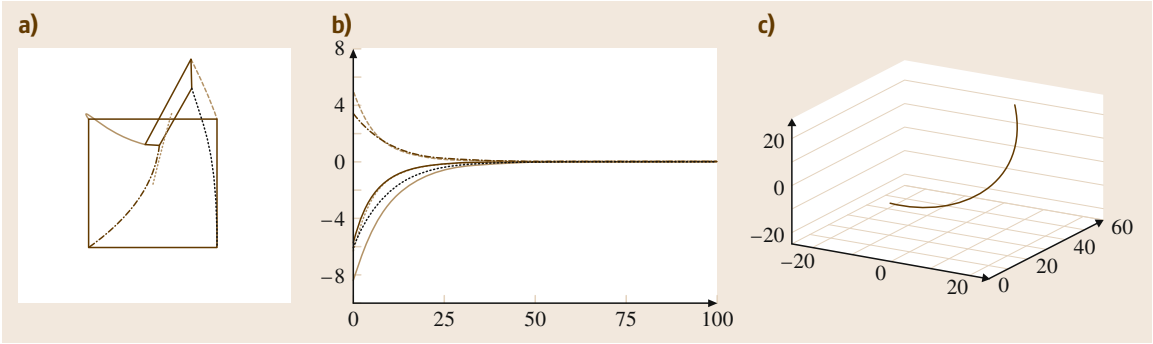


Fig. 34.9 PBVS system behavior using $\boldsymbol{s} = ({}^{c}\boldsymbol{t}_{o}, \theta\boldsymbol{u})$ (refer to Fig. 34.2a-c for description)

In this case, the interaction matrix related to $\boldsymbol{e}$ is given by

$$\mathbf{L}_{\boldsymbol{e}} = \begin{pmatrix} -\mathbf{I}_3 & [{}^{c}\boldsymbol{t}_{o}]_{\times} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}} \end{pmatrix} , \tag{34.20}$$

in which $\mathbf{I}_3$ is the 3×3 identity matrix and $\mathbf{L}_{\theta\boldsymbol{u}}$ is given by [34.36]

$$\mathbf{L}_{\theta\boldsymbol{u}} = \mathbf{I}_3 + \frac{\theta}{2}[\boldsymbol{u}]_{\times} + \left(1 - \frac{\operatorname{sinc}\theta}{\operatorname{sinc}^2\theta/2}\right)[\boldsymbol{u}]_{\times}^2 , \tag{34.21}$$

where $\operatorname{sinc} x$ is the sinus cardinal defined such that $x \operatorname{sinc} x = \sin x$ and $\operatorname{sinc} 0 = 1$.

Following the development in Sect. 34.1, we obtain the control scheme

$$\boldsymbol{v}_{c} = -\lambda \widehat{\mathbf{L}_{\boldsymbol{e}}^{-1}} \boldsymbol{e}$$

since the dimension k of $\boldsymbol{s}$ is six, that is, the number of camera degrees of freedom. By setting

$$\widehat{\mathbf{L}_{\boldsymbol{e}}^{-1}} = \begin{pmatrix} -\mathbf{I}_3 & [{}^{c}\boldsymbol{t}_{o}]_{\times}\mathbf{L}_{\theta\boldsymbol{u}}^{-1} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}}^{-1} \end{pmatrix} , \tag{34.22}$$

we obtain after simple developments

$$\begin{cases} \boldsymbol{v}_{c} = -\lambda\left[{}^{c^*}\boldsymbol{t}_{o} - {}^{c}\boldsymbol{t}_{o}\right] + [{}^{c}\boldsymbol{t}_{o}]_{\times}\,\theta\boldsymbol{u} \\ \boldsymbol{\omega}_{c} = -\lambda\theta\boldsymbol{u} \end{cases} . \tag{34.23}$$

since $\mathbf{L}_{\theta\boldsymbol{u}}$ is such that

$$\mathbf{L}_{\theta\boldsymbol{u}}^{-1}\theta\boldsymbol{u} = \theta\boldsymbol{u} .$$

Ideally, that is, if the pose parameters are perfectly estimated, the behavior of $\boldsymbol{e}$ will be the expected one ($\dot{\boldsymbol{e}} = -\lambda\boldsymbol{e}$). The choice of $\boldsymbol{e}$ causes the rotational motion to follow a geodesic with an exponential decreasing speed and causes the translational parameters involved in $\boldsymbol{s}$ to decrease at the same speed. This explains the nice exponential decrease of the camera velocity components in Fig. 34.9. Furthermore, the trajectory in the image of the origin of the object frame follows a straight line (here the center of the four points has been selected as this origin). On the other hand, the camera trajectory does not follow a straight line (VIDEO 62).

Another PBVS scheme can be designed by using $\boldsymbol{s} = ({}^{c^*}\boldsymbol{t}_{c}, \theta\boldsymbol{u})$. In this case, we have $\boldsymbol{s}^* = \mathbf{0}$, $\boldsymbol{e} = \boldsymbol{s}$, and

$$\mathbf{L}_{\boldsymbol{e}} = \begin{pmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}} \end{pmatrix} . \tag{34.24}$$

Note the decoupling between translational and rotational motions, which allows us to obtain a simple control scheme

$$\begin{cases} \boldsymbol{v}_{c} &= -\lambda\mathbf{R}^{\mathrm{T}}\,{}^{c^*}\boldsymbol{t}_{c} \\ \boldsymbol{\omega}_{c} &= -\lambda\,\theta\boldsymbol{u} \end{cases} . \tag{34.25}$$

In this case, as can be seen in Fig. 34.10 and in VIDEO 63, if the pose parameters involved in (34.25) are estimated perfectly, the camera trajectory is a straight line, while the image trajectories are less satisfactory than before. Some particular configurations can be found which will lead to some points leaving the camera field of view during the robot motion.

The stability properties of PBVS seem quite attractive. Since $\mathbf{L}_{\theta\boldsymbol{u}}$ given in (34.21) is nonsingular when $\theta \neq 2k\pi$, $\forall k \in Z^*$, we obtain from (34.13) the global asymptotic stability of the system since $\mathbf{L}_{\boldsymbol{e}}\widehat{\mathbf{L}_{\boldsymbol{e}}^{-1}} = \mathbf{I}_6$, under the strong hypothesis that all the pose parameters are perfect. This is true for both methods presented above, since the interaction matrices given in (34.20) and (34.24) are full rank when $\mathbf{L}_{\theta\boldsymbol{u}}$ is nonsingular.

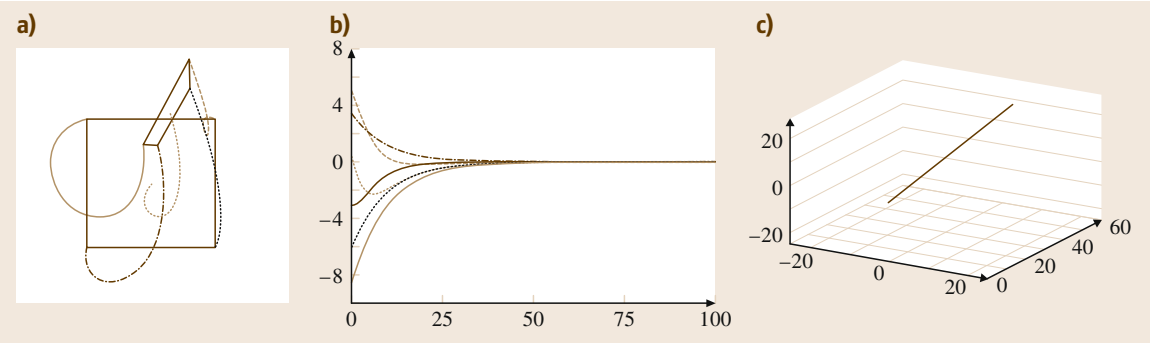


Fig. 34.10 PBVS system behavior using $\boldsymbol{s} = ({}^{c^*}\boldsymbol{t}_c, \theta\boldsymbol{u})$ (refer to Fig. 34.2a-c for description)

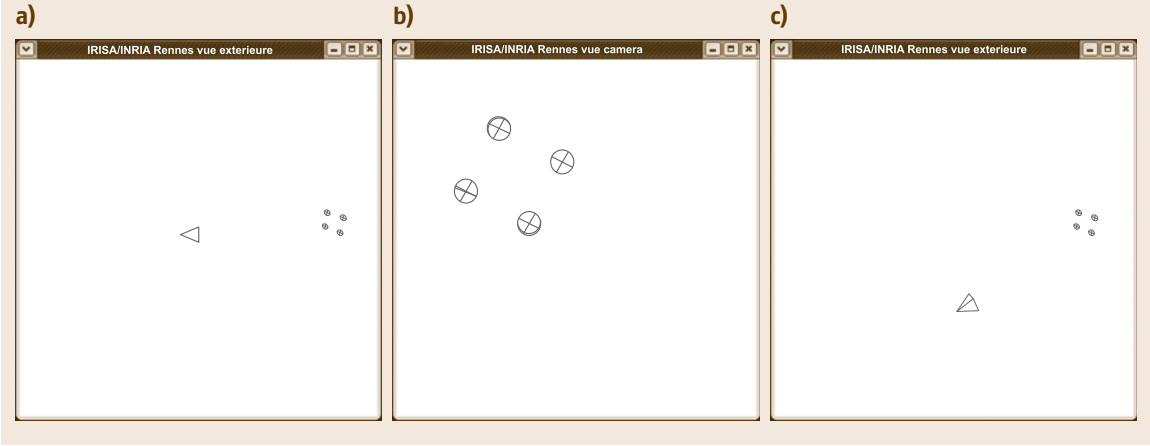


Fig.34.11a–c Two different camera poses **(a,c)** that provide almost the same image of four coplanar points shown overlaid in **(b)**

With regard to robustness, feedback is computed using *estimated* quantities that are a function of the image measurements and the system calibration parameters. For the first method presented in Sect. 34.3 (the analysis for the second method is analogous), the interaction matrix given in (34.20) corresponds to perfectly estimated pose parameters, while the real one is unknown since the estimated pose parameters may be biased due to calibration errors, or inaccurate and unstable due to noise [34.13]. The true positivity condition (34.13) should in fact be written

$$\mathbf{L}_{\hat{e}}\widehat{\mathbf{L}_{\hat{e}}^{-1}} > 0\,, \tag{34.26}$$

where $\widehat{\mathbf{L}_{\hat{e}}^{-1}}$ is given by (34.22) but where $\mathbf{L}_{\hat{e}}$ is unknown, and not given by (34.20). Indeed, even small errors in computing the position of points in the image can lead to pose errors which will significantly impact the accuracy and the stability of the system (Fig. 34.11).

34.4 Advanced Approaches

34.4.1 Hybrid VS

Suppose we have access to a control law for $\boldsymbol{\omega}_c$, such as the one used in PBVS ((34.23) or (34.25))

$$\boldsymbol{\omega}_c = -\lambda\, \theta \boldsymbol{u}\,. \tag{34.27}$$

How could we use this in conjunction with IBVS?

Considering a feature vector $\boldsymbol{s}_t$ and an error $\boldsymbol{e}_t$ devoted to control the translational degrees of freedom, we can partition the interaction matrix as follows

$$\begin{aligned}\dot{\boldsymbol{s}}_t &= \mathbf{L}_{\boldsymbol{s}_t} \boldsymbol{v}_c \\ &= \begin{pmatrix}\mathbf{L}_v & \mathbf{L}_{\boldsymbol{\omega}}\end{pmatrix}\begin{pmatrix}\boldsymbol{v}_c \\ \boldsymbol{\omega}_c\end{pmatrix} \\ &= \mathbf{L}_v \boldsymbol{v}_c + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c\,.\end{aligned}$$

Now, setting $\dot{\boldsymbol{e}}_t = -\lambda \boldsymbol{e}_t$, we can solve for the desired translational control input as

$$\begin{aligned}-\lambda \boldsymbol{e}_t = \dot{\boldsymbol{e}}_t = \dot{\boldsymbol{s}}_t = \mathbf{L}_v \boldsymbol{v}_c + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c\,, \\ \Rightarrow \boldsymbol{v}_c = -\mathbf{L}_v^{+}(\lambda \boldsymbol{e}_t + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c)\,.\end{aligned} \tag{34.28}$$

We can think of the quantity $(\lambda \boldsymbol{e}_t + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c)$ as a modified error term, one that combines the original error with the error that would be induced by the rotational motion due to $\boldsymbol{\omega}_c$. The translational control input $\boldsymbol{v}_c = -\mathbf{L}_v^{+}(\lambda \boldsymbol{e}_t + \mathbf{L}_{\boldsymbol{\omega}} \boldsymbol{\omega}_c)$ will drive this error to zero. The method known as 2.5-D visual servo [34.36] was the first to exploit such a partitioning in combining IBVS and PBVS. More precisely, in [34.36], $\boldsymbol{s}_t$ has been selected as the coordinates of an image point, and the logarithm of its depth, so that $\mathbf{L}_v$ is a triangular always invertible matrix. More precisely, we have $\boldsymbol{s}_t = (\boldsymbol{x}, \log Z)$, $\boldsymbol{s}_t^* = (\boldsymbol{x}^*, \log Z^*)$, $\boldsymbol{e}_t = (\boldsymbol{x} - \boldsymbol{x}^*, \log \rho_Z)$ where $\rho_Z = Z/Z^*$, and

$$\begin{aligned}\mathbf{L}_v &= \frac{1}{Z^* \rho_Z}\begin{pmatrix}-1 & 0 & x \\ 0 & -1 & y \\ 0 & 0 & -1\end{pmatrix}\mathbf{L}_{\boldsymbol{\omega}} \\ &= \begin{pmatrix}xy & -(1+x^2) & y \\ 1+y^2 & -xy & -x \\ -y & x & 0\end{pmatrix}\,.\end{aligned}$$

Note that the ratio ρ_Z can be obtained directly from the partial pose estimation algorithm that will be described in Sect. 34.6.

If we come back to the usual global representation of visual servo control schemes, we have $\boldsymbol{e} = (\boldsymbol{e}_t, \theta \boldsymbol{u})$ and $\mathbf{L}_e$ given by

$$\mathbf{L}_e = \begin{pmatrix}\mathbf{L}_v & \mathbf{L}_{\boldsymbol{\omega}} \\ \mathbf{0} & \mathbf{L}_{\theta u}\end{pmatrix}\,,$$

from which we immediately obtain the control law (34.27) and (34.28) by applying (34.5).

The behavior obtained using this choice for $\boldsymbol{s}_t$ is shown in Fig. 34.12 and VIDEO 64. Here, the point that has been considered in $\boldsymbol{s}_t$ is the center of gravity $\boldsymbol{x}_g$ of the target. We note the image trajectory of that point, which is a straight line as expected, and the nice decreasing of the camera velocity components, which makes this scheme very similar to the first PBVS one.

As for stability, it is clear that this scheme is globally asymptotically stable in perfect conditions. Furthermore, thanks to the triangular form of the interaction matrix $\mathbf{L}_e$, it is possible to analyze the stability of this scheme in the presence of calibration errors using the partial pose-estimation algorithm that will be described in Sect. 34.6 [34.37]. Finally, the only unknown constant parameter involved in this scheme, that is Z^*, can be estimated online using adaptive techniques [34.38].

Other hybrid schemes can be designed. For instance, in [34.39], the third component of $\boldsymbol{s}_t$ is different and has been selected so that all the target points remain in the camera field of view as far as possible. Another example has been proposed in [34.40]. In that case, $\boldsymbol{s}$ is selected as $\boldsymbol{s} = ({}^{c^*}\boldsymbol{t}_c, \boldsymbol{x}_g, \theta u_z)$ which provides with a block-triangular interaction matrix of the form

$$\mathbf{L}_e = \begin{pmatrix}\mathbf{R} & \mathbf{0} \\ \mathbf{L}'_v & \mathbf{L}'_{\boldsymbol{\omega}}\end{pmatrix}\,,$$

where $\mathbf{L}'_v$ and $\mathbf{L}'_{\boldsymbol{\omega}}$ can easily be computed. This scheme is such that, under perfect conditions, the camera trajectory is a straight line (since ${}^{c^*}\boldsymbol{t}_c$ is a part of $\boldsymbol{s}$), and the image trajectory of the center of gravity of the object is also a straight line (since $\boldsymbol{x}_g$ is also a part of $\boldsymbol{s}$). The translational camera degrees of freedom are devoted to realize the 3-D straight line, while the rotational camera degrees of freedom are devoted to realize the 2-D straight line and also compensate the 2-D motion of $\boldsymbol{x}_g$ due to the translational motion. As can be seen in Fig. 34.13 and VIDEO 65, this scheme is particularly satisfactory in practice.

Finally, it is possible to combine 2-D and 3-D features in different ways. For instance, in [34.41], it has been proposed to use in $\boldsymbol{s}$ the 2-D homogeneous coordinates of a set of image points expressed in pixels multiplied by their corresponding depth:

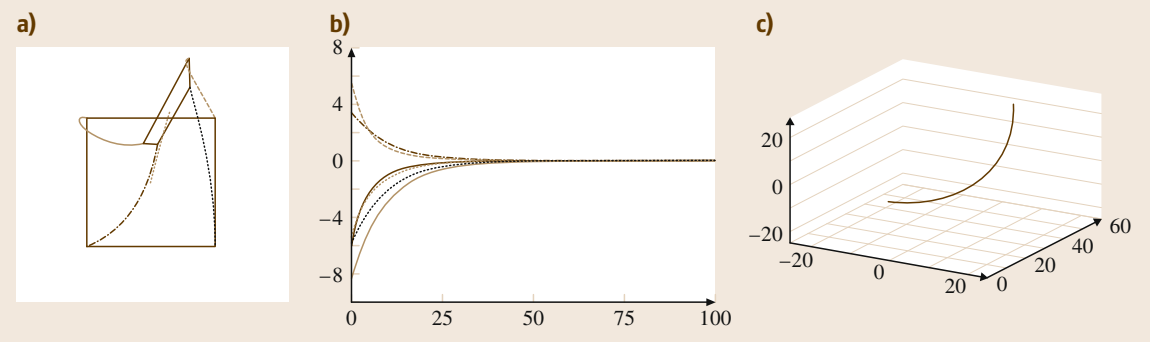


Fig. 34.12 2.5-D VS system behavior using $\boldsymbol{s} = (\mathbf{x}_g, \log(Z_g), \theta\boldsymbol{u})$ (refer to Fig. 34.2a-c for description)

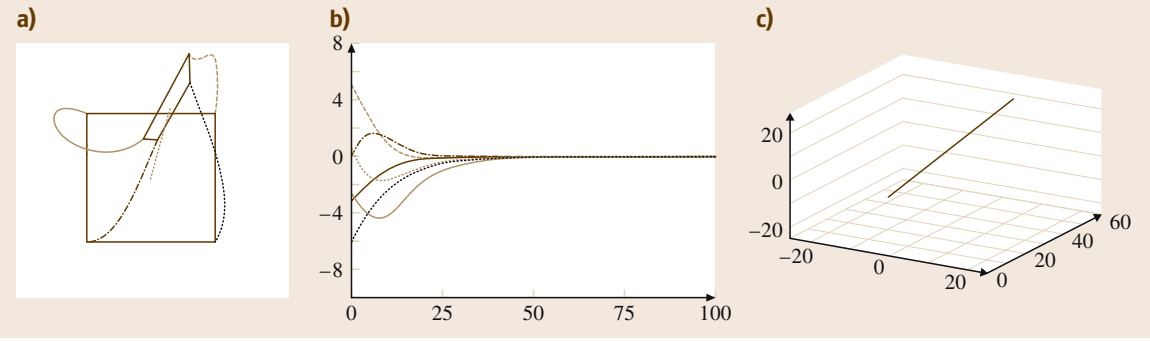


Fig. 34.13 2.5-D VS system behavior using $\boldsymbol{s} = ({}^{c^*}\mathbf{t}_c, \mathbf{x}_g, \theta u_z)$ (refer to Fig. 34.2a-c for description)

$\boldsymbol{s} = (u_1 Z_1, v_1 Z_1, Z_1, \cdots, u_n Z_n, v_n Z_n, Z_n)$. As for classical IBVS, we obtain in this case a set of redundant features, since at least three points have to be used to control the six camera degrees of freedom (here $k \geq 9$). However, it has been demonstrated in [34.42] that this selection of redundant features is free of attractive local minima.

34.4.2 Partitioned VS

The hybrid visual servo schemes described above have been designed to decouple the rotational motions from the translational ones by selecting adequate visual features defined in part in 2-D, and in part in 3-D (which is why they have been called 2.5-D visual servoing). This work has inspired some researchers to find features that exhibit similar decoupling properties but using only features expressed directly in the image. More precisely, the goal is to find six features such that each is related to only one degree of freedom (in which case the interaction matrix is a diagonal matrix). The Grail is to find a diagonal interaction matrix whose elements are constant, as near as possible to the identity matrix, leading to a pure, direct, and simple linear control problem.

The first work in this area partitioned the interaction matrix to isolate motion related to the optic axis [34.12]. Indeed, whatever the choice of $\boldsymbol{s}$, we have

$$\begin{aligned}\dot{\boldsymbol{s}} &= \mathbf{L}_{\boldsymbol{s}} \boldsymbol{v}_c \\ &= \mathbf{L}_{xy} \boldsymbol{v}_{xy} + \mathbf{L}_z \boldsymbol{v}_z \\ &= \dot{\boldsymbol{s}}_{xy} + \dot{\boldsymbol{s}}_z\end{aligned}$$

in which $\mathbf{L}_{xy}$ includes the first, second, fourth, and fifth columns of $\mathbf{L}_{\boldsymbol{s}}$, and $\mathbf{L}_z$ includes the third and sixth columns of $\mathbf{L}_{\boldsymbol{s}}$. Similarly, $\boldsymbol{v}_{xy} = (v_x, v_y, \omega_x, \omega_y)$ and $\boldsymbol{v}_z = (v_z, \omega_z)$. Here, $\dot{\boldsymbol{s}}_z = \mathbf{L}_z \boldsymbol{v}_z$ gives the component of $\dot{\boldsymbol{s}}$ due to the camera motion along and rotation about the optic axis, while $\dot{\boldsymbol{s}}_{xy} = \mathbf{L}_{xy} \boldsymbol{v}_{xy}$ gives the component of $\dot{\boldsymbol{s}}$ due to velocity along and rotation about the camera x and y axes.

Proceeding as above, by setting $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$ we obtain

$$-\lambda \boldsymbol{e} = \dot{\boldsymbol{e}} = \dot{\boldsymbol{s}} = \mathbf{L}_{xy} \boldsymbol{v}_{xy} + \mathbf{L}_z \boldsymbol{v}_z \,,$$

which leads to

$$\boldsymbol{v}_{xy} = -\mathbf{L}_{xy}^{+} [\lambda \boldsymbol{e}(t) + \mathbf{L}_z \boldsymbol{v}_z] \,.$$

As before, we can consider $[\lambda \boldsymbol{e}(t) + \mathbf{L}_z \boldsymbol{v}_z]$ as a modified error that incorporates the original error while taking into account the error that will be induced by $\boldsymbol{v}_z$.

Given this result, all that remains is to choose $\boldsymbol{s}$ and $\boldsymbol{v}_z$. As for basic IBVS, the coordinates of a collection of image points can be used in $\boldsymbol{s}$, while two new image features can be defined to determine $\boldsymbol{v}_z$:

- Define α, with $0 \leq \alpha < 2\pi$, as the angle between the horizontal axis of the image plane and the directed line segment joining two feature points.
  It is clear that α is closely related to the rotation around the optic axis.
- Define σ^2 to be the area of the polygon defined by these points. Similarly, σ^2 is closely related to the translation along the optic axis.

Using these features, $\boldsymbol{v}_z$ has been defined in [34.12] as

$$\begin{cases} v_z &= \lambda_{v_z} \ln \frac{\sigma^*}{\sigma} \,, \\ \omega_z &= \lambda_{\omega_z}(\alpha^* - \alpha) \,. \end{cases}$$

34.5 Performance Optimization and Planning

In some sense, partitioned methods represent an effort to optimize system performance by assigning distinct features and controllers to individual degrees of freedom. In this way, the designer performs a sort of offline optimization when allocating controllers to degrees of freedom. It is also possible to explicitly design controllers that optimize various system performance measures. We describe a few of these in this section.

34.5.1 Optimal Control and Redundancy Framework

An example of such an approach is given in [34.43] and [34.44], in which linear quadratic Gaussian (LQG) control design is used to choose gains that minimize a linear combination of state and control inputs. This approach explicitly balances the trade-off between tracking errors (since the controller attempts to drive $\boldsymbol{s} - \boldsymbol{s}^*$ to zero) and robot motion. A similar control approach is proposed in [34.45] where joint limit avoidance is considered simultaneously with the positioning task.

It is also possible to formulate optimality criteria that explicitly express the observability of robot motion in the image. For example, the singular value decomposition of the interaction matrix reveals which degrees of freedom are most apparent and can thus be easily controlled, while the condition number of the interaction matrix gives a kind of global measure of the visibility of motion. This concept has been called resolvability in [34.46] and motion perceptibility in [34.47]. By selecting features and designing controllers that maximize these measures, either along specific degrees of freedom or globally, the performance of the visual servo system can be improved.

The constraints considered to design the control scheme using the optimal control approach may be contradictory in some cases, leading the system to fail due to local minima in the objective function to be minimized. For example, it may happen that the motion produced to move away from a robot joint limit is exactly the opposite of the motion produced toward the desired pose, which results in a zero global motion. To avoid this potential problem, it is possible to use the gradient projection method, which is classical in robotics. Applying this method to visual servoing has been proposed in [34.3, 19]. The approach consists of projecting the secondary constraints $\boldsymbol{e}_s$ onto the null space of the vision-based task $\boldsymbol{e}$ so that they have no effect on the regulation of $\boldsymbol{e}$ to 0

$$\boldsymbol{e}_{\mathrm{g}} = \widehat{\mathbf{L}_e^+}\boldsymbol{e} + \mathbf{P}_e\,\boldsymbol{e}_s \,,$$

where $\boldsymbol{e}_{\mathrm{g}}$ is the new global task considered and

$$\mathbf{P}_e = \left(\mathbf{I}_6 - \widehat{\mathbf{L}_e^+}\widehat{\mathbf{L}_e}\right)$$

is such that

$$\widehat{\mathbf{L}_e}\mathbf{P}_e\boldsymbol{e}_s = 0 \,, \forall \boldsymbol{e}_s \,.$$

Avoiding the robot joint limits using this approach has been presented in [34.48]. However, when the vision-based task constrains all the camera degrees of freedom, the secondary constraints cannot be considered since, when $\widehat{\mathbf{L}_e}$ is of full rank 6, we have $\mathbf{P}_e\boldsymbol{e}_s = 0, \forall \boldsymbol{e}_s$. In this case, it is necessary to insert the constraints into a global objective function, such as navigation functions that are free of local minima [34.49, 50].

34.5.2 Switching Schemes

The partitioned methods described previously attempt to optimize performance by assigning individual controllers to specific degrees of freedom. Another way to use multiple controllers to optimize performance is to design switching schemes that select at each moment

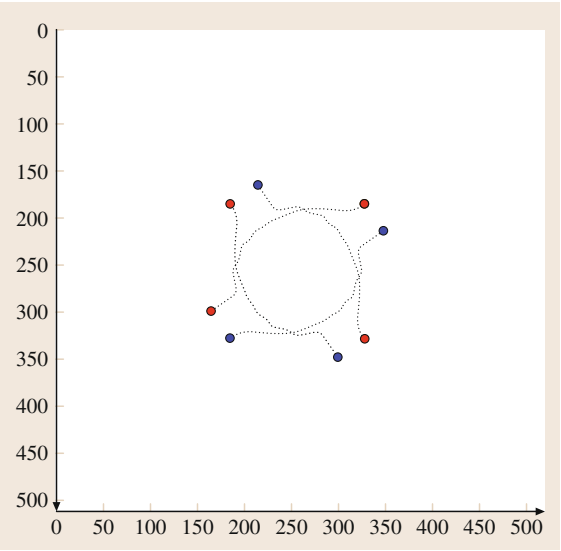


Fig. 34.14 Image feature trajectories for a rotation of 160° about the optical axis using a switched control scheme (initial point positions in *blue*, and desired points position in *red*)

in time which controller to use based on criteria to be optimized.

A simple switching controller can be designed using an IBVS and a PBVS controller as follows [34.51]. Let the system begin by using the IBVS controller. Consider the Lyapunov function for the PBVS controller given by $\mathcal{L}_\mathrm{P} = \frac{1}{2}\|\boldsymbol{e}_\mathrm{P}(t)\|^2$, with $\boldsymbol{e}_\mathrm{P}(t) = ({}^\mathrm{c}\boldsymbol{t}_\mathrm{o} - {}^{\mathrm{c}^*}\boldsymbol{t}_\mathrm{o}, \theta\boldsymbol{u})$. If at any time the value of this Lyapunov function exceeds a threshold γ_P, the system switches to the PBVS controller. While using the PBVS controller, if at any time the value of the Lyapunov function $\mathcal{L}_\mathrm{I}$ for the IBVS controller exceeds a threshold, $\mathcal{L}_\mathrm{I} = \frac{1}{2}\|\boldsymbol{e}_\mathrm{I}(t)\|^2 > \gamma_\mathrm{I}$, the system switches to the IBVS controller. With this scheme, when the Lyapunov function for a particular controller exceeds a threshold, that controller is invoked, which in turn reduces the value of the corresponding Lyapunov function. If the switching thresholds are selected appropriately, the system is able to exploit the relative advantages of IBVS and PBVS, while avoiding their shortcomings.

An example of such a system is shown in Fig. 34.14 for the case of a rotation by 160° about the optical axis. Note that the system begins in IBVS mode and the features initially move on straight lines toward their goal positions in the image. However, as the camera retreats, the system switches to PBVS, which allows the camera to reach its desired position by combining a rotational motion around its optic axis and a forward translational motion, producing the circular trajectories observed in the image.

Other examples of temporal switching schemes can be found, such as the one developed in [34.52], to ensure the visibility of the target observed.

34.5.3 Feature Trajectory Planning

It is also possible to treat the optimization problem offline, during a planning stage, if we have sufficient knowledge of the system and world. In this case, several constraints can be taken into account simultaneously, such as obstacle avoidance [34.53], joint limit and occlusions avoidance, and ensuring the visibility of the target [34.54]. The feature trajectories $\boldsymbol{s}^*(t)$ that allow the camera to reach its desired pose while ensuring that the constraints are satisfied are determined using path planning techniques, such as the well-known potential field approach [34.54] or linear matrix inequality optimizations [34.55].

Coupling path planning with trajectory following also allows the robustness of the visual servo with respect to modeling errors to be significantly improved. Indeed, modeling errors may have large effects when the error $\boldsymbol{s} - \boldsymbol{s}^*$ is large, but have little effect when $\boldsymbol{s} - \boldsymbol{s}^*$ is small. Once the desired features trajectories $\boldsymbol{s}^*(t)$ such that $\boldsymbol{s}^*(0) = \boldsymbol{s}(0)$ have been designed during the planning stage, it is easy to adapt the control scheme to take into account the fact that $\boldsymbol{s}^*$ is varying, and to make the error $\boldsymbol{s} - \boldsymbol{s}^*$ remain small. More precisely, we now have

$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{s}} - \dot{\boldsymbol{s}}^* = \mathbf{L}_{\boldsymbol{e}}\boldsymbol{v}_\mathrm{c} - \dot{\boldsymbol{s}}^* \,,$$

from which we deduce, by selecting as usual $\dot{\boldsymbol{e}} = -\lambda\boldsymbol{e}$ as the desired behavior,

$$\boldsymbol{v}_\mathrm{c} = -\lambda\widehat{\mathbf{L}_{\boldsymbol{e}}^+}\boldsymbol{e} + \widehat{\mathbf{L}_{\boldsymbol{e}}^+}\dot{\boldsymbol{s}}^* \,.$$

The new second term of this control law anticipates the variation of $\boldsymbol{s}^*$, removing the tracking error it would produce. We will see in the 34.8 that the form of the control law is similar when tracking of a moving target is considered.

34.6 Estimation of 3-D Parameters

All the control schemes described in the previous sections use 3-D parameters that are not directly available from the image measurements. As for IBVS, we recall that the range of the object with respect to the camera appears in the coefficients of the interaction matrix related to the translational degrees of freedom. Noticeable exceptions are the schemes based on a numerical estimation of $\mathbf{L}_{\boldsymbol{e}}$ or of $\mathbf{L}_{\boldsymbol{e}}^{+}$ (Sect. 34.2.8). Another exception is the IBVS scheme that uses the constant matrix $\widehat{\mathbf{L}_{e^*}^{+}}$ in the control scheme, in which only the depth for the desired pose is required, which is not so difficult to obtain in practice. As for PBVS and hybrid schemes that combine 2-D and 3-D data in $\boldsymbol{e}$, 3-D parameters appear both in the error $\boldsymbol{e}$ and in the interaction matrix. A correct estimation of the 3-D parameters involved is thus important for IBVS since they will have an effect on the camera motion during the task execution (they appear in the stability conditions (34.13) and (34.14)), while a correct estimation is crucial in PBVS and hybrid schemes since they will have also an effect on the accuracy of the pose reached after convergence.

If a calibrated stereo vision system is used, all 3-D parameters can be easily determined by triangulation, as mentioned in Sect. 34.2.5 and described in Chap. 32. Similarly, if a 3-D model of the object is known, all 3-D parameters can be computed from a pose-estimation algorithm. However, we recall that such an estimation can be quite unstable due to image noise (Sect. 34.3). As already said, IBVS does not require full pose estimation, simply the range of the object with respect to the camera. When image points are involved in $\boldsymbol{s}$, the range is expressed as the scalar depth Z or distance R of the corresponding world points, which appears in the interaction matrix (34.12), (34.16) and (34.17). This can be considered as a parameter estimation problem, that is, estimating the 3-D parameters from knowledge of the analytical form of the interaction matrix, and measurements of camera motion and visual feature position and velocity [34.56–58].

It is also possible to estimate 3-D parameters by using the epipolar geometry that relates the images of the same scene observed from different viewpoints. Indeed, in visual servoing, two images are generally available: the current one and the desired one. Given a set of matches between the image measurements in the current image and in the desired one, the fundamental matrix, or the essential matrix if the camera is calibrated, can be recovered [34.6], and then used in visual servoing [34.59]. Indeed, from the essential matrix, the rotation and the translation up to a scalar factor between the two views can be estimated. However, near the convergence of the visual servo, that is, when the current and desired images are similar, the epipolar geometry becomes degenerate and it is not possible to estimate accurately the partial pose between the two views. For this reason, using homography is generally preferred.

Let $\boldsymbol{x}_i$ and $\boldsymbol{x}_i^*$ denote the homogeneous image coordinates for a point in the current and desired images. Then $\boldsymbol{x}_i$ is related to $\boldsymbol{x}_i^*$ by

$$\boldsymbol{x}_i = \mathbf{H}_i \boldsymbol{x}_i^*$$

in which $\mathbf{H}_i$ is a homography matrix.

If all feature points lie on a 3-D plane, then there is a single homography matrix $\mathbf{H}$ such that $\boldsymbol{x}_i = \mathbf{H}\boldsymbol{x}_i^*$ for all i. This homography can be estimated using the position of four matched points in the desired and the current images. If all the features points do not belong to the same 3-D plane, then three points can be used to define such a plane and five supplementary points are needed to estimate $\mathbf{H}$ [34.60].

Once $\mathbf{H}$ is available, it can be decomposed as

$$\mathbf{H} = \mathbf{R} + \frac{\boldsymbol{t}}{d^*}\boldsymbol{n}^{*\mathrm{T}}, \tag{34.29}$$

in which $\mathbf{R}$ is the rotation matrix relating the orientation of the current and desired camera frames, $\boldsymbol{n}^*$ is the normal to the chosen 3-D plane expressed in the desired frame, d^* is the distance to the 3-D plane from the desired frame, and $\boldsymbol{t}$ is the translation between current and desired frames. From $\mathbf{H}$, it is thus possible to recover $\mathbf{R}$, $\boldsymbol{t}/d^*$, and $\boldsymbol{n}$. In fact, two solutions for these quantities exist [34.61], but it is quite easy to select the correct one using some knowledge about the desired pose. It is also possible to estimate the depth of any target point up to a common scale factor [34.54]. The unknown depth of each point that appears in classical IBVS can thus be expressed as a function of a single, constant parameter whatever the number of points. Similarly, the pose parameters required by PBVS can be recovered up to a scalar factor as for the translation term. The PBVS schemes described previously can thus be revisited using this approach, with the new error defined as the translation up to a scalar factor and the angle/axis parameterization of the rotation. This approach has also been used for the hybrid visual servoing schemes described in Sect. 34.4.1. In that case, using such homography estimation, it has been possible to analyze the stability of hybrid visual servoing schemes in the presence of calibration errors [34.36]. Finally, it is also possible to directly use the homography in the control scheme, avoiding thus its decomposition as a partial pose [34.62].

34.7 Determining s^* and Matching Issues

All visual servo methods require knowledge of the desired feature values $\boldsymbol{s}^*$ which implicitly define constraints on the desired camera or robot pose with respect to the target. Three common approaches are employed. The first is when the task is directly specified as a desired value of some features to be reached. This is the case for instance when a target has to be centered in the image. This is also the case when the task is specified as a particular pose to reach and a PBVS is chosen. In that case however, the camera will reach its desired pose only if the camera is perfectly calibrated. Indeed, a coarse camera calibration will induce a biased estimation of the pose, which will make the final pose different from the desired one.

For IBVS and hybrid schemes, the second approach is to use knowledge of the object and camera projection models to compute $\boldsymbol{s}^*$ for the desired relative pose. Once again, the accuracy of the system directly depends on the camera calibration, since the camera intrinsic parameters are involved to compute $\boldsymbol{s}^*$.

Finally, the third approach is to simply record the feature values $\boldsymbol{s}^*$ when the camera or robot has the desired pose with respect to the target. This is usually done during an off-line teaching step. When it is possible in practice, this approach is very efficient since the positioning accuracy does not depend anymore on the camera calibration. This is still true for PBVS since, even if the pose estimated from the desired image is biased due to calibration errors, the same biased pose will be estimated once the robot will have converged so that the final image acquired by the camera will be the desired one.

So far we have not commented on the matching issues involved in visual servo methods. Two cases can be differentiated in function of the nature of the components of $\boldsymbol{s}$. When some components of $\boldsymbol{s}$ come from a pose estimation, it is necessary to match the measurements in the image (usually some image points) to the model of the object (usually some world points). Incorrect association will lead to an erroneous estimation of the pose. In all other cases, the calculation of the error vector $\boldsymbol{e}$ defined in (34.1) necessitates a matching between the measurements $\boldsymbol{m}(t)$ in the current image and $\boldsymbol{m}^*$ in the desired image. For instance, in all IBVS examples presented above, the camera observes four image points and we need to determine which of the four desired points to associate with each observed point so that the visual features $\boldsymbol{s}$ are correctly associated to their desired value $\boldsymbol{s}^*$. Incorrect association will lead to incorrect final camera pose and possibly a configuration which can not be achieved from any real camera pose. If we want to use the epipolar geometry or estimate a homography matrix, a similar matching process is necessary (Sect. 34.6).

This matching process is a classical computer vision problem. Note that it may be particularly difficult for the very first image, especially when the robot displacement to achieve is large, which generally implies large disparities between the initial and desired images. Once the association has been correctly performed for the very first image, the matching is greatly simplified since it transforms to a visual tracking problem where the results obtained for the previous image can be used as initialization for the current one.

34.8 Target Tracking

We now consider the case of a moving target and a constant desired value $\boldsymbol{s}^*$ for the features, the generalization to varying desired features $\boldsymbol{s}^*(t)$ being immediate. The time variation of the error is now given by

$$\dot{\boldsymbol{e}} = \mathbf{L}_e \boldsymbol{v}_c + \frac{\partial \boldsymbol{e}}{\partial t} , \tag{34.30}$$

where the term $\partial \boldsymbol{e}/\partial t$ expresses the time variation of $\boldsymbol{e}$ due to the generally unknown target motion. If the control law is still designed to try to ensure an exponential decoupled decrease of $\boldsymbol{e}$ (that is, once again $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$), we now obtain using (34.30)

$$\boldsymbol{v}_c = -\lambda \widehat{\mathbf{L}_e^+} \boldsymbol{e} - \widehat{\mathbf{L}_e^+} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} , \tag{34.31}$$

where $\widehat{\partial \boldsymbol{e}/\partial t}$ is an estimation or an approximation of $\partial \boldsymbol{e}/\partial t$. This term must be introduced into the control law to compensate for the target motion.

Closing the loop, that is, inserting (34.31) into (34.30), we obtain

$$\dot{\boldsymbol{e}} = -\lambda \mathbf{L}_e \widehat{\mathbf{L}_e^+} \boldsymbol{e} - \mathbf{L}_e \widehat{\mathbf{L}_e^+} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} + \frac{\partial \boldsymbol{e}}{\partial t} . \tag{34.32}$$

Even if $\mathbf{L}_e \widehat{\mathbf{L}_e^+} > 0$, the error will converge to zero only if the estimation of $\widehat{\partial \boldsymbol{e}/\partial t}$ is sufficiently accurate so that

$$\mathbf{L}_e \widehat{\mathbf{L}_e^+} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \frac{\partial \boldsymbol{e}}{\partial t} , \tag{34.33}$$

otherwise tracking errors will be observed. Indeed, by just solving the scalar differential equation $\dot{e} = -\lambda e + b$, which is a simplification of (34.32), we obtain $e(t) = e(0)\exp(-\lambda t) + b/\lambda$, which converges towards b/λ. On one hand, setting a high gain λ will reduce the tracking error, but on the other hand, setting the gain too high can make the system unstable. It is thus necessary to make b as small as possible.

Of course, if the system is known to be such that $\partial \boldsymbol{e}/\partial t = 0$ (that is, the camera observes a motionless object, as described in Sect. 34.1), no tracking error will appear with the most simple estimation given by $\widehat{\partial \boldsymbol{e}/\partial t} = 0$. Otherwise, a classical method in automatic control to cancel tracking errors consists of compensating the target motion through an integral term in the control law. In this case, wehave

$$\widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \mu \sum_j \boldsymbol{e}(j) \,,$$

where μ is the integral gain that has to be tuned. This scheme allows the tracking errors to be canceled only if the target has a constant velocity. Other methods, based on feedforward control, estimate the term $\widehat{\partial \boldsymbol{e}/\partial t}$ directly through the image measurements and the camera velocity, when it is available. Indeed, from (34.30), we obtain

$$\widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \widehat{\dot{\boldsymbol{e}}} - \widehat{\mathbf{L}_e}\,\widehat{\boldsymbol{v}}_c \,,$$

where $\widehat{\dot{\boldsymbol{e}}}$ can, for instance, be obtained as $\widehat{\dot{\boldsymbol{e}}}(t) = [\boldsymbol{e}(t) - \boldsymbol{e}(t-\Delta t)]/\Delta t$, Δt being the duration of the control loop. A Kalman filter [34.63] or more-elaborate filtering methods [34.64] can then be used to improve the estimated values obtained. If some knowledge about the target velocity or the target trajectory is available, it can of course be used to smooth or predict the motion [34.65–67]. For instance, in [34.68], the periodic motion of the heart and breathing are compensated for an application of visual servoing in medical robotics. Finally, other methods have been developed to remove the perturbations induced by the target motion as fast as possible [34.43], using for instance predictive controllers [34.69].

34.9 Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space

In the previous sections, we have considered the six components of the camera velocity as the input of the robot controller. As soon as the robot is not able to realize this motion, for instance, because it has fewer than six degrees of freedom, the control scheme must be expressed in the joint space. In this section, we describe how this can be done, and in the process develop a formulation for eye-to-hand systems.

In the joint space, the system equations for both the eye-to-hand and eye-in-hand configurations have the same form

$$\dot{\boldsymbol{s}} = \mathbf{J}_s\,\dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{s}}{\partial t} \,. \tag{34.34}$$

Here, $\mathbf{J}_s \in \mathbb{R}^{k\times n}$ is the feature Jacobian matrix, which can be linked to the interaction matrix, and n is the number of robot joints.

For an eye-in-hand system (Fig. 34.15a), $\partial \boldsymbol{s}/\partial t$ is the time variation of $\boldsymbol{s}$ due to a potential object motion, and $\mathbf{J}_s$ is given by

$$\mathbf{J}_s = \mathbf{L}_s\,{}^{c}\boldsymbol{X}_N \mathbf{J}(\boldsymbol{q}) \,, \tag{34.35}$$

where:

- ${}^{c}\boldsymbol{X}_N$ is the spatial motion transform matrix (as defined in Chap. 02 and recalled in (34.15)) from the vision sensor frame to the end-effector frame. It is usually a constant matrix (as long as the vision sensor is rigidly attached to the end-effector). Thanks to the robustness of closed-loop control schemes, a coarse approximation of this transform matrix is sufficient in visual servoing. If needed, an accurate estimation is possible through classical hand–eye calibration methods [34.70].
- $\mathbf{J}(\boldsymbol{q})$ is the robot Jacobian expressed in the end-effector frame (as defined in Chap. 02)

For an eye-to-hand system (Fig. 34.15b), $\partial \boldsymbol{s}/\partial t$ is now the time variation of $\boldsymbol{s}$ due to a potential vision sensor motion and $\mathbf{J}_s$ can be expressed as

$$\mathbf{J}_s = -\mathbf{L}_s\,{}^{c}\boldsymbol{X}_N\,{}^{N}\mathbf{J}(\boldsymbol{q}) \,, \tag{34.36}$$

$$= -\mathbf{L}_s\,{}^{c}\boldsymbol{X}_0\,{}^{0}\mathbf{J}(\boldsymbol{q}) \,. \tag{34.37}$$

In (34.36), the classical robot Jacobian ${}^{N}\mathbf{J}(\boldsymbol{q})$ expressed in the end-effector frame is used but the spatial motion transform matrix ${}^{c}\boldsymbol{X}_N$ from the vision sensor frame to the end-effector frame changes all along the robot mo-

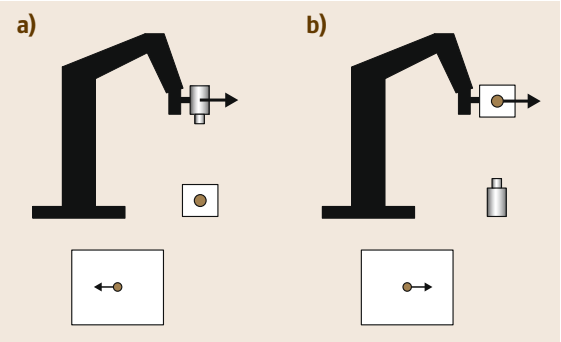


Fig.34.15a,b **(a)** Eye-in-hand system, **(b)** eye-to-hand system: system schematic (*top*) and opposite image motion produced by the same robot motion (*bottom*)

tion, and it has to be estimated at each iteration of the control scheme, usually using pose-estimation methods.

In (34.37), the robot Jacobian ${}^0\mathbf{J}(\boldsymbol{q})$ is expressed in the robot reference frame, and the spatial motion transform matrix ${}^c\boldsymbol{X}_0$ from the vision sensor frame to that reference frame is constant as long as the camera does not move. In this case, which is convenient in practice, a coarse approximation of ${}^c\boldsymbol{X}_0$ is usually sufficient.

Once the modeling step is finished, it is quite easy to follow the procedure that has been used above to design a control scheme expressed in the joint space, and to determine the sufficient condition to ensure the stability of the control scheme. We obtain, considering again $\boldsymbol{e} = \boldsymbol{s} - \boldsymbol{s}^*$, and an exponential decoupled decrease of $\boldsymbol{e}$

$$\dot{\boldsymbol{q}} = -\lambda \widehat{\mathbf{J}_e^+} \boldsymbol{e} - \widehat{\mathbf{J}_e^+} \frac{\widehat{\partial \boldsymbol{e}}}{\partial t} \,. \tag{34.38}$$

If $k = n$, considering as in Sect. 34.1 the Lyapunov function $\mathcal{L} = \frac{1}{2}\|\boldsymbol{e}(t)\|^2$, a sufficient condition to ensure the global asymptotic stability is given by

$$\mathbf{J}_e \widehat{\mathbf{J}_e^+} > 0 \,. \tag{34.39}$$

If $k > n$, we obtain similarly to Sect. 34.1

$$\widehat{\mathbf{J}_e^+} \mathbf{J}_e > 0 \tag{34.40}$$

to ensure the local asymptotic stability of the system. Note that the actual extrinsic camera parameters appear in $\mathbf{J}_e$ while the estimated ones are used in $\widehat{\mathbf{J}_e^+}$. It is thus possible to analyze the robustness of the control scheme with respect to the camera extrinsic parameters. It is also possible to estimate directly the numerical value of $\mathbf{J}_e$ or $\mathbf{J}_e^+$ using the methods described in Sect. 34.2.8.

Finally, to remove tracking errors, we have to ensure that

$$\mathbf{J}_e \widehat{\mathbf{J}_e^+} \frac{\widehat{\partial \boldsymbol{e}}}{\partial t} = \frac{\partial \boldsymbol{e}}{\partial t} \,.$$

Let us note that, even if the robot has six degrees of freedom, it is generally not equivalent to first compute $\boldsymbol{v}_c$ using (34.5) and then deduce $\dot{\boldsymbol{q}}$ using the robot inverse Jacobian, and to compute directly $\dot{\boldsymbol{q}}$ using (34.38). Indeed, it may occur that the robot Jacobian $\mathbf{J}(\boldsymbol{q})$ is singular while the feature Jacobian $\mathbf{J}_s$ is not (that may occur when $k < n$). Furthermore, the properties of the pseudo-inverse ensure that using (34.5), $\|\boldsymbol{v}_c\|$ is minimal while using (34.38), $\|\dot{\boldsymbol{q}}\|$ is minimal. As soon as

$$\mathbf{J}_e^+ \neq \mathbf{J}^+(\boldsymbol{q})^N \boldsymbol{X}_c \mathbf{L}_e^+ \,,$$

the control schemes will be different and will induce different robot trajectories. The choice of the state space is thus important.

34.10 Under Actuated Robots

Many useful robots are under actuated, that is, they cannot move instantaneously in all directions because of the number or configuration of their actuators (Chap. 52). For instance, a quadrotor flying robot is under actuated since it has only four actuators while the dimension of its configuration space is six. Many other useful robots are subject to non-holonomic constraints (Chap. 49), leading to similar motion inability. A car for instance has only two degrees of freedom (for velocity and steering) while the dimension of its configuration space is three (position and orientation in the ground plane).

A consequence of under actuation is that time varying manoeuvres may be required in order to achieve particular goal states. For example, if a quadrotor has to move forward, it must first change its attitude, pitching down, so that a component of its thrust vector is able to accelerate the vehicle forward. For a visual servo system this can be problematic since the initial attitude change will affect the value of the visual features even before the vehicle has moved. In fact the attitude change increases the error and this is ultimately destabilising.

A common and expedient solution [34.71] is to *de-rotate* the image, that is, to use information from

a non-vision attitude sensor such as an inertial measurement unit (IMU) (Chap. 29) to correct the feature coordinates as if they had been viewed by a virtual camera whose optical axis has a constant orientation in space, typically, straight down. As discussed in Sect. 34.2.8, the spherical projection model is well suited for such image transformation, thanks to its invariance properties with respect to rotational motion.

For a non-holonomic vehicle, there are several solutions. In the non-general case where the vehicle is able to follow a smooth path from its initial to goal configuration, a controller based on visually estimated relative pose (range, heading angle and lateral offset) can be used (Chap. 49), following a PBVS strategy. Particular controllers based on the epipolar geometry or the trifocal tensor have also been designed [34.72, 73]. For the more general case, switching control laws can be used. If possible in practice, another common solution is to add a controlled DOF between the vehicle and the camera in order to bypass the non-holonomic constraint through redundancy. It is thus possible to control the full camera configuration, but not the robot one.

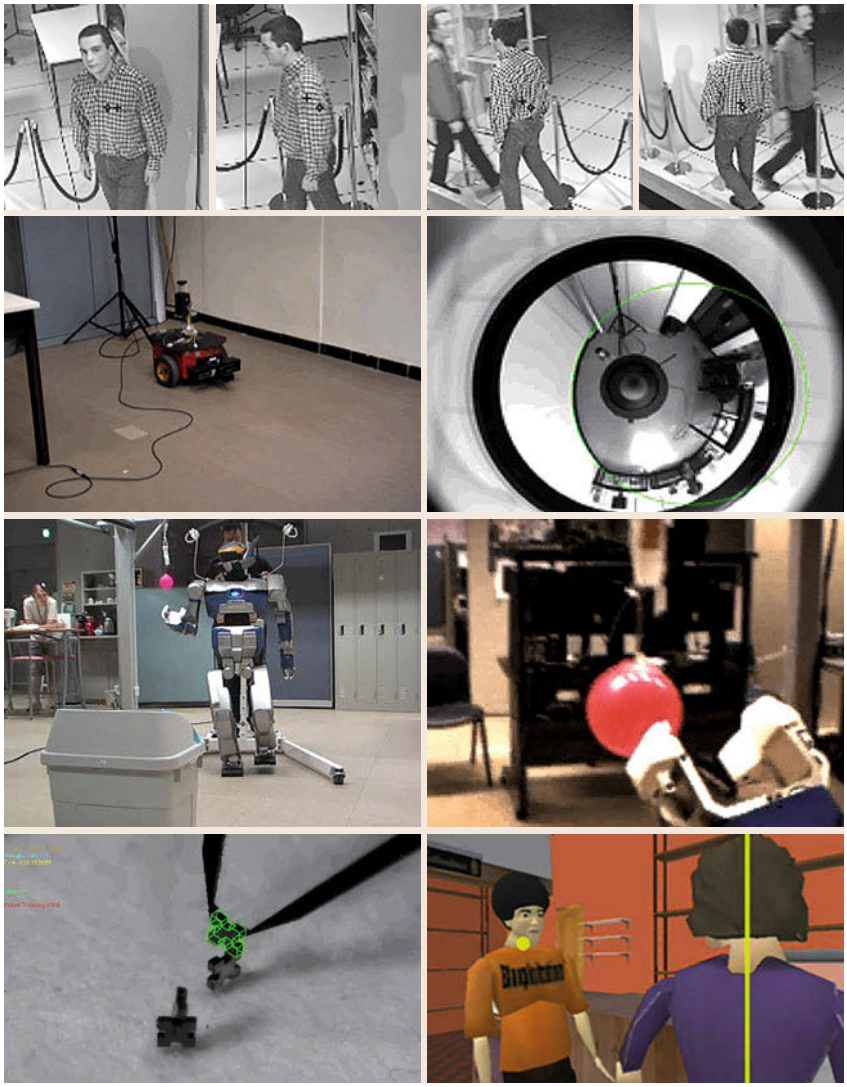


Fig. 34.16 Few applications of visual servoing: Gaze control for target tracking, navigation of a mobile robot to follow a wall using an omnidirectional vision sensor, grasping a ball with a humanoid robot, assembly of MEMS and film of a dialogue within the constraints of a script in animation

34.11 Applications

Applications of visual servoing in robotics are numerous. It can be used as soon as a vision sensor is available and a task is assigned to a dynamic system to control its motion. A non exhaustive list of examples are (Fig. 34.16):

- The control of a pan-tilt-zoom camera for target tracking
- Grasping using a robot arm
- Locomotion and dextrous manipulation with a humanoid robot
- Micro or nano manipulation of microelectromechanical system (MEMS) or biological cells
- Accurate positioning of a parallel robot
- Pipe inspection by an underwater autonomous vehicle
- Autonomous navigation of a mobile robot in indoor or outdoor environment
- Aircraft landing
- Autonomous satellite rendezvous
- Biopsy using ultrasound probes or heart motion compensation in medical robotics
- Virtual cinematography in animation.

Software are available to help any user interested in developing visual servoing applications. We recommend the Robotics, Machine Vision Toolbox for Matlab [34.74] and the ViSP C++ library [34.11].

34.12 Conclusions

In this chapter we have only considered velocity controllers, which are convenient for most classical robot arms. However, the dynamics of the robot must of course be taken into account for high-speed tasks. Features related to the image motion [34.75], image intensity [34.76], or coming from other vision sensors (RGB-D sensors, ultrasonic probes [34.77], etc.) necessitate reconsideration of the modeling issues to select adequate visual features. Finally, fusing visual features with data coming from other sensors (force sensor, proximity sensors, etc.) at the level of the control scheme will allow new research topics to be addressed. The end of fruitful research in the field of visual servo is thus nowhere yet in sight.

Video-References

VIDEO 59 IBVS on a 6 DOF robot arm (1);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/59

VIDEO 60 IBVS on a 6 DOF robot arm (2);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/60

VIDEO 61 IBVS on a 6 DOF robot arm (3);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/61

VIDEO 62 PBVS on a 6 DOF robot arm (1);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/62

VIDEO 63 PBVS on a 6 DOF robot arm (2);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/63

VIDEO 64 2.5-D VS on a 6 DOF robot arm (1);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/64

VIDEO 65 2.5-D VS on a 6 DOF robot arm (2);
available from http://handbookofrobotics.org/view-chapter/34/videodetails/65

References

34.1 L. Weiss, A. Sanderson, C. Neuman: Dynamic sensor-based control of robots with visual feedback, IEEE J. Robot. Autom. **3**, 404–417 (1987)

34.2 S. Hutchinson, G. Hager, P. Corke: A tutorial on visual servo control, IEEE Trans. Robot. Autom. **12**, 651–670 (1996)

34.3 B. Espiau, F. Chaumette, P. Rives: A new approach to visual servoing in robotics, IEEE Trans. Robot. Autom. **8**, 313–326 (1992)

34.4 J. Feddema, O. Mitchell: Vision-guided servoing with feature-based trajectory generation, IEEE Trans. Robot. Autom. **5**, 691–700 (1989)

34.5 D. Forsyth, J. Ponce: *Computer Vision: A Modern Approach* (Prentice Hall, Upper Saddle River 2003)
34.6 Y. Ma, S. Soatto, J. Kosecka, S. Sastry: *An Invitation to 3-D Vision: From Images to Geometric Models* (Springer, New York 2003)
34.7 P. Corke: *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* (Springer, Berlin, Heidelberg 2011)
34.8 H. Michel, P. Rives: Singularities in the Determination of the Situation of a Robot Effector from the Perspective View of Three Points. Res. Rep. RR-1850 (INRIA 1993)
34.9 M. Fischler, R. Bolles: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Communications ACM **24**, 381–395 (1981)
34.10 E. Malis: Improving vision-based control using efficient second-order minimization techniques, IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp. 1843–1848
34.11 E. Marchand, F. Spindler, F. Chaumette: ViSP for visual servoing: A generic software platform with a wide class of robot control skills, IEEE Robot. Autom. Mag. **12**(4), 40–52 (2005), https://team.inria.fr/lagadic/visp/visp.html
34.12 P. Corke, S. Hutchinson: A new partitioned approach to image-based visual servo control, IEEE Trans. Robot. Autom. **17**, 507–515 (2001)
34.13 F. Chaumette: Potential problems of stability and convergence in image-based and position-based visual servoing, Lect. Note. Contr. Inform. Sci. **237**, 66–78 (1998)
34.14 E. Malis: Visual servoing invariant to changes in camera intrinsic parameters, IEEE Trans. Robot. Autom. **20**, 72–81 (2004)
34.15 A. Isidori: *Nonlinear Control Systems*, 3rd edn. (Springer, Berlin, Heidelberg 1995)
34.16 G. Hager, W. Chang, A. Morse: Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination, IEEE Control Syst. Mag. **15**, 30–39 (1995)
34.17 M. Iwatsuki, N. Okiyama: A new formulation of visual servoing based on cylindrical coordinate system, IEEE Trans. Robot. Autom. **21**, 266–273 (2005)
34.18 F. Chaumette, P. Rives, B. Espiau: Classification and realization of the different vision-based tasks, Robot. Autom. Syst. **7**, 199–228 (1993)
34.19 A. Castano, S. Hutchinson: Visual compliance: Task directed visual servo control, IEEE Trans. Robot. Autom. **10**, 334–342 (1994)
34.20 G. Hager: A modular system for robust positioning using feedback from stereo vision, IEEE Trans. Robot. Autom. **13**, 582–595 (1997)
34.21 F. Chaumette: Image moments: A general and useful set of features for visual servoing, IEEE Trans. Robot. Autom. **20**, 713–723 (2004)
34.22 O. Tahri, F. Chaumette: Point-based and region-based image moments for visual servoing of planar objects, IEEE Trans. Robot. **21**, 1116–1127 (2005)
34.23 C. Geyer, K. Daniilidis: Catadioptric projective geometry, Int. J. Comput. Vis. **45**(3), 223–243 (2001)
34.24 T. Hamel, R. Mahony: Visual servoing of an under-actuated dynamic rigid-body system: An image-based approach, IEEE Trans Robot. **18**(2), 187–198 (2002)
34.25 I. Suh: Visual servoing of robot manipulators by fuzzy membership function based neural networks. In: *Visual Servoing*, Robotics and Automated Systems, Vol. 7, ed. by K. Hashimoto (World Scientific, Singapore 1993) pp. 285–315
34.26 G. Wells, C. Venaille, C. Torras: Vision-based robot positioning using neural networks, Image Vis. Comput. **14**, 75–732 (1996)
34.27 J.T. Lapresté, F. Jurie, F. Chaumette: An efficient method to compute the inverse jacobian matrix in visual servoing, IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp. 727–732
34.28 K. Hosada, M. Asada: Versatile visual servoing without knowledge of true jacobian, IEEE/RSJ Int. Conf. Intell. Robots Syst., München (1994) pp. 186–193
34.29 M. Jägersand, O. Fuentes, R. Nelson: Experimental evaluation of uncalibrated visual servoing for precision manipulation, IEEE Int. Conf. Robot. Autom., Albuquerque (1997) pp. 2874–2880
34.30 J. Piepmeier, G.M. Murray, H. Lipkin: Uncalibrated dynamic visual servoing, IEEE Trans. Robot. Autom. **20**, 143–147 (2004)
34.31 K. Deguchi: Direct interpretation of dynamic images and camera motion for visual servoing without image feature correspondence, J. Robot. Mechatron. **9**(2), 104–110 (1997)
34.32 W. Wilson, C. Hulls, G. Bell: Relative end-effector control using cartesian position based visual servoing, IEEE Trans. Robot. Autom. **12**, 684–696 (1996)
34.33 B. Thuilot, P. Martinet, L. Cordesses, J. Gallice: Position based visual servoing: Keeping the object in the field of vision, IEEE Int. Conf. Robot. Autom., Washington (2002) pp. 1624–1629
34.34 D. Dementhon, L. Davis: Model-based object pose in 25 lines of code, Int. J. Comput. Vis. **15**, 123–141 (1995)
34.35 D. Lowe: Three-dimensional object recognition from single two-dimensional images, Artif. Intell. **31**(3), 355–395 (1987)
34.36 E. Malis, F. Chaumette, S. Boudet: 2-1/2 D visual servoing, IEEE Trans. Robot. Autom. **15**, 238–250 (1999)
34.37 E. Malis, F. Chaumette: Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods, IEEE Trans. Robot. Autom. **18**, 176–186 (2002)
34.38 J. Chen, D. Dawson, W. Dixon, A. Behal: Adaptive homography-based visual servo tracking for fixed camera-in-hand configurations, IEEE Trans. Control Syst. Technol. **13**, 814–825 (2005)
34.39 G. Morel, T. Leibezeit, J. Szewczyk, S. Boudet, J. Pot: Explicit incorporation of 2-D constraints in vision-based control of robot manipulators, Lect. Note. Contr. Inform. Sci. **250**, 99–108 (2000)

34.40 F. Chaumette, E. Malis: 2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings, IEEE Int. Conf. Robot. Autom., San Fransisco (2000) pp. 630–635

34.41 E. Cervera, A.D. Pobil, F. Berry, P. Martinet: Improving image-based visual servoing with three-dimensional features, Int. J. Robot. Res. **22**, 821–840 (2004)

34.42 F. Schramm, G. Morel, A. Micaelli, A. Lottin: Extended 2-D visual servoing, IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp. 267–273

34.43 N. Papanikolopoulos, P. Khosla, T. Kanade: Visual tracking of a moving target by a camera mounted on a robot: A combination of vision and control, IEEE Trans. Robot. Autom. **9**, 14–35 (1993)

34.44 K. Hashimoto, H. Kimura: LQ optimal and nonlinear approaches to visual servoing, Robot. Autom. Syst. **7**, 165–198 (1993)

34.45 B. Nelson, P. Khosla: Strategies for increasing the tracking region of an eye-in-hand system by singularity and joint limit avoidance, Int. J. Robot. Res. **14**, 225–269 (1995)

34.46 B. Nelson, P. Khosla: Force and vision resolvability for assimilating disparate sensory feedback, IEEE Trans. Robot. Autom. **12**, 714–731 (1996)

34.47 R. Sharma, S. Hutchinson: Motion perceptibility and its application to active vision-based servo control, IEEE Trans. Robot. Autom. **13**, 607–617 (1997)

34.48 E. Marchand, F. Chaumette, A. Rizzo: Using the task function approach to avoid robot joint limits and kinematic singularities in visual servoing, IEEE/RSJ Int. Conf. Intell. Robots Syst., Osaka (1996) pp. 1083–1090

34.49 E. Marchand, G. Hager: Dynamic sensor planning in visual servoing, IEEE Int. Conf. Robot. Autom., Leuven (1998) pp. 1988–1993

34.50 N. Cowan, J. Weingarten, D. Koditschek: Visual servoing via navigation functions, IEEE Trans. Robot. Autom. **18**, 521–533 (2002)

34.51 N. Gans, S. Hutchinson: An asymptotically stable switched system visual controller for eye in hand robots, IEEE/RSJ Int. Conf. Intell. Robots Syst., Las Vegas (2003) pp. 735–742

34.52 G. Chesi, K. Hashimoto, D. Prattichizio, A. Vicino: Keeping features in the field of view in eye-in-hand visual servoing: a switching approach, IEEE Trans. Robot. Autom. **20**, 908–913 (2004)

34.53 K. Hosoda, K. Sakamato, M. Asada: Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3-D reconstruction, IEEE/RSJ Int. Conf. Intell. Robots Syst. 3, Pittsburgh (1995) pp. 29–34

34.54 Y. Mezouar, F. Chaumette: Path planning for robust image-based control, IEEE Trans. Robot. Autom. **18**, 534–549 (2002)

34.55 G. Chesi: Visual servoing path-planning via homogeneous forms and LMI optimizations, IEEE Trans. Robot. **25**(2), 281–291 (2009)

34.56 L. Matthies, T. Kanade, R. Szeliski: Kalman filter-based algorithms for estimating depth from image sequences, Int. J. Comput. Vis. **3**(3), 209–238 (1989)

34.57 C.E. Smith, N. Papanikolopoulos: Computation of shape through controlled active exploration, IEEE Int. Conf. Robot. Autom., San Diego (1994) pp. 2516–2521

34.58 A. De Luca, G. Oriolo, P. Robuffo Giordano: Feature depth observation for image-based visual servoing: Theory and experiments, Int. J. Robot. Res. **27**(10), 1093–1116 (2008)

34.59 R. Basri, E. Rivlin, I. Shimshoni: Visual homing: Surfing on the epipoles, Int. J. Comput. Vis. **33**, 117–137 (1999)

34.60 E. Malis, F. Chaumette, S. Boudet: 2 1/2 D visual servoing with respect to unknown objects through a new estimation scheme of camera displacement, Int. J. Comput. Vis. **37**, 79–97 (2000)

34.61 O. Faugeras: *Three-Dimensional Computer Vision: A Geometric Viewpoint* (MIT Press, Cambridge 1993)

34.62 G. Silveira, E. Malis: Direct visual servoing: Vision-based estimation and control using only nonmetric information, IEEE Trans. Robot. **28**(4), 974–980 (2012)

34.63 P. Corke, M. Goods: Controller design for high performance visual servoing, 12th World Congr. IFAC'93, Sydney (1993) pp. 395–398

34.64 F. Bensalah, F. Chaumette: Compensation of abrupt motion changes in target tracking by visual servoing, IEEE/RSJ Int. Conf. Intell. Robots Syst., Pittsburgh (1995) pp. 181–187

34.65 P. Allen, B. Yoshimi, A. Timcenko, P. Michelman: Automated tracking and grasping of a moving object with a robotic hand-eye system, IEEE Trans. Robot. Autom. **9**, 152–165 (1993)

34.66 K. Hashimoto, H. Kimura: Visual servoing with non linear observer, IEEE Int. Conf. Robot. Autom., Nagoya (1995) pp. 484–489

34.67 A. Rizzi, D. Koditschek: An active visual estimator for dexterous manipulation, IEEE Trans. Robot. Autom. **12**, 697–713 (1996)

34.68 R. Ginhoux, J. Gangloff, M. de Mathelin, L. Soler, M.A. Sanchez, J. Marescaux: Active filtering of physiological motion in robotized surgery using predictive control, IEEE Trans. Robot. **21**, 67–79 (2005)

34.69 J. Gangloff, M. de Mathelin: Visual servoing of a 6-DOF manipulator for unknown 3-D profile following, IEEE Trans. Robot. Autom. **18**, 511–520 (2002)

34.70 R. Tsai, R. Lenz: A new technique for fully autonomous efficient 3-D robotics hand-eye calibration, IEEE Trans. Robot. Autom. **5**, 345–358 (1989)

34.71 N. Guenard, T. Hamel, R. Mahony: A practical visual servo control for an unmanned aerial vehicle, IEEE Trans. Robot. **24**(2), 331–340 (2008)

34.72 G.L. Mariottini, G. Oriolo, D. Prattichizo: Image-based visual servoing for nonholonomic mobile robots using epipolar geometry, IEEE Trans. Robot. **23**(1), 87–100 (2007)

34.73 G. Lopez-Nicolas, J.J. Guerrero, C. Sagues: Visual control through the trifocal tensor for nonholonomic robots, Robot. Auton. Syst. **58**(2), 216–226 (2010)

34.74 P. Corke: *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, Springer Tracts in Advanced Robotics, Vol. 73 (Springer, Berlin, Heidelberg 2011)
34.75 A. Crétual, F. Chaumette: Visual servoing based on image motion, Int. J. Robot. Res. **20**(11), 857–877 (2001)
34.76 C. Collewet, E. Marchand: Photometric visual servoing, IEEE Trans. Robot. **27**(4), 828–834 (2011)
34.77 R. Mebarki, A. Krupa, F. Chaumette: 2D ultrasound probe complete guidance by visual servoing using image moments, IEEE Trans. Robot. **26**(2), 296–306 (2010)

Multimedia Contents

35. Multisensor Data Fusion

Hugh Durrant-Whyte, Thomas C. Henderson

Multisensor data fusion is the process of combining observations from a number of different sensors to provide a robust and complete description of an environment or process of interest. Data fusion finds wide application in many areas of robotics such as object recognition, environment mapping, and localization.

This chapter has three parts: methods, architectures, and applications. Most current data fusion methods employ probabilistic descriptions of observations and processes and use Bayes' rule to combine this information. This chapter surveys the main probabilistic modeling and fusion techniques including grid-based models, Kalman filtering, and sequential Monte Carlo techniques. This chapter also briefly reviews a number of nonprobabilistic data fusion methods. Data fusion systems are often complex combinations of sensor devices, processing, and fusion algorithms. This chapter provides an overview of key principles in data fusion architectures from both a hardware and algorithmic viewpoint. The applications of data fusion are pervasive in robotics and underly the core problem of sensing, estimation, and perception. We highlight two example applications that bring out these features. The first describes a navigation or self-tracking application for an autonomous vehicle. The second describes an application in mapping and environment modeling.

35.1 **Multisensor Data Fusion Methods** 867
35.1.1 Bayes' Rule 868
35.1.2 Probabilistic Grids 869
35.1.3 The Kalman Filter 872
35.1.4 Sequential Monte Carlo Methods 876
35.1.5 Alternatives to Probability 878
35.2 **Multisensor Fusion Architectures** 880
35.2.1 Architectural Taxonomy 880
35.2.2 Centralized, Local Interaction, and Hierarchical 882
35.2.3 Decentralized, Global Interaction, and Heterarchical 882
35.2.4 Decentralized, Local Interaction, and Hierarchical 883
35.2.5 Decentralized, Local Interaction, and Heterarchical 884
35.3 **Applications** 885
35.3.1 Dynamic System Control 885
35.3.2 ANSER II: Decentralized Data Fusion 888
35.3.3 Recent Developments 889
35.4 **Conclusions** 889
Video-References 889
References 890

The essential algorithmic tools of data fusion are reasonably well established. However, the development and use of these tools in realistic robotics applications is still developing.

35.1 Multisensor Data Fusion Methods

The most widely used data fusion methods employed in robotics originate in the fields of statistics, estimation, and control. However, the application of these methods in robotics has a number of unique features and challenges. In particular, autonomy is most often the goal and so results must be presented and interpreted in a form from which autonomous decisions can be made, for recognition or navigation, for example.

In this section we review the main data fusion methods employed in robotics. These are very often based on probabilistic methods, which are indeed now considered the standard approach to data fusion in all robotics applications [35.1]. Probabilistic data fusion methods are generally based on Bayes' rule for combining prior and observation information. Practically, this may be implemented in a number of ways: through the use of the Kalman and extended Kalman filters, through sequential Monte Carlo methods, or through the use of functional density estimates. Each of these is reviewed. There are a number of alternatives to probabilistic methods. These include the theory of evidence and interval methods. Such alternative techniques are not as widely used as they once were, however they have some special features that can be advantageous in specific problems. These, too, are briefly reviewed.

35.1.1 Bayes' Rule

Bayes' rule lies at the heart of most data fusion methods. In general, Bayes' rule provides a means to make inferences about an object or environment of interest described by a state $\boldsymbol{x}$, given an observation z.

Bayesian Inference

Bayes' rule requires that the relationship between $\boldsymbol{x}$ and z be encoded as a joint probability or joint probability distribution $P(\boldsymbol{x}, z)$ for discrete and continuous variables, respectively. The chain rule of conditional probabilities can be used to expand a joint probability in two ways

$$P(\boldsymbol{x}, z) = P(\boldsymbol{x} \mid z)P(z) = P(z \mid \boldsymbol{x})P(\boldsymbol{x}). \tag{35.1}$$

Rearranging in terms of one of the conditionals, Bayes' rule is obtained

$$P(\boldsymbol{x} \mid z) = \frac{P(z \mid \boldsymbol{x})P(\boldsymbol{x})}{P(z)}. \tag{35.2}$$

The value of this result lies in the interpretation of the probabilities $P(\boldsymbol{x} \mid z)$, $P(z \mid \boldsymbol{x})$, and $P(\boldsymbol{x})$. Suppose it is necessary to determine the various likelihoods of different values of an unknown state $\boldsymbol{x}$. There may be prior beliefs about what values of $\boldsymbol{x}$ might be expected, encoded in the form of relative likelihoods in the *prior probability* $P(\boldsymbol{x})$. To obtain more information about the state $\boldsymbol{x}$ an observation z is made. These observations are modeled in the form of a conditional probability $P(z \mid \boldsymbol{x})$ that describes, for each fixed state $\boldsymbol{x}$, the probability that the observation z will be made, i. e., the probability of z given $\boldsymbol{x}$. The new likelihoods associated with the state $\boldsymbol{x}$ are computed from the product of the original prior information and the information gained by observation. This is encoded in the *posterior probability* $P(\boldsymbol{x} \mid z)$, which describes the likelihoods associated with $\boldsymbol{x}$ given the observation z. In this fusion process, the marginal probability $P(z)$ simply serves to normalize the posterior and is not generally computed. The marginal $P(z)$ plays an important role in model validation or data association as it provides a measure of how well the observation is predicted by the prior, because $P(z) = \int P(z \mid \boldsymbol{x})P(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$. The value of Bayes' rule is that it provides a principled means of combining observed information with prior beliefs about the state of the world.

Sensor Models and Multisensor Bayesian Inference

The conditional probability $P(z \mid \boldsymbol{x})$ serves the role of a *sensor model* and can be thought of in two ways. First, in building a sensor model, the probability is constructed by fixing the value of $\boldsymbol{x} = x$ and then asking what probability density $P(z \mid \boldsymbol{x} = x)$ on z results. Conversely, when this sensor model is used and observations are made, $z = z$ is fixed and a *likelihood function* $P(z = z \mid \boldsymbol{x})$ on $\boldsymbol{x}$ is inferred. The likelihood function, while not strictly a probability density, models the relative likelihood that different values of $\boldsymbol{x}$ gave rise to the observed value of z. The product of this likelihood with the prior, both defined on $\boldsymbol{x}$, gives the posterior or observation update $P(\boldsymbol{x} \mid z)$. In a practical implementation of (35.2), $P(z \mid \boldsymbol{x})$ is constructed as a function of both variables (or a matrix in discrete form). For each fixed value of $\boldsymbol{x}$, a probability density on z is defined. Therefore, as $\boldsymbol{x}$ varies, a family of likelihoods on z is created.

The multisensor form of Bayes' rule requires conditional independence

$$\begin{aligned} P(z_1, \cdots, z_n \mid \boldsymbol{x}) &= P(z_1 \mid \boldsymbol{x}) \cdots P(z_n \mid \boldsymbol{x}) \\ &= \prod_{i=1}^{n} P(z_i \mid \boldsymbol{x})\,, \end{aligned} \tag{35.3}$$

so that

$$P(\boldsymbol{x} \mid \boldsymbol{Z}^n) = CP(\boldsymbol{x}) \prod_{i=1}^{n} P(z_i \mid \boldsymbol{x})\,, \tag{35.4}$$

where C is a normalizing constant. Equation 35.4 is known as the *independent likelihood pool* [35.2]. This states that the posterior probability on $\boldsymbol{x}$ given all observations $\boldsymbol{Z}^n$, is simply proportional to the product of prior probability and individual likelihoods from each information source.

The recursive form of Bayes' rule is

$$P(\boldsymbol{x} \mid \boldsymbol{Z}^k) = \frac{P(z_k \mid \boldsymbol{x}) P(\boldsymbol{x} \mid \boldsymbol{Z}^{k-1})}{P\left(z_k \mid \boldsymbol{Z}^{k-1}\right)} . \tag{35.5}$$

The advantage of (35.5) is that we need to compute and store only the posterior density $P(\boldsymbol{x} \mid \boldsymbol{Z}^{k-1})$, which contains a complete summary of all past information. When the next piece of information $P(z_k \mid \boldsymbol{x})$ arrives, the previous posterior takes on the role of the current prior and the product of the two becomes, when normalized, the new posterior.

Bayesian Filtering

Filtering is concerned with the sequential process of maintaining a probabilistic model for a state which evolves over time and which is periodically observed by a sensor. Filtering forms the basis for many problems in tracking and navigation. The general filtering problem can be formulated in Bayesian form. This is significant because it provides a common representation for a range of discrete and continuous data fusion problems without recourse to specific target or observation models.

Define $\boldsymbol{x}_t$ as the value of a state of interest at time t. This may, for example, describe a feature to be tracked, the state of a process being monitored, or the location of a platform for which navigation data is required. For convenience, and without loss of generality, time is defined at discrete (asynchronous) times $t_k \triangleq k$. At a time instant k, the following quantities are defined:

- $\boldsymbol{x}_k$: The state vector to be estimated at time k.
- $\boldsymbol{u}_k$: A control vector, assumed known, and applied at time $k-1$ to drive the state from $\boldsymbol{x}_{k-1}$ to $\boldsymbol{x}_k$ at time k.
- z_k: An observation taken of the state $\boldsymbol{x}_k$ at time k.

In addition, the following sets are also defined:

- The history of states: $\boldsymbol{X}^k = \{\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_k\} = \{\boldsymbol{X}^{k-1}, \boldsymbol{x}_k\}$.
- The history of control inputs: $\boldsymbol{U}^k = \{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_k\} = \{\boldsymbol{U}^{k-1}, \boldsymbol{u}_k\}$.
- The history of state observations: $\boldsymbol{Z}^k = \{z_1, z_2, \cdots, z_k\} = \{\boldsymbol{Z}^{k-1}, z_k\}$.

In probabilistic form, the general data fusion problem is to find the posterior density

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^k, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \tag{35.6}$$

for all times k given the recorded observations and control inputs up to and including time k together (possibly) with knowledge of the initial state $\boldsymbol{x}_0$. Bayes' rule can be used to write (35.6) in terms of a sensor model $P(z_k \mid \boldsymbol{x}_k)$ and a predicted probability density $P(\boldsymbol{x}_k \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0)$ based on observations up to time $k-1$ as

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^k, \boldsymbol{U}^k, \boldsymbol{x}_0\right) = \frac{P(z_k \mid \boldsymbol{x}_k) P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right)}{P\left(z_k \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k\right)} . \tag{35.7}$$

The denominator in (35.7) is independent of the state and following (35.4) can be set to some normalizing constant C. The sensor model makes use of the conditional independence assumption from (35.3).

The total probability theorem can be used to rewrite the second term in the numerator of (35.7) in terms of the state transition model and the joint posterior from time step $k-1$ as

$$\begin{aligned} & P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \\ &= \int P\left(\boldsymbol{x}_k, \boldsymbol{x}_{k-1} \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \mathrm{d}\boldsymbol{x}_{k-1} \\ &= \int P\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \\ &\quad \times P\left(\boldsymbol{x}_{k-1} \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^k, \boldsymbol{x}_0\right) \mathrm{d}\boldsymbol{x}_{k-1} \\ &= \int P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_k) \\ &\quad \times P\left(\boldsymbol{x}_{k-1} \mid \boldsymbol{Z}^{k-1}, \boldsymbol{U}^{k-1}, \boldsymbol{x}_0\right) \mathrm{d}\boldsymbol{x}_{k-1} , \end{aligned} \tag{35.8}$$

where the last equality implies that the future state depends only on the current state and the control exerted at this time. The state transition model is described in terms of a probability distribution in the form $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_k)$. That is, the state transition may reasonably be assumed to be a Markov process in which the next state $\boldsymbol{x}_k$ depends only on the immediately proceeding state $\boldsymbol{x}_{k-1}$ and the applied control $\boldsymbol{u}_k$, and is independent of both the observations and preceding states.

Equations (35.7) and (35.8) define a recursive solution to (35.6). Equation (35.8) is the *time update* or prediction step for the full Bayes data fusion algorithm. A graphical description of this equation is shown in Fig. 35.1. Equation (35.7) is the *observation update* step for the full Bayes data fusion algorithm. A graphical description of this equation is shown in Fig. 35.2. The Kalman filter, grid-based methods, and sequential Monte Carlo methods, to be described, are specific implementations of these general equations.

35.1.2 Probabilistic Grids

Probabilistic grids are conceptually the simplest way of implementing Bayesian data fusion methods. They can

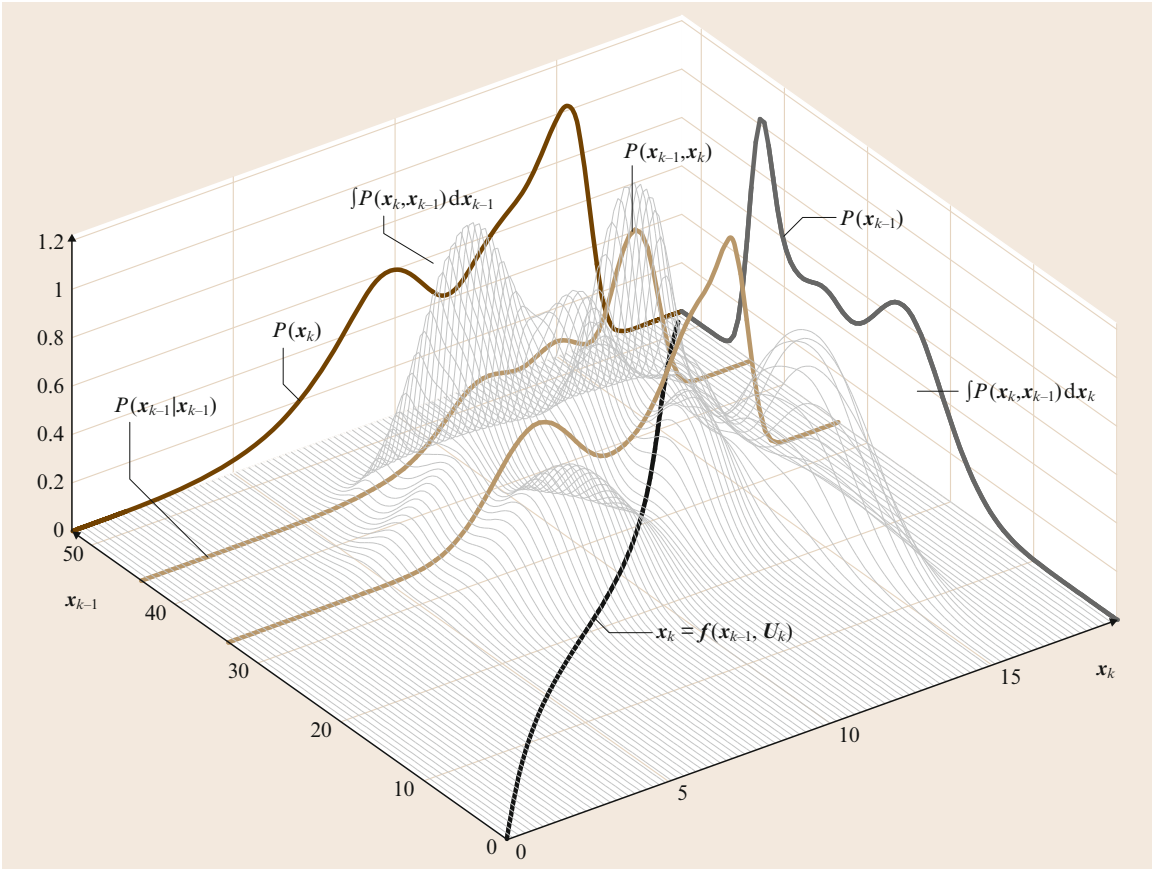


Fig. 35.1 Time update step for the full Bayes filter. At a time $k-1$, knowledge of the state $\boldsymbol{x}_{k-1}$ is summarized in a probability distribution $P(\boldsymbol{x}_{k-1})$. A vehicle model, in the form of a conditional probability density $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$, then describes the stochastic transition of the vehicle from a state $\boldsymbol{x}_{k-1}$ at a time $k-1$ to a state $\boldsymbol{x}_k$ at a time k. Functionally, this state transition may be related to an underlying kinematic state model in the form $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k)$. The figure shows two typical conditional probability distributions $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$ on the state $\boldsymbol{x}_k$ given fixed values of $\boldsymbol{x}_{k-1}$. The product of this conditional distribution with the marginal distribution $P(\boldsymbol{x}_{k-1})$, describing the prior likelihood of values of $\boldsymbol{x}_k$, gives the the joint distribution $P(\boldsymbol{x}_k, \boldsymbol{x}_{k-1})$ shown as the surface in the figure. The total marginal density $P(\boldsymbol{x}_k)$ describes knowledge of $\boldsymbol{x}_k$ after state transition has occurred. The marginal density $P(\boldsymbol{x}_k)$ is obtained by integrating (projecting) the joint distribution $P(\boldsymbol{x}_k, \boldsymbol{x}_{k-1})$ over all $\boldsymbol{x}_{k-1}$. Equivalently, using the total probability theorem, the marginal density can be obtained by integrating (summing) all conditional densities $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$ weighted by the prior probability $P(\boldsymbol{x}_{k-1})$ of each $\boldsymbol{x}_{k-1}$. The process can equally be run in reverse (a retroverse motion model) to obtain $P(\boldsymbol{x}_{k-1})$ from $P(\boldsymbol{x}_k)$ given a model $P(\boldsymbol{x}_{k-1} \mid \boldsymbol{x}_k)$

be applied both to problems in mapping [35.3, 4] and tracking [35.5].

In mapping applications, the environment of interest is divided into a grid of equally sized spatial cells. Each cell is indexed and labeled with a property, thus the state $\boldsymbol{x}_{ij}$ may describe a two-dimensional world indexed by ij and having the property x. Interest is focused on maintaining a probability distribution on possible state values $P(\boldsymbol{x}_{ij})$ *at each grid cell*. Typically, in navigation and mapping problems, the property of interest has only two values O and E, *occupied* and *empty*, respectively, and it is then usual to assume that $P(\boldsymbol{x}_{ij} = O) = 1 - P(\boldsymbol{x}_{ij} = E)$. However, there is no particular constraint on the property encoded by the state $\boldsymbol{x}_{ij}$ which could have many values (green, red, blue, for example) and indeed be continuous (the temperature at a cell, for example).

Once the state has been defined, Bayesian methods require that a sensor model or likelihood function for the sensor be established. In theory, this requires specification of a probability distribution $P(\boldsymbol{z} \mid \boldsymbol{x}_{ij} = x_{ij})$ mapping each possible grid state x_{ij} to a distribution on

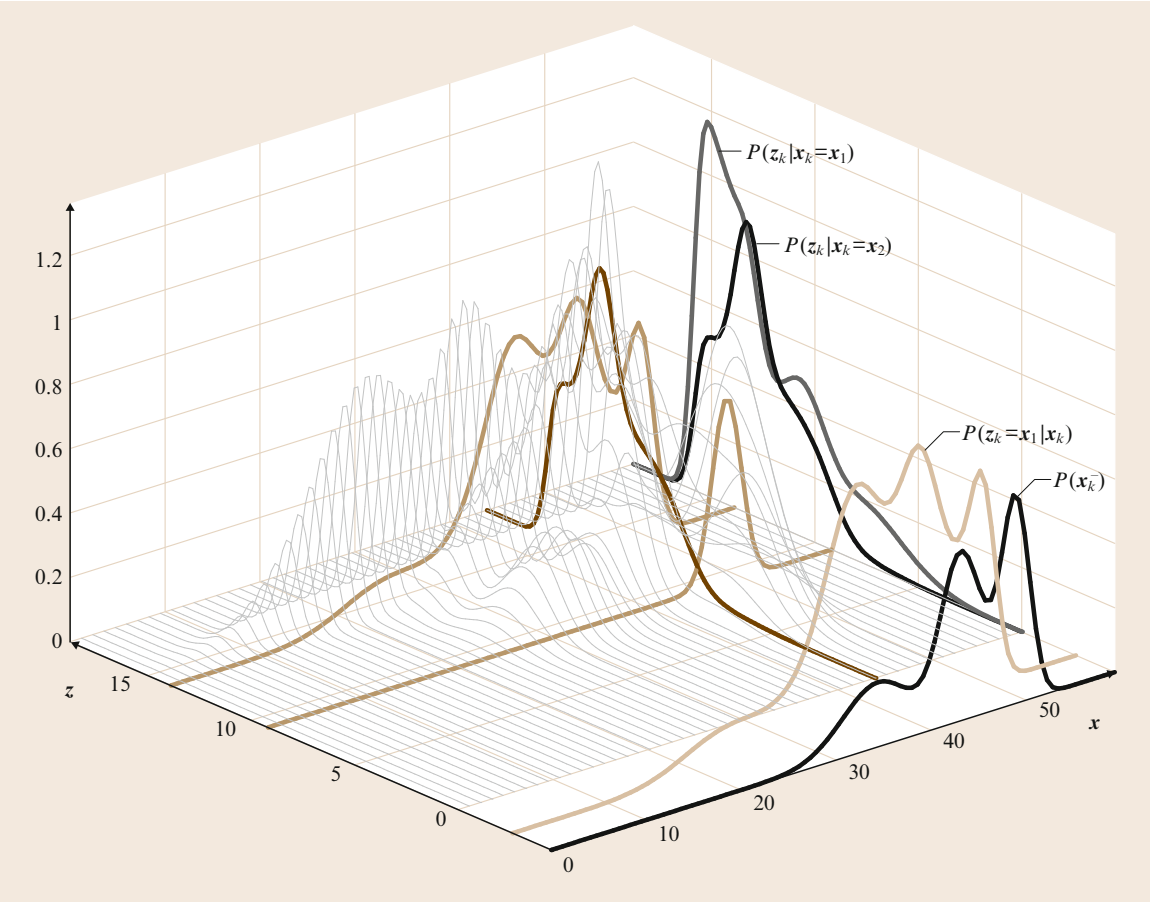


Fig. 35.2 Observation update for the full Bayes filter. Prior to observation, an observation model in the form of the conditional density $P(z_k \mid \boldsymbol{x}_k)$ is established. For a fixed value of $\boldsymbol{x}_k$, equal to $\boldsymbol{x}_1$ or $\boldsymbol{x}_2$ for example, a density function $P(z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_1)$ or $P(z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_2)$ is defined describing the likelihood of making the observation z_k. Together the density $P(z_k \mid \boldsymbol{x}_k)$ is then a function of both z_k and $\boldsymbol{x}_k$. This conditional density then defines the observation model. Now, in operation, a specific observation $z_k = \boldsymbol{x}_1$ is made and the resulting distribution $P(z_k = \boldsymbol{x}_1 \mid \boldsymbol{x}_k)$ defines a density function (now termed the likelihood function) on $\boldsymbol{x}_k$. This density is then multiplied by the prior density $P(\boldsymbol{x}_k^-)$ and normalized to obtain the posterior distribution $P(\boldsymbol{x}_k \mid z_k)$ describing knowledge in the state after observation

observations. Practically, however, this is implemented simply as another *observation* grid so that for a specific observation $\boldsymbol{z} = z$ (taken from a specific location), a grid of likelihoods on the states $\boldsymbol{x}_{ij}$ is produced in the form $P(\boldsymbol{z} = z \mid \boldsymbol{x}_{ij}) = \Lambda(\boldsymbol{x}_{ij})$. It is then trivial to apply Bayes' rule to update the property value at each grid cell as

$$P^+(\boldsymbol{x}_{ij}) = C\Lambda(\boldsymbol{x}_{ij})P(\boldsymbol{x}_{ij}) \,, \quad \forall i, j \,, \tag{35.9}$$

where C us a normalizing constant obtained by summing posterior probabilities to one *at node ij only*. Computationally, this is a simple pointwise multiplication of two grids. Some care needs to be taken that the two grids appropriately overlap and align with each other at the right scale. In some instances it is also valuable to encode the fact that spatially adjacent cells will influence each other; that is, if we knew the value of the property (occupancy, temperature, for example) at ij we will have some belief also of the value of this property at adjacent nodes $i+1, j$, $i, j+1$, etc. Different sensors and the fusion of different sensor outputs is accommodated simply by building appropriate sensor models $\Lambda(\boldsymbol{x}_{ij})$.

Grids can also be used for tracking and self-tracking (localization). The state $\boldsymbol{x}_{ij}$ in this case is the location of the entity being tracked. This is a qualitatively different definition of state from that used in mapping. The probability $P(\boldsymbol{x}_{ij})$ must now be interpreted as the probability that the object being tracked occupies the grid cell ij. In the case of mapping, the sum of property probabilities at each grid cell is one, whereas in the case of tracking, the

sum of location probabilities over *the whole grid* must sum to one. Otherwise, the procedure for updating is very similar. An observation grid is constructed which, when instantiated with an observation value, provides a location likelihood grid

$$P(\boldsymbol{z} = z \mid \boldsymbol{x}_{ij}) = \Lambda(\boldsymbol{x}_{ij}) \, .$$

Bayes' rule is then applied to update the location probability at each grid cell in the same form as (35.9) except that now the normalization constant C is obtained by summing posterior probabilities over *all* ij grid cells. This can become computationally expensive, especially if the grid has three or more dimensions. One major advantage of grid-based tracking is that it is easy to incorporate quite complex prior information. For example, if it is known that the object being tracked is on a road, then the probability location values for all off-road grid cells can simply be set to zero.

Grid-based fusion is appropriate to situations where the domain size and dimension are modest. In such cases, grid-based methods provide straightforward and effective fusion algorithms. Grid-based methods can be extended in a number of ways; to hierarchical (quad-tree) grids, or to irregular (triangular, pentagonal) grids. These can help reduce computation in larger spaces. Monte Carlo and particle filtering methods (Sect. 35.1.4) may be considered as grid-based methods, where the grid cells themselves are sample of the underlying probability density for the state.

35.1.3 The Kalman Filter

The Kalman filter is a recursive linear estimator that successively calculates an estimate for a continuous valued state, that evolves over time, on the basis of periodic observations of the state. The Kalman filter employs an explicit statistical model of how the parameter of interest $\boldsymbol{x}(t)$ evolves over time and an explicit statistical model of how the observations $\boldsymbol{z}(t)$ that are made are related to this parameter. The gains employed in a Kalman filter are chosen to ensure that, with certain assumptions about the observation and process models used, the resulting estimate $\hat{\boldsymbol{x}}(t)$ minimizes mean-squared error and is thus the conditional mean $\hat{\boldsymbol{x}}(t) = \mathrm{E}[\boldsymbol{x}(t) \mid \boldsymbol{Z}^t]$: an average, rather than a most likely value.

The Kalman filter has a number of features which make it ideally suited to dealing with complex multisensor estimation and data fusion problems. In particular, the explicit description of process and observations allows a wide variety of different sensor models to be incorporated within the basic algorithm. In addition, the consistent use of statistical measures of uncertainty makes it possible to quantitatively evaluate the role each sensor plays in overall system performance. Further, the linear recursive nature of the algorithm ensures that its application is simple and efficient. For these reasons, the Kalman filter has found widespread application in many different data fusion problems [35.6–9].

In robotics, the Kalman filter is most suited to problems in tracking, localization, and navigation, and less so to problems in mapping. This is because the algorithm works best with well-defined state descriptions (positions, velocities, for example), and for states where observation and time-propagation models are also well understood.

Observation and Transition Models

The Kalman filter may be considered a specific instance of the recursive Bayesian filter of (35.7) and (35.8) for the case where the probability densities on states are Gaussian. The starting point for the Kalman Filter algorithm is to define a model for the states to be estimated in the standard state-space form

$$\dot{\boldsymbol{x}}(t) = \mathbf{F}(t)\boldsymbol{x}(t) + \mathbf{B}(t)\boldsymbol{u}(t) + \mathbf{G}(t)\boldsymbol{v}(t) \, , \qquad (35.10)$$

where $\boldsymbol{x}(t)$ is the state vector of interest, $\boldsymbol{u}(t)$ is a known control input, $\boldsymbol{v}(t)$ is a random variable describing uncertainty in the evolution of the state, and $\mathbf{F}(t)$, $\mathbf{B}(t)$, and $\mathbf{G}(t)$ are matrices describing the contribution of states, controls, and noise to state transition [35.7]. An observation (output) model is also defined in standard state-space form

$$\boldsymbol{z}(t) = \mathbf{H}(t)\boldsymbol{x}(t) + \mathbf{D}(t)\boldsymbol{w}(t) \, , \qquad (35.11)$$

where $\boldsymbol{z}(t)$ is the observation vector, $\boldsymbol{w}(t)$ is a random variable describing uncertainty in the observation, and where $\mathbf{H}(t)$ and $\mathbf{D}(t)$ are matrices describing the contribution of state and noise to the observation.

These equations define the evolution of a continuous-time system with continuous observations being made of the state. However, the Kalman Filter is almost always implemented in discrete time $t_k = k$. It is straightforward [35.8] to obtain a discrete-time version of (35.10) and (35.11) in the form

$$\boldsymbol{x}(k) = \mathbf{F}(k)\boldsymbol{x}(k-1) + \mathbf{B}(k)\boldsymbol{u}(k) + \mathbf{G}(k)\boldsymbol{v}(k) \, , \qquad (35.12)$$

$$\boldsymbol{z}(k) = \mathbf{H}(k)\boldsymbol{x}(k) + \mathbf{D}(k)\boldsymbol{w}(k) \, . \qquad (35.13)$$

A basic assumption in the derivation of the Kalman filter is that the random sequences $\boldsymbol{v}(k)$ and $\boldsymbol{w}(k)$ describing process and observation noise are all Gaussian, temporally uncorrelated, and zero-mean

$$\mathrm{E}[\boldsymbol{v}(k)] = \mathrm{E}[\boldsymbol{w}(k)] = \mathbf{0} \, , \quad \forall k \, , \qquad (35.14)$$

with known covariance

$$\mathrm{E}[\boldsymbol{v}(i)\boldsymbol{v}^{\mathrm{T}}(j)] = \delta_{ij}\mathbf{Q}(i)\,, \quad \mathrm{E}[\boldsymbol{w}(i)\boldsymbol{w}^{\mathrm{T}}(j)] = \delta_{ij}\mathbf{R}(i)\,. \tag{35.15}$$

It is also generally assumed that the process and observation noises are also uncorrelated

$$\mathrm{E}[\boldsymbol{v}(i)\boldsymbol{w}^{\mathrm{T}}(j)] = \mathbf{0}\,, \quad \forall i,j\,. \tag{35.16}$$

These are equivalent to a Markov property requiring observations and successive states to be conditionally independent. If the sequences $\boldsymbol{v}(k)$ and $\boldsymbol{w}(k)$ are temporally correlated, a shaping filter can be used to whiten the observations, again making the assumptions required for the Kalman filter valid [35.8]. If the process and observation noise sequences are correlated, then this correlation can also be accounted for in the Kalman filter algorithm [35.10]. If the sequence is not Gaussian, but is symmetric with finite moments, then the Kalman filter will still produce good estimates. If however, the sequence has a distribution which is skewed or otherwise pathological, results produced by the Kalman filter will be misleading and there will be a good case for using a more-sophisticated Bayesian filter [35.5].

Filtering Algorithm

The Kalman filter algorithm produces estimates that minimize mean-squared estimation error conditioned on a given observation sequence and so is the conditional mean

$$\hat{\boldsymbol{x}}(i\,|\,j) \triangleq \mathrm{E}[\boldsymbol{x}(i)\,|\,z(1),\cdots,z(j)] \triangleq \mathrm{E}[\boldsymbol{x}(i)\,|\,\boldsymbol{Z}^j]\,. \tag{35.17}$$

The estimate variance is defined as the mean-squared error in this estimate

$$\boldsymbol{P}(i\,|\,j) \triangleq \mathrm{E}\{[\boldsymbol{x}(i)-\hat{\boldsymbol{x}}(i\,|\,j)][\boldsymbol{x}(i)-\hat{\boldsymbol{x}}(i\,|\,j)]^T\,|\,\boldsymbol{Z}^j\}\,. \tag{35.18}$$

The estimate of the state at a time k given all information up to time k is written $\hat{\boldsymbol{x}}(k\,|\,k)$. The estimate of the state at a time k given only information up to time $k-1$ is called a one-step-ahead prediction (or just a prediction) and is written $\hat{\boldsymbol{x}}(k\,|\,k-1)$.

The Kalman filter algorithm is now stated without proof. Detailed derivations can be found in many books on the subject, for example [35.7, 8]. The state is assumed to evolve in time according to (35.12). Observations of this state are made at regular time intervals according to (35.13). The noise processes entering the system are assumed to obey (35.14–35.16). It is also assumed that an estimate $\hat{\boldsymbol{x}}(k-1\,|\,k-1)$ of the state $\boldsymbol{x}(k-1)$ at time $k-1$ based on all observations made up to and including time $k-1$ is available, and that this estimate is equal to the conditional mean of the true state $\boldsymbol{x}(k-1)$ conditioned on these observations. The conditional variance $\boldsymbol{P}(k-1\,|\,k-1)$ in this estimate is also assumed known. The Kalman filter then proceeds recursively in two stages (Fig. 35.3).

Prediction. A prediction $\hat{\boldsymbol{x}}(k\,|\,k-1)$ of the state at time k and its covariance $\boldsymbol{P}(k\,|\,k-1)$ is computed according to

$$\hat{\boldsymbol{x}}(k\,|\,k-1) = \mathbf{F}(k)\hat{\boldsymbol{x}}(k-1\,|\,k-1) + \mathbf{B}(k)\boldsymbol{u}(k)\,, \tag{35.19}$$

$$\boldsymbol{P}(k\,|\,k-1) = \mathbf{F}(k)\boldsymbol{P}(k-1\,|\,k-1)\mathbf{F}^{\mathrm{T}}(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{G}^{\mathrm{T}}(k)\,. \tag{35.20}$$

Update. At time k an observation $\boldsymbol{z}(k)$ is made and the updated estimate $\hat{\boldsymbol{x}}(k\,|\,k)$ of the state $\boldsymbol{x}(k)$, together with the updated estimate covariance $\boldsymbol{P}(k\,|\,k)$, is computed from the state prediction and observation according to

$$\hat{\boldsymbol{x}}(k\,|\,k) = \hat{\boldsymbol{x}}(k\,|\,k-1) + \mathbf{W}(k)[\boldsymbol{z}(k) - \mathbf{H}(k)\hat{\boldsymbol{x}}(k\,|\,k-1)]\,, \tag{35.21}$$

$$\boldsymbol{P}(k\,|\,k) = \boldsymbol{P}(k\,|\,k-1) - \mathbf{W}(k)\mathbf{S}(k)\mathbf{W}^{\mathrm{T}}(k)\,, \tag{35.22}$$

where the gain matrix $\mathbf{W}(k)$ is given by

$$\mathbf{W}(k) = \boldsymbol{P}(k\,|\,k-1)\mathbf{H}(k)\mathbf{S}^{-1}(k)\,, \tag{35.23}$$

where

$$\mathbf{S}(k) = \mathbf{R}(k) + \mathbf{H}(k)\boldsymbol{P}(k\,|\,k-1)\mathbf{H}(k) \tag{35.24}$$

is the innovation covariance. The difference between the observation $\boldsymbol{z}(k)$ and the predicted observation $\mathbf{H}(k)\hat{\boldsymbol{x}}(k\,|\,k-1)$ is termed the *innovation* or residual $\boldsymbol{v}(k)$

$$\boldsymbol{v}(k) = \boldsymbol{z}(k) - \mathbf{H}(k)\hat{\boldsymbol{x}}(k\,|\,k-1)\,. \tag{35.25}$$

The innovation is an important measure of the deviation between the filter estimates and the observation sequence. Indeed, because the true states are not usually available for comparison with the estimated states, the innovation is often the only measure of how well the estimator is performing. The innovation is particularly important in data association.

The Extended Kalman Filter

The extended Kalman filter (EKF) is a form of the Kalman filter that can be employed when the state model and/or the observation model are nonlinear. The EKF is briefly described in this section.

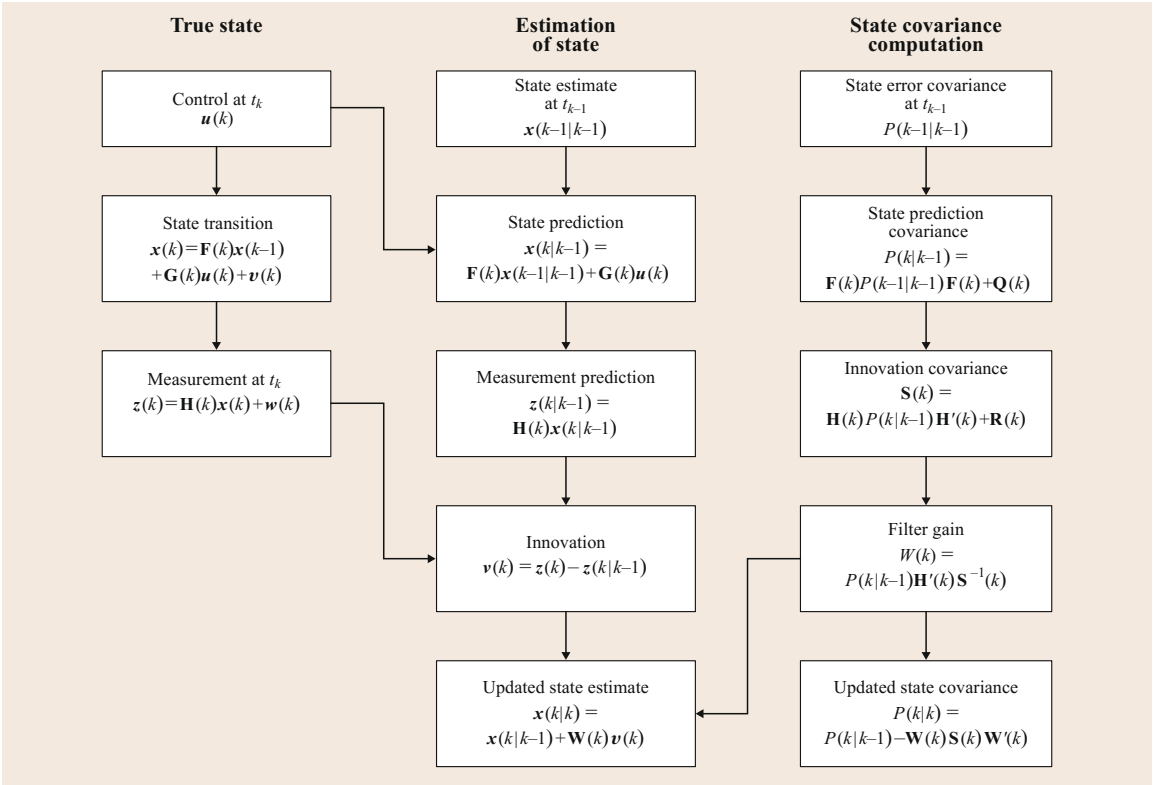


Fig. 35.3 Block diagram of the Kalman filter cycle (after *Bar-Shalom* and *Fortmann* [35.7])

The state models considered by the EKF are described in state-space notation by a first-order nonlinear vector differential equation or state model of the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}[\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), t] \,, \tag{35.26}$$

where $\boldsymbol{f}[\cdot,\cdot,\cdot,\cdot]$ is now a general nonlinear mapping of state and control input to state transition. The observation models considered by the EKF are described in state-space notation by a nonlinear vector function in the form

$$\boldsymbol{z}(t) = \boldsymbol{h}[\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{w}(t), t] \,, \tag{35.27}$$

where $\boldsymbol{h}[\cdot,\cdot,\cdot,\cdot]$ is now a general nonlinear mapping of state and control input to observations.

The EKF, like the Kalman filter, is almost always implemented in discrete time. By integration and with appropriate identification of discrete time states and observations, the state model is written

$$\boldsymbol{x}(k) = \boldsymbol{f}[\boldsymbol{x}(k-1), \boldsymbol{u}(k), \boldsymbol{v}(k), k] \,, \tag{35.28}$$

and the observation model as

$$\boldsymbol{z}(k) = \boldsymbol{h}[\boldsymbol{x}(k), \boldsymbol{w}(k)] \,. \tag{35.29}$$

Like the Kalman filter, it is assumed that the noises $\boldsymbol{v}(k)$ and $\boldsymbol{w}(k)$ are all Gaussian, temporally uncorrelated, and zero-mean with known variance as defined in (35.14–35.16). The EKF aims to minimize the mean-squared error and thereby compute an approximation to the conditional mean. It is assumed therefore that an estimate of the state at time $k-1$ is available which is approximately equal to the conditional mean, $\hat{\boldsymbol{x}}(k-1 \mid k-1) \approx \mathrm{E}[\boldsymbol{x}(k-1) \mid Z^{k-1}]$. The EKF algorithm will now be stated without proof. Detailed derivations may be found in any number of books on the subject. The principle stages in the derivation of the EKF follow directly from those of the linear Kalman filter with the additional step that the process and observation models are linearized as a Taylor series about the estimate and prediction, respectively. The algorithm has two stages: prediction and update.

Prediction. A prediction $\hat{\boldsymbol{x}}(k \mid k-1)$ of the state at time k and its covariance $\boldsymbol{P}(k \mid k-1)$ is computed according to

$$\hat{\boldsymbol{x}}(k \mid k-1) = \boldsymbol{f}[\hat{\boldsymbol{x}}(k-1 \mid k-1), \boldsymbol{u}(k)] \,, \tag{35.30}$$

$$\boldsymbol{P}(k \mid k-1) = \nabla \boldsymbol{f}_{\boldsymbol{x}}(k) \boldsymbol{P}(k-1 \mid k-1) \nabla^{\mathrm{T}} \boldsymbol{f}_{\boldsymbol{x}}(k) + \nabla \boldsymbol{f}_{\boldsymbol{v}}(k) \mathbf{Q}(k) \nabla^{\mathrm{T}} \boldsymbol{f}_{\boldsymbol{v}}(k) \,. \tag{35.31}$$

Update. At time k an observation $\boldsymbol{z}(k)$ is made and the updated estimate $\hat{\boldsymbol{x}}(k \mid k)$ of the state $\boldsymbol{x}(k)$, together with the updated estimate covariance $\boldsymbol{P}(k \mid k)$ is computed from the state prediction and observation according to

$$\hat{\boldsymbol{x}}(k \mid k) = \hat{\boldsymbol{x}}(k \mid k-1) + \mathbf{W}(k)\{\boldsymbol{z}(k) - \boldsymbol{h}[\hat{\boldsymbol{x}}(k \mid k-1)]\} , \quad (35.32)$$

$$\boldsymbol{P}(k \mid k) = \boldsymbol{P}(k \mid k-1) - \mathbf{W}(k)\mathbf{S}(k)\mathbf{W}^{\mathrm{T}}(k) , \quad (35.33)$$

where

$$\mathbf{W}(k) = \boldsymbol{P}(k \mid k-1)\nabla^{\mathrm{T}}\boldsymbol{h}_{\boldsymbol{x}}(k)\mathbf{S}^{-1}(k) , \quad (35.34)$$

and

$$\mathbf{S}(k) = \nabla\boldsymbol{h}_{w}(k)\mathbf{R}(k)\nabla^{\mathrm{T}}\boldsymbol{h}_{w}(k) + \nabla\boldsymbol{h}_{\boldsymbol{x}}(k)\boldsymbol{P}(k \mid k-1)\nabla^{\mathrm{T}}\boldsymbol{h}_{\boldsymbol{x}}(k) , \quad (35.35)$$

and where the Jacobian $\nabla\boldsymbol{f}_{\cdot}(k)$ is evaluated at $\boldsymbol{x}(k-1) = \hat{\boldsymbol{x}}(k-1 \mid k-1)$ and $\nabla\boldsymbol{h}_{\cdot}(k)$ is evaluated at $\boldsymbol{x}(k) = \hat{\boldsymbol{x}}(k \mid k-1)$.

A comparison of (35.19–35.24) with (35.30–35.35) makes it clear that the EKF algorithm is very similar to the linear Kalman filter algorithm, with the substitutions $\mathbf{F}(k) \to \nabla\boldsymbol{f}_{\boldsymbol{x}}(k)$ and $\mathbf{H}(k) \to \nabla\boldsymbol{h}_{\boldsymbol{x}}(k)$ being made in the equations for the variance and gain propagation. Thus, the EKF is, in effect, a linear estimator for a state error which is described by a *linear* equation and which is being observed according to a *linear* equation of the form of (35.13).

The EKF works in much the same way as the linear Kalman filter with some notable caveats:

- The Jacobians $\nabla\boldsymbol{f}_{\boldsymbol{x}}(k)$ and $\nabla\boldsymbol{h}_{\boldsymbol{x}}(k)$ are typically not constant, being functions of both state and time step. This means that unlike the linear filter, the covariances and gain matrix must be computed online as estimates and predictions are made available, and will not in general tend to constant values. This significantly increases the amount of computation which must be performed online by the algorithm.
- As the linearized model is derived by perturbing the true state and observation models around a predicted or nominal trajectory, great care must be taken to ensure that these predictions are always close enough to the true state that second order terms in the linearization are indeed insignificant. If the nominal trajectory is too far away from the true trajectory then the true covariance will be much larger than the estimated covariance and the filter will become poorly matched. In extreme cases the filter may also become unstable.
- The EKF employs a linearized model that must be computed from an approximate knowledge of the state. Unlike the linear algorithm this means that the filter must be accurately initialized at the start of operation to ensure that the linearized models obtained are valid. If this is not done, the estimates computed by the filter will simply be meaningless.

The Information Filter

The information filter is mathematically equivalent to a Kalman filter. However, rather than generating state estimates $\hat{\boldsymbol{x}}(i \mid j)$ and covariances $\boldsymbol{P}(i \mid j)$ it uses information state variables $\hat{\boldsymbol{y}}(i \mid j)$ and information matrices $\mathbf{Y}(i \mid j)$, which are related to each other through the relationships

$$\hat{\boldsymbol{y}}(i \mid j) = \boldsymbol{P}^{-1}(i \mid j)\hat{\boldsymbol{x}}(i \mid j) ,$$
$$\mathbf{Y}(i \mid j) = \boldsymbol{P}^{-1}(i \mid j) . \quad (35.36)$$

The information filter has the same prediction–update structure as the Kalman filter.

Prediction. A prediction $\hat{\boldsymbol{y}}(k \mid k-1)$ of the information state at time k and its information matrix $\mathbf{Y}(k \mid k-1)$ is computed according to (Joseph form [35.8])

$$\hat{\boldsymbol{y}}(k \mid k-1) = (\mathbf{1} - \boldsymbol{\Omega}\mathbf{G}^{\mathrm{T}})\mathbf{F}^{-\mathrm{T}}\hat{\boldsymbol{y}}(k-1 \mid k-1) + \mathbf{Y}(k \mid k-1)\mathbf{B}\boldsymbol{u}(k) , \quad (35.37)$$

$$\mathbf{Y}(k \mid k-1) = \mathbf{M}(k) - \boldsymbol{\Omega}\boldsymbol{\Sigma}\boldsymbol{\Omega}^{\mathrm{T}} , \quad (35.38)$$

respectively, where

$$\mathbf{M}(k) = \mathbf{F}^{-\mathrm{T}}\mathbf{Y}(k-1 \mid k-1)\mathbf{F}^{-1} ,$$
$$\boldsymbol{\Sigma} = \mathbf{G}^{\mathrm{T}}\mathbf{M}(k)\mathbf{G} + \mathbf{Q}^{-1} ,$$

and

$$\boldsymbol{\Omega} = \mathbf{M}(t_k)\mathbf{G}\boldsymbol{\Sigma}^{-1} .$$

It should be noted that $\boldsymbol{\Sigma}$, whose inverse is required to compute $\boldsymbol{\Omega}$, has the same dimensionality as the process driving noise, which is normally considerably smaller than the state dimension. Further, the matrix $\mathbf{F}^{-1}$ is the state-transition matrix evaluated backwards in time and so must always exist.

Update. At time k an observation $\boldsymbol{z}(k)$ is made and the updated information state estimate $\hat{\boldsymbol{y}}(k \mid k)$ together with the updated information matrix $\mathbf{Y}(k \mid k)$ is computed from

$$\hat{\boldsymbol{y}}(k \mid k) = \hat{\boldsymbol{y}}(k \mid k-1) + \mathbf{H}(k)\mathbf{R}^{-1}(k)\boldsymbol{z}(k) , \quad (35.39)$$

$$\mathbf{Y}(k \mid k) = \mathbf{Y}(k \mid k-1) + \mathbf{H}(k)\mathbf{R}^{-1}(k)\mathbf{H}^{\mathrm{T}}(k) . \quad (35.40)$$

We emphasize that (35.38) and (35.37) are mathematically identical to (35.19) and (35.20), and that (35.39) and (35.40) are mathematically identical to (35.21) and (35.22). It will be noted that there is a duality between information and state space forms [35.10]. This duality is evident from the fact that $\mathbf{\Omega}$ and $\mathbf{\Sigma}$ in the prediction stage of the information filter play an equivalent role to the gain matrix $\mathbf{W}$ and innovation covariance $\mathbf{S}$ in the update stage of the Kalman filter. Furthermore, the simple linear update step for the information filter is mirrored in the simple linear prediction step for the Kalman filter.

The main advantage of the information filter over the Kalman filter in data fusion problems is the relative simplicity of the update stage. For a system with n sensors, the fused information state update is *exactly* the linear sum of information contributions from all sensors as

$$\hat{\boldsymbol{y}}(k \mid k) = \hat{\boldsymbol{y}}(k \mid k-1) + \sum_{i=1}^{n} \mathbf{H}_i(k)\mathbf{R}_i^{-1}(k)\boldsymbol{z}_i(k) \,,$$
$$\mathbf{Y}(k \mid k) = \mathbf{Y}(k \mid k-1) + \sum_{i=1}^{n} \mathbf{H}_i(k)\mathbf{R}_i^{-1}(k)\mathbf{H}_i^{\mathrm{T}}(k) \,. \tag{35.41}$$

The reason such an expression exists in this form is that the information filter is essentially a log-likelihood expression of Bayes' rule, where products of likelihoods (35.4) are turned into sums. No such simple expression for multisensor updates exists for the Kalman filter. This property of the information filter has been exploited for data fusion in robotic networks [35.11, 12] and more recently in robot navigation and localization problems [35.1]. One substantial disadvantage of the information filter is the coding of nonlinear models, especially for the prediction step.

When to Use a Kalman or Information Filter

Kalman or information filters are appropriate to data fusion problems where the entity of interest is well defined by a continuous parametric state. This would include estimation of the position, attitude, and velocity of a robot or other object, or the tracking of a simple geometric feature such as a point, line or curve. Kalman and information filters are inappropriate for estimating properties such as spatial occupancy, discrete labels, or processes whose error characteristics are not easily parameterized.

35.1.4 Sequential Monte Carlo Methods

Monte Carlo (MC) filter methods describe probability distributions as a set of weighted samples of an underlying state space. MC filtering then uses these samples to simulate probabilistic inference usually through Bayes' rule. Many samples or simulations are performed. By studying the statistics of these samples as they progress through the inference process, a probabilistic picture of the process being simulated can be built up.

Representing Probability Distributions

In sequential Monte Carlo methods, probability distributions are described in terms of a set of support points (state-space values) $\boldsymbol{x}^i$, $i = 1, \cdots, N$, together with a corresponding set of normalized weights w^i, $i = 1, \cdots, N$, where $\sum_i w^i = 1$. The support points and weights can be used to define a probability density function in the form

$$P(\boldsymbol{x}) \approx \sum_{i=1}^{N} w^i \delta(\boldsymbol{x} - \boldsymbol{x}^i) \,. \tag{35.42}$$

A key question is how these support points and weights are selected to obtain a faithful representation of the probability density $P(\boldsymbol{x})$. The most general way of selecting support values is to use an *importance density* $q(\boldsymbol{x})$. The support values $\boldsymbol{x}^i$ are drawn as samples from this density; where the density has high probability, more support values are chosen, and where the density has low probability, few support support vectors are selected. The weights in (35.42) are then computed from

$$w^i \propto \frac{P(\boldsymbol{x}^i)}{q(\boldsymbol{x}^i)} \,. \tag{35.43}$$

Practically, a sample $\boldsymbol{x}^i$ is drawn from the importance distribution. The sample is then instantiated in the underlying probability distribution to yield the value $P(\boldsymbol{x} = \boldsymbol{x}^i)$. The ratio of the two probability values, appropriately normalized, then becomes the weight.

There are two instructive extremes of the importance sampling method:

1. At one extreme, the importance density could be taken to be a uniform distribution, so the support values $\boldsymbol{x}^i$ are uniformly distributed on the state space in a close approximation to a grid. The probabilities $q(\boldsymbol{x}^i)$ are also therefore equal. The weights computed from (35.43) are then simply proportional to the probabilities $w^i \propto P(\boldsymbol{x} = \boldsymbol{x}^i)$. The result is a model for the distribution which looks very like the regular grid model.
2. At the other extreme, we could choose an importance density equal to the probability model $q(\boldsymbol{x}) = P(\boldsymbol{x})$. Samples of the support values $\boldsymbol{x}^i$ are now drawn from this density. Where the density is high

there will be many samples, where the density is low there will be few samples. However, if we substitute $q(\boldsymbol{x}^i) = P(\boldsymbol{x}^i)$ into (35.43), it is clear that the weights all become equal $w^i = \frac{1}{N}$. A set of samples with equal weights is known as a *particle* distribution.

It is, of course, possible to mix these two representations to describe a probability distribution both in terms of a set of weights and in terms of a set of support values. The complete set of samples and weights describing a probability distribution $\{\boldsymbol{x}^i, w^i\}_{i=1}^N$ is termed a *random measure*.

The Sequential Monte Carlo Method

Sequential Monte Carlo (SMC) filtering is a simulation of the recursive Bayes update equations using sample support values and weights to describe the underlying probability distributions.

The starting point is the recursive or sequential Bayes observation update given in (35.7) and (35.8). The SMC recursion begins with an a posterior probability density represented by a set of support values and weights $\{\boldsymbol{x}_{k-1}^i, w_{k-1|k-1}^i\}_{i=1}^{N_{k-1}}$ in the form

$$P\left(\boldsymbol{x}_{k-1} \mid \boldsymbol{Z}^{k-1}\right) = \sum_{i=1}^{N_{k-1}} w_{k-1}^i \delta\left(\boldsymbol{x}_{k-1} - \boldsymbol{x}_{k-1}^i\right) . \tag{35.44}$$

The prediction step requires that (35.44) is substituted into (35.8) where the joint density is marginalized. Practically however, this complex step is avoided by implicitly assuming that the importance density is exactly the transition model as

$$q_k\left(\boldsymbol{x}_k^i\right) = P\left(\boldsymbol{x}_k^i \mid \boldsymbol{x}_{k-1}^i\right) . \tag{35.45}$$

This allows new support values $\boldsymbol{x}_k^i$ to be drawn on the basis of old support values $\boldsymbol{x}_{k-1}^i$ while leaving the weights unchanged $w_k^i = w_{k-1}^i$. With this, the prediction becomes

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^{k-1}\right) = \sum_{i=1}^{N_k} w_{k-1}^i \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_k^i\right) . \tag{35.46}$$

The SMC observation update step is relatively straightforward. An observation model $P(\boldsymbol{z}_k \mid \boldsymbol{x}_k)$ is defined. This is a function on both variables, $\boldsymbol{z}_k$ and $\boldsymbol{x}_k$, and is a probability distribution on $\boldsymbol{z}_k$ (integrates to unity). When an observation or measurement is made, $\boldsymbol{z}_k = z_k$, the observation model becomes a function of state $\boldsymbol{x}_k$ only. If samples of the state are taken $\boldsymbol{x}_k = \boldsymbol{x}_k^i$, $i = 1 \cdots, N_k$, the observation model $P(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^i)$ becomes a set of scalars describing the likelihood that the sample $\boldsymbol{x}_k^i$ could have given rise to the observation z_k. Substituting these likelihoods and (35.46) into (35.7) gives

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^k\right) = C \sum_{i=1}^{N_k} w_{k-1}^i P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^i\right) \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_k^i\right) . \tag{35.47}$$

This is normally implemented in the form of an updated set of normalized weights

$$w_k^i = \frac{w_{k-1}^i P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^i\right)}{\sum_{j=1}^{N_k} w_{k-1}^j P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^j\right)} , \tag{35.48}$$

and so

$$P\left(\boldsymbol{x}_k \mid \boldsymbol{Z}^k\right) = \sum_{i=1}^{N_k} w_k^i \delta\left(\boldsymbol{x}_k - \boldsymbol{x}_k^i\right) . \tag{35.49}$$

Note that the support values in (35.49) are the same as those in (35.46), only the weights have been changed by the observation update.

The implementation of the SMC method requires the enumeration of models for both the state transition $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$ and the observation $P(\boldsymbol{z}_k \mid \boldsymbol{x}_k)$. These need to be presented in a form that allows instantiation of values for $\boldsymbol{z}_k$, $\boldsymbol{x}_k$, and $\boldsymbol{x}_{k-1}$. For low-dimensional state spaces, interpolation in a lookup table is a viable representation. For high-dimensional state spaces, the preferred method is to provide a representation in terms of a function.

Practically, (35.46) and (35.49) are implemented as follows.

Time Update. A process model is defined in the usual state-space form as $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{w}_{k-1}, k)$, where $\boldsymbol{w}_k$ is an independent noise sequence with known probability density $P(\boldsymbol{w}_k)$. The prediction step is now implemented as follows: N_k samples $\boldsymbol{w}_k^i$, $i = 1, \cdots, N_k$ are drawn from the distribution $P(\boldsymbol{w}_k)$. The N_k support values $\boldsymbol{x}_{k-1}^i$ together with the samples $\boldsymbol{w}_k^i$ are passed through the process model as

$$\boldsymbol{x}_k^i = \boldsymbol{f}\left(\boldsymbol{x}_{k-1}^i, \boldsymbol{w}_{k-1}^i, k\right) , \tag{35.50}$$

yielding a new set of support vectors $\boldsymbol{x}_k^i$. The weights for these support vectors w_{k-1}^i are not changed. In effect, the process model is simply used to do N_k simulations of state propagation.

Observation Update. The observation model is also defined in the usual state-space form as $\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{v}_k, k)$, where $\boldsymbol{v}_k$ is an independent noise sequence with known probability density $P(\boldsymbol{v}_k)$. The observation step is now implemented as follows. A measurement $\boldsymbol{z}_k = z_k$ is made. For each support value $\boldsymbol{x}_k^i$, a likelihood is computed as

$$\Lambda\left(\boldsymbol{x}_k^i\right) = P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k = \boldsymbol{x}_k^i\right) . \tag{35.51}$$

Practically, this requires that the observation model be in an equational form (such as a Gaussian) which allows computation of the likelihood in the error between the measured value z_k and the observations predicted by each particle $\boldsymbol{h}(\boldsymbol{x}_k^i, k)$. The updated weights after observation are just

$$w_k^i \propto w_{k-1}^i P\left(\boldsymbol{z}_k = z_k \mid \boldsymbol{x}_k^i\right) . \tag{35.52}$$

Resampling

After the weights are updated it is usual to resample the measure $\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N$. This focuses the samples in on those areas that have most probability density. The decision to resample is made on the basis of the *effective number*, N_{eff} of particles in the sample, approximately estimated from

$$N_{\text{eff}} = \frac{1}{\sum_i (w_k^i)^2} .$$

The sampling importance resampling (SIR) algorithm resamples at every cycle so that the weights are always equal. One of the key problems with resampling is that the sample set fixates on a few highly likely samples. This problem of fixating on a few highly likely particles during resampling is known as *sample impoverishment*. Generally, it is good to resample when N_{eff} falls to some fraction of the actual samples (say $1/2$).

When to Use Monte Carlo Methods

Monte Carlo (MC) methods are well suited to problems where state-transition models and observation models are highly nonlinear. This is because sample-based methods can represent very general probability densities. In particular multimodal or multiple hypothesis density functions are well handled by Monte Carlo techniques. One caveat to note however is that the models $P(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$ and $P(\boldsymbol{z}_k \mid \boldsymbol{x}_k)$ must be enumerable in all cases and must typically be in a simple parametric form. MC methods also span the gap between parametric and grid-based data fusion methods.

Monte Carlo methods are inappropriate for problems where the state space is of high dimension. In general the number of samples required to faithfully model a given density increases exponentially with state-space dimension. The effects of dimensionality can be limited by marginalizing out states that can be modeled without sampling, a procedure known as Rao–Blackwellization.

35.1.5 Alternatives to Probability

The representation of uncertainty is so important to the problem of information fusion that a number of alternative modeling techniques have been proposed to deal with perceived limitations in probabilistic methods.

There are three main perceived limitations of probabilistic modeling techniques:

1. Complexity: the need to specify a large number of probabilities to be able to apply probabilistic reasoning methods correctly.
2. Inconsistency: the difficulties involved in specifying a consistent set of beliefs in terms of probability and using these to obtain consistent deductions about states of interest.
3. Precision of models: the need to be precise in the specification of probabilities for quantities about which little is known.
4. Uncertainty about uncertainty: the difficulty in assigning probability in the face of uncertainty, or ignorance about the source of information.

There are three main techniques put forward to address these issues: interval calculus, fuzzy logic, and the theory of evidence (Dempster–Shafer methods). We briefly discuss each of these in turn.

Interval Calculus

The representation of uncertainty using an interval to bound true parameter values has a number of potential advantages over probabilistic techniques. In particular, intervals provide a good measure of uncertainty in situations where there is a lack of probabilistic information, but in which sensor and parameter error is known to be bounded. In interval techniques, the uncertainty in a parameter x is simply described by a statement that the true value of the state x is known to be bounded from below by a, and from above by b; $x \in [a, b]$. It is important that no other additional probabilistic structure is implied; in particular the statement $x \in [a, b]$ does not necessarily imply that x is equally probable (uniformly distributed) over the interval $[a, b]$.

There are a number of simple and basic rules for the manipulation of interval errors. These are described in detail in the book by *Moore* [35.13] (whose analysis was originally aimed at understanding limited-precision computer arithmetic). Briefly, with $a, b, c, d \in \mathbb{R}$, addi-

tion, subtraction, multiplication, and division are defined by the following algebraic relations

$$[a,b]+[c,d]=[a+c,b+d]\,,$$
$$[a,b]-[c,d]=[a-d,b-c]\,, \tag{35.53}$$
$$[a,b]\times[c,d]=[\min(ac,ad,bc,bd),\ \max(ac,ad,bc,bd)]\,, \tag{35.54}$$
$$\frac{[a,b]}{[c,d]}=[a,b]\times\left[\frac{1}{d},\frac{1}{c}\right]\,,\quad 0\notin[c,d]\,. \tag{35.55}$$

It can be shown that interval addition and multiplication are both associative and commutative. Interval arithmetic admits an obvious metric distance measure

$$d([a,b],[c,d])=\max(|a-c|,|b-d|)\,. \tag{35.56}$$

Matrix arithmetic using intervals is also possible, but substantially more complex, particularly when matrix inversion is required.

Interval calculus methods are sometimes used for detection. However, they are not generally used in data fusion problems since:

1. It is difficult to get results that converge to anything of value (it is too pessimistic), and
2. It is hard to encode dependencies between variables which are at the core of many data fusion problems.

Fuzzy Logic

Fuzzy logic has found widespread popularity as a method for representing uncertainty particularly in applications such as supervisory control and high-level data fusion tasks. It is often claimed that fuzzy logic provides an ideal tool for inexact reasoning, particularly in rule-based systems. Certainly, fuzzy logic has had some notable success in practical application.

A great deal has been written about fuzzy sets and fuzzy logic (e.g. [35.14] and the discussion in [35.15, Chap. 11]). Here we briefly describe the main definitions and operations without any attempt to consider the more advanced features of fuzzy-logic methods.

Consider a universal set consisting of the elements x; $\mathcal{X}=\{x\}$. Consider a proper subset $\mathcal{A}\subseteq\mathcal{X}$ such that

$$\mathcal{A}=\{x\mid x \text{ has some specific property}\}\,.$$

In conventional logic systems, we can define a membership function $\mu_A(x)$ (also called the *characteristic function* which reports if a specific element $x\in\mathcal{X}$ is a member of this set

$$\mathcal{A}\rightleftharpoons\mu_A(x)=\begin{cases}1 & \text{if } x\in A\\ 0 & \text{if } x\notin A\end{cases}\,.$$

For example $\mathcal{X}$ may be the set of all aircraft. The set $\mathcal{A}$ may be the set of all supersonic aircraft. In the fuzzy logic literature, this is known as a *crisp* set. In contrast, a fuzzy set is one in which there is a *degree of membership*, ranging between 0 and 1. A fuzzy membership function $\mu_A(x)$ then defines the degree of membership of an element $x\in\mathcal{X}$ to the set $\mathcal{A}$. For example, if $\mathcal{X}$ is again the set of all aircraft, $\mathcal{A}$ may be the set of all *fast* aircraft. Then the fuzzy membership function $\mu_A(x)$ assigns a value between 0 and 1 indicating the degree of membership of every aircraft x to this set. Formally

$$\mathcal{A}\rightleftharpoons\mu_A\mapsto[0,1]\,.$$

Composition rules for fuzzy sets follow the composition processes for normal crisp sets, for example,

$$\mathcal{A}\cap\mathcal{B}\rightleftharpoons\mu_{A\cap B}(x)=\min[\mu_A(x),\mu_B(x)]\,,$$
$$\mathcal{A}\cup\mathcal{B}\rightleftharpoons\mu_{A\cup B}(x)=\max[\mu_A(x),\mu_B(x)]\,.$$

The normal properties associated with binary logic now hold: commutativity, associativity, idempotence, distributivity, De Morgan's law, and absorption. The only exception is that the law of the excluded middle is no longer true

$$\mathcal{A}\cup\overline{\mathcal{A}}\neq\mathcal{X}\,,\quad \mathcal{A}\cap\overline{\mathcal{A}}\neq\emptyset\,.$$

Together these definitions and laws provide a systematic means of reasoning about inexact values.

The relationship between fuzzy set theory and probability is still hotly debated.

Evidential Reasoning

Evidential reasoning (often called the Dempster–Shafer theory of evidence after the originators of these ideas) has seen intermittent success particularly in automated reasoning applications. Evidential reasoning is qualitatively different from either probabilistic methods or fuzzy set theory in the following sense. Consider a universal set $\mathcal{X}$. In probability theory or fuzzy set theory, a belief mass may be placed on any element $x_i\in\mathcal{X}$ and indeed on any subset $\mathcal{A}\subseteq\mathcal{X}$. In evidential reasoning, belief mass can not only be placed on elements and sets, but also sets of sets. Specifically, while the domain of probabilistic methods is all possible subsets $\mathcal{X}$, the domain of evidential reasoning is the power set $2^{\mathcal{X}}$.

As an example, consider the mutually exclusive set $\mathcal{X}=\{\text{occupied},\text{empty}\}$. In probability theory we

might assign a probability to each possible event, for example, $P(\text{occupied}) = 0.3$, and thus $P(\text{empty}) = 0.7$. In evidential reasoning, we construct the set of all subsets

$$2^{\mathcal{X}} = \{\{\text{occupied}, \text{empty}\}, \{\text{occupied}\}, \{\text{empty}\}, \emptyset\} ,$$

and belief mass is assigned to all elements of this set as

$$\begin{aligned} m(\{\text{occupied}, \text{empty}\}) &= 0.5 , \\ m(\{\text{occupied}\}) &= 0.3 , \\ m(\{\text{empty}\}) &= 0.2 , \\ m(\emptyset) &= 0.0 , \end{aligned}$$

(the null set $\emptyset$ is assigned a belief mass of zero for normalization purposes). The interpretation of this is that there is a 30% chance of occupied, a 20% chance of empty, and a 50% chance of either occupied or empty. In effect, the measure placed on the set containing both occupied and empty is a measure of ignorance or inability to distinguish between the two alternatives. See [35.16] for a more-detailed example of applying the evidential method to certainty-grid navigation.

Evidential reasoning thus provides a method of capturing ignorance or an inability to distinguish between alternatives. In probability theory, this would be dealt with in a very different manner by assigning an equal or uniform probability to each alternative. Yet, stating that there is a 50% chance of occupancy is clearly *not* the same as saying that it is unknown whether it will occupied or not. The use of the power set as the *frame of discernment* allows a far richer representation of beliefs. However, this comes at the cost of a substantial increase in complexity. If there are n elements in the original set $\mathcal{X}$, then there will be 2^n possible subsets on which a belief mass will be assigned. For large n, this is clearly intractable. Further, when the set is continuous, the set of all subsets is not even measurable.

Evidential reasoning methods provide a means of assigning, and combining belief masses on sets. Methods also exist for obtaining related measures called *support* and *plausibility* which, in effect, provide upper and lower probability bounds in agreement with Dempster's original formulation of this method.

Evidential reasoning can play an important role in discrete data fusion systems, particularly in areas such as attribute fusion and situation assessment, where information may be unknown or ambiguous. Its use in lower-level data fusion problems is challenging as the assignment of belief mass to the power set scales exponentially with state cardinality.

35.2 Multisensor Fusion Architectures

The multisensor fusion methods described in the previous section provide the algorithmic means by which sensor data and their associated uncertainty models can be used to construct either implicit or explicit models of the environment. However, a multisensor fusion system must include many other functional components to manage and control the fusion process. The organization of these is termed a *multisensor fusion architecture*.

35.2.1 Architectural Taxonomy

Multisensor systems architectures can be organized in various ways. The military community has developed a layout of functional architectures based on the joint directors of the laboratories (JDL) model for multisensor systems. This approach views multisensor fusion in terms of signal, feature, threat and situation analysis levels (so-called JDL levels). The assessment of such systems is specified in terms of tracking performance, survivability, efficiency and bandwidth. Such measures are not generally appropriate in robotics applications and so the JDL model is not discuss this further here (see [35.17, 18] for details). Other classification schemes distinguish between low- and high-level fusion [35.19], or centralized versus decentralized processing or data versus variable [35.20].

A general architectural framework for multisensor robotic systems has been developed and described in detail by *Makarenko* [35.21], and we will base our discussion on this approach. A system architecture is defined as follows.

Meta-Architecture

A set of high-level considerations that strongly characterize the system structure. The selection and organization of the system elements may be guided by aesthetics, efficiency, or other design criteria and goals (for example, system and component comprehensibility, modularity, scalability, portability, interoperability, (de)centralization, robustness, fault tolerance).

Algorithmic Architecture

A specific set of information fusion and decision-making methods. These methods address data heterogeneity, registration, calibration, consistency, informa-

tion content, independence, time interval and scale, and relationships between models and uncertainty.

Conceptual Architecture

The granularity and functional roles of components (specifically, mappings from algorithmic elements to functional structures).

Logical Architecture

Detailed canonical component types (i. e., object-oriented specifications) and interfaces to formalize intercomponent services. Components may be ad hoc or regimented, and other concerns include granularity, modularity, reuse, verification, data structures, semantics, etc. Communication issues include hierarchical versus heterarchical organization, shared memory versus message passing, information-based characterizations of subcomponent interactions, pull/push mechanisms, subscribe–publish mechanisms, etc. Control involves both the control of actuation systems within the multisensor fusion system, as well as control of information requests and dissemination within the system, and any external control decisions and commands.

Execution Architecture

Defines mapping of components to execution elements. This includes internal or external methods of ensuring correctness of the code (i. e., that the environment and sensor models have been correctly transformed from mathematical or other formal descriptions into computer implementations), and also validation of the models (i. e., ensure that the formal descriptions match physical reality to the required extent).

In any closed-loop control system, sensors are used to provide the feedback information describing the current status of the system and its uncertainties. Building a sensor system for a given application is a system engineering process that includes the analysis of system requirements, a model of the environment, the determination of system behavior under different conditions, and the selection of suitable sensors [35.22]. The next step in building the sensor system is to assemble the hardware components and develop the necessary software modules for data fusion and interpretation. Finally, the system is tested, and the performance is analyzed. Once the system is built, it is necessary to monitor the different components of the system for the purpose of testing, debugging, and analysis. The system also requires quantitative measures in terms of time complexity, space complexity, robustness, and efficiency.

In addition, designing and implementing real-time systems are becoming increasingly complex owing to many added features such as graphical user interfaces (GUIs), visualization capabilities, and the use of many sensors of different types. Therefore, many software engineering issues such as reusability and the use of commercial off-the-shelf (COTS) components [35.23], real-time issues [35.24–26], sensor selection [35.27], reliability [35.28–30], and embedded testing [35.31] are now receiving more attention from system developers.

Each sensor type has different characteristics and functional descriptions. Consequently, some approaches aim to develop general methods of modeling sensor systems in a manner that is independent of the physical sensors used. In turn, this enables the performance and robustness of multisensor systems to be studied in a general way. There have been many attempts to provide *the* general model, along with its mathematical basis and description. Some of these modeling techniques concern error analysis and fault tolerance of multisensor systems [35.32–37]. Other techniques are model based, and require a priori knowledge of the sensed object and its environment [35.38–40]. These help fit data to a model, but do not always provide the means to compare alternatives. Task-directed sensing is another approach to devising sensing strategies [35.41–43]. General sensor modeling work has had a considerable influence on the evolution of multisensor fusion architectures.

Another approach to modeling sensor systems is to define sensori-computational systems associated with each sensor to allow design, comparison, transformation, and reduction of any sensory system [35.44]. In this approach, the concept of an information invariant is used to define a measure of information complexity. This provides a computational theory allowing analysis, comparison, and reduction of sensor systems.

In general terms, multisensor fusion architectures may be classified according to the choice along four independent design dimensions:

1. Centralized–decentralized,
2. Local–global interaction of components,
3. Modular–monolithic, and
4. Heterarchical–hierarchical.

The most prevalent combinations are:

- Centralized, global interaction, and hierarchical,
- Decentralized, global interaction, and heterarchical,
- Decentralized, local interaction, and hierarchical,
- Decentralized, local interaction, and heterarchical.

In some cases explicit modularity is also desirable. Most existing multisensor architectures fit reasonably well into one of these categories. These categories make no general commitment to the algorithmic architecture.

If the algorithmic architecture is the predominant feature of a system, it will be characterized as part of multisensor fusion theory in Sect. 35.1; otherwise, it merely differentiates methods within one of the four meta-architectures.

35.2.2 Centralized, Local Interaction, and Hierarchical

Centralized, local interaction and hierarchical architectures encompass a number of system philosophies. Least representationally demanding is the *subsumption architecture* initially proposed by *Braitenberg* [35.45] and popularized by *Brooks* [35.46]. The subsumption multisensor architecture defines behaviors as the basic components, and employs a layered set of behaviors to embody one program (monolithic). Any behavior may utilize the output of other behaviors, and may also inhibit other behaviors. The hierarchy is defined by the layers, although this is not always clear-cut. The major design philosophy is to develop behaviors directly from perception–action loops without recourse to brittle, environment representations. This leads to robustness in operation, but a lack of composite behavior predictability.

A more sophisticated (representationally) behavior-based system is the *distributed field robot architecture* (DFRA) [35.47]. This is a generalization of the sensor fusion effects (SFX) architecture [35.48]. This approach exploits modularity, and aims to achieve both behavior-based and deliberative action, reconfigurability and interoperability through the use of Java, Jini, and XML, fault tolerance, adaptability, longevity, consistent interfaces, and dynamic components. The algorithmic architecture is based on fuzzy logic controllers. Experiments have been demonstrated on outdoor mobile robot navigation.

Other similar architectures of this type include perception action networks *Lee* and *Ro* [35.49, 50], while *Draper* et al. [35.51] focuses on types of information needed to perform tasks (higher-level integration); see also [35.52].

Another approach to this type of sensor fusion is to use artificial neural networks. The advantage is that the user, at least in principle, does not need to understand how sensor modalities relate, nor model the uncertainties, nor in fact determine the structure of the system more than to specify the number of layers in the network and the number of nodes per layer. The neural network is presented with a set of training examples, and must determine through the weights on the neuron connections the optimal mapping from inputs to desired outputs (classifications, control signals, etc.) [35.53, 54].

Various other methods exist; for example, *Hager* and *Mintz* [35.42, 43] defines a task-oriented approach to sensor fusion based on Bayesian decision theory and develops an object-oriented programming framework. *Joshi* and *Sanderson* [35.55] describe a

> *methodology for addressing model selection and multisensor fusion issues using representation size (description length) to choose (1) model class and number of parameters, (2) model parameter resolution (3) subset of observed features to model, and (4) correspondence to map features to models.*

Their approach is broader than an architecture and uses a minimization criterion to synthesize a multisensor fusion system to solve specific two-dimensional (2-D) and three-dimensional (3-D) object recognition problems.

35.2.3 Decentralized, Global Interaction, and Heterarchical

The major example of the decentralized, global interaction meta-architecture is the blackboard system. There have been many examples of blackboard systems developed for data fusion applications. For example, the SEPIA system of *Berge-Cherfaoui* and *Vachon* [35.56] uses logical sensors (see below) in the form of modular agents that post results to a blackboard. The overall architectural goals for blackboards include efficient collaboration and dynamic configuration. Experiments are reported on an indoor robot moving from room to room.

The MESSIE (multi expert system for scene interpretation and evaluation) system [35.57] is a scene interpretation system based on multisensor fusion; it has been applied to the interpretation of remotely sensed images. A typology of the multisensor fusion concepts is presented, and the consequences of modeling problems for objects, scene, and strategy are derived. The proposed multispecialist architecture generalized the ideas of their previous work by taking into account the knowledge of sensors, the multiple viewing notion (shot), and the uncertainty and imprecision of models and data modeled with possibility theory. In particular, generic models of objects are represented by concepts independent of sensors (geometry, materials, and spatial context). Three kinds of specialists are present in the architecture: generic specialists (scene and conflict), semantic object specialists, and low-level specialists. A blackboard structure with a centralized control is used. The interpreted scene is implemented as a matrix of pointers enabling conflicts to be detected very easily. Under the control of the scene specialist, the conflict

specialist resolves conflicts using the spatial context knowledge of objects. Finally, an interpretation system with SAR (synthetic aperture radar)/SPOT sensors is described, and an example of a session concerned with bridge, urban area, and road detection is shown.

35.2.4 Decentralized, Local Interaction, and Hierarchical

One of the earliest proposals for this type of architecture is the real-time control system (RCS) [35.58]. RCS is presented as a cognitive architecture for intelligent control, but essentially uses multisensor fusion to achieve complex control. RCS focuses on task decomposition as the fundamental organizing principle. It defines a set of nodes, each comprised of a sensor processor, a world model, and a behavior generation component. Nodes communicate with other nodes, generally in a hierarchical manner, although across-layer connections are allowed. The system supports a wide variety of algorithmic architectures, from reactive behavior to semantic networks. Moreover, it maintains signals, images, and maps, and allows tight coupling between iconic and symbolic representations. The architecture does not generally allow dynamic reconfiguration, but maintains the static module connectivity structure of the specification. RCS has been demonstrated in unmanned ground vehicles [35.59]. Other object-oriented approaches have been reported [35.34, 60].

An early architectural approach which advocated strong programming semantics for multisensor systems is the logical sensor system (LSS). This approach exploits functional (or applicative) language theory to achieve that.

The most developed version of LSS is instrumented LSS (ILLS) [35.22]. The ILLS approach is based on the LSS introduced by *Shilcrat* and *Henderson* [35.61]. The LSS methodology is designed to specify any sensor in a way that hides its physical nature. The main goal behind LSS was to develop a coherent and efficient presentation of the information provided by many sensors of different types. This representation provides a means for recovery from sensor failure, and also facilitates reconfiguration of the sensor system when adding or replacing sensors [35.62].

ILSS is defined as an extension to LSS, and is comprised of the following components (Fig. 35.4):

1. *ILS* name: uniquely identifies a module
2. *Characteristic output vector (COV)*: strongly typed output structure, with one output vector and zero or more input vectors
3. *Commands*: input commands to the module, and output commands to the other modules
4. *Select function*: a selector that detects the failure of an alternate and switches to another alternate if possible
5. *Alternate subnets*: alternative ways of producing the COV_{out}; it is these implementations of one or more algorithms that carry the main functions of the module
6. *Control command interpreter (CCI)*: interpreter of the commands to the module
7. *Embedded tests*: self-testing routines that increase robustness and facilitate debugging
8. *Monitors*: modules that check the validity of the resulting COVs
9. *Taps*: hooks on the output lines to view different COV values.

These components identify the system behavior and provide mechanisms for online monitoring and debugging. In addition, they provide handles for measuring the runtime performance of the system. Monitors are validity check stations that filter the output and alert the user to any undesired results. Each monitor is equipped with a set of rules (or constraints) that governs the behavior of the COV under different conditions.

Embedded testing is used for online checking and debugging purposes. *Weller* et al. proposed a sensor-processing model with the ability to detect measurement errors and to recover from these errors [35.31]. This method is based on providing each system module with verification tests to verify certain characteristics in the measured data, and to verify the internal and output data resulting from the sensor-module algorithm. The recovery strategy is based on rules that are local to the

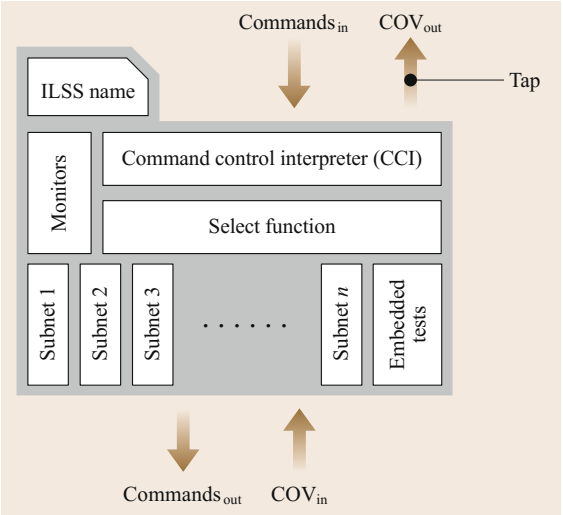


Fig. 35.4 Instrumented logical sensor module

Table 35.1 Canonical components and the roles they play. Multiple ✓ in the same row indicate that some interrole relationships are internalized within a component. *Frame* does not participate in information fusion or decision making but is required for localization and other platform-specific tasks (after [35.21])

Component	Belief			Plan			Action	
Type	Source	Fuse/dist	Sink	Source	Fuse/dist	Sink	Source	Sink
Sensor	✓							
Node		✓			✓			
Actuator								✓
Planner			✓	✓		✓	✓	
UI	✓		✓			✓	✓	
Frame								

different sensor modules. ILSS uses a similar approach called *local embedded testing*, in which each module is equipped with a set of tests based on the semantic definition of that module. These tests generate input data to check different aspects of the module, then examine the output of the module using a set of constraints and rules defined by the semantics. These tests can also take input from other modules to check the operation of a group of modules. Examples are given of a wall-posed estimation system comprised of a Labmate platform with a camera and sonars. Many extensions have been proposed for LSS [35.63, 64].

35.2.5 Decentralized, Local Interaction, and Heterarchical

The best example of this meta-architecture is the active sensor network (ASN) framework for distributed data fusion developed by *Makarenko* et al. [35.21, 65]. The distinguishing features of the various architectures are now described.

Meta-architecture

The distinguishing features of ASN are its commitment to decentralization, modularity, and strictly local interactions (this may be physical or by type). Thus, these are communicating processes. Decentralized means that no component is central to operation of the system, and the communication is peer to peer. Also, there are no central facilities or services (e.g., for communication, name and service lookup or timing). These features lead to a system that is scalable, fault tolerant, and reconfigurable.

Local interactions mean that the number of communication links does not change with the network size. Moreover, the number of messages should also remain constant. This makes the system scalable as well as reconfigurable.

Modularity leads to interoperability derived from interface protocols, reconfigurability, and fault tolerance: failure may be confined to individual modules.

Algorithmic Architecture

There are three main algorithmic components: belief fusion, utility fusion, and policy selection. Belief fusion is achieved by communicating all beliefs to neighboring platforms. A belief is defined as a probability distribution of the world state space.

Utility fusion is handled by separating the individual platform's partial utility into the team utility of belief quality and local utilities of action and communication. The downside is that the potential coupling between individual actions and messages is ignored because the utilities of action and communication remain local.

The communication and action policies are chosen by maximizing expected values. The selected approach is to achieve point maximization for one particular state and follows the work of *Durrant-Whyte* et al. [35.11, 66].

Conceptual Architecture

The data types of the system include:

1. *Beliefs*: Current world beliefs
2. *Plans*: Future planned world beliefs
3. *Actions*: Future planned actions.

The definition of component roles leads to a natural partition of the system.

The information fusion task is achieved through the definition of four component roles for each data type: *source*, *sink*, *fuser*, and *distributor*. (Note that the data type action does not have fuser or distributor component roles.)

Connections between distributors form the backbone of the ASN framework, and the information exchanged is in the form of their local beliefs. Similar considerations are used to determine component roles for the decision-making and system configuration tasks.

Logical Architecture

A detailed architecture specification is determined from the conceptual architecture. It is comprised of

six canonical component types as described in Table 35.1 [35.21].

Makarenko then describes how to combine the components and interfaces to realize the use cases of the problem domain in ASN.

Execution Architecture

The execution architecture traces the mapping of logical components to runtime elements such as processes and shared libraries. The deployment view shows the mapping of physical components onto the nodes of the physical system. The source code view explains how the software implementing the system is organized. At the architectural level, three items are addressed: execution, deployment, and source code organization.

The experimental implementation of the ASN framework has proven to be flexible enough to accommodate a variety of system topologies, platform, and sensor hardware, and environment representations. Several examples are given with a variety of sensors, processors, and hardware platforms.

35.3 Applications

Multisensor fusion systems have been applied to a wide variety of problems in robotics (see the references for this chapter and VIDEO 132, VIDEO 638 and VIDEO 639), but the two most general areas are *dynamic system control* and *environment modeling*. Although there is some overlap between these, they may generally be characterized as:

- *Dynamic system control*: The problem is to use appropriate models and sensors to control the state of a dynamic system (e.g., industrial robot, mobile robot, autonomous vehicle, surgical robot). Usually such systems involve real-time feedback control loops for steering, acceleration, and behavior selection. In addition to state estimation, uncertainty models are required. Sensors may include force/torque sensors, gyros, global positioning system (GPS), position encoders, cameras, range finders, etc..
- *Environment modeling*: The problem is to use appropriate sensors to construct a model of some aspect of the physical environment. This may be a particular object, e.g., a cup, a physical part, a face, or a larger part of the surroundings, e.g., the interior of a building, part of a city or an extended remote or underground area. Typical sensors include cameras, radar, 3-D range finders, infrared (IR), tactile sensors and touch probes (CMMs), etc. The result is usually expressed as geometry (points, lines, surfaces), features (holes, sinks, corners, etc.), or physical properties. Part of the problem includes the determination of optimal sensor placement.

35.3.1 Dynamic System Control

The EMS-Vision system [35.67] is an outstanding exemplar of this application domain. The goal is to develop a robust and reliable perceptual system for autonomous vehicles. The development goals, as stated by the EMS-Vision team, are:

- COTS components
- Wide variety of objects modeled and incorporated into behaviors
- Inertial sensors for ego-state estimation
- Peripheral/foveal/saccadic vision
- Knowledge and goal driven behavior
- State tracking for objects
- 25 Hz real-time update rate.

The approach has been in development since the 1980s. Figure 35.5 shows the first vehicle to drive fully autonomously on the German autobahn for 20 km and at speeds up to 96 km/h.

Information from inertial and vision sensors is combined to produce a road scene tree (Fig. 35.6). A four-

Fig. 35.5 First fully autonomous vehicle on German autobahn

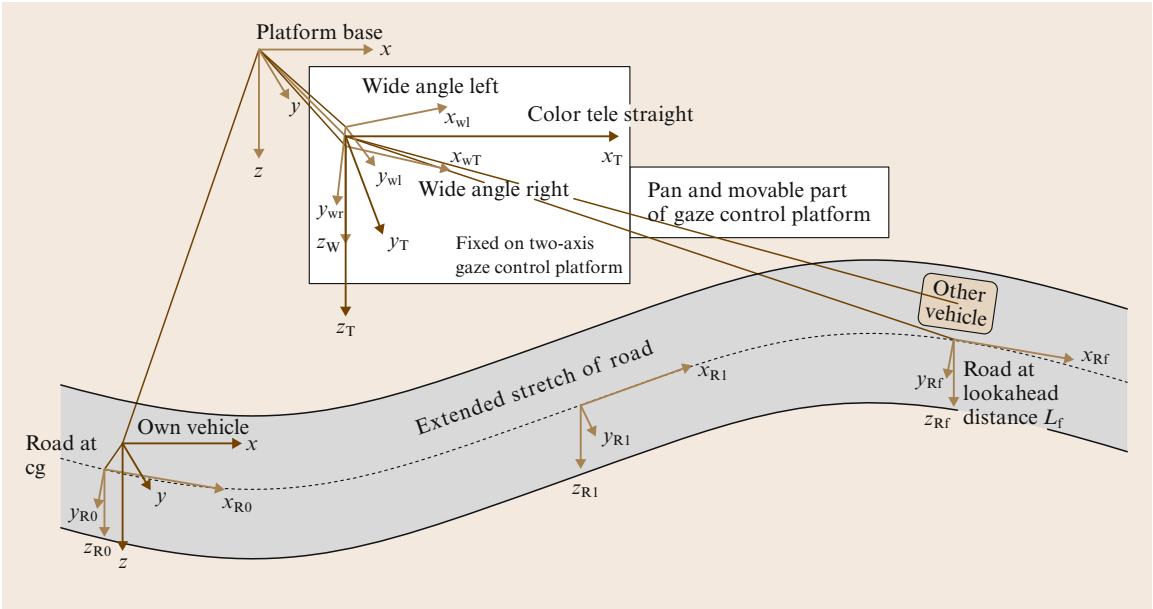


Fig. 35.6 EMS-Vision road scene tree

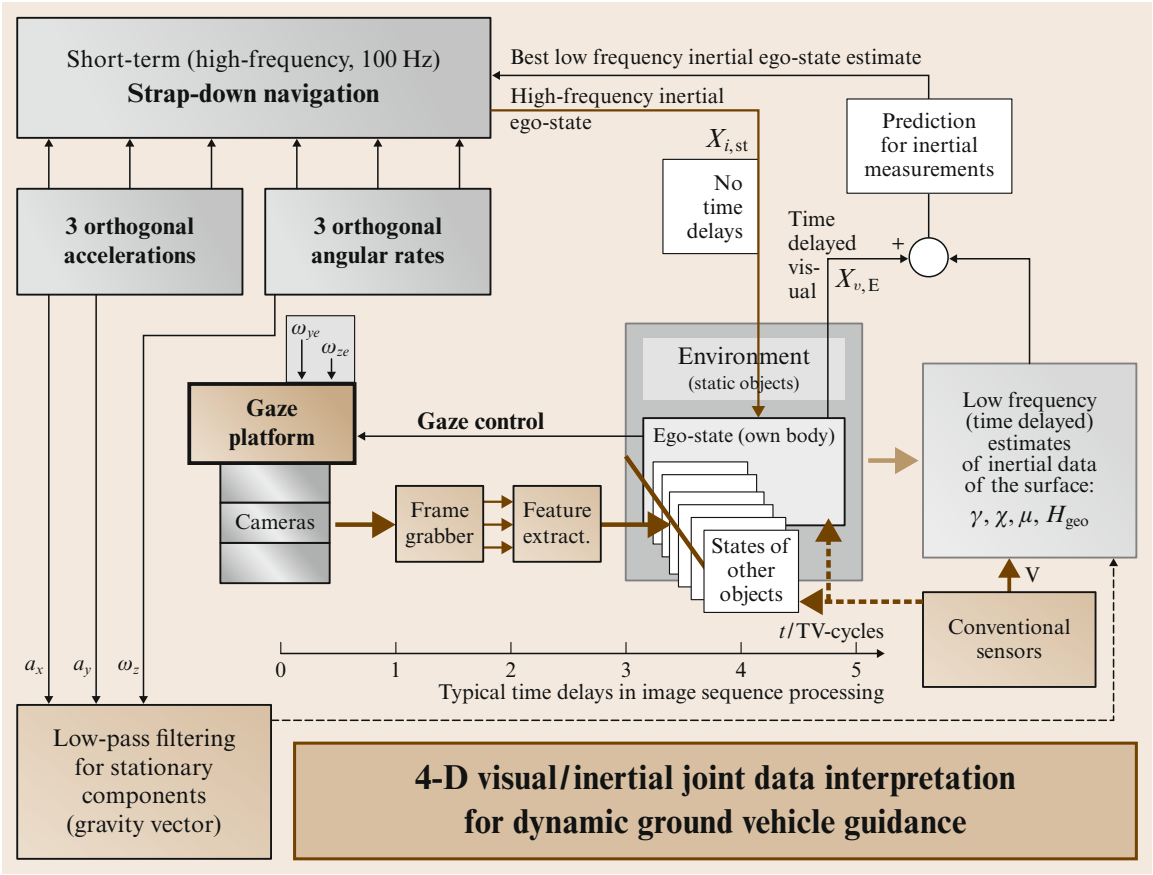


Fig. 35.7 EMS-Vision guidance system

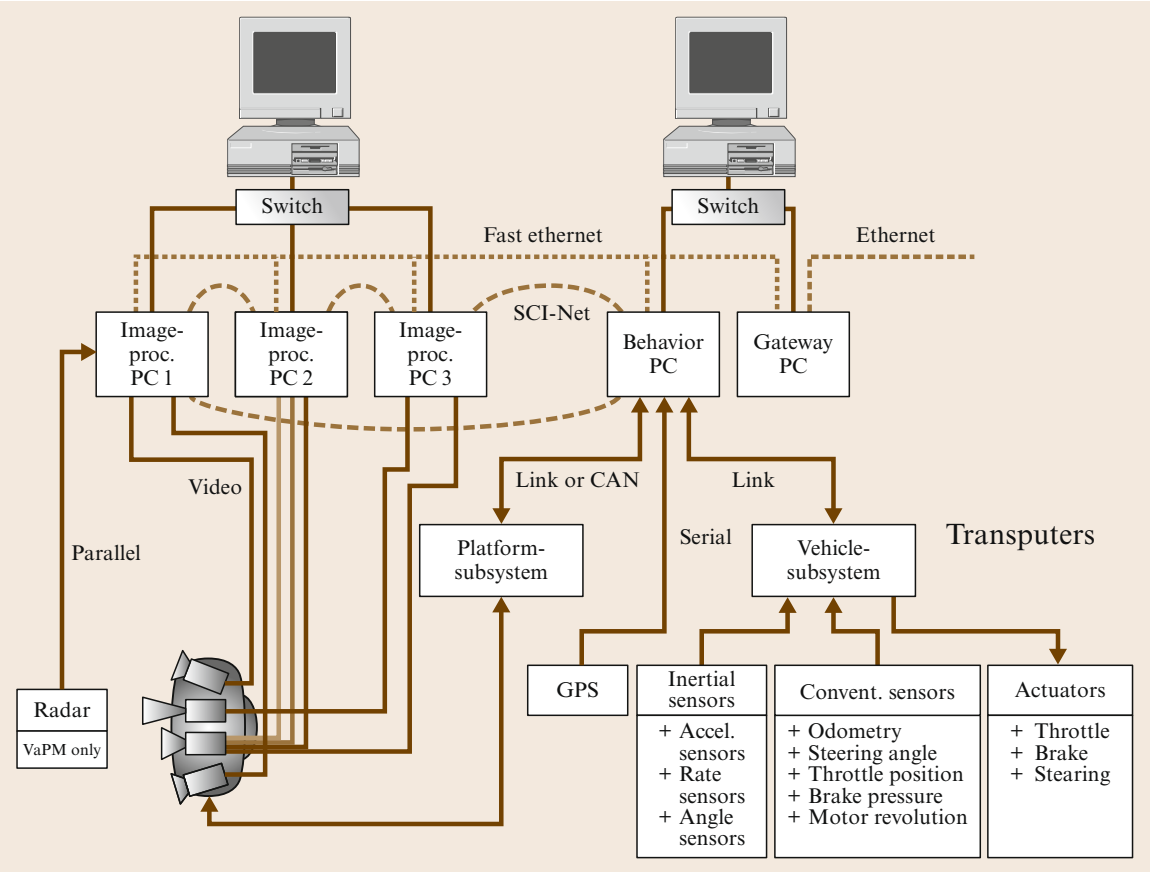


Fig. 35.8 EMS-Vision hardware layout (CAN: controller area network)

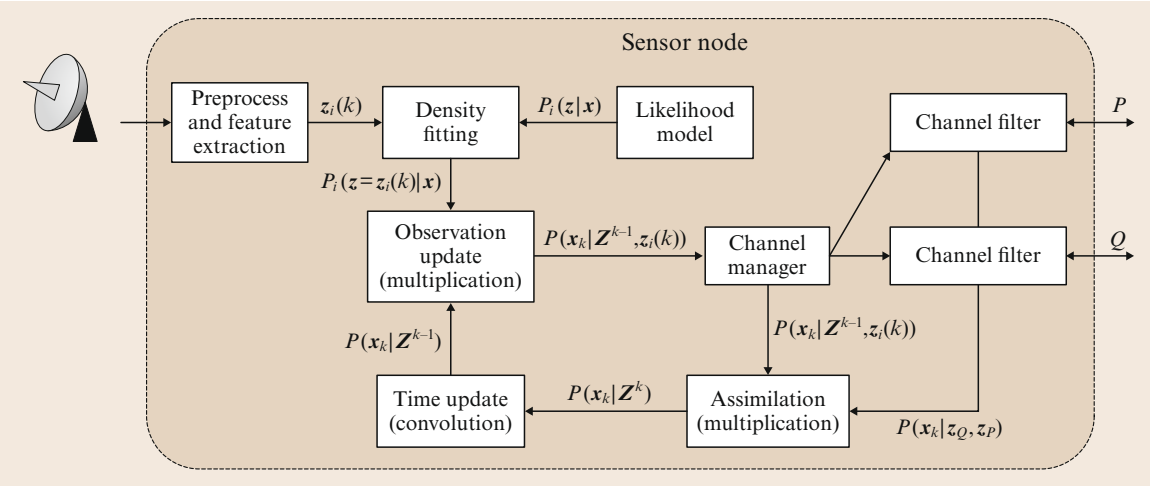


Fig. 35.9 Mathematical structure of a decentralized data fusion node

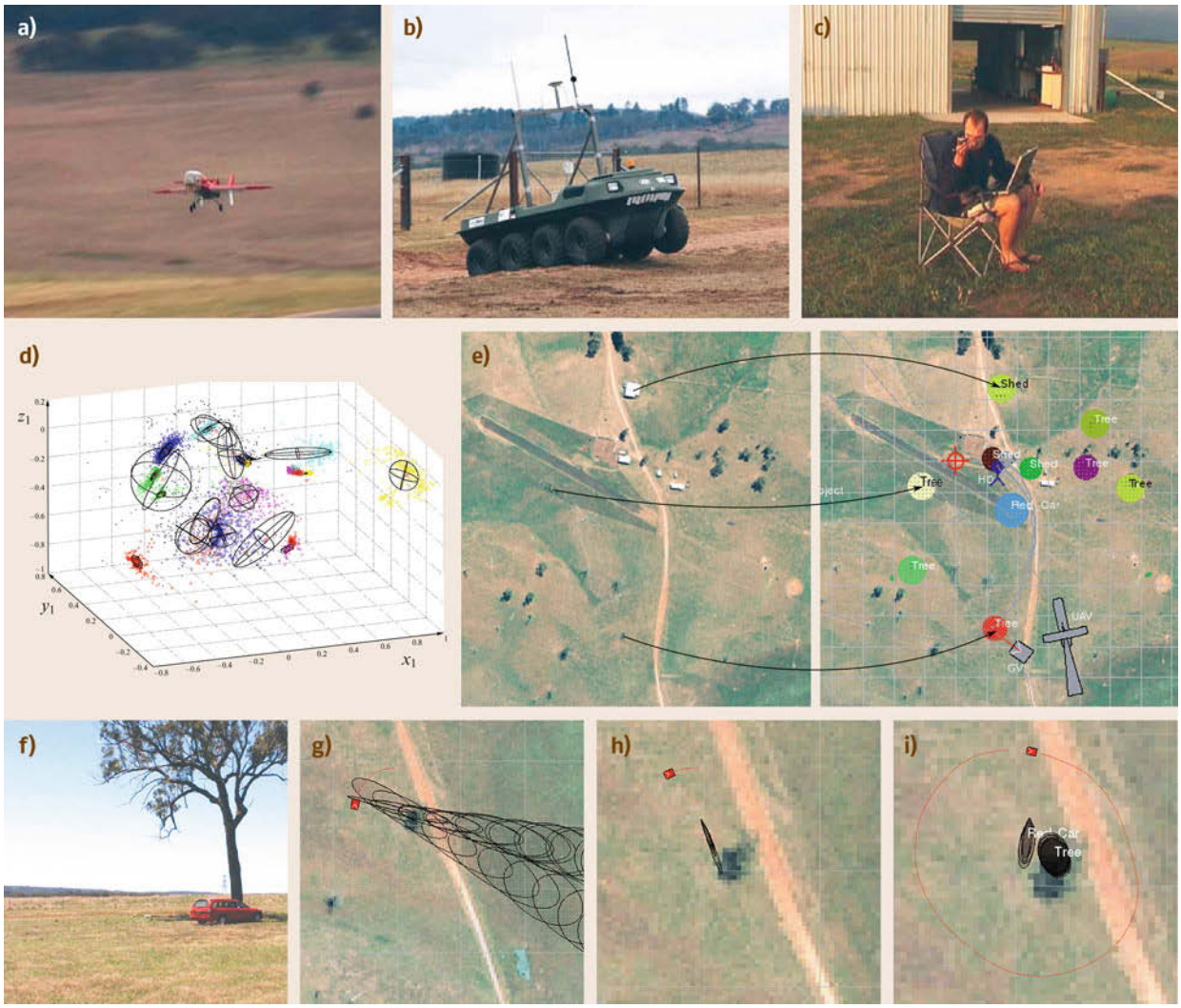


Fig.35.10a–i A synopsis of the ANSER II autonomous network and its operation. **(a–c)** Main system components: **(a)** air vehicle, **(b)** ground vehicle, **(c)** human operative. **(d–e)** The perception process: **(d)** top three dimensions of features discovered from ground-based visual sensor data along with the derived mixture model describing these feature properties, **(e)** sector of the overall picture obtained from fusing air vehicle (UAV), ground vehicle (GV), and human operator (HO) information. Each set of ellipses corresponds to a particular feature and the labels represent the identity state with highest probability. **(f–i)** Sequential fusion process for two close landmarks: **(f)** a tree and a red car, **(g)** bearing-only visual observations of these landmarks are successively fused, **(h)** to determine location and identity **(i)**. Note the Gaussian mixture model for the bearing measurement likelihood

dimensional (4-D) generic object representation is built which includes background knowledge of the object (e.g., roads), its behavioral capabilities, object state and variances, and shape and aspect parameters. Figure 35.7 shows the 4-D inertial/vision multisensor guidance system, while Fig. 35.8 shows the hardware aspects.

In summary, the EMS-Vision system is an interesting and powerful demonstration of multisensor fusion for dynamic system control.

35.3.2 ANSER II: Decentralized Data Fusion

Decentralized data fusion (DDF) methods were initially motivated by the insight that the *information* or canonical form of the conventional Kalman filter data fusion algorithm could be implemented by simply adding information contributions from observations as shown in (35.41). As these (vector and matrix) additions are commutative, the update or data fusion process can

be optimally distributed amongst a network of sensors [35.11, 12, 68]. The aim of the ANSER II project was to generalize the DDF method to deal with non-Gaussian probabilities for observations and states, and to incorporate information from a diversity of sources including uninhabited air and ground vehicles, terrain databases, and human operatives.

The mathematical structure of a DDF sensor node is shown in Fig. 35.9. The sensor is modeled directly in the form of a likelihood function. Once instantiated with an observation, the likelihood function is input to a local fusion loop which implements a local form of the Bayesian time and observation update of (35.7) and (35.8). Network nodes accumulate probabilistic information from observation or communication and exchange mutual information (information gain) with other nodes in the network [35.21]. This mutual information is transmitted to and assimilated by other nodes in the network in an ad hoc manner. The result is that all nodes in the network obtain a single integrated posterior-probability-based all-node observations.

The ANSER II system consists of a pair of autonomous air vehicles equipped with infrared and visual sensors, a pair of unmanned ground vehicles equipped with visual and radar sensors, and additional information provided by geometric and hyperspectral databases, along with information input by human operatives [35.69]. The likelihood functions for single-sensor features are obtained through a semisupervised machine learning method [35.70]. The resulting probabilities are modeled in the form of a mixture of Gaussians. Each platform then maintains a bank of decentralized, non-Gaussian Bayesian filters for the observed features, and transmits this information to all other platforms. The net result is that each platform maintains a complete map of all features observed by all nodes in the network. Multiple observations of the same feature, possibly by different platforms, result in an increasingly accurate estimate of the feature location for all nodes. A corresponding discrete probability measure is used for Fig. 35.10 shows a synopsis of the operation of the ANSER II system.

The ANSER II system demonstrates a number of general principles in Bayesian data fusion methods, specifically the need to model sensors appropriately through the likelihood function, and the possibility of building very different data fusion architectures from the essential Bayesian form.

35.3.3 Recent Developments

There have been recent developments in multisensor fusion methods: see [35.71–73] for more theoretical work, vision and biomedical applications [35.74–77], tracking and terrain classification [35.78–81], and robotics and automation [35.82–87].

35.4 Conclusions

Multisensor data fusion has progressed greatly in the last few decades; further advances in the field will be documented in the robotics and multisensor fusion and integration conference and journal literature. Robust applications are being fielded based on the body of theory and experimental knowledge produced by the research community. Current directions of interest include:

1. Large-scale, ubiquitous sensor systems,
2. Bio-based or biomimetic systems,
3. Medical in situ applications,
4. Wireless sensor networks.

Representative large-scale examples include intelligent vehicle and road systems, as well as instrumented contexts such as cities. Biological principles may provide fundamentally distinct approaches to the exploitation of dense, redundant, correlated, noisy sensors, especially when considered as part of a Gibbsian framework for behavioral response to environmental stimuli. Another issue here is the development of a theoretical understanding of sensor system development, adaptivity, and learning with respect to the particular context in which the system is deployed.

Further pushing the envelope of both technology and theory will permit the introduction of micro- and nanosensors into the human body and allow the monitoring and locally adaptive treatment of various illnesses. Finally, a more-complete theoretical framework that encompasses system models for wireless sensor networks is still required. This should include models of the physical phenomena being monitored, as well as operational and network issues. Finally, numerical analysis of the algorithmic properties of data-driven systems with sensor data error sources must be unified with the analysis of truncation, roundoff, and other errors.

A firm foundation exists upon which to build these new theories, systems, and applications. It will be a vibrant area of research for years to come.

Video-References

VIDEO 132 AnnieWay
available from http://handbookofrobotics.org/view-chapter/35/videodetails/132
VIDEO 638 Application of visual odometry for sewer inspection robots
available from http://handbookofrobotics.org/view-chapter/35/videodetails/638
VIDEO 639 Multisensor remote surface inspection
available from http://handbookofrobotics.org/view-chapter/35/videodetails/639

References

35.1 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)
35.2 J.O. Berger: *Statistical Decision Theory and Bayesian Analysis* (Springer, Berlin, Heidelberg 1985)
35.3 A. Elfes: Sonar-based real-world mapping and navigation, IEEE Trans. Robotics Autom. **3**(3), 249–265 (1987)
35.4 L. Matthies, A. Elfes: Integration of sonar and stereo range data using a grid-based representation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1988) pp. 727–733
35.5 L.D. Stone, C.A. Barlow, T.L. Corwin: *Bayesian Multiple Target Tracking* (Artech House, Norwood 1999)
35.6 Y. Bar-Shalom: *Multi-Target Multi-Sensor Tracking* (Artec House, Norwood 1990)
35.7 Y. Bar-Shalom, T.E. Fortmann: *Tracking and Data Association* (Academic, New York 1988)
35.8 P.S. Maybeck: *Stochastic Models, Estimaton and Control* (Academic, New York 1979)
35.9 W. Sorensen: Special issue on the applications of the Kalman filter, IEEE Trans. Autom. Control **28**(3), 254–255 (1983)
35.10 B.D.O. Anderson, J.B. Moore: *Optimal Filtering* (Prentice Hall, Englewood Cliffs 1979)
35.11 J. Manyika, H.F. Durrant-Whyte: *Data Fusion and Sensor Management: An Information-Theoretic Approach* (Ellis Horwood, New York 1994)
35.12 S. Sukkarieh, E. Nettleton, J.H. Kim, M. Ridley, A. Goktogan, H. Durrant-Whyte: The ANSER project: Data fusion across multiple uninhabited air vehicles, Int. J. Robotics Res. **22**(7), 505–539 (2003)
35.13 R.E. Moore: *Interval Analysis* (Prentice Hall, Englewood Cliffs 1966)
35.14 D. Dubois, H. Prade: *Fuzzy Sets and Systems: Theory and Applications* (Academic, New York 1980)
35.15 S. Blackman, R. Popoli: *Design and Analysis of Modern Tracking Systems* (Artec House, Boston 1999)
35.16 D. Pagac, E.M. Nebot, H. Durrant-Whyte: An evidential approach to map-building for autonomous vehicles, IEEE Trans. Robotics Autom. **14**(4), 623–629 (1998)
35.17 D. Hall, J. Llinas: *Handbook of Multisensor Data Fusion* (CRC, Boca Raton 2001)
35.18 E.L. Waltz, J. Llinas: *Sensor Fusion* (Artec House, Boston 1991)
35.19 M. Kam, Z. Zhu, P. Kalata: Sensor fusion for mobile robot navigation, Proceedings IEEE **85**, 108–119 (1997)
35.20 H. Carvalho, W. Heinzelman, A. Murphy, C. Coelho: A general data fusion architecture, Proc. 6th Int. Conf. Inf. Fusion, Cairns (2003)
35.21 A. Makarenko: A Decentralized Architecture for Active Sensor Networks, Ph.D. Thesis (University of Sydney, Sydney 2004)
35.22 M. Dekhil, T. Henderson: Instrumented logical sensors systems, Int. J. Robotics Res. **17**(4), 402–417 (1998)
35.23 J.A. Profeta: Safety-critical systems built with COTS, Computer **29**(11), 54–60 (1996)
35.24 H. Hu, J.M. Brady, F. Du, P. Probert: Distributed real-time control of a mobile robot, J. Intell. Autom. Soft Comput. **1**(1), 63–83 (1995)
35.25 S.A. Schneider, V. Chen, G. Pardo: ControlShell: A real-time software framework, AIAA Conf. Intell. Robotics Field Fact. Serv. Space (1994)
35.26 D. Simon, B. Espiau, E. Castillo, K. Kapellos: Computer-aided design of a generic robot controller handling reactivity and real-time issues, IEEE Trans. Control Syst. Technol. **4**(1), 213–229 (1993)
35.27 C. Giraud, B. Jouvencel: Sensor selection in a fusion process: a fuzzy approach, Proc. IEEE Int. Conf. Multisens. Fusion Integr., Las Vegas (1994) pp. 599–606
35.28 R. Kapur, T.W. Williams, E.F. Miller: System testing and reliability techniques for avoiding failure, Computer **29**(11), 28–30 (1996)
35.29 K.H. Kim, C. Subbaraman: Fault-tolerant real-time objects, Communication ACM **40**(1), 75–82 (1997)
35.30 D.B. Stewart, P.K. Khosla: Mechanisms for detecting and handling timing errors, Communication ACM **40**(1), 87–93 (1997)
35.31 G. Weller, F. Groen, L. Hertzberger: A sensor processing model incorporating error detection and recovery. In: *Traditional and Non-Traditional Robotic Sensors*, ed. by T. Henderson (Springer, Berlin, Heidelberg 1990) pp. 351–363
35.32 R.R. Brooks, S. Iyengar: *Averaging Algorithm for Multi-Dimensional Redundant Sensor Arrays: Resolving Sensor Inconsistencies*, Tech. Rep. (Louisiana State University, Baton Rouge 1993)
35.33 T.C. Henderson, M. Dekhil: *Visual Target Based Wall Pose Estimation*, Tech. Rep. UUCS-97-010 (University of Utah, Salt Lake City 1997)

35.34 S. Iyengar, D. Jayasimha, D. Nadig: A versatile architecture for the distributed sensor integration problem, Computer **43**, 175–185 (1994)
35.35 D. Nadig, S. Iyengar, D. Jayasimha: A new architecture for distributed sensor integration, Proc. IEEE Southeastcon (1993)
35.36 L. Prasad, S. Iyengar, R.L. Kashyap, R.N. Madan: Functional characterization of fault tolerant integration in distributed sensor networks, IEEE Trans. Syst. Man Cybern. **25**, 1082–1087 (1991)
35.37 L. Prasad, S. Iyengar, R. Rao, R. Kashyap: Fault-tolerence sensor integration using multiresolution decomposition, Am. Phys. Soc. **49**(4), 3452–3461 (1994)
35.38 H.F. Durrant-Whyte: *Integration, Coordination, and Control of Multi-Sensor Robot Systems* (Kluwer, Boston 1987)
35.39 F. Groen, P. Antonissen, G. Weller: Model based robot vision, IEEE Instrum. Meas. Technol. Conf. (1993) pp. 584–588
35.40 R. Joshi, A.C. Sanderson: Model-based multisensor data fusion: A minimal representation approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994)
35.41 A.J. Briggs, B.R. Donald: Automatic sensor configuration for task-directed planning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 1345–1350
35.42 G. Hager: *Task Directed Sensor Fusion and Planning* (Kluwer, Boston 1990)
35.43 G. Hager, M. Mintz: Computational methods for task-directed sensor data fusion and sensor planning, Int. J. Robotics Res. **10**(4), 285–313 (1991)
35.44 B. Donald: On information invariants in robotics, Artif. Intell. **72**, 217–304 (1995)
35.45 V. Braitenberg: *Vehicles: Experiments in Synthetic Psychology* (MIT Press, Cambridge 1984)
35.46 R.A. Brooks: A robust layered control system for a mobile robot, IEEE Trans. Robotics Autom. **2**(1), 14–23 (1986)
35.47 K.P. Valavanis, A.L. Nelson, L. Doitsidis, M. Long, R.R. Murphy: *Validation of a Distributed Field Robot Architecture Integrated with a Matlab Based Control Theoretic Environment: A Case Study of Fuzzy Logic Based Robot Navigation*, CRASAR Tech. Rep. 25 (University of South Florida, Tampa 2004)
35.48 R.R. Murphy: *Introduction to AI Robotics* (MIT Press, Cambridge 2000)
35.49 S. Lee: Sensor fusion and planning with perception-action network, Proc. IEEE Conf. Multisens. Fusion Integr. Intell. Syst., Washington (1996)
35.50 S. Lee, S. Ro: Uncertainty self-management with perception net based geometric data fusion, Proc. IEEE Conf. Robotics Autom. (ICRA), Albuquerque (1997)
35.51 B.A. Draper, A.R. Hanson, S. Buluswar, E.M. Riseman: Information acquisition and fusion in the mobile perception laboratory, Proc. SPIE Sens. Fusion VI (1993)
35.52 S.S. Shafer, A. Stentz, C.E. Thorpe: An architecture for sensor fusion in a mobile robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 2002–2007
35.53 S. Nagata, M. Sekiguchi, K. Asakawa: Mobile robot control by a structured hierarchical neural network, IEEE Control Syst. Mag. **10**(3), 69–76 (1990)
35.54 M. Pachter, P. Chandler: Challenges of autonomous control, IEEE Control Syst. Mag. **18**(4), 92–97 (1998)
35.55 R. Joshi, A.C. Sanderson: *Multisensor Fusion* (World Scientific, Singapore 1999)
35.56 V. Berge-Cherfaoui, B. Vachon: Dynamic configuration of mobile robot perceptual system, Proc. IEEE Conf. Multisens. Fusion Integr. Intell. Syst., Las Vegas (1994)
35.57 V. Clement, G. Giraudon, S. Houzelle, F. Sandakly: *Interpretation of Remotely Sensed Images in a Context of Multisensor Fusion Using a Multi-Specialist Architecture*, Rapp. Rech.: No. 1768 (INRIA, Sophia-Antipolis 1992)
35.58 J. Albus: RCS: A cognitive architecture for intelligent multi-agent systems, Proc. IFAC Symp. Intell. Auton. Veh., Lisbon (2004)
35.59 R. Camden, B. Bodt, S. Schipani, J. Bornstein, R. Phelps, T. Runyon, F. French: *Autonomous Mobility Technology Assessment*, Interim Rep., ARL-MR 565 (Army Research Laboratory, Washington 2003)
35.60 T. Queeney, E. Woods: A generic architecture for real-time multisensor fusion tracking algorithm development and evaluation, Proc. SPIE Sens. Fusion VII, Vol. 2355 (1994) pp. 33–42
35.61 T. Henderson, E. Shilcrat: Logical sensor systems, J. Robotics Syst. **1**(2), 169–193 (1984)
35.62 T. Henderson, C. Hansen, B. Bhanu: The specification of distributed sensing and control, J. Robotics Syst. **2**(4), 387–396 (1985)
35.63 J.D. Elliott: Multisensor Fusion within an Encapsulated Logical Device Architecture, Master's Thesis (University of Waterloo, Waterloo 2001)
35.64 M.D. Naish: Elsa: An Intelligent Multisensor Integration Architecture for Industrial Grading Tasks, Master's Thesis (University of Western Ontario, London 1998)
35.65 A. Makarenko, A. Brooks, S. Williams, H. Durrant-Whyte, B. Grocholsky: A decentralized architecture for active sensor networks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 1097–1102
35.66 B. Grocholsky, A. Makarenko, H. Durrant-Whyte: Information-theoretic coordinated control of multiple sensor platforms, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Taipei (2003) pp. 1521–1527
35.67 R. Gregor, M. Lützeler, M. Pellkofer, K.-H. Siedersberger, E. Dickmanns: EMS-Vision: A perceptual system for autonomous vehicles, IEEE Trans. Intell. Transp. Syst. **3**(1), 48–59 (2002)
35.68 B. Rao, H. Durrant-Whyte, A. Sheen: A fully decentralized multi-sensor system for tracking and surveillance, Int. J. Robotics Res. **12**(1), 20–44 (1993)
35.69 B. Upcroft: Non-gaussian state estimation in an outdoor decentralised sensor network, Proc. IEEE Conf. Decis. Control (CDC) (2006)
35.70 S. Kumar, F. Ramos, B. Upcroft, H. Durrant-Whyte: A statistical framework for natural feature repre-

sentation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Edmonton (2005) pp. 1–6
35.71 S. Gao, Y. Zhong, W. Li: Random weighting method for multisensor data fusion, IEEE Sens. J. **11**(9), 1955–1961 (2011)
35.72 M. Lhuillier: Incremental fusion of structure-from-motion and GPS using constrained bundle adjustments, IEEE Trans. Pattern Anal. Mach. Intell. **34**(12), 2489–2495 (2012)
35.73 S. Yu, L. Tranchevent, X. Liu, W. Glanzel, J. Suykens, B. DeMoor, Y. Moreau: Optimized data fusion for kernel k-means clustering, IEEE Trans. Pattern Anal. Mach. Intell. **34**(5), 1031–1039 (2012)
35.74 K. Kolev: Fast joint estimation of silhouettes and dense 3D geometry from multiple images, IEEE Trans. Pattern Anal. Mach. Intell. **34**(3), 493–505 (2012)
35.75 C. Loy: Incremental activity modeling in multiple disjoint cameras, IEEE Trans. Pattern Anal. Mach. Intell. **34**(9), 1799–1813 (2012)
35.76 N. Poh, J. Kittler: A unified framework for biometric expert fusion incorporating quality measures, IEEE Trans. Pattern Anal. Mach. Intell. **34**(1), 3–18 (2012)
35.77 M.-F. Weng, Y.-Y. Chuang: Cross-domain multicue fusion for concept-based video indexing, IEEE Trans. Pattern Anal. Mach. Intell. **34**(10), 1927–1941 (2012)
35.78 M. Hwangbo, J.-S. Kim, T. Kanade: Gyro-aided feature tracking for a moving camera: fusion, auto-calibration and GPU implementation, Intl. J. Robotics Res. **30**(14), 1755–1774 (2011)
35.79 H. Seraji, N. Serrano: A multisensor decision fusion system for terrain safety assessment, IEEE Trans. Robotics **25**(1), 99–108 (2009)
35.80 H. Himberg, Y. Motai, A. Bradley: Interpolation volume calibration: a multisensor calibration technique for electromagnetic trackers, IEEE Trans. Robotics **28**(5), 1120–1130 (2012)
35.81 K. Zhou, S.I. Roumeliotis: Optimal motion strategies for range-Only constrained multisensor target tracking, IEEE Trans. Robotics **24**(5), 1168–1185 (2008)
35.82 N.R. Ahmed, E.M. Sample, M. Campbell: Bayesian multicategorical soft data fusion for human–robot collaboration, IEEE Trans. Robotics **PP**(99), 1–18 (2012)
35.83 H. Frigui, L. Zhang, P.D. Gader: Context-dependent multisensor fusion and its application to land mine detection, IEEE Trans. Geosci. Remote Sens. **48**(6), 2528–2543 (2010)
35.84 J.G. Garcia, A. Robertson, J.G. Ortega, R. Johansson: Sensor fusion for compliant robot motion control, IEEE Trans. Robotics **24**(2), 430–441 (2008)
35.85 R. Heliot, B. Espiau: Multisensor input for CPG-based sensory–motor coordination, IEEE Trans. Robotics **24**(1), 191–195 (2008)
35.86 S. Liu, R.X. Gao, D. John, J.W. Staudenmayer, P.S. Freedson: Multisensor data fusion for physical activity assessment, IEEE Trans. Bio-Med. Eng. **59**(3), 687–696 (2012)
35.87 A. Martinelli: Vision and IMU data fusion: closed-Form solutions for attitude, speed, absolute scale, and bias determination, IEEE Trans. Robotics **28**(1), 44–60 (2012)

Multimedia Contents

Part D Manipulation

Part D Manipulation and Interfaces

Ed. by Makoto Kaneko

36 Motion for Manipulation Tasks
James Kuffner, Pittsburgh, USA
Jing Xiao, Charlotte, USA

37 Contact Modeling and Manipulation
Imin Kao, Stony Brook, USA
Kevin M. Lynch, Evanston, USA
Joel W. Burdick, Pasadena, USA

38 Grasping
Domenico Prattichizzo, Siena, Italy
Jeffrey C. Trinkle, Troy, USA

39 Cooperative Manipulation
Fabrizio Caccavale, Potenza, Italy
Masaru Uchiyama, Sendai, Japan

40 Mobility and Manipulation
Oliver Brock, Berlin, Germany
Jaeheung Park, Suwon, Korea
Marc Toussaint, Stuttgart, Germany

41 Active Manipulation for Perception
Anna Petrovskaya, Stanford, USA
Kaijen Hsiao, Palo Alto, USA

42 Haptics
Blake Hannaford, Seattle, USA
Allison M. Okamura, Stanford, USA

43 Telerobotics
Günter Niemeyer, Glendale, USA
Carsten Preusche, Wessling, Germany
Stefano Stramigioli, Enschede, The Netherlands
Dongjun Lee, Seoul, Korea

44 Networked Robots
Dezhen Song, College Station, USA
Ken Goldberg, Berkeley, USA
Nak-Young Chong, Ishikawa, Japan

Part D, Manipulation and Interfaces, is separated into two subparts; the first half is concerned with manipulation where frameworks of modeling, motion planning, and control of grasp and manipulation of an object are addressed, and the second half is concerned with interfaces where physical human–robot interactions are handled. Humans can achieve grasping and manipulation of an object dexterously through hand–arm coordination. An optimum control skill for such a redundant system is naturally and gradually acquired through experience in our daily life. Especially, fingers play an important role for expressing human dexterity. Without dexterous fingers, it is hard for us to handle any daily tool, such as a pencil, keyboard, cup, knife, or fork. This dexterity is supported with active and passive compliance as well as the multiple sensory organs existing at the fingertip. Such dexterous manipulation enables us to clearly differentiate humans from other animals. Thus, manipulation is one of the most important functions for humans. We eventually acquired the current shape of finger, the sensory organs, and skill for manipulation, through a long history of evolution, over more than six million years. While humans and robots are largely different in terms of actuators, sensors, and mechanisms, achieving dexterous manipulation like that of a human in a robot is a challenging subject in robotics. As we overview current robot technology, however, we observe that the dexterity of robots is still far behind that of humans. With this overview, we now provide a brief synopsis of each chapter in the first half of Part D.

Chapter 36, Motion for Manipulation Tasks, discusses algorithms that generate motion for manipulation tasks at the arm level, especially in an environment, by using the configuration space formalism. While in previous chapters (6 and 7) the focus was on specific algorithmic techniques for robot motion, this chapter is focused on a specific application for robot manipulation. The important example of assembly motion is discussed through the analysis of contact states and compliant motion control.

Chapter 37, Contact Modeling and Manipulation, provides the contact modeling on rigid contact with and without friction, and also the modeling on soft contact, such as elastic and viscoelastic contact interfaces. Kinematics and mechanics with friction are precisely handled under rigid-body contact. The selection matrix $\mathbf{H}$ is introduced to understand the force and velocity constraints at the contact interface. Pushing manipulation is also addressed by using the concept of the friction limit surface.

Chapter 38, Grasping, discusses based on the closure property, grasping with many examples, supposing multifingered robotic hands. A strong constraint for grasping is the unilateral characteristic,where a fingertip can push but not pull an object through a contact point. The rigid-body model is further extended for considering compliance in grasping with robotic hands having a low number of degrees of freedom (DOFs). Kinematics and closure issues are also addressed under this unilateral constraint.

Chapter 39, Cooperative Manipulators, addresses the strategies for controlling both the motion of cooperative system and the interaction forces between the manipulators and the grasped object when two manipulator arms firmly grasp a common object. It should be noted that this chapter allows the bilateral constraint where both directional force and moment are permissible. A general cooperative task-space formulation is introduced for different classes of multiarm robotic systems.

Chapter 40, Mobile Manipulation, focuses on research conducted on an experimental platform that combines capabilities in mobility and manipulation. Furthermore, it involves the interaction between the robots and real-world environment, unstructured environments. The main research objective here is to maximize task generality of autonomous robotic systems, under minimizing the dependence on task-specific, hard-coded, or narrowly-relevant information.

Chapter 41, Active Manipulation for Perception, covers perceptual methods in which manipulation is an integral part of perception. There are advantages to use manipulation rather than vision for perception. For example, manipulation can be used to sense in poor-visibility conditions, and also to determine properties that require physical interaction. This chapter includes the methods that have been developed for inference, planning, recognition, and modeling in sensing-via-manipulation approaches are cover.

Without dexterity like that of humans, future robots will not be able to work instead of humans in environments where human cannot enter. In this sense, the implementation of dexterity into robots is one of the highlights of future robot design. Chapters 36–41 provide a good hint for enhancing dexterity for robots.

The second half of Part D addresses interfaces where humans control a robot or multiple robots through direct or indirect contact with robot(s).We now

provide a brief synopsis of each chapter in the second half of Part D.

Chapter 42, Haptics, discusses robotics devices that allow human operators to experience the sense of touch in remote or virtual environment. Two classes of force feedback haptic devices are discussed. One is an admittance device that senses the force applied by the operator and constrains the operator's position to match the appropriate deflection of a simulated object or surface; the other is an impedance haptic device that senses the position of the operator and then applies a force vector to the operator according to the computed behavior of the simulated object or surface. Haptic rendering from real time three-dimensional (3-D) image is also introduced in this chapter.

Chapter 43, Telerobotics, starts with a discussion on the classification of three different concepts: direct control where all slave motions are directly controlled by the user via the master interface, shared control where task execution is shared between direct control and local sensory control, and supervisory control where the user and slave are connected loosely with strong local autonomy. Various control issues such as lossy communication with Internet and operation with mobile robots, are also addressed.

Chapter 44, Networked Robots, focuses on the framework of computer networks which offer extensive computing, memory, and other resources that can dramatically improve performance. The chapter covers a wide span from the history of networked robots as it evolves from teleoperation to cloud robotics to how to build a networked robot. The very recent progress on cloud robotics and potential topics for future research are included later in the chapter.

In Haptics (Chap. 42) direct contact between human and robot is made, while in both Telerobotics (Chap. 43) and Networked Robots (Chap. 44), an appropriate distance between the human and robot is kept. One of the main issues is how to maintain appropriate control performance of the system in the presence of humans or a time lag between humans and robots.

Multimedia Contents

36. Motion for Manipulation Tasks

James Kuffner, Jing Xiao

This chapter serves as an introduction to Part D by giving an overview of motion generation and control strategies in the context of robotic manipulation tasks. Automatic control ranging from the abstract, high-level task specification down to fine-grained feedback at the task interface are considered. Some of the important issues include modeling of the interfaces between the robot and the environment at the different time scales of motion and incorporating sensing and feedback. Manipulation planning is introduced as an extension to the basic motion planning problem, which can be modeled as a hybrid system of continuous configuration spaces arising from the act of grasping and moving parts in the environment. The important example of assembly motion is discussed through the analysis of contact states and compliant motion control. Finally, methods aimed at integrating global planning with state feedback control are summarized.

36.1 **Overview** 898

36.2 **Task-Level Control** 900
- 36.2.1 Operational Space Control 900
- 36.2.2 Combined Force and Position Control 901
- 36.2.3 Operational Space Control of Redundant Mechanisms 901
- 36.2.4 Combining Mobility and Manipulation 902
- 36.2.5 Combining Multiple Task Behaviors 902

36.3 **Manipulation Planning** 904
- 36.3.1 Configuration Space Formalism . 904
- 36.3.2 Example of a Three-DOF Planar Manipulator 907
- 36.3.3 Inverse Kinematics Considerations 908
- 36.3.4 Continuum Manipulation 910

36.4 **Assembly Motion** 911
- 36.4.1 Topological Contact States 912
- 36.4.2 Passive Compliance 912
- 36.4.3 Active Compliant Motion 914
- 36.4.4 Compliant Motion Under Manipulator Constraints .. 917

36.5 **Unifying Feedback Control and Planning** 918
- 36.5.1 Feedback Motion Planning 918
- 36.5.2 Augmenting Global Plans with Feedback 920

36.6 **Conclusions and Further Reading** 920
- 36.6.1 General Contact Models 921
- 36.6.2 Cooperative Manipulation Control 921
- 36.6.3 Control of Branching Mechanisms 921
- 36.6.4 Nonholonomic Mobile Manipulation 921
- 36.6.5 Learning Models with Uncertainty 921
- 36.6.6 Grasping and Regrasp Planning 921
- 36.6.7 Multiple Parts 922
- 36.6.8 Planning for Multiple Robots 922
- 36.6.9 Planning for Closed Kinematic Chains 922
- 36.6.10 Planning with Movable Obstacles 922
- 36.6.11 Nonprehensile Manipulation 922
- 36.6.12 Assembly Motion Extensions 922

Video-References 923

References 923

36.1 Overview

Part D of this handbook is concerned with the interfaces that connect robots to their environment. We differentiate three such interfaces, depicted in Fig. 36.1. The first interface, between the robot and the computer, is primarily concerned with the automatic generation of motion for performing a task. The second interface is concerned with the physical interaction between the robot and the environment. This chapter relates both interfaces in the context of manipulation tasks. It is focused on automatic specification, planning, and execution of motion of the robot manipulator or the manipulated object to meet a task goal under the constraints of the physical environment. Chapters 37–39, 40, and 42 address other important issues pertaining to the second interface, such as the physical characteristics of contacts between the robot and the environment, grasping and active manipulation (i. e., the interaction between the robot and the object for manipulation), the interaction between cooperative manipulators, and the interaction between mobility and manipulation. The third interface, described in Chaps. 41, 43, and 44, lies between humans and robots. Key issues that need to be addressed involve displaying appropriate sensor information to humans, or permitting humans to interactively specify the task or motion a robot is supposed to perform.

Manipulation refers to the process of moving or rearranging objects in the environment [36.1]. To perform a *manipulation task*, a robot establishes physical contact with objects in the environment and subsequently moves these objects by exerting forces and moments. The object to be manipulated can be large or small, and may serve various purposes in relation to the manipulation task (e.g., flipping a switch, opening a door, polishing a surface). In the context of automated assembly and industrial manipulation, the object to be manipulated is often referred to as a *part*. The *end-effector* generally refers to the link of the *manipulator* that makes contact and applies forces to the *part*. The *end-effector* is so named as it is often the end link in a serial kinematic chain. However, complex manipulation tasks may require the simultaneous exertion of multiple forces and moments at different contact points as well as the execution of sequences of force applications to several objects.

To understand the challenge of generating the motion required for the execution of a manipulation task, we will consider the classic example of inserting a peg into a hole. The mobile manipulator shown in Fig. 36.2 is commanded to perform the peg insertion *manipulation task*, whose general problem structure arises in a number of contexts such as compliant assembly. For simplicity, we will only consider the *transfer motion* (Sect. 36.3) and assume that the robot has already established a stable grasp (Chap. 38) and is holding the peg. If the clearance between the peg and the hole is very small, the peg may not be able to enter the hole easily, and often some contact occurs. In order to guide the peg successfully into the hole, the robot has to deal with the possible contact states between the peg and the hole (Sect. 36.4) and select an appropriate sequence of control strategies that will result in successful completion of the task.

The peg-and-hole example illustrates that successful robotic manipulation involves dealing with models of object geometry and contact in the presence of uncertainty. There are various kinds of uncertainty, including modeling, actuation, and sensing uncertainty. To plan motion strategies in the presence of uncertainty poses difficult computational challenges. Historically, motion planning for manipulation has been divided into *gross motion planning* and *fine motion planning*. The former involves reasoning about the manipulator motion on the macro scale and considers its overall global movement

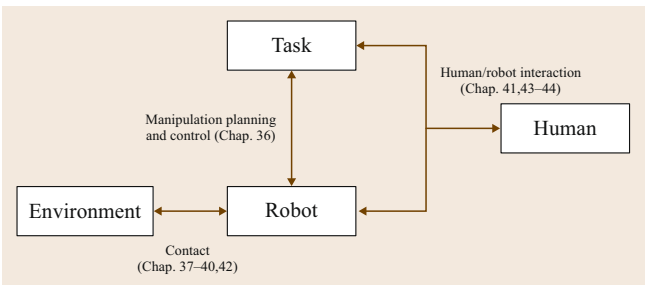


Fig. 36.1 Overview of Part D

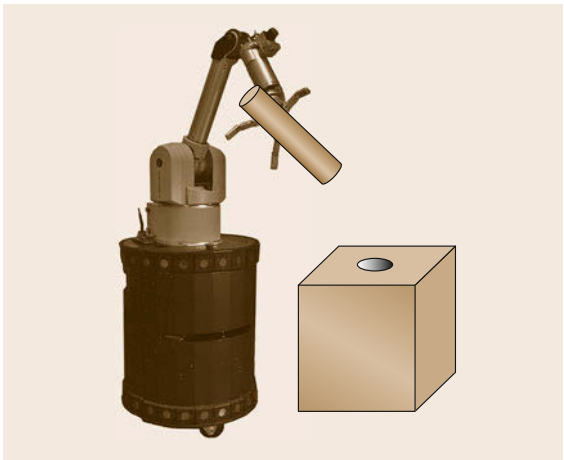


Fig. 36.2 A mobile manipulator inserting a peg into a hole

strategy, while the latter considers how to deal with uncertainty to accomplish a task requiring high precision robustly. The primary reason for the historical division stems from attempts to simplify the various aspects of the problem into something computationally tractable. Assuming that the uncertainty in actuator position, object models, and environment obstacle shapes can be bounded, gross motion planning can be formulated primarily in terms of geometry (Sect. 36.3). As the robot or the manipulated object makes contact with the environment, such as in the example of the insertion task, fine motion planning techniques are employed to take into account contact geometry, forces, friction, and uncertainty effectively (see *Mason*'s book [36.1] and [36.2, 3] for an overview and historical perspective on fine motion strategies for manipulation tasks).

This peg-and-hole insertion example also illustrates that motion in the context of manipulation tasks is subject to *constraints*. These constraints have to be maintained to perform the task successfully. The specific constraints depend on the type of manipulation task, but they may include contact constraints, position, and force constraints for the point at which the manipulator makes contact with the environment, kinematic and dynamic constraints imposed by the mechanism and its actuation capabilities, posture constraints that specify behavior to be performed in addition to the manipulation task, reactive obstacle avoidance in an unpredictably changing environment, and global motion constraints to ensure that a specific goal location is attained. These motion constraints are imposed by the task, by the kinematics of the mechanism performing the task, by the actuation capabilities of the mechanism, and by the environment.

To ensure that constraints are satisfied in spite of uncertainty, sensory information (see Part C – Sensing and Perception) is considered in the context of feedback control loops (Chaps. 8 and 47). Depending on the type of constraint, this feedback has to be considered at various time scales. For example, the exertion of a constant force on an object in the environment requires high-frequency feedback at rates up to 1000 Hz. At the other end of the time scale, we consider changes to the global connectivity of the environment. These changes, resulting from opening doors or moving obstacles, for example, occur relatively slowly or infrequently and feedback about these changes only has to be considered a few times per second. Figure 36.3 graphically illustrates a task-dependent ordering of motion constraints encountered in manipulation tasks and their associated feedback requirements.

This chapter is concerned with algorithms that generate motion for manipulation tasks. Previous chapters already discussed planning algorithms (Chap. 7 and control methods (Chap. 8). While in these two former chapters the focus was on specific algorithmic techniques for robot motion, the current chapter is focused on a specific application, namely robotic manipulation. This application dictates motion constraints and their feedback requirements that will have to be satisfied during the motion for a manipulation task. In contrast, the methods discussed in the two aforementioned chapters each only address a subset of the motion constraints, as indicated in Fig. 36.3.

This chapter first discusses task-level control in Sect. 36.2. Task-level control describes a set of techniques to control a robotic mechanism in terms of its contact points with the environment. These contact points are controlled to perform motion that accomplishes a specific manipulation task. This means that, instead of directly controlling the mechanism, task-relevant points on the mechanism, so-called *operational points*, are controlled. This indirect control provides an intuitive means to specify desired motion for manipulation tasks.

Section 36.3 gives an overview of the configuration space formalization for manipulation planning, and how it can be cast as a classical motion planning problem (Chap. 7). Through this formalism, we can gain an understanding and intuition regarding how the geometry and relative positions of the manipulator, the manipulated objects, and the obstacles in the workspace determine both the geometry and topological structure of the manipulation configuration space. As we shall see, there are often an infinite number of possible ways to accomplish a manipulation task. Thus, the challenge in manipulation planning is to develop methods that deal effectively with the combinatorics of the search over all possible motions.

Section 36.4 addresses planning motion for assembly tasks, for which the peg-in-hole insertion is a typical example. The focus is on motion constrained by contact, called *compliant motion*. There are two broad

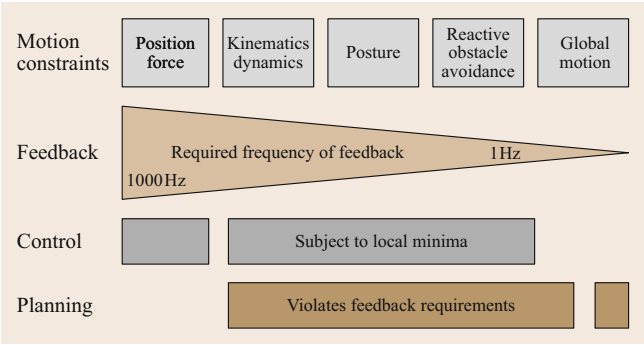


Fig. 36.3 Examples of motion constraints encountered in manipulation tasks and their feedback requirements

classes of assembly strategies involving compliant motion; one class is through special mechanism or control, called *passive compliance*, and the other is through active reasoning of contact states, called *active compliant motion*. Fine motion planning is discussed within the class of active compliant motion.

Methods from task-level control and manipulation planning address aspects of motion in a complementary fashion. While control methods are able to satisfy the feedback requirements of manipulation tasks, they often do not account for the overall global progress in accomplishing the manipulation task. Manipulation planners, on the other hand, reason about the global nature of the task at the high level, but are often computationally too expensive to deal with uncertainty or satisfy the high-frequency feedback requirements of manipulation tasks. A robust and skillful manipulator must satisfy all feedback requirements while at the same time ensuring the accomplishment of the manipulation task. Section 36.5 discusses various techniques that offer a unified view of planning and control for manipulation tasks.

36.2 Task-Level Control

To perform a manipulation task, a robot has to establish contact with the environment. Through contact points the robot is able to apply forces and moments to objects. By controlling the position and velocity of the contact points as well as the forces acting at them, the robot causes the desired motion of objects, thereby performing the manipulation task. The programming of such a task is most conveniently accomplished by directly specifying positions, velocities, and forces at the contact points, rather than by specifying the joint positions and velocities required to achieve them.

Consider the manipulation task of serving a cup of water shown in Fig. 36.2. This task can easily be specified by providing a trajectory for the cup's motion. To determine a joint space trajectory that achieves the same motion, however, would be much more complex. One would have to rely on inverse kinematics (Sect. 2.9), which can be computationally challenging. More importantly, the task constraints imposed on the cup's motion do not uniquely specify the cup's trajectory. For example, while the cup is not allowed to tilt, it can be delivered to the goal location in any vertical orientation. Such task constraints can easily be specified in terms of the object motion – at the task level – but would be very difficult to characterize in terms of joint trajectories. Operational space control is therefore a natural choice for performing manipulation tasks. We will see in Sect. 36.2.3 that the operational space framework provides important additional advantages over joint space control in the context of redundant manipulators.

The advantages associated with task-level control come at the cost of depending on the manipulator Jacobian. For singular configurations of the manipulator (Chap. 4), the manipulator Jacobian is not well defined. In these configurations, task-level control of a manipulator becomes unstable. Special care has to be applied to prevent the manipulator from entering these configurations, or to resort to specialized controllers in the proximity of these configurations. Most commonly, the manipulator is controlled so as to avoid singular configurations.

36.2.1 Operational Space Control

Task-level control – also called operational space control [36.4, 5] (Sect. 8.2) – specifies the behavior of a robot in terms of operational points rather than joint positions and velocities. An operational point is an arbitrary point on the robot that is required to perform a particular motion or to exert a specified force to accomplish a manipulation task. In a serial-chain manipulator arm the operational point is most commonly chosen to coincide with the end-effector. More complex, branching mechanisms, such as humanoid robots, but also redundant serial-chain mechanisms, can have several operational points. To specify positions, velocities, accelerations, forces, and moments at the operational point, it is convenient to define a coordinate system with an origin that coincides with the operation point. This coordinate frame is called the operational frame. The orientation of the frame should be chosen in accordance with the task. For now we will ignore the orientation, however, and only consider tasks that require positioning of the operational point. In Sect. 36.2.2, we will consider more general tasks, combining position and force control.

Let us consider an operational point $\boldsymbol{x}$ on a robot. The relationship between the joint velocities $\dot{\boldsymbol{q}}$ and the velocity of the operational point $\dot{\boldsymbol{x}}$ is given by the Jacobian matrix of the mechanism (Sect. 2.8) at the operational point $\boldsymbol{x}$

$$\dot{\boldsymbol{x}} = \mathbf{J}_x(\boldsymbol{q})\dot{\boldsymbol{q}} \,. \tag{36.1}$$

Note that the Jacobian $\mathbf{J}$ depends on the placement of the operational point $\boldsymbol{x}$, and on the current configura-

tion $\boldsymbol{q}$ of the robot. For simplicity, we will omit this dependency from our notation and just write $\dot{\boldsymbol{x}} = \mathbf{J}\dot{\boldsymbol{q}}$. From (36.1) we can derive an expression for the instantaneous torques $\boldsymbol{\tau}$ acting at the joints when the forces and moments described by $\boldsymbol{F}$ is applied at the operational point $\boldsymbol{x}$

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{F}_{\boldsymbol{x}} \,. \tag{36.2}$$

The vector $\boldsymbol{F}$ captures the task to be performed at operational point $\boldsymbol{x}$. If the task specifies the complete position and orientation of the operational frame, $\boldsymbol{F}$ will be given by $\boldsymbol{F} = (f_x, f_y, f_z, u_x, u_y, u_z)$, where f and u designate forces along and moments about the respective axes. (For easy of presentation, we will refer to $\boldsymbol{F}$ as a force vector, even if it describes forces and moments. The execution of such a task requires a manipulator with at least six degrees of freedom. If a task does not specify one of the components of $\boldsymbol{F}$, this component can be omitted and the corresponding column of the Jacobian matrix is dropped (or a manipulator with fewer than six degrees of freedom can be used to accomplish the task). For example, dropping u_z would indicate that the orientation about the z-axis is not specified by the task. The motion behavior in the dimensions of the task space that have been dropped is unspecified and the robot will float in these dimensions [36.4]. In Sect. 36.2.3 we will discuss how additional behavior can be performed in these unspecified dimensions of the task space.

Similarly to joint space control, we have to account for manipulator dynamics to achieve acceptable performance in operational space control. Using (36.1), we can project the joint space dynamics into operational space to obtain

$$\boldsymbol{F}_{\boldsymbol{x}} = \boldsymbol{\Lambda}(\boldsymbol{q}\ldots\ddot{\boldsymbol{x}} + \boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{\eta}(\boldsymbol{q}) \,, \tag{36.3}$$

where $\boldsymbol{\Lambda}$ is the operational space inertial matrix, $\boldsymbol{\Gamma}$ captures centrifugal and Coriolis forces in operational space, and $\boldsymbol{\eta}$ compensates for gravity forces. This equation pertains to a particular operational point. Intuitively speaking, the operational space inertia matrix $\boldsymbol{\Lambda}$ captures the resistance of the operational point to acceleration along and about different axes. More details about operational space control and its relationship to joint space control are provided in Sect. 8.2 (motion control, joint space versus operational space control).

36.2.2 Combined Force and Position Control

Let us consider the example task of controlling the motion of a peg inside a hole. We assume the peg is already inserted into the hole and rigidly connected to the robot's end-effector. We attach the operational frame to a point on the peg so that its z-axis is aligned with the desired direction of motion of the peg inside the hole. To perform this task, the robot has to control the position of the peg along the z-axis of the operational frame and it also has to control contact forces between the peg and the hole to avoid jamming. Thus, the task of moving a peg inside a hole requires that position and force control be seamlessly integrated.

To address contact forces within the operational space framework, we rewrite (36.3) as

$$\boldsymbol{F}_{\boldsymbol{x}} = \boldsymbol{F}_{\mathrm{c}} + \boldsymbol{\Lambda}(\boldsymbol{q})\boldsymbol{F}_{\mathrm{m}} + \boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{\eta}(\boldsymbol{q}) \,, \tag{36.4}$$

where $\boldsymbol{F}_{\mathrm{c}}$ represents the contact forces acting at the end-effector [36.4]. We can replace $\ddot{\boldsymbol{x}}$ with $\boldsymbol{F}_{\mathrm{m}}$ because $\ddot{\boldsymbol{x}}$ is acting on a dynamically decoupled system, i. e., a system that acts like a unit point mass. We can now control forces and motion in the operational frame by selecting the following control structure

$$\boldsymbol{F}_{\boldsymbol{x}} = \boldsymbol{F}_{\mathrm{m}} + \boldsymbol{F}_{\mathrm{c}} \,, \tag{36.5}$$

where

$$\boldsymbol{F}_{\mathrm{m}} = \boldsymbol{\Lambda}(\boldsymbol{q})\,\boldsymbol{\Omega}\boldsymbol{F}'_{\mathrm{m}} + \boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{\eta}(\boldsymbol{q}) \,, \tag{36.6}$$

$$\boldsymbol{F}_{\mathrm{c}} = \boldsymbol{\Lambda}(\boldsymbol{q})\,\bar{\boldsymbol{\Omega}}\boldsymbol{F}'_{\mathrm{c}} \,, \tag{36.7}$$

where $\boldsymbol{\Omega}$ and $\bar{\boldsymbol{\Omega}}$ represent complementary task specification matrices [36.4] that determine along which directions the end-effector is position controlled and along which it is force controlled. By selecting $\boldsymbol{\Omega}$ appropriately, the combination of position and force control can be tailored to the task. In its simplest form, $\boldsymbol{\Omega}$ and $\bar{\boldsymbol{\Omega}} = I - \boldsymbol{\Omega}$ are diagonal matrices. The i-th diagonal entry of $\boldsymbol{\Omega}$ is 1 if the i-th operational coordinate of the end-effector is position controlled and 0 if it is force controlled. In this simplest case, positions and forces are controlled along axes of the same coordinate frame. The concept of task specification matrix can be extended to include differently oriented coordinate frames for position and force control [36.4].

Once the force $\boldsymbol{F}_{\boldsymbol{x}}$ has been computed as given by (36.5), the corresponding joint torques used to control the robot are computed based on (36.1).

36.2.3 Operational Space Control of Redundant Mechanisms

The full expressiveness of task-level control comes to bear in the context of redundant manipulators. A manipulator is considered redundant with respect to the task it is performing if has more degrees of freedom than required by the task. For example, the task of holding a cup of water only specifies two degrees of freedom,

namely the two rotations about the axes spanning the horizontal plane – this task has two degrees of freedom. The mobile manipulator shown in Fig. 36.2 has ten degrees of freedom, leaving eight redundant degrees of freedom with respect to the task.

The operational space framework for task-level control of redundant manipulators decomposes the overall motion behavior into two components. The first component is given by the task, specified in terms of forces and moments $\boldsymbol{F}_{\text{task}}$ acting at an operational point. This vector $\boldsymbol{F}$ is translated into a joint torque based on (36.2): $\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\boldsymbol{F}_{\text{task}}$. For a redundant manipulator, however, the torque vector $\boldsymbol{\tau}$ is not uniquely specified and we can select from a set of task-consistent torque vectors. The operational space framework performs this selection by considering a secondary task, the so-called posture behavior, making up the second component of the overall motion behavior. Posture behavior can be specified by an arbitrary torque vector $\boldsymbol{\tau}_{\text{posture}}$. To ensure that this additional torque does not affect the task behavior ($\boldsymbol{F}_{\text{task}}$), the torque $\boldsymbol{\tau}_{\text{posture}}$ is projected into the null space $\boldsymbol{N}$ [36.6] of the task Jacobian $\mathbf{J}$. The null space of the Jacobian is the space orthogonal to the one spanned by $\mathbf{J}$ and will be of rank $N_J - k$, where N_J is the number of degrees of freedom of the manipulator and $k = \text{rank}(\mathbf{J})$. The torque $\boldsymbol{N}^{\mathrm{T}}\boldsymbol{\tau}_{\text{posture}}$ resulting from the null space projection is task consistent, i. e., it is guaranteed not to affect the behavior of the operational point. Since the vector $\boldsymbol{N}^{\mathrm{T}}\boldsymbol{\tau}_{\text{posture}}$ may not lie entirely within the null space of $\mathbf{J}$, however, the execution of the posture behavior cannot be guaranteed.

The operational space task ($\mathbf{J}^{\mathrm{T}}\boldsymbol{F}_{\text{task}}$) and the posture behavior ($\boldsymbol{N}^{\mathrm{T}}\boldsymbol{\tau}_{\text{posture}}$) are combined to obtain the general expression for the torque-level decomposition of the overall motion behavior

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\boldsymbol{F}_{\text{task}} \ldots\ldots + \boldsymbol{N}^{\mathrm{T}}\boldsymbol{\tau}_{\text{posture}} \,. \tag{36.8}$$

The null-space projection $\boldsymbol{N}$ associated with the task Jacobian $\mathbf{J}$ can be obtained by

$$\boldsymbol{N} = \mathbf{I} - \bar{\mathbf{J}}\mathbf{J} \,, \tag{36.9}$$

where $\mathbf{I}$ is the identity matrix, and $\bar{\mathbf{J}}$ is the dynamically consistent generalized inverse of $\mathbf{J}$, given by

$$\bar{\mathbf{J}} = \mathbf{H}^{-1}\mathbf{J}^{\mathrm{T}}\boldsymbol{\Lambda} \,, \tag{36.10}$$

where $\mathbf{H}$ is the joint space inertia matrix and $\boldsymbol{\Lambda}$ is the operational space inertia matrix for the operational point for which $\mathbf{J}$ is defined. The use of this specific inverse to compute the null-space projection results in the selection of the task-consistent torque vector that minimizes kinetic energy.

36.2.4 Combining Mobility and Manipulation

The operational space framework does not distinguish between degrees of freedom used for mobility and those used for manipulation. The end-effector-centric perspective of this approach considers all degrees of freedom to be in service to position and move the end-effector. An explicit coordination of manipulation and mobility for task-level control is not necessary.

A computationally efficient approach for coordinating manipulation and mobility was introduced in [36.7]. A similar, schema-based coordination of manipulation and mobility in the presence of dynamic obstacles was demonstrated in [36.8]. This approach is based on (36.2) and projects the forces that are derived from various schemas into the configuration space of the robot to determine its motion. Another coordination approach for manipulation and mobility considers a given end-effector path and generates a path of the base that maintains manipulability criteria [36.9]. This underlying representation of mobile manipulators can also be used to generate obstacle avoidance behavior of the base for a given end-effector path [36.10].

The coordination of a nonholonomic mobility platform and a cart-pushing task is addressed in [36.11]. In [36.12], an analysis of the task space for a mobile base with two manipulators is presented. Such a system with two manipulator arms can be viewed as a branching kinematic chain. Section 36.6 discusses extensions to the operational space framework to such branching mechanisms.

36.2.5 Combining Multiple Task Behaviors

The concept of decomposing a redundant robot's overall motion into task and posture behavior can be generalized to an arbitrary number of behaviors [36.13, 14]. Assume a set of n tasks T_i, where T_i has higher priority than T_j if $i < j$. Every task T_i is associated with a force vector $\boldsymbol{F}_i$ and the corresponding joint torque $\boldsymbol{\tau}_i$. These tasks can be performed simultaneously by projecting the joint torque associated with each task into the combined null space of all tasks with higher priority, just as task and posture behavior were combined in the previous section. The null-space projection ensures that the execution of task i does not affect the execution of tasks with higher priority, given by the set $\text{prec}(i)$.

Given a task i, the joint torque consistent with the set of higher priority tasks $\text{prec}(i)$ can be obtained by projecting $\boldsymbol{\tau}_i$ into the combined null space of $\text{prec}(i)$,

$$\boldsymbol{\tau}_{i|\text{prec}(i)} = \boldsymbol{N}^{\mathrm{T}}_{\text{prec}(i)}\boldsymbol{\tau}_i \,, \tag{36.11}$$

where the combined null space $\boldsymbol{N}^{\mathrm{T}}_{\mathrm{prec}(i)}$ is computed as

$$\boldsymbol{N}_{\mathrm{prec}(n)} = \boldsymbol{N}_{n-1}\boldsymbol{N}_{n-2}\cdots\boldsymbol{N}_1 \,. \tag{36.12}$$

We say that $\boldsymbol{\tau}_{i|\mathrm{prec}(i)}$ is the torque for task i given the preceding tasks $\mathrm{prec}(i)$. Applying this torque to the robot will not affect the execution of higher-priority tasks. The original torque vector $\boldsymbol{\tau}_i$ can be computed based on (36.2).

Given the ability to compute a task torque that is consistent with preceding tasks we are able to combine an arbitrary number of tasks into a single motion behavior

$$\boldsymbol{\tau} = \boldsymbol{\tau}_1 + \boldsymbol{\tau}_{2|\mathrm{prec}(2)} + \boldsymbol{\tau}_{3|\mathrm{prec}(3)} + \cdots \boldsymbol{\tau}_{n|\mathrm{prec}(n)} \,, \tag{36.13}$$

where $\boldsymbol{\tau}$ is the torque used to control the robot.

Note that $\boldsymbol{\tau}_1$ is not projected into any null space, since it is the task with the highest priority. Task 1 is therefore performed in the full space defined by the robot's kinematics. If the robot has N degrees of freedom, this space has N dimensions. As the number of tasks performed on the robot increases, the null-space projections $\boldsymbol{N}_{\mathrm{prec}(i)}$ will project the N-dimensional torque vectors associated with lower-priority tasks into a subspace of decreasing dimension. Eventually, the null space will be reduced to the trivial case, when all torque vectors are projected onto the null vector. This prevents lower-priority tasks from being executed. The dimensionality of the null space of tasks $\mathrm{prec}(i)$ can be computed by counting the nonzero singular values [36.6] of $\boldsymbol{N}_{\mathrm{prec}(i)}$, permitting a programmatic assessment of task feasibility.

If the execution of a particular task has to be guaranteed, it should be associated with task 1 in (36.13). Therefore, the highest-priority task is generally used for hard constraints that under no circumstances should be violated. These constraints may include contact constraints, joint limits, and balancing in the case of humanoid robots, for example (Chap. 67).

The projection of (36.11) maps the torque $\boldsymbol{\tau}_i$ of task i into the null space of all preceding tasks. This projection also scales the vector, which may significantly impair the execution of task i. This effect can be reduced by scaling the desired accelerations $\ddot{\boldsymbol{x}}$ at the operational point with the task-consistent inertia matrix $\boldsymbol{\Lambda}_{i|\mathrm{prec}(i)}$, given by

$$\boldsymbol{\Lambda}_{i|\mathrm{prec}(i)} = \left(\mathbf{J}_{i|\mathrm{prec}(i)}\mathbf{H}^{-1}\mathbf{J}^{\mathrm{T}}_{i|\mathrm{prec}(i)}\right)^{-1} \,, \tag{36.14}$$

where $\mathbf{H}$ is the joint space inertia matrix and

$$\mathbf{J}_{i|\mathrm{prec}(i)} = \mathbf{J}_i\boldsymbol{N}_{\mathrm{prec}(i)} \tag{36.15}$$

is the task Jacobian consistent with all preceding tasks. This Jacobian projects operational space forces into joint torques that do not cause any acceleration at the operational points of tasks in $\mathrm{prec}(i)$. The task-consistent inertia matrix $\boldsymbol{\Lambda}_{i|\mathrm{prec}(i)}$ then captures the inertia perceived at the operational point when its motion is limited to the subspace consistent with all preceding tasks.

Using the task-consistent inertia matrix, we can obtain the task-consistent operational space dynamics by projecting the joint space dynamics into the space defined by the dynamically consistent inverse of $\mathbf{J}_{i|\mathrm{prec}(i)}$. Inverting (36.2) and replacing $\boldsymbol{\tau}$ based on (36.2) we obtain

$$\bar{\mathbf{J}}^{\mathrm{T}}_{i|\mathrm{prec}(i)}\left(\mathbf{H}(\boldsymbol{q}) + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{\tau}_g(\boldsymbol{q}) = \boldsymbol{\tau}_{k|\mathrm{prec}(i)}\right) , \tag{36.16}$$

which yields the task-consistent equivalent to (36.3)

$$\boldsymbol{F}_{i|\mathrm{prec}(i)} = \boldsymbol{\Lambda}_{i|\mathrm{prec}(i)}\ddot{\boldsymbol{x}} + \boldsymbol{\Gamma}_{i|\mathrm{prec}(i)}\dot{\boldsymbol{x}} + \boldsymbol{\eta}_{i|\mathrm{prec}(i)} \,, \tag{36.17}$$

where $\boldsymbol{\Lambda}_{i|\mathrm{prec}(i)}$ is the task-consistent operational space inertial matrix, $\boldsymbol{\Gamma}_{i|\mathrm{prec}(i)}$ captures centrifugal and Coriolis forces in operational space, and $\boldsymbol{\eta}_{i|\mathrm{prec}(i)}$ compensates for gravity forces. Equation 36.17 can be used to control a task at a particular operational point in a task-consistent manner, i. e., without causing acceleration at the operational points associated with the tasks in $\mathrm{prec}(i)$. Note that in practice the terms $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ and $\boldsymbol{\tau}_g(\boldsymbol{q})$ can be computed in the joint space; they only need to be accounted for once in the case of multiple tasks.

Referring back to Fig. 36.3, we can analyze the effectiveness of the operational space framework. The control of operational points directly addresses force and position constraints on motion in manipulation tasks. Additional behaviors with lower priority than the task can be used to ensure the maintenance of kinematic and dynamic constraints, implement posture behavior, or perform reactive obstacle avoidance based on artificial potential fields [36.15]. The control of all of these aspects of manipulation tasks is computationally efficient. As a consequence, the operational space framework is able to maintain these motion constraints while satisfying feedback frequency requirements. However, the operational space framework does not include the consideration of global motion constraints, such as those imposed by the connectivity of the work space. Therefore, the operational space framework is susceptible to local minima, similarly to the artificial potential field method. Such local

minima can occur for all of the aforementioned motion constraints and would prevent the robot from accomplishing its task. To overcome this problem, planning methods have to be applied (Chap. 7 and Sect. 36.3).

The control basis approach [36.16] provides an alternative, compact formalism for combining multiple behaviors. Each individual behavior is described by a controller, represented by a potential function $\phi \in \Phi$, sensors $\sigma \in \Sigma$, and effectors $\tau \in \Upsilon$. A particular controller ϕ can be parameterized with sensors and effectors. A particular instance of a behavior can be written as $\phi|_{\tau}^{\sigma}$. These controllers can be combined using null-space projections, similar to (36.13). In the control basis framework, (36.13) would be written more compactly as

$$\phi_n \triangleleft \cdots \triangleleft \phi_3 \triangleleft \phi_2 \triangleleft \phi_1 \,, \tag{36.18}$$

where we abbreviate $\phi_i|_{\tau_i}^{\sigma_i}$ with ϕ_i. The relation $\phi_i \triangleleft \phi_j$ means that ϕ_i is projected into the null space of ϕ_j. We say that ϕ_i is executed *subject to* ϕ_j. If more than two behaviors are combined, as shown in (36.18), each controller ϕ_i is projected into the combined null space of all superior tasks.

The control basis represents a general framework for combining controllers and is not limited to the notion of task-level control. In addition to the aforementioned formalism for representing combinations of controllers, the control basis also encompasses discrete structures that define transitions between combinations of controllers. Such transitions can be specified by the user or learned with techniques from reinforcement learning [36.16]. The control basis can compose multiple task-level controllers to implement manipulation tasks that cannot be described in terms of a single task-level controller.

36.3 Manipulation Planning

In order to appreciate the computational challenges inherent to gross motion planning for manipulation, it is useful to adopt a mathematical formalism for analysis. The chapter on planning (Chap. 7) introduced the *configuration space* (*C-space*), which represents the space of allowable transformations of the robot. In general, the set of all allowable transformations forms an n-dimensional *manifold*, in which n is the total number of degrees of freedom (DOF) of the mechanical system. For a robotic manipulator, the configuration space is defined by its kinematic structure (Chap. 2). One common class of manipulators is *serial-chain* manipulators, which consist of a series of revolute or prismatic joints connected to form a linear kinematic chain. The discussion of manipulation planning in this section focuses on configuration spaces arising from single serial-chain manipulators. However, the mathematical formulation generalizes to more complex robots such as humanoids (Fig. 36.4; Chap. 67), whose upper body consists of two cooperating manipulator arms attached to a movable base (Chap. 39). The model considered here greatly simplifies the problems of grasping, stability, friction, mechanics, and uncertainties and instead focuses on the geometric aspects. In this section, we will consider a simple example of a manipulation task that involves a classic pick-and-place operation. Section 36.4 further considers grasping and contact states in the context of assembly operations, and Sect. 36.5 gives an overview of techniques to unify feedback control and planning for more complex manipulation tasks.

Fig. 36.4 Simulation of the H6 humanoid robot manipulating an object

36.3.1 Configuration Space Formalism

The basic motion planning problem is defined as finding a path in the configuration space connecting a start and a goal configuration while avoiding static obstacles in the environment (Chap. 7). Manipulation planning involves additional complexity because manipulation tasks fundamentally require contact with objects in the environment. When objects are grasped, regrasped, and moved, the structure and topology of the free configuration space can change drastically. However, manipulation planning shares the same basic theoretical foundations and computational complexity as classical path planning, such as PSPACE-hardness (Chap. 7).

To be more precise, we will utilize a formalization for manipulation planning that is primarily due to *Alami*

et al. [36.17], and adapted in *LaValle*'s book [36.18]. Our goal is to gain understanding and intuition regarding the search space for manipulation tasks and the different situations that can arise. Throughout this discussion, bear in mind that the geometry and relative positions of the manipulator, the manipulated objects, and the obstacles in the workspace determine both the geometry and topological structure of the free configuration space. Within this framework, manipulation planning can be intuitively placed in the context of planning for *hybrid systems* [36.19]. Hybrid systems involve a mixture of continuous variables and discrete *modes*, whose transitions are defined by switching functions (for a discussion of hybrid systems in a planning context, see Chap. 7 of *LaValle*'s book [36.18]). In the case of manipulation planning, the effect of grasping, repositioning, and releasing objects in the environment corresponds to switching between different continuous configuration spaces.

Let us consider a simple example of a simple *pick-and-place manipulation task* that requires a robot manipulator $\mathcal{A}$ to reposition a single movable rigid object (part) $\mathcal{P}$ from its current location in the workspace to some desired goal location. Note that accomplishing the task only requires that the part $\mathcal{P}$ reach the goal location, without regard to precisely *how* the manipulator does so. Because the part cannot move by itself, it must either be transported by the robot or be placed at rest in some stable intermediate configuration. There are some configurations in which the robot is able to grasp and move the part to other locations in the environment. There are also forbidden configurations in which the robot or the part collides with obstacles in the environment. The solution of a manipulation planning problem consists of a sequence of subpaths for the robot moving alone or moving while grasping the part. The primary constraints on the allowable motions are:

1. The robot manipulator must not collide with any obstacles in the environment while reaching for, grasping, or regrasping the part.
2. The robot must adequately grasp the part while transporting it (for details on grasping and metrics for stable grasps see Chap. 38).
3. The robot manipulator and movable part must not collide with any obstacles in the environment while the part is being transported.
4. If the part is not being transported by the robot, it must be placed at rest in some stable intermediate placement.
5. The part must end up at the desired goal location in the workspace.

The two modes of operation give rise to the hybrid systems formulation: either the robot moves alone, or the robot moves while grasping the part. In the literature, motions of the robot holding the object at a fixed grasp are called *transfer paths*, and motions of the robot while the part stays at a stable placement are called *transit paths* [36.17, 20, 21]. Solving a manipulation planning problem involves searching for a connected sequence of *transit* and *transfer* paths for the manipulator that will accomplish the task. For an autonomous robot, the ideal is to utilize planning algorithms that will automatically produce correct sequences of transfer and transit paths separated by grasping and ungrasping operations.

Admissible Configurations

Building on the notation introduced in Chap. 7, we define a robot manipulator $\mathcal{A}$ with n degrees of freedom operating in a Euclidean *world* or *workspace*, $\mathcal{W} = \mathbb{R}^N$, in which $N = 2$ or $N = 3$. Let C^A represent the n-dimensional C-space of $\mathcal{A}$, and q^A be a configuration. C^A is called the *manipulator configuration space*. Let $\mathcal{P}$ denote a movable part, which is a solid rigid body modeled with geometric primitives. We will assume that $\mathcal{P}$ is allowed to undergo rigid-body transformations. This gives rise to a *part configuration space*, $C^P = SE(2)$ or $C^P = SE(3)$. Let $q^P \in C^P$ denote a part configuration. The area or volume in the workspace occupied by the transformed part model is denoted by $\mathcal{P}(q^P)$.

We define the combined (robot and part) configuration space C as the Cartesian product

$$C = C^A \times C^P \,,$$

in which each configuration $q \in C$ is of the form $q = (q^A, q^P)$. Note that the set of admissible configurations for manipulation planning purposes is more restrictive than simply C_{free}, the space free of collisions with obstacle regions as defined in Chap. 7. We must incorporate the constraints on the allowable motions by removing inadmissable configurations. Figure 36.5 illustrates examples of some of the important subsets of C for manipulation planning.

We start by removing all configurations that involve collisions with obstacles. Let the subset of configurations in which the manipulator collides with obstacles be given by

$$C^A_{\text{obs}} = \{(q^A, q^P) \in C | \mathcal{A}(q^A) \cap \mathcal{O} \neq \emptyset\} \,.$$

We also want to remove configurations for which the part collides with obstacles. However, we will allow the surface of the part to make contact with obstacle surfaces. This happens for example, when the part is

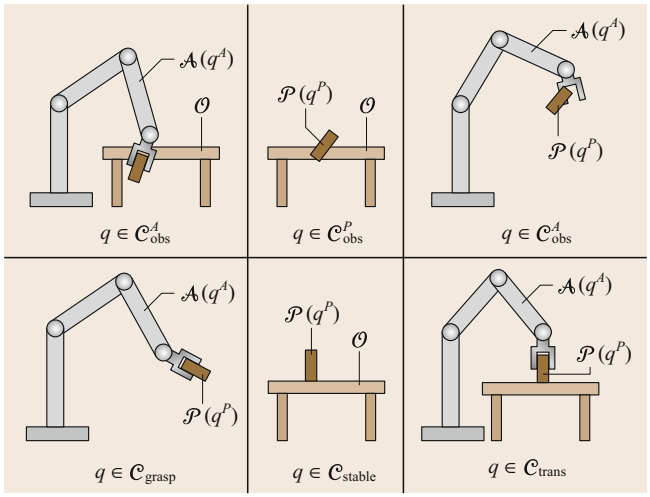


Fig. 36.5 Examples of the various configuration space subsets defined for manipulation planning

resting on a shelf or table, or when a peg-shaped part is being inserted into a hole. Therefore, let

$$C^P_{\text{obs}} = \{(q^A, q^P) \in C | \text{int}(\mathcal{P}(q^P)) \cap \mathcal{O} \neq \emptyset\}$$

denote the open set for which $\mathcal{O}$ intersects the interior volume $\text{int}(\mathcal{P}(q^P))$ of the part for any configuration q^P. If the interior of the part penetrates $\mathcal{O}$, then clearly these configurations should be avoided.

We can now define $C \setminus (C^A_{\text{obs}} \cup C^P_{\text{obs}})$, which is the set of all configurations in which both the robot and the part do not inappropriately collide with $\mathcal{O}$. Next consider the interaction between $\mathcal{A}$ and $\mathcal{P}$. The manipulator must be allowed to make contact with the surface of the part, but penetration is once again not allowed. Therefore, let all configurations in which the interior of the part overlaps with the robot geometry be defined as

$$C^{PA}_{\text{obs}} = \left\{(q^A, q^P) \in C | A(q^A) \cap \text{int}(\mathcal{P}(q^P)) \neq \emptyset\right\} .$$

Finally, we are able to define

$$C_{\text{adm}} = C \setminus \left(C^A_{\text{obs}} \cup C^P_{\text{obs}} \cup C^{PA}_{\text{obs}}\right) ,$$

which represents the set of configurations remaining after subtracting away all unwanted configurations. We call this the set of *admissible configurations*.

Stable and Grasped Configurations

By definition, all configurations in the set C_{adm} are free of penetrative collisions between the robot, the part, and obstacles in the environment. However, many of these configurations define the part as floating in space or on the verge of falling. We now consider two important subsets of C_{adm} that are used in manipulation planning (Fig. 36.5). Let $C^P_{\text{stable}} \subseteq C^P$ denote the subset of *stable part configurations*, which are configurations at which the part can safely rest without any forces being applied by the manipulator. Examples of stable configurations include the part resting on a table or inserted into a larger assembly of other parts. The criteria for inclusion into the set of stable configurations depends on properties such as the part geometry, friction, mass distribution, and contacts with the environment. Our formalism does not consider these issues directly, but assumes that some method for evaluating the stability of a part configuration is available. Given C^P_{stable}, we define $C_{\text{stable}} \subseteq C_{\text{adm}}$ to be the corresponding stable configurations of the part–robot system

$$C_{\text{stable}} = \left\{(q^A, q^P) \in C_{\text{adm}} | q^P \in C^P_{\text{stable}}\right\} .$$

The other important subset of C_{adm} is the set of all configurations in which the robot is grasping the part and is capable of manipulating it according to some defined criteria. Let $C_{\text{grasp}} \subseteq C_{\text{adm}}$ denote the set of *grasp configurations*. For every configuration, $(q^A, q^P) \in C_{\text{grasp}}$, the manipulator touches the part, which implies that $\mathcal{A}(q^A) \cap \mathcal{P}(q^P) \neq \emptyset$. Just as before, penetration between the robot and part geometry is not allowed because $C_{\text{grasp}} \subseteq C_{\text{adm}}$. In general, many configurations at which $\mathcal{A}(q^A)$ contacts $\mathcal{P}(q^P)$ will not necessarily be in C_{grasp}. The criteria for a configuration to lie in C_{grasp} depends on the particular characteristics of the manipulator, the part, and the contact surfaces between them. For example, a typical manipulator would not be able to pick up a part by making contact at only a single point. (For additional information, see Sect. 36.4 for contact state identification, Chap. 37, and grasping criteria, such as *force closure* models in Chaps. 38 and 19).

For any robot and part configuration $q = (q^A, q^P) \in C$ that we consider for manipulation planning we must always ensure that either $q \in C_{\text{stable}}$ or $q \in C_{\text{grasp}}$. We therefore define $C_{\text{free}} = C_{\text{stable}} \cup C_{\text{grasp}}$, to reflect the subset of C_{adm} that is permissible for manipulation planning. In the context of hybrid systems, this gives rise to the two distinct *modes* for manipulation planning: the *transit mode* and the *transfer mode*. Recall that in the transit mode, the manipulator is not carrying the part (i. e., only the robot moves), which requires that $q \in C_{\text{stable}}$. In the transfer mode, the manipulator carries the part (i. e., both the robot and the part move), which requires that $q \in C_{\text{grasp}}$. Based on these conditions, the only way the mode can change is if $q \in C_{\text{stable}} \cap C_{\text{grasp}}$. When the manipulator is in these configurations, it has two available actions: (1) continue to grasp and move the part, or (2) release the part in its currently stable configuration. In all other configura-

tions the mode remains unchanged. For convenience, we define $C_{\text{trans}} = C_{\text{stable}} \cap C_{\text{grasp}}$ to denote the set of *transition configurations*, which are the places in which the mode may change and the robot may grasp or release the part.

Manipulation Planning Task Definition

Finally, the basic pick-and-place manipulation planning task can now be defined. An *initial part configuration*, $q^P_{\text{init}} \in C^P_{\text{stable}}$, and a *goal part configuration*, $q^P_{\text{goal}} \in C^P_{\text{stable}}$, are specified. Recall that this high-level task specification is only defined in terms of repositioning the part, without regard to how the manipulator actually accomplishes the task. Let $C_{\text{init}} \subseteq C_{\text{free}}$ be the set of all configurations in which the part configuration is q^P_{init}

$$C_{\text{init}} = \left\{(q^A, q^P) \in C_{\text{free}} | q^P = q^P_{\text{init}}\right\} .$$

We define $C_{\text{goal}} \subseteq C_{\text{free}}$ similarly as the set of all configurations in which the part configuration is q^P_{goal}

$$C_{\text{goal}} = \left\{(q^A, q^P) \in C_{\text{free}} | q^P = q^P_{\text{goal}}\right\} .$$

If the initial and final configurations of the manipulator are specified, then let $q_{\text{init}} = (q^A_{\text{init}}, q^P_{\text{init}}) \in C_{\text{init}}$ and $q_{\text{goal}} = (q^A_{\text{goal}}, q^P_{\text{goal}}) \in C_{\text{goal}}$. The objective of the planner is to compute a path τ such that

$$\tau : [0, 1] \mapsto C_{\text{free}} ,$$
$$\tau(0) = q_{\text{init}}; \tau(1) = q_{\text{goal}} .$$

If the initial and final positions of the manipulator are unspecified, we implicitly define the objective of the planner as computing a path $\tau : [0, 1] \mapsto C_{\text{free}}$ such that $\tau(0) \in C_{\text{init}}$ and $\tau(1) \in C_{\text{goal}}$. In either case, a solution is an alternating sequence of transit paths and transfer paths, whose names follow from the mode. In between each transfer path, the part is placed in a stable intermediate configuration while the manipulator moves alone (transit path) in order to regrasp the part. This sequence continues until the part is ultimately placed to rest in its final goal configuration. This is depicted in Fig. 36.6.

36.3.2 Example of a Three-DOF Planar Manipulator

For manipulators with three or fewer degrees of freedom (DOF), we can construct visualizations of the C-space in order to gain an intuition of the structure. Throughout this section, we will use the example of a three-DOF serial-chain manipulator whose end-effector operates a planar workspace ($\mathbb{R}^2$).

Figure 36.7 shows a simple redundant robot arm originally illustrated in [36.22]. The robot has three revolute joints with parallel axes and joint limits of $-\pi, \pi$ radians, providing three degrees of freedom. Since the end-effector of the robot can only reach positions in a plane, the arm is redundant in this plane and since it has only three DOF and limits imposed on the joint angles, C can be visualized as a cube with an edge length of 2π. Generally, the C-space manifold of a manipulator with n revolute joints is homeomorphic to an n-dimensional *hypertorus* if no joint limits are present, and homeomorphic to an n-dimensional *hypercube* if joint limits are present.

Recall that obstacles in the workspace are mapped to regions in the C-space called C-obstacles (Chap. 7), which represent the set of all joint configurations of the robot that cause an intersection of the geometry of the robot and the obstacle

$$C^A_{\text{obs}} = \left\{(q^A, q^P) \in C | A(q^A) \cap \mathcal{O} \neq \emptyset\right\} .$$

In general, the structure of C^A_{obs} can be highly complex and involve multiple connected components of C. An example visualization of a configuration space for the planar three-DOF manipulator corresponding

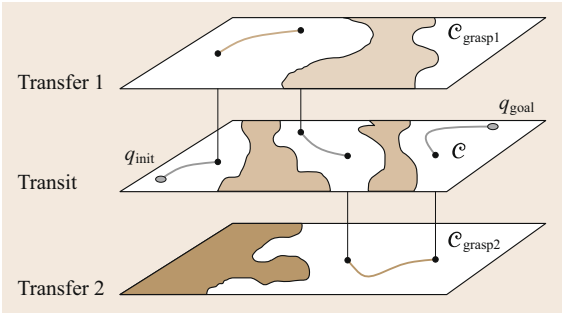


Fig. 36.6 Manipulation planning involves searching for a sequence of transfer and transit paths in a set of hybrid continuous configuration spaces. In this example, transfer paths in different C-spaces arise from different ways of rigidly grasping a part

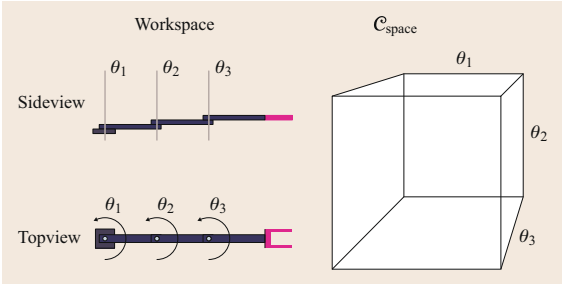


Fig. 36.7 Planar 3 DOF manipulator

to a workspace with two differently colored box-shaped obstacles is given in Fig. 36.8. The C-obstacles are illustrated using the same color as the corresponding box-shaped obstacle in the workspace. The red, green, and blue coordinate axes correspond to the three joint angles $\theta_{\{1,2,3\}}$ of the manipulator, respectively.

Self-collisions induce special C-obstacle regions that do not involve environment geometry, but only the geometric interference of the links of the robot itself. The robot shown in Fig. 36.7 is, by design, self-collision-free. Figure 36.9 shows a modified version of the manipulator, where self-collision is actually possible, and a visualization of the corresponding C-obstacles.

C-obstacles can cover large parts of the configuration space and even cause a complete disconnection between different components of C_{free}. Figure 36.10 shows how the addition of only a small post as an obstacle can cause a wide range of configurations to produce collisions and disconnect regions of C_{free}: the C-obstacle corresponding to the post forms a *wall* of configurations parallel to the θ_2–θ_3-plane. This wall represents the range of values for θ_1 for which the first link of the manipulator intersects with the post, splitting C_{free} into two disconnected components. Having disconnected components $C_1, C_2 \in C_{\text{free}}$ means that there is no collision-free path that starts in C_1 and ends in C_2.

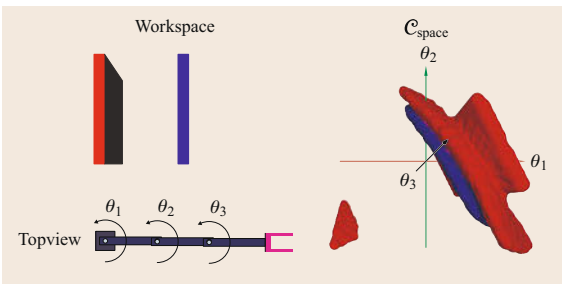


Fig. 36.8 Planar 3 DOF manipulator in a workspace with box-shaped obstacles and associated configuration space obstacles

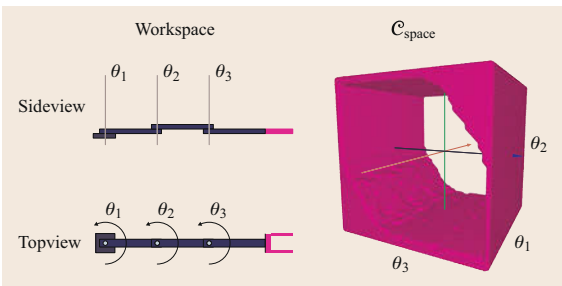


Fig. 36.9 Non-planar 3 DOF manipulator and corresponding self-collision C-obstacle

36.3.3 Inverse Kinematics Considerations

The typical formulation of a path planning problem requires a goal configuration q_{goal} as input (Chap. 7). For the case of manipulation planning, this configuration is usually computed from the desired workspace pose of the end-effector by an inverse kinematics (IK) solver (Sect. 2.7). Apart from special cases, there currently exist no known analytical methods for solving the inverse kinematics of a general redundant mechanism (greater than six degrees of freedom). Iterative, numerical techniques (such as Jacobian-based methods or gradient-based optimization) are typically used to calculate solutions in this case. Note that the numerical computation of inverse kinematics solutions can suffer from poor performance or even nonconvergence, depending on the quality of the initial guess, the distribution of singularities in the C-space or a combination of these effects (Sect. 2.7).

Most IK solvers are only able to compute one configuration for a given end-effector pose. For a redundant manipulator, however, an infinite, continuous range of configurations that cause the end-effector to assume the desired pose usually exists. In the context of path planning, this means that the IK solver will select one out of an infinite number of configurations as q_{goal}. Note that in some cases, the selected q_{goal} might not be collision-free, i. e., it is not part of C_{free}. Such a configuration cannot be used as a goal for computing a global reaching strategy because no solution exists for the corresponding path planning query in the classical sense (Chap. 7).

Another possibility is that the selected q_{goal} might be disconnected from q_{init}. In this case, q_{goal} is unsuitable for planning despite the fact that it is a member of C_{free}, because it is in a component of C_{free} that is disconnected from the component in which q_{init} lies. Consider the planar manipulator shown in Fig. 36.7.

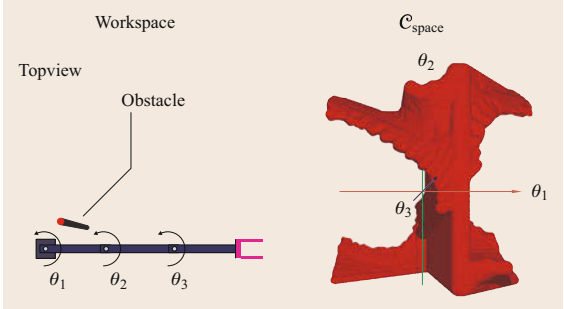


Fig. 36.10 Workspace mapping to a configuration space with topologically disconnected components

Figure 36.11 shows two possible solutions for a certain end-effector pose. The corresponding configurations $q_a = (0, \pi/2, 0)$ and $q_b = (\pi/2, -\pi/2, \pi/2)$ are marked by green spheres in the C-space visualization. In this case, the IK solver might select one of the two solutions as q_{goal}. If the same obstacle as in Fig. 36.10 is added to the workspace, the two solutions will lie in disconnected components of C_{free}, as shown in Fig. 36.12.

Let $q_{init} = (0, 0, 0)$ be the zero configuration, marked by the point of intersection of the C-space axes in Fig. 36.12. Let the component of C_{free} in which it lies be denoted by C_1 and the other component by C_2. Since $q_b \in C_2$ and $C_1 \cap C_2 = \emptyset$, no solution for the planning problem can exist if the IK solver selects q_b as q_{goal}. This means that every planning attempt with this q_{goal} will invariably fail.

Since computing a complete representation of the connectivity of C is a problem of similar complexity to path planning itself, it is virtually impossible to detect a disconnection between q_{goal} and q_{init} before the planning attempt has failed.

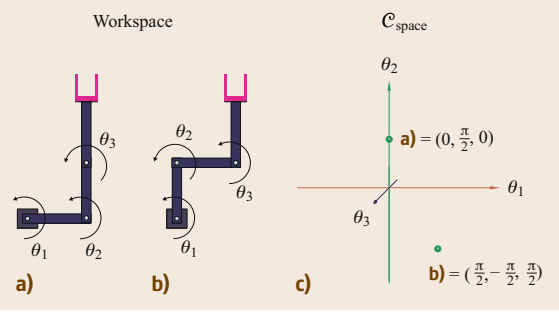


Fig.36.11a–c Two inverse kinematics solutions for the same end-effector pose of a planar 3 DOF arm; **(a)** solution 1, **(b)** solution 2, **(c)** solution shown in C space

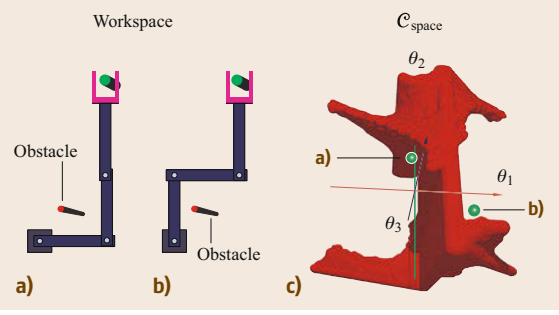


Fig.36.12a–c Two inverse kinematics solutions lying in disconnected components of C_{free} due to a small obstacle in the workspace; **(a)** solution 1, **(b)** solution 2, **(c)** the two solutions shown in C space, which are in disconnected components

Recall that the workspace goal for a classical manipulation planning problem is defined by a desired end-effector pose computed using inverse kinematics. Note however, that the purpose of a manipulation task may not require moving the end-effector to that particular location, but rather moving it to *any* feasible location which enables the desired manipulation task to be solved. Thus, it becomes important to devise computational methods and control schemes that are focused around the task, and allow some flexibility and freedom with regards to the end-effector pose and manipulator configuration. This is one of the motivations and advantages of task space or operational space control techniques (Sect. 8.2).

As an example, consider the task of grasping a cylindrical part with the planar three-DOF manipulator. Figure 36.11 shows two possible solution configurations yielding the same end-effector pose. The addition of obstacles to this workspace may cause one of the configurations to become disconnected from the zero configuration. These two configurations, however, may not be the only valid configurations that allow the manipulator to grasp the cylindrical part. Figure 36.13a shows several other configurations that allow the manipulator to grasp the part. By interpolating between these, an infinite number of different workspace poses that provide solutions to the grasping problem is generated. The inverse kinematics solutions associated with these workspace poses are contained in the continuous subset $C_{goal} \subseteq C$, as shown in Fig. 36.13b, which represents the actual solution space to the planning problem for this reaching task.

Adding the obstacle from Fig. 36.10 to the workspace of this manipulation task produces the configuration space depicted in Fig. 36.14. Note that the C-obstacle corresponding to collisions between the manipulator and the cylinder itself has been disregarded in this visualization for clarity. Approximately half of the

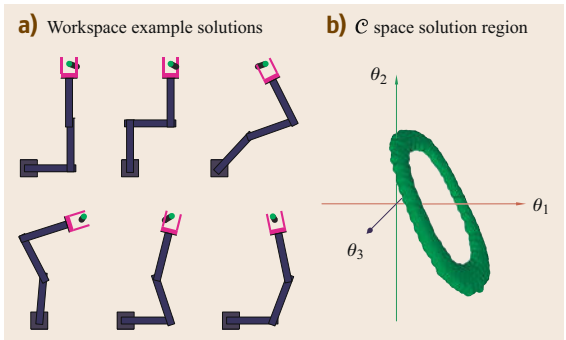


Fig. 36.13 **(a)** Multiple arm configurations that allow the manipulator to grasp the cylinder and **(b)** C-space visualization of the set of solution configurations

possible solutions are disconnected from the zero configuration and a quarter are in collision, which leaves only one quarter as reachable and therefore suitable candidates for q_{goal} in a classical planning query.

36.3.4 Continuum Manipulation

As introduced in Chap. 20, a manipulator with far more degrees of freedom than that of an object to be manipulated is a hyperredundant manipulator. Hyperredundant manipulators can be further classified into vertebrate-like rigid-link manipulators, such as snakes, and invertebrate-like continuum manipulators, such as octopus arms or elephant trunks. Many vertebrate-like manipulators inherit the kinematics of articulated arms, and like the articulated arms, they are often used as carriers of the actual manipulators: hands or grippers, such as in surgical applications.

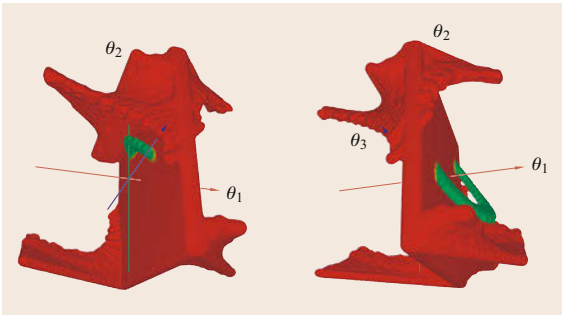


Fig. 36.14 Two views of the set of goal configurations (*green*) for grasping a cylindrical part in a disconnected *C*-space

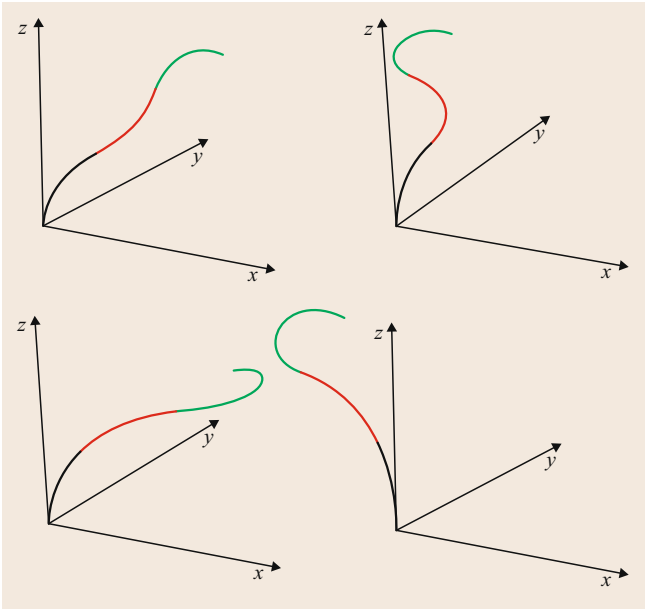


Fig. 36.15 Different configurations of the OctArm manipulator

Continuum manipulators [36.23], on the other hand, consist of deformable sections and are often used for whole-arm manipulation without a divide between the *arm* and *hand/gripper*, taking advantage of their compliance and flexibility. They can be viewed as the extreme case of hyperredundancy but are fundamentally different from articulated arms. A *grasping* configuration for a continuum manipulator affects the whole arm as it wraps around an object. The smooth and compliant nature of a continuum manipulator allows it to gently interact with the object by adapting its shape to the object it wraps. The OctArm [36.24], with multiple constant-curvature sections, is representative of the general class of continuum manipulators.

Although each section i of a continuum manipulator is deformable, and thus has infinite degrees of freedom theoretically, it often has three controllable degrees of freedom, which describes the configuration of the section as a truncated torus when undeformed or a cylinder if the curvature is zero. length s_i, curvature κ_i, and the orientation angle ϕ_i between section $i-1$ and section i. By changing those three variables for each section, the configuration of the whole manipulator can be changed. Figure 36.15 shows different configurations of a three-section OctArm.

Planning for continuum, whole-arm manipulation is significantly different from manipulation and grasping planning for conventional articulated arms with hands/grippers. Here the inverse kinematics problem is to find end positions of each section for the manipulator, given the configuration variables of the section [36.25]. However, planning motion for whole-arm manipulation has to consider directly a mixture of configuration variables and Cartesian-space variables (i. e., the end position of each section), because the shapes and grasping capabilities of the end-effector (which can be viewed as the portion of the continuum manipulator contacting the target object) depend on both.

Given a target object, its location and size decide if there exists a configuration that a continuum manipulator can wrap around the object to grasp it. If wrapping around the object along one circle is sufficient, that bounding circle of the object centered at position c with radius r_c determines the k ($k \leq n$) sections that are needed to wrap it, starting from the tip n-th section backward. The poses of the remaining $n-k$ sections must satisfy the intersection constraints to form feasible whole-arm configurations for grasping [36.26]. A feasible grasping configuration can also be determined heuristically as an integral part of collision-free motion planning [36.27]. Multiple grasping solutions provide flexibility to avoid obstacles (VIDEO 357).

A more practical approach [36.28] is to first make a contact between the manipulator and the target object at the m-th section ($m < n$), and then generate wrapping configurations for the m to n sections of the manipulator one by one without overlap to form a tight, spiral wrap (i. e., along multiple object bounding circles). In a cluttered environment, progressive wrapping of an object can be achieved through moving all arm sections together led by the tip of the arm to surroun. the object [36.29], which requires minimum free space for the arm to manuever. Those approaches can generate a better force-closure grasp and facilitate sensing-based manipulation based on contact detection.

Finally, strategies for planning continuum manipulation could also benefit whole-arm manipulation with vertebrate-like hyperredundant manipulators in that the planned spatial curves for poses of a continuum manipulator could be used as desired shapes for a hyperredundant manipulator, such as a snake robot, to fit its joint configurations. Indeed, curve design and fitting for hyperredundant manipulators has long been introduce. for planning manipulator poses [36.30].

In this section, we have covered the basics of manipulation planning, developed a mathematical formulation for the kinds of search spaces that arise, and discussed some of the important geometric and topological considerations, as well as different types of manipulators and manipulation. Section 36.6 provides an overview of additional topics and extensions with pointers to further reading.

36.4 Assembly Motion

Assembly or an *assembly task* defines the process of putting together manufactured parts to make a complete product, such as a machine. It is a major operation in the manufacturing process of any product. Assembly automation with robots aims to reduce cost and increase the quality and efficiency of the operation. In environments hazardous to humans, such as in space, having robots perform assembly tasks could save human lives. Assembly has long been not only an important but also one of the most challenging applications for robotics. There are many significant research issues related to the broad scope of assembly automation, from design for assembly [36.31] to tolerance analysis [36.32], assembly sequence planning [36.33], fixture design [36.34], etc. This section is only focused on the issue of robotic motion for assembly.

The concerned *assembly motion* is that of a robot manipulator holding a part and moving it to reach a certain *assembled state*, i. e., a required spatial arrangement or contact against another part. The main difficulty of assembly motion is due to the requirement for high precision or low tolerance between the parts in an assembled state. As a result, the assembly motion has to overcome uncertainty to be successful. *Compliant motion* is defined as motion constrained by the contact between the held part and another part in the environment. As it reduces uncertainty through reducing the degrees of freedom of the held part, compliant motion is desirable in assembly.

Consider the peg-in-hole insertion example introduced earlier (Fig. 36.2). If the clearance between the peg and the hole is very small, the effect of uncertainty will most likely cause a downward insertion motion of the peg to fail, i. e., the peg ends up colliding with the entrance of the hole in some way without reaching the desired assembled state. Therefore, a successful assembly motion has to move the peg out of such an unintended contact situation and lead it to reach the desired assembled state eventually. To make this transition, compliant motion is preferred. Often a sequence of contact transitions via compliant motion is necessary before the desired assembled state can be reached. Figure 36.16 shows a typical sequence of contact transitions for the peg-in-hole task.

Assembly motion strategies that incorporate compliant motion can be broadly classified into two groups: *passive compliance* and *active compliant motion*, and both groups of strategies require certain informa-

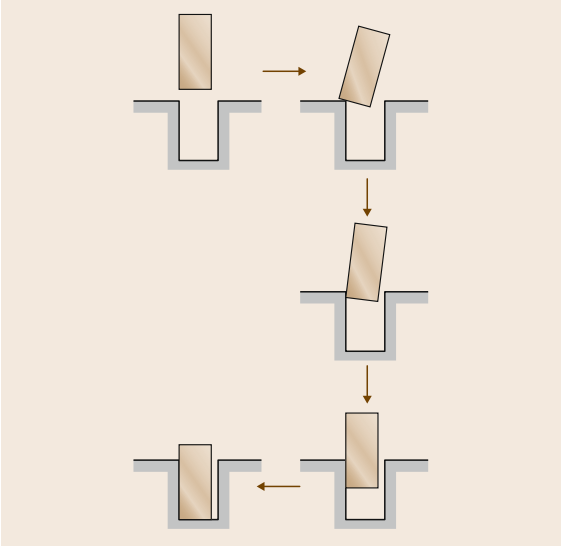


Fig. 36.16 A sequence of contact transitions for the peg-in-hole insertion task

tion characterizing *topological contact states* between parts.

36.4.1 Topological Contact States

When a part A contacts a part B, the configuration of A (or B) is a *contact configuration*. Often a set of contact configurations share the same high-level contact characteristics. For example, *a cup is on the table* is a high-level description shared by all the contact configurations of the cup when its bottom are on the table. Such a description is often what really matters in assembly motion as it characterizes a spatial arrangement that could be either an assembled state or just a *contact state* between a part and another part.

For contacting polyhedral objects, it is common to describe a *contact state* topologically as a set of *primitive contacts*, each of which is defined by a pair of contacting surface elements in terms of faces, edges, and vertices. Different contact state representations essentially differ only in how primitive contacts are defined. One common representation [36.35] defines primitive contacts as point contacts in terms of vertex–edge contacts for two-dimensional (2-D) polygons, and vertex–face and edge–edge contacts for three-dimensional (3-D) polyhedra. Another representation [36.36] defines a primitive contact as between a pair of *topological surface elements* (i. e., faces, edges, and vertices). Here a contact primitive can characterize a contact region that is either a point, a line segment, or a planar face, unlike the point-contact notion. From the viewpoint of contact identification via sensing, however, both representations can result in states that are different by definition but indistinguishable in identification due to uncertainties. Figure 36.17 shows such an example.

The notion of a *principal contact* [36.37, 38] presents a high-level primitive contact that is more robust to recognition. A *principal contact* (PC) denotes a contact between a pair of topological surface elements that are not boundary elements of other contacting topological surface elements. The boundary elements of a face are the edges and vertices bounding it, and the boundary elements of an edge are the vertices bounding it. Different PCs between two objects correspond to different degrees of freedom of the objects, which often also correspond to significant differences in the contact forces and moments. As shown in Fig. 36.17, the indistinguishable states in terms of the other contact primitives are grouped as a single contact state in terms of PCs. Thus, there are also fewer contact states in terms of PCs, leading to a more concise characterization of contact states. In fact, every contact state between two convex polyhedral objects is described by a single PC.

36.4.2 Passive Compliance

Passive compliance refers to strategies that incorporate compliant motion for error correction during the assembly motion without requiring active and explicit recognition and reasoning of contact states between parts.

Remote Center Compliance

In the 1970s, a *remote center compliance* (RCC) device was developed to assist high-precision peg-in-hole insertion [36.39–41]. The RCC is a mechanical spring structure used as a tool attached to the end-effector of a robot manipulator to hold a round peg when it is inserted into a round hole. The RCC is designed to have high stiffness along the direction of insertion but high lateral and angular compliances K_x and K_θ, and it projects the center of compliance near the tip of the peg (hence the name remote center compliance) to overcome small lateral and angular errors in the peg's position and orientation in response to the contact forces applied to the peg by the hole during insertion (Fig. 36.18). The large lateral and angular compliance of the RCC also helps to avoid *wedging* and *jamming*, conditions that cause the peg to stick in two-point contacts due to contact forces in direct opposition (wedging) or ill-proportioned contact forces/moments (jamming). Because the RCC is low cost and reliable for fast insertions, it has long seen successful industrial applications.

However, the RCC has limitations. It only applies to round peg-in-hole insertions, and a particular RCC works better for a peg and hole of a particular size. Attempts have been made to expand the applicability of the RCC. Some attempts try to make certain param-

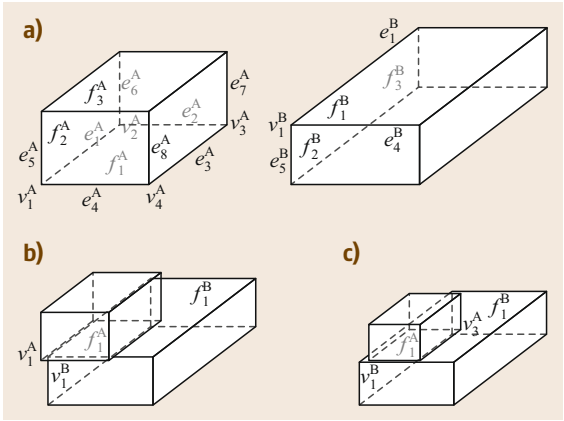


Fig.36.17a–c Indistinguishable contact states due to positional uncertainty of the top part; **(a)** two objects, **(b)** state 1, **(c)** state 2

eters of an RCC adjustable to achieve variable compliance or projection (i. e., the position of the remote center) [36.42–44]. A spatial RCC (SRCC) [36.45] was proposed to achieve square-peg-in-hole insertions. A notable difference with this extension is the combinatorial explosion of the number of possible contact states, given the range of the position and orientation uncertainty of the held part that have to be considered in the design of the device. Unlike the case of the round peg and hole, which is essentially a 2-D problem with a handful of different contact states, there are hundreds of possible different contact states between a square peg and a square hole. From each of these possible contact states, a motion leading the held part to the goal assembled state has to be realized by the device. This makes it difficult to design an RCC device that works for parts of more general and complex shapes with more possible contact states between them. This is a major limiting factor that prevent more extensions of the RCC to assembly of different parts.

Admittance Matrix

As an alternative method to the RCC, a particular form of manipulator force control, damping control, was proposed to achieve compliant motion of the held part by the manipulator and to correct small location errors of the held part during assembly. This approach eliminates the need to build a mechanical device like an RCC to achieve error correction. Among the force control laws [36.46], damping control is a common strategy, where a commanded velocity of the held part is modified based on the sensed force caused by contact between the held part and the environment. The resulted actual velocity leads to reduction and hopefully eventual correction of small position or orientation errors of the held part. Let $\boldsymbol{v}$ be a six-dimensional (6-D) vector representing the actual translational and angular velocity of the held part, $\boldsymbol{v}_0$ be the six-dimensional commanded velocity, and $\boldsymbol{f}$ be a six-dimensional vector representing the sensed force and moment. A linear damping control law is described as

$$\boldsymbol{v} = \boldsymbol{v}_0 + \mathbf{A}\boldsymbol{f} \,,$$

where $\mathbf{A}$ is a 6×6 matrix, called an *admittance matrix* or accommodation matrix.

The effectiveness of such a damping control law depends on the existence and finding of a proper admittance matrix $\mathbf{A}$. There is considerable research on the design of a single $\mathbf{A}$ that can make an assembly operation successful regardless of what contact states the held peg may encounter in the process [36.47–51]. This is aimed at cases where a single commanded velocity would be sufficient to achieve an assembly operation when there were no uncertainty or error, such as certain peg-in-hole insertion operations. One main approach to design $\mathbf{A}$ is based on explicit kinematic and static analysis of contact conditions under all possible contact states and mating requirements, which result in a set of linear inequalities as constraints on $\mathbf{A}$. Learning is also used [36.50] to obtain an $\mathbf{A}$ that minimizes the force $\boldsymbol{f}$ without causing instability. Another approach [36.51] applies perturbations to the end-effector during insertion in order to obtain richer force information.

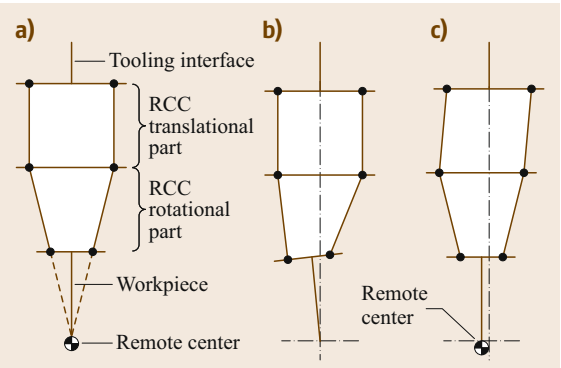


Fig. 36.18 (a) Planar representation of RCC. (b) Rotational part of RCC allowing workpiece to rotate. (c) Translational part of RCC allowing workpiece to translate

Learning Control for Assembly

Another category of approaches is to learn proper control for a particular assembly operation through stochastic or neural-network-based methods [36.52–56]. The essence of most of these approaches is to learn to map a reaction force upon the held object caused by contact to the next commanded velocity in order to reduce errors and to achieve an assembly operation successfully. A more recent approach [36.56] maps fused sensory data of pose and vision obtained during human demonstration of assembly tasks to compliant motion signals for successful assembly.

A different approach observes assembly tasks performed by human operators through vision [36.57] or in a virtual environment [36.58, 59] and generates a motion strategy necessary for the success of the task that consists of a sequence of recognized contact state transitions and associated motion parameters.

As a sequence of commanded velocities can be generated, unlike RCC or strategies based on a single admittance matrix described above, can be applied to cases with large uncertainties. However, the learned controllers are task dependent.

More recently, an approach combining vision and force sensing for assembly was introduced [36.60] based on measuring the deformation of parts through vision to adjust desired forces for mating and achieve

force control through force sensing and an impedance controller. The approach was applied to the task of mating two power connectors. The effect was similar to having an RCC device on the end-effector.

All of the above assembly motion strategies do not require explicit recognition of contact states during their execution.

36.4.3 Active Compliant Motion

Active compliant motion is characterized by error correction based on online identification or recognition of contact states in addition to feedback of contact forces. The capability for active compliant motion allows a robot the flexibility to deal with a much broader range of assembly tasks with large uncertainties and tasks beyond assembly where compliance is required. An active compliant motion system generally requires the following components: a planner to plan compliant motion commands, an identifier to recognize contact states, state transitions, and other state information during task operation, and a controller to execute compliant motion plans based on both low-level feedback provided by sensors and high-level feedback provided by the identifier. Research on each component is described below.

Fine Motion Planning

Fine motion planning refers to planning fine-scale motions to make an assembly task successful in spite of significant uncertainties. A general approach was proposed [36.61] based on the concept of preimages in configuration space (*C*-space) [36.35] to devise motion strategies that would not fail in the presence of uncertainties. Given a goal state of the held part defined by a region of goal configurations, a *preimage* of the goal state encodes those configurations from which a commanded velocity will guarantee that the held part reaches the goal state recognizably in spite of location and velocity uncertainties (Fig. 36.19).

Given the initial location of the held part and the goal state, the preimage approach generates a motion plan in backward chaining by finding the preimage of the goal state associated with a commanded velocity and then the preimage of the preimage, and so on, until a preimage that includes the initial configuration of the held part is found. Figure 36.20 shows an example.

The sequence of commanded velocities associated with the sequence of preimages (starting from the initial preimage that includes the initial location) forms the motion plan that guarantees the success of the task. Starting from the initial preimage, each subsequent preimage in the sequence can be viewed as a subgoal state. In this approach, compliant motions are preferred wherever possible because they typically produce larger preimages [36.61] than pure positioning motions, and subgoal states are often contact states. The approach was further extended [36.62] to include modeling uncertainties of objects.

However, the computability of the approach is a major problem. The requirement of both goal reachability and recognizability under sensing uncertainties, which are intertwined issues, complicates the computation of preimages. It has been shown [36.63] that the time complexity of generating a plan can be double exponential in nmr, where n is the number of plan steps, m is the environment complexity (of the physical space), and r is the dimension of the configuration space. By separating reachability and recognizability and restricting the recognizability power of the original preimage model, computability was improved [36.64, 65].

As an alternative, a two-phase approach is to simplify the fine motion planning into (1) global and offline nominal path planning assuming no uncertainty and (2) local and online replanning to deal with unintended contacts due to uncertainties. Different variations of the two-phase approach have been proposed [36.66–72]. The success of such an approach depends on successful online identification of contact states (Sect. 36.4.3).

Several researchers also studied the representation and propagation of uncertainties and constraints that have to be satisfied for the success of a task [36.73–77].

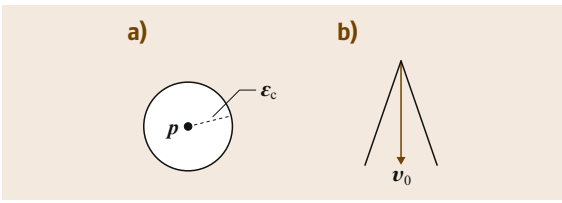


Fig.36.19a,b Location and velocity uncertainties (in the *C*-space). **(a)** The actual configuration of a part could be inside an uncertainty ball centered at an observed configuration. **(b)** The actual velocity of a part could be inside an uncertainty cone of a commanded velocity

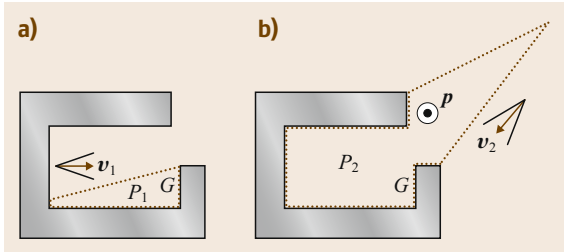


Fig.36.20a,b Backward chaining of preimages: P_1 is the preimage of the goal region G **(a)**, and P_2 is the preimage of P_1 **(b)**

Compliant Motion Planning

Compliant motion planning focuses on planning motions of objects always in contact. *Hopcroft* and *Wilfong* [36.78] proved that, if two arbitrary objects in contact can be moved to another configuration where they are also in contact, then there is always a path of contact configurations (on the boundary of configuration space obstacles, or C-obstacles) connecting the first and second contact configurations. Therefore, not only is compliant motion desirable in many cases, but it is always possible given initial and goal contact configurations.

Because compliant motion occurs on the boundary of C-obstacles, planning compliant motion poses special challenges not present in collision-free motion planning: it requires the exact information of contact configurations on the boundary of C-obstacles. Unfortunately computing C-obstacles exactly remains a formidable task to date. While there are exact descriptions of C-obstacles for polygons [36.79, 80], which are three dimensional, there are only approximations of C-obstacles for polyhedra [36.81, 82], which are six dimensional. If m and n indicate the complexity of two polyhedral objects in contact, the complexity of the corresponding C-obstacle is $\Theta(m^6n^6)$ [36.83].

Researchers have focused on either reducing the dimensionality and scope of the problem or getting around the problem of computing C-obstacles altogether. Some researchers have studied compliant motion planning on C-obstacles of reduced dimensions [36.72, 84, 85]. It is more popular to plan compliant motions based on a predetermined graph of topological contact states to avoid the problem of computing C-obstacles [36.86–89]. In particular, one approach [36.87, 88] models an assembly task of polygonal parts as a discrete event system using Petri nets. However, contact states and transitions are often generated manually, which is awfully tedious for even assembly tasks of simple geometry [36.45] and is practically infeasible for complex tasks due to the huge number of different contact states.

Therefore, automatic generation of a contact state graph is desirable and even necessary. A method was first developed to enumerate all possible contact states and their connections between two convex polyhedral objects [36.90]. Recall that a contact state between two convex polyhedra can be described by a single principal contact (Sect. 36.4.1), and any principal contact between two topological surface elements of two convex polyhedra describes a geometrically valid contact state. These are nice properties and greatly simplify the problem of contact state graph generation. In general, however, to construct a contact state graph between two objects automatically requires the handling of two rather difficult issues:

1. How to generate *valid* contact states, i. e., how to tell if a set of principal contacts corresponds to a geometrically valid contact state, given the geometries of objects, and
2. How to link one valid contact state to another in the graph, i. e., how to find the neighboring (or adjacency) relations among the regions of contact configurations belonging to different contact states in the contact configuration space.

A general and efficient divide-and-merge approach was introduced [36.38] for automatically generating a contact state graph between two arbitrary polyhedra. Each node in the graph denotes a contact state, described by a topological *contact formation* (CF) [36.37] as a set of principal contacts and a configuration satisfying the CF. Each edge connects the nodes of two neighboring contact states. Figure 36.21 shows an example contact state graph between two planar parts.

The approach handled the above two issues simultaneously by directly exploiting both topological and geometrical knowledge of contacts in the *physical space* of objects and by dividing the problem into simpler subproblems of generating and merging special subgraphs. Specifically, the approach takes advantage of the fact that a contact state graph can be divided into special subgraphs called the *goal-contact relaxation* (GCR) graphs, where each GCR graph is defined by a locally most constrained valid contact state, called the seed, and its less-constrained neighboring valid contact states, which is easier to generate because of several properties. The main properties include:

- Given a valid contact state CS_i all of its less constrained neighboring contact states can be hy-

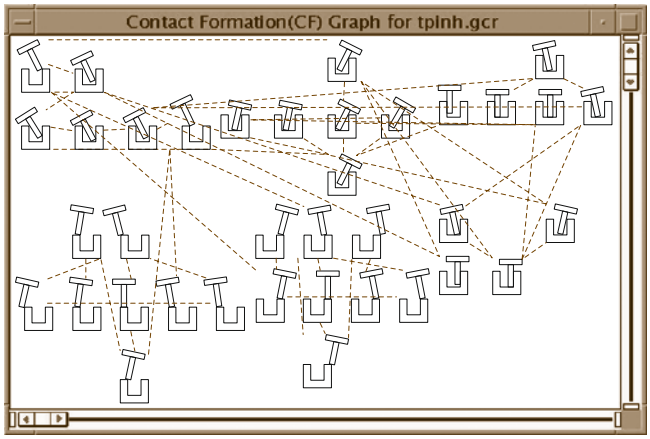


Fig. 36.21 A contact state graph between two planar parts

pothesized topologically from the principal contacts in CS_i.
- A hypothesized less constrained neighboring contact state CS_j is valid, if and only if there exists a compliant motion to relax certain constraints of CS_i to obtain CS_j which does not result in any other contact state, called neighboring relaxation.
- Neighboring relaxation can often be achieved by instantaneous compliant motion.

With this approach, a contact state graph of several hundreds or thousands of nodes and links can be generated in a few seconds.

With a contact state graph, the problem of compliant motion planning can be decomposed into two simpler subproblems at two levels: (1) high level: graph search for state transitions from one node to another in a contact state graph, and (2) low level: contact motion planning within the set of contact configurations constrained by the *same* contact state (and the same contact formation), called *CF-compliant motion planning*. A general contact motion plan crossing several contact states can be considered as consisting of segments of CF-compliant motions in different contact states. One approach plans [36.91] CF-compliant motions based on random sampling of CF-compliant configurations and extending the probabilistic roadmap motion planning technique [36.92].

Contact State Identification

Successful execution of a fine-motion or compliant motion plan depends on correct online identification of contact states during execution. Contact state identification uses both model information of contact states (including topological, geometrical and physical information) and sensory information of location and force/torque of the end-effector. The presence of sensing uncertainties make the identification problem nontrivial. Approaches for contact state identification are different in ways of dealing with uncertainties.

One class of approaches obtain the mapping between sensory data (in the presence of sensing uncertainties) and corresponding contact states through learning. Models for learning include hidden Markov models [36.93, 94], thresholds [36.95, 96], neural network structures, and fuzzy classifiers [36.97–101]. Training data are obtained either in an unsupervised way or by human task demonstration. Such approaches are task dependent: a new task or environment requires new training.

Another class of approaches are based on analytical models for contact states. A common strategy is to predetermine the set of configurations, constraints on configurations, or the set of forces/torques or force/torque constraints that are possible for each contact state and match such information against the sensed data to identify contact states online. Some approaches do not consider the effect of uncertainties [36.102, 103], while others consider uncertainties modeled either by uncertainty bounds or by probability distributions [36.36, 89, 104–108]. An alternative strategy is based on checking the distance between contacting elements in Cartesian space: either by growing the objects with pose uncertainty and intersecting the obtained regions [36.109], or by combining the distance between the objects with several other factors such as the instantaneous approach direction [36.110].

In terms of how sensory data are used, some identification schemes are rather static because they do not consider the prior motion and state identifications before the current contact state is reached, while others use prior history or even use new motion or *active sensing* to help current identification. The latter include approaches that perform simultaneous contact state identification and parameter estimation. In such approaches, an analytical contact state model is expressed as a function of uncertain parameters, such as the pose or dimension of an object in contact. At each contact state, some of the uncertain parameters can be observed or estimated with reduced uncertainty. For example, if the held object is at a face–face contact with another object, parameters describing the normals of the contacting faces can be estimated with the help of force/torque sensing. During a task execution (i. e., the execution of a motion plan), the uncertain parameters are estimated along with the identification of contact states with increasing accuracy, which in turn makes the subsequent contact state identifications more accurate. This also improves force control and contact state transition monitoring. The increased performance, however, comes at a higher computational cost.

Simultaneous contact state identification and parameter estimation is most often done by ruling out contact state models that are inconsistent with sensed data or parameter estimation results [36.111–117]. One notable exception [36.118] performs a truly simultaneous estimation by describing the system as a hybrid joint probability distribution of contact states and parameters.

Active sensing is about deliberate use of applied force/torque or motion to better aid contact state identification and parameter estimation. Earlier research focused on simple force strategy [36.119] or movability tests [36.36, 110]. More recently, methods were introduced [36.120] to design sequences of contact state transitions and compliant motion strategies that were optimized for active sensing to determine all uncertain

geometric parameters while achieving certain goal contact state.

From a different perspective, an approach was introduced to analyze contact state distinguishability and unknown/uncertain parameter identifiability [36.121] during the design phase of a contact state identifier or parameter estimator.

36.4.4 Compliant Motion Under Manipulator Constraints

A general compliant motion plan (as the output of a compliant motion or fine motion planner) usually consists of a sequence of contact states, and within each contact state, a path of contact configurations compliant to the contact state and leads to the next contact state in the sequence. Each contact configuration is described in terms of the parts in contact. However, it takes a robot manipulator to move one part against the others, especially in assembly tasks. Therefore, whether a contact state can be formed and whether a compliant motion and a contact state transition is possible is subject to the constraints of the manipulator.

Feasibility of Contact States and Compliant Motions

A contact state graph should be re-visited by considering manipulator constraints [36.122]. Contact configurations compliant to a given contact state are sampled and applied inverse kinematics to check if there is a feasible manipulator joint-space configuration to realize the contact state. If no feasible manipulator configuration can be found after a number of samples are considered, the contact state is considered at least very difficult to reach, and is discarded from the graph as an infeasible state.

Given two feasible adjacent contact states, a compliant motion for their transition should also be checked for feasibility under manipulator constraints. The main issue here is how to deal with possible singularities along the compliant path (of contact configurations). Certain singularities can be escaped if the manipulator is redundant (i. e., with more than six degrees of freedom) [36.123]. With inescapable singularities, the compliant path of the moved object is not feasible and has to be either altered to avoid singularity [36.124] or abandoned. However, an altered path may no longer be compliant to the desired contact states. In practice, it is preferred to check the feasibility of a compliant path starting from the goal contact state of an assembly task backward to make sure that the feasible portion leading to the goal as well as the feasible contact states encountered are discovered and preserved [36.125].

Execution of Feasible Compliant Motion Plans

Depending on whether the manipulator is equipped with a force/torque sensor or not, different control schemes can be considered for executing a compliant motion plan that includes multiple contact state transitions (VIDEO 356).

If the manipulator has a force/torque sensor, a hybrid position/force controller can be used. It requires a specification that separates dimensions of force control from those of position/velocity control. Such specifications not only have to vary for different contact states but may have to vary from one contact configuration to another within the same contact state, i. e., a control specification is a function of time or configuration along a compliant trajectory in general. Moreover, it is far from trivial to make control specifications for complex contact states involving multiple principal contacts [36.126].

An approach was introduced [36.127] to convert automatically a compliant path of contact configurations across a sequence of contact states into wrench, twist, and position control signals $\boldsymbol{w}(t)$, $\boldsymbol{t}(t)$, and $\boldsymbol{p}(t)$ for a hybrid controller to execute the plan. The approach was experimentally verified successfully.

More recently, an approach was introduced [36.128] to achieve force control by estimating contact force/torque from the joint torques without using a force sensor for the end-effector force/torque or torque sensors for individual joints. By reducing the integral gain of the joint PID (proportional–integral–derivative) controller, the approach acts like a high-pass filter to detect sharp force/torque changes more effectively, which often indicates contact state changes. Combining such detection with position control, the approach was able to execute an assembly motion plan successfully, even though compliant motion could not always be maintained with position control.

Other approaches to realize force control without a force sensor requires the dynamic model of the manipulator [36.128–130].

However, there is not yet a full integration of planning, online contact state identification, online replanning (to deal with contact states that are off the preplanned path due to uncertainty), and compliant motion control.

36.5 Unifying Feedback Control and Planning

In previous sections we discussed task-level control methods and planning algorithms. Both task-level control and planning methods are able to address specific constraints imposed on robot motion in the context of manipulation tasks. However, as illustrated in Fig. 36.3, neither of these categories of methods is able to address all constraints completely. Control remains susceptible to local minima and thus cannot guarantee that a particular motion will lead to the desired result. Planning methods, on the other hand, overcome the problem of local minima by projecting possible motions into the future to predict if a sequence of motion will lead to success. Given certain assumptions, the resulting plan can be guaranteed to succeed. However, the computations associated with this process are generally too computationally complex to satisfy the feedback requirements of manipulation tasks. Unfortunately, this means that control and motion planning by themselves are unable to address the problem of moving robots for general manipulations tasks.

In this final section of this chapter, we will review efforts to combine the advantages of control methods and planning methods into a single approach to determine the motion of a robot. These efforts aim to create methods that avoid the susceptibility to local minima while satisfying the feedback requirements. Early attempts of integrating planning and control occurred in the early 1990s. Not all of these efforts are specifically directed at manipulation but many contain insights relevant to the topic. In this section, we will review these efforts and also discuss some of the more recent research efforts aimed at unifying motion planning and feedback control.

Motion planning (Chap. 7) and motion control (Chap. 8) have traditionally been regarded as two distinct areas of research. However, these areas have many features in common. Both areas are concerned with the motion of robotic mechanisms, and more importantly planning and control methods both determine a representation that maps robot state to robot motion. In the case of feedback control, this representation is a potential function defined for a given region of the state space. The gradient of the potential function at a specific state encodes the motion command. In contrast, motion planning determines motion plans. These motion plans also encode motions for a set of states.

Motion planning and feedback control also differ in certain aspects. A motion planner generally makes stronger assumptions than a controller about the environment, the ability to assess its state, and the changes that can occur in the environment. Motion planning also requires the ability to project the robot's state into the future, given its current state and a particular action. By using this ability together with global information about the environment, motion planners can determine motions that are not susceptible to local minima. This positive characteristic of motion planners, however, results in significantly increased computational cost (Chap. 7). The large discrepancy in computational requirements for planning and control and the resulting divergence of computational techniques may explain the current separation of the two fields. Researchers are beginning to attempt to reverse this separation by devising a unified theory of planning and control.

36.5.1 Feedback Motion Planning

Feedback motion planning combines planning and feedback into a single motion strategy. A planner considers global information to compute a feedback motion plan free of local minima. Such a feedback motion plan can be interpreted as a potential function or vector field whose gradient will lead the robot to the goal state from any reachable part of the state space [36.18]. Local minima-free potential functions are also called navigation functions [36.131, 132]. Given the current state of the robot and the global navigation function, feedback control is used to determine the robot's motion. The consideration of feedback about the robot's state in the context of a global navigation function reduces the susceptibility to sensing and actuation uncertainty.

Feedback motion planning, in principle, can address the entire spectrum of motion constraints and their feedback requirements (Fig. 36.3). Given a global navigation function that considers all motion constraints, the feedback requirements can easily be satisfied, since the feedback motion plan already specifies the desired motion command for the entire state space. Obviously, the major challenge in feedback motion planning is the computation of such a navigation function or feedback motion plan. The problem becomes particularly difficult in the context of a manipulation task, since the state space (or configuration space) changes each time the robot grasps or releases an object in the environment (Sect. 36.3). This change makes it necessary to recompute the feedback motion plan. Frequent recomputation is also necessary in dynamic environments, where the motion of obstacles can repeatedly invalidate a previously computed feedback motion plan.

In the remainder of this section, we will review a variety of methods for computing navigation functions. In general, the problem of efficiently computing feedback motion plans for manipulation tasks remains unsolved. We therefore also present methods that do not explicitly

consider manipulation. These methods can be divided into three categories: (1) exact methods, (2) approximate methods based on dynamic programming, and (3) approximate methods based on composing and sequencing simpler potential functions.

The earliest exact methods for the computation of navigation functions are applicable to simple environments with obstacles of specific shapes [36.131–133]. Approximate methods based on discretized spaces (grids) overcome this limitation but possess an exponential computational complexity in the number of dimensions of the state space. These approximate navigation functions are called numerical navigation functions [36.134]. These navigation functions are often used for applications in mobile robotics. Due to the low-dimensional configuration space associated with mobile robots, they can generally solve motion planning problems robustly and efficiently.

Some physical processes, such as heat transfer or fluid flow, can be described by a specific type of differential equation, called harmonic functions. These functions possess properties that make them suitable as navigation functions [36.135–139]. Navigation functions based on harmonic functions are most commonly computed in an approximate, iterative fashion. The requirement for iterative computation increases the computational cost relative to the simpler numerical navigation functions [36.134].

More recent methods for the computation of numerical navigation functions consider differential motion constraints at a significantly reduced computational cost [36.140]. These methods rely on classical numerical dynamic programming techniques and numerical optimal control. However, in spite of their reduced computational complexity, these methods remain too computationally costly to be applied to manipulation tasks with many degrees of freedom in dynamic environments.

Navigation functions can also be computed by composing local potential functions based on global information. This is illustrated in Fig. 36.22. The goal is to compute a navigation function for the entire configuration space C. This is accomplished by sequencing overlapping funnels. Each of the funnels represents a simple, local potential function. By following the gradient of this potential function, motion commands can be determined for a subset of the configuration space. If the funnels are sequenced correctly, the composition of local funnels can yield a global feedback plan. This feedback plan can be viewed as a hybrid system [36.141] in which the sequencing of funnels represents a discrete transition structure, whereas the individual controllers operate in a continuous domain. The composition of funnels is considered planning. The consideration of global information during the planning permits to determine a funnel composition that avoids local minima.

One of the earliest methods based on the global composition of local funnels was proposed by *Choi* and *Latombe* [36.142]. In this method, the state space of the robot is decomposed into convex regions. The connectivity of these regions is analyzed to determine global information about the state space. This information can be used to combine simple, local potential functions, one for each convex regions of state space, into a local minima-free potential function. More rigorous approaches based on this idea have been developed. These approaches can consider the dynamics of the robot and nonholonomic motion constraints [36.143–145]. These methods have only been applied to low-dimensional state spaces and cannot easily be applied to manipulation tasks.

The random neighborhood graph [36.146] is a sampling-based method of computing a decomposition of the overall state space. As before, a navigation function can be computed by analyzing the global connectivity of the decomposition and imposing adequate local potential functions for each subdivision. A specialized method following the same principle for planar robots in polygonal environments has also been proposed [36.147]. The idea of composing local potential functions has also been applied successfully to a complex robot control task [36.148]. Finally, in [36.149] a general and efficient method of computing a smooth feedback plan over a cylindrical algebraic decomposition of configuration space has been proposed.

The computation of a navigation function over the entire configuration space quickly becomes intractable as the dimensionality of the space increases. To overcome this challenge, in particular in the context of autonomous mobile manipulation, workspace heuristics have been employed to determine a navigation function efficiently. This navigation function does not cover the entire configuration space but only those regions heuristically determined to be relevant to the motion problem [36.150]. This method of generating feedback

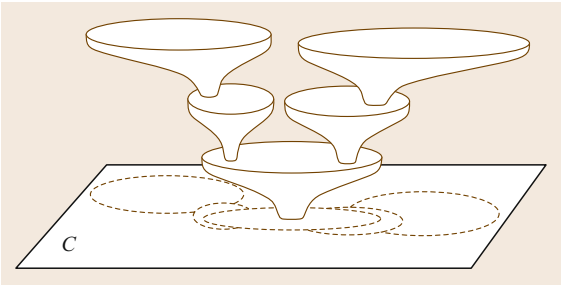


Fig. 36.22 Composing funnels into a navigation function

plans is able to satisfy the various motion constraints depicted in Fig. 36.3 and their respective feedback requirements.

36.5.2 Augmenting Global Plans with Feedback

The manipulation planning techniques described in Sect. 36.3 are not susceptible to local minima, as they consider global state space information. Due to the computational complexity associated with the consideration of global information, these planning techniques are not able to satisfy the feedback requirements of manipulation tasks. However, the required frequency of feedback about global motion are relatively low: the global connectivity of the configuration space changes relatively infrequently (Fig. 36.3). It would therefore be possible to consider feedback for global motion at the slow rates the planner can accommodate, while considering feedback for other motion constraints at higher frequencies. To achieve this, global motion plans have to be augmented with reactive components that incrementally modify the global plan in response to feedback from the environment. As long as the global connectivity information captured by the plan remains valid, the incremental modifications can ensure that all other motion constraints, ranging from task requirements to reactive obstacles avoidance, are satisfied.

The elastic-band framework [36.151] augments global plans with reactive obstacle avoidance. A global configuration space path, determined by a planner, is covered with local potential functions, each of which is derived from the local distribution of obstacles around the path. These local potentials cause the path to deform so as to maintain a minimum distance from obstacles. Visually, the path behaves as an elastic band that is deformed by the motion of obstacles. The local potential functions, together with the global path, can be viewed as a navigation function for a local region of the configuration space. Integrated with a global planner and replanner, the elastic-band framework permits real-time obstacle avoidance that is not susceptible to local minima. However, the feedback frequency for global motion remains limited by the global motion planner. Specific task constraints have not been integrated into the elastic-band framework; consequently, its application to manipulation tasks is limited.

In its original formulation, the elastic-band framework assumed that all degrees of freedom of the robot are holonomic. An extended formulation augments motion paths for nonholonomic platforms with reactive components [36.152].

The elastic-strip framework [36.153] also augments global motion plans with reactive obstacle avoidance. In addition to reactive obstacle avoidance, however, the elastic-strip framework can accommodate task constraints. Similarly to an elastic band, an elastic strip covers a global path with local potential functions. In contrast to the elastic-band framework, these potential functions are based on task-level controllers (Sect. 36.2) and therefore allow the task-consistent modification of the global path. The elastic-strip framework is therefore well suited for the execution of manipulation plans in dynamic environments. An elastic strip will be incrementally modified to represent a constraint-consistent trajectory, as long as the global information captured by the underlying plan remains valid. The elastic-strip framework has been applied to a variety of manipulation tasks on a mobile manipulation platform.

Extending the elastic-band and elastic-strips frameworks, the elastic-roadmap framework combines reactive task-level control with efficient global motion planning [36.150]. The elastic roadmap represents a hybrid system of task-level controllers that are composed into a navigation function, thereby satisfying the motion constraints depicted in Fig. 36.3 and their respective feedback requirements.

36.6 Conclusions and Further Reading

In this chapter, an overview of motion generation and control strategies in the context of robotic manipulation tasks has been provided. Issues related to modeling the interfaces between the robot and the environment at the different time scales of motion and incorporating sensing and feedback were considered. Manipulation planning was introduced as an extension to the basic motion planning problem, which can be modeled as a hybrid system of continuous configuration spaces arising from the act of grasping and moving parts in the environment. The important example of assembly motion has been discussed through the analysis of contact states and compliant motion control. The operational space framework, as described in Sect. 36.2, permits the position and force control of operational points on a serial-chain manipulator. A number of extensions have been proposed in the literature, extending the framework to situations with multiple concurrent contact points, cooperative manipulation scenarios, and branching kinematic chains. For manipulation plan-

ning, extensions involving grasp and regrasp planning, multiple robots, multiple parts, and movable obstacles have been considered. In this section we will briefly discuss these extensions and refer to the appropriate literature.

36.6.1 General Contact Models

The hybrid force/motion control described by *Raibert* and *Craig* [36.154] has been shown to have shortcomings [36.155]. A general contact model for dynamically decoupled force/motion control in the context of operational space control that overcomes these problems has been presented in [36.155]. This general contact model has been extended into a compliant motion control framework to enable force control for multiple contact points [36.126] in nonrigid environments, and to enable force control for multiple contact points on different links of a kinematic chain [36.156]. The operational space framework for task-level control has been applied for the real-time simulation of complex dynamic environments [36.157]. Contact points between objects in the environment are modeled as operation points, resulting in a mathematically elegant framework for resolving impulses and contact constraints for moving bodies.

36.6.2 Cooperative Manipulation Control

If multiple task-level controlled robots collaborate to manipulate an object, they form a closed kinematic chain that is connected through the object. The dynamics of the entire system can be described using the notion of an augmented object [36.158, 159], in which the dynamics of the manipulators and the object are combined to form a model of the overall system. The internal forces that occur at the grasp points during motion can be modeled using the virtual linkage framework [36.160]. A number of alternative strategies for the cooperative manipulation of multiple robots outside of the operational space framework have been proposed [36.161–168].

36.6.3 Control of Branching Mechanisms

So far, we have implicitly assumed that the task-controlled robot consists of a single kinematic chain. This assumption does not hold in the case of kinematically more complex mechanisms, such as humanoid robots (Chap. 67). These robots can consist of multiple branching kinematic chains. If we consider the torso of a humanoid robot to be the robot's base, for example, then the legs, arms, and the head represent five kinematic chains attached to this base. We call such a mechanism a branching mechanism if it does not contain any closed kinematic loops. Tasks performed by a branching mechanism may require the specification of operational points on any one of those branches; for example, while the legs perform locomotion the hands achieve a manipulation task and the head is oriented to maintain visibility of the manipulated object.

The operational space framework has been extended to task-level control of branching mechanisms [36.169]. This extension combined operational points and associated Jacobians and computes an operational space inertia matrix that combines all operational points. This matrix can be computed efficiently with an algorithm that in practice is linear in the number of operational points [36.170, 171]. More recently, the operational space framework has been applied to quadrupedal locomotion [36.172].

36.6.4 Nonholonomic Mobile Manipulation

All previously discussed work on task-level control assumes that the degrees of freedom are holonomic (Chap. 40). For manipulator arms this is generally a valid assumption. The most common type of mobility platform, however, is based on either differential drives, synchro-drives, or Ackerman steerings (point to a section), all of which are subject to nonholonomic constraints. The operational space framework has been extended to mobile manipulation platforms that combine a nonholonomic mobility platform with a holonomic manipulator arm [36.173–175], enabling the task-level control of a large class of mobile manipulation platforms.

36.6.5 Learning Models with Uncertainty

The efficacy of operational space control depends on the accuracy of the dynamic model of the robot. In particular when multiple behaviors are executed using null-space projections, modeling errors can have significant effects. To overcome the reliance on accurate dynamic models, reinforcement learning can be used to learn operational space controllers [36.176].

36.6.6 Grasping and Regrasp Planning

Deciding the intermediate stable configurations for a part necessary to complete a manipulation task and selecting proper grasps are both difficult problems in themselves that operate over a continuum of possible solutions. *Simeon* et al. developed a manipulation planning framework that considers continuous grasps and placements [36.177]. However, selecting from among all possible grasp configurations and deciding when

to regrasp is still an open research problem. For an overview of various grasp quality metrics, see *Miller* and *Allen* [36.178, 179] and Chap. 38.

36.6.7 Multiple Parts

The manipulation planning framework nicely generalizes to multiple parts, $\mathcal{P}_1, \ldots, \mathcal{P}_k$. Each part has its own C-space, and C is formed by taking the Cartesian product of all the part C-spaces with the manipulator C-space. The set C_{adm} is defined in a similar way, but now part–part collisions also have to be removed, in addition to part–manipulator, manipulator–obstacle, and part–obstacle collisions. The definition of C_{stable} requires that all parts be in stable configurations; the parts may even be allowed to stack on top of each other. The definition of C_{grasp} requires that one part is grasped and all other parts are stable. There are still two modes, depending on whether the manipulator is grasping a part. Once again, transitions occur only when the robot is in $C_{\text{trans}} = C_{\text{stable}} \cap C_{\text{grasp}}$.

36.6.8 Planning for Multiple Robots

Generalizing to k robots would lead to $2k$ modes, in which each mode indicates whether each robot is grasping a part. Multiple robots may even be allowed to grasp the same part, which leads to the interesting problem of planning for closed kinematic chains and cooperative motion (Chap. 39). *Koga* addressed multi-arm manipulation planning where the same object must be grasped and moved by several manipulator arms [36.21]. Another generalization could allow a single robot to grasp more than one part simultaneously.

36.6.9 Planning for Closed Kinematic Chains

The subspace of C that results from maintaining kinematic closure arises when multiple robots grasp the same part, or even when multiple fingers of the same hand grasp a single part (Chap. 19). Planning in this context requires that paths remain on a lower-dimensional variety for which a parameterization is not available. Planning the motion of parallel mechanisms or other systems with loops typically requires maintaining multiple closure constraints simultaneously (Chap. 18).

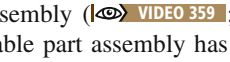
36.6.10 Planning with Movable Obstacles

In some cases, the robot may be allowed to reposition obstacles in the environment rather than simply avoid them. *Wilfong* first addressed motion planning in the presence of movable obstacles [36.180]. Wilfong proved that the problem is PSPACE-hard even in two-dimensional environments where the final positions of all movable objects are specified. *Erdmann* and *Lozano-Perez* considered coordinated planning for multiple moving objects [36.181]. *Alami* et al. presented a general algorithm for the case of one robot and one movable object and formulated the space of grasping configurations into a finite number of cells [36.17]. *Chen* and *Hwang* developed a planner for a circular robot that is allowed to push objects aside as it moves [36.182]. *Stilman* and *Kuffner* considered movable obstacles in the context of navigation planning [36.183], and manipulation planning [36.184]. *Nieuwenhuisen* et al. also developed a general framework for planning with movable obstacles [36.185].

36.6.11 Nonprehensile Manipulation

Lynch and *Mason* explored scenarios in which grasping operations are replaced by pushing operations [36.186]. The space of stable pushing directions imposes nonholonomic constraints on the motion of the robot, which opens up issues related to controllability. Related issues also arise in the context of part-feeders and manipulation for manufacturing and assembly (Chap. 54).

36.6.12 Assembly Motion Extensions

The work in assembly motion described in this chapter has focused on rigid-part assembly, and the parts considered are mostly polyhedral or of simple nonpolyhedral shapes, such as round pegs and holes. More recent work on contact state analysis for curved objects can be found in [36.187]. An emerging field in assembly is micro/nanoassembly ( ; Chap. 27). Flexible or deformable part assembly has also begun to attract attention [36.188–190]. Virtual assembly [36.191] for simulation and prototyping is another interesting area of study.

Video-References

VIDEO 356 Reducing uncertainty in robotics surface assembly tasks
available from http://handbookofrobotics.org/view-chapter/36/videodetails/356

VIDEO 357 Autonomous continuum grasping
available from http://handbookofrobotics.org/view-chapter/36/videodetails/357

VIDEO 358 Robotic assembly of emergency stop buttons
available from http://handbookofrobotics.org/view-chapter/36/videodetails/358

VIDEO 359 Meso-scale manipulation: System, modeling, planning and control
available from http://handbookofrobotics.org/view-chapter/36/videodetails/359

VIDEO 360 Grasp and multi-fingers-3 cylindrical-peg-in-hole demonstration using manipulation primitives
available from http://handbookofrobotics.org/view-chapter/36/videodetails/360

VIDEO 361 Demonstration of multi-sensor integration in industrial manipulation
available from http://handbookofrobotics.org/view-chapter/36/videodetails/361

VIDEO 363 Control pre-imaging for multifingered grasp synthesis
available from http://handbookofrobotics.org/view-chapter/36/videodetails/363

VIDEO 364 Robust and fast manipulation of objects with multi-fingered hands
available from http://handbookofrobotics.org/view-chapter/36/videodetails/364

VIDEO 366 Whole quadruped manipulation
available from http://handbookofrobotics.org/view-chapter/36/videodetails/366

VIDEO 367 The mobipulator
available from http://handbookofrobotics.org/view-chapter/36/videodetails/367

VIDEO 368 Handling of a single object by multiple mobile robots based on caster-like dynamics
available from http://handbookofrobotics.org/view-chapter/36/videodetails/368

VIDEO 369 Rollin' Justin – Mobile platform with variable base
available from http://handbookofrobotics.org/view-chapter/36/videodetails/369

VIDEO 370 Learning to place new objects
available from http://handbookofrobotics.org/view-chapter/36/videodetails/370

References

36.1 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT Press, Cambridge 2001)

36.2 H. Inoue: *Force Feedback in Precise Assembly Tasks*, Tech. Rep. Vol. 308 (Artificial Intelligence Laboratory, MIT, Cambridge 1974)

36.3 T. Lozano-Pérez, M. Mason, R.H. Taylor: Automatic synthesis of fine-motion strategies for robots, Int. J. Robotics Res. **31**(1), 3–24 (1984)

36.4 O. Khatib: A unified approach to motion and force control of robot manipulators: The operational space formulation, Int. J. Robotics Autom. **3**(1), 43–53 (1987)

36.5 O. Khatib, K. Yokoi, O. Brock, K.-S. Chang, A. Casal: Robots in human environments, Arch. Control Sci. **11**(3/4), 123–138 (2001)

36.6 G. Strang: *Linear Algegra and Its Applications* (Brooks Cole, New York 1988)

36.7 H. Seraji: An on-line approach to coordinated mobility and manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta, Vol. 1 (1993) pp. 28–35

36.8 J.M. Cameron, D.C. MacKenzie, K.R. Ward, R.C. Arkin, W.J. Book: Reactive control for mobile manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta, Vol. 3 (1993) pp. 228–235

36.9 M. Egerstedt, X. Hu: Coordinated trajectory following for mobile manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000)

36.10 P. Ögren, M. Egerstedt, X. Hu: Reactive mobile manipulation using dyanmic trajectory tracking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francicso (2000) pp. 3473–3478

36.11 J. Tan, N. Xi, Y. Wang: Integrated task planning and control for mobile manipulators, Int. J. Robotics Res. **22**(5), 337–354 (2003)

36.12 Y. Yamamoto, X. Yun: Unified analysis on mobility and manipulability of mobile manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Detroit (1999) pp. 1200–1206

36.13 L. Sentis, O. Khatib: Control of free-floating humanoid robots through task prioritization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Barcelona (2005)

36.14 L. Sentis, O. Khatib: Synthesis of whole-body behaviors through hierarchical control of behavioral primitives, Int. J. Hum. Robotics **2**(4), 505–518 (2005)

36.15 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robotics Res. **5**(1), 90–98 (1986)

36.16 M. Huber, R.A. Grupen: A feedback control structure for on-line learning tasks, Robotics Auton. Syst. **22**(3-4), 303–315 (1997)

36.17 R. Alami, J.P. Laumond, T. Siméon: Two manipulation planning algorithms, Proc. Workshop Algorithm. Found. Robotics (1994)

36.18 S.M. LaValle: *Planning Algorithms* (Cambridge Univ., Cambridge 2006)

36.19 R. Grossman, A. Nerode, A. Ravn, H. Rischel (Eds.): *Hybrid Systems* (Springer, Berlin, Heidelberg 1993)

36.20 J.M. Ahuactzin, K. Gupta, E. Mazer: Manipulation planning for redundant robots: A practical approach, Int. J. Robotics Res. **17**(7), 731–747 (1998)

36.21 Y. Koga: On Computing Multi-Arm Manipulation Trajectories, Ph.D. Thesis (Stanford University, Stanford 1995)

36.22 D. Bertram, J.J. Kuffner, T. Asfour, R. Dillman: A unified approach to inverse kinematics and path planning for redundant manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1874–1879

36.23 D. Trivedi, C.D. Rahn, W.M. Kier, I.D. Walker: Soft robotics: Biological inspiration, state of the art, and future research, Appl. Bionics Biomech. **5**(3), 99–117 (2008)

36.24 B.A. Jones, I.D. Walker: Kinematics for multisection continuum robots, IEEE Trans. Robotics **22**(1), 43–55 (2006)

36.25 S. Neppalli, M.A. Csencsits, B.A. Jones, I. Walker: A geometrical approach to inverse kinematics for continuum manipulators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 3565–3570

36.26 J. Li, J. Xiao: Determining *grasping* configurations for a spatial continuum manipulator, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 4207–4214

36.27 J. Xiao, R. Vatcha: Real-time adaptive motion planning for a continuum manipulator, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 5919–5926

36.28 J. Li, J. Xiao: Progressive generation of force-closure grasps for an n-section continuum manipulator, Proc. IEEE Int. Conf. Robotics Auto. (ICRA) (2013) pp. 4016–4022

36.29 J. Li, J. Xiao: Progressive, continuum grasping in cluttered space, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 4563–4568

36.30 G.S. Chirikjian, J.W. Burdick: Kinematically optimal hyperredundant manipulator configurations, IEEE Trans. Robotics Autom. **11**(6), 794–798 (1995)

36.31 G. Boothroyd: *Assembly Automation and Product Design*, 2nd edn. (Taylor Francis, Boca Raton 2005)

36.32 D. Whitney, O.L. Gilbert, M. Jastrzebski: Representation of geometric variations using matrix transforms for statistical tolerance analysis in assemblies, Res. Eng. Des. **64**, 191–210 (1994)

36.33 P. Jimenez: Survey on assembly sequencing: A combinatorial and geometrical perspective, J. Intell. Manuf. **l24**, 235–250 (2013)

36.34 B. Shirinzadeh: Issues in the design of the reconfigurable fixture modules for robotic assembly, J. Manuf. Syst. **121**, 1–14 (1993)

36.35 T. Lozano-Pérez: Spatial planning: A configuration space approach, IEEE Trans. Comput. **C-32**(2), 108–120 (1983)

36.36 R. Desai: On Fine Motion in Mechanical Assembly in Presence of Uncertainty, Ph.D. Thesis (University of Michigan, Ann Arbor 1989)

36.37 J. Xiao: Automatic determination of topological contacts in the presence of sensing uncertainties, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 65–70

36.38 J. Xiao, X. Ji: On automatic generation of high-level contact state space, Int. J. Robotics Res. **20**(7), 584–606 (2001)

36.39 S.N. Simunovic: Force information in assembly processes, Proc. 5th Int. Symp. Ind. Robots (1975) pp. 415–431

36.40 S.H. Drake: Using Compliance in Lieu of Sensory Feedback for Automatic Assembly, Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 1989)

36.41 D.E. Whitney: Quasi-static assembly of compliantly supported rigid parts, ASME J. Dyn. Syst. Meas. Control **104**, 65–77 (1982)

36.42 R.L. Hollis: A six-degree-of-freedom magnetically levitated variable compliance fine-motion wrist: Design, modeling, and control, IEEE Trans. Robotics Autom. **7**(3), 320–332 (1991)

36.43 S. Joo, F. Miyazaki: Development of variable RCC and ITS application, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Victoria, Vol. 3 (1998) pp. 1326–1332

36.44 H. Kazerooni: Direct-drive active compliant end effector (active RCC), IEEE J. Robotics Autom. **4**(3), 324–333 (1988)

36.45 R.H. Sturges, S. Laowattana: Fine motion planning through constraint network analysis, Proc. Int. Symp. Assem. Task Plan. (1995) pp. 160–170

36.46 D.E. Whitney: Historic perspective and state of the art in robot force control, Int. J. Robotics Res. **6**(1), 3–14 (1987)

36.47 M. Peshkin: Programmed compliance for error corrective assembly, IEEE Trans. Robotics Autom. **6**(4), 473–482 (1990)

36.48 J.M. Schimmels, M.A. Peshkin: Admittance matrix design for force-guided assembly, IEEE Trans. Robotics Autom. **8**(2), 213–227 (1992)

36.49 J.M. Schimmels: A linear space of admittance control laws that guarantees force assembly with friction, IEEE Trans. Robotics Autom. **13**(5), 656–667 (1997)

36.50 S. Hirai, T. Inatsugi, K. Iwata: Learning of admittance matrix elements for manipulative operations, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1996) pp. 763–768

36.51 S. Lee, H. Asada: A perturbation/correlation method for force guided robot assembly, IEEE Trans. Robotics Autom. **15**(4), 764–773 (1999)

36.52 H. Asada: Representation and learning of nonlinear compliance using neural nets, IEEE Trans. Robotics Autom. **9**(6), 863–867 (1993)

36.53 J. Simons, H. Van Brussel, J. De Schutter, J. Verhaert: A self-learning automaton with variable resolution for high precision assembly by industrial robots, IEEE Trans. Autom. Control **27**(5), 1109–1113 (1982)

36.54 V. Gullapalli, J.A. Franklin, H. Benbrahim: Acquiring robot skills via reinforcement learning, IEEE Control Syst. **14**(1), 13–24 (1994)

36.55 Q. Wang, J. De Schutter, W. Witvrouw, S. Graves: Derivation of compliant motion programs based on human demonstration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 2616–2621

36.56 R. Cortesao, R. Koeppe, U. Nunes, G. Hirzinger: Data fusion for robotic assembly tasks based on human skills, IEEE Trans. Robotics Autom. **20**(6), 941–952 (2004)

36.57 K. Ikeuchi, T. Suehiro: Toward an assembly plan from observation. Part I: Task recognition with polyhedral objects, IEEE Trans. Robotics Autom. **10**(3), 368–385 (1994)

36.58 H. Onda, H. Hirokawa, F. Tomita, T. Suehiro, K. Takase: Assembly motion teaching system using position/forcesimulator-generating control program, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 938–945

36.59 H. Onda, T. Suehiro, K. Kitagaki: Teaching by demonstration of assembly motion in VR – non-deterministic search-type motion in the teaching stage, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002) pp. 3066–3072

36.60 Y. Kobari, T. Nammoto, J. Kinugawa, K. Kosuge: Vision based compliant motion control for part assembly, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 293–298

36.61 T. Lozano-Pérez, M.T. Mason, R.H. Taylor: Automatic synthesis of fine-motion strategies for robot, Int. J. Robotics Res. **31**(1), 3–24 (1984)

36.62 B.R. Donald: *Error Detection and Recovery in Robotics* (Springer, Berlin, Heidelberg 1989)

36.63 J. Canny: On computability of fine motion plans, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 177–182

36.64 M. Erdmann: Using backprojections for fine motion planning with uncertainty, Int. J. Robotics Res. **5**(1), 19–45 (1986)

36.65 J.C. Latombe: *Robot Motion Planning* (Kluwer, Dordrecht 1991)

36.66 B. Dufay, J.C. Latombe: An approach to automatic programming based on inductive learning, Int. J. Robotics Res. **3**(4), 3–20 (1984)

36.67 H. Asada, S. Hirai: Towards a symbolic-level force feedback: Recognition of assembly process states, Proc. Int. Symp. Robotics Res. (1989) pp. 290–295

36.68 J. Xiao, R. Volz: On replanning for assembly tasks using robots in the presence of uncertainties, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 638–645

36.69 J. Xiao: Replanning with compliant rotations in the presence of uncertainties, Proc. Int. Symp. Intell. Control, Glasgow (1992) pp. 102–107

36.70 G. Dakin, R. Popplestone: Simplified fine-motion planning in generalized contact space, Proc. Int. Symp. Intell. Control (1992) pp. 281–287

36.71 G. Dakin, R. Popplestone: Contact space analysis for narrow-clearance assemblies, Proc. Int. Symp. Intell. Control (1993) pp. 542–547

36.72 J. Rosell, L. Basañez, R. Suárez: Compliant-motion planning and execution for robotic assembly, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 2774–2779

36.73 R. Taylor: The Synthesis of Manipulator Control Programs from Task-Level Specifications, Ph.D. Thesis (Stanford University, Stanford 1976)

36.74 R.A. Brooks: Symbolic error analysis and robot planning, Int. J. Robotics Res. **1**(4), 29–68 (1982)

36.75 R.A. Smith, P. Cheeseman: On the representation and estimation of spatial uncertaity, Int. J. Robotics Res. **5**(4), 56–68 (1986)

36.76 S.-F. Su, C. Lee: Manipulation and propagation of uncertainty and verification of applicability of actions in assembly tasks, IEEE Trans. Syst. Man Cybern. **22**(6), 1376–1389 (1992)

36.77 S.-F. Su, C. Lee, W. Hsu: Automatic generation of goal regions for assembly tasks in the presence of uncertainty, IEEE Trans. Robotics Autom. **12**(2), 313–323 (1996)

36.78 J. Hopcroft, G. Wilfong: Motion of objects in contact, Int. J. Robotics Res. **4**(4), 32–46 (1986)

36.79 F. Avnaim, J.D. Boissonnat, B. Faverjon: A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1988) pp. 1656–1661

36.80 R. Brost: Computing metric and topological properties of c-space obstacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 170–176

36.81 B. Donald: A search algorithm for motion planning with six degrees of freedom, Artif. Intell. **31**(3), 295–353 (1987)

36.82 L. Joskowicz, R.H. Taylor: Interference-free insertion of a solid body into a cavity: An algorithm and a medical application, Int. J. Robotics Res. **15**(3), 211–229 (1996)

36.83 H. Hirukawa: On motion planning of polyhedra in contact, Proc. Workshop Algorithm. Found. Robotics, Toulouse (1996) pp. 381–391

36.84 S.J. Buckley: Planning compliant motion strategies, Proc. Int. Symp. Intell. Control (1988) pp. 338–343

36.85 E. Sacks: Path planning for planar articulated robots using configuration spaces and compliant motion, IEEE Trans. Robotics Autom. **19**(3), 381–390 (2003)

36.86 C. Laugier: Planning fine motion strategies by reasoning in the contact space, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 653–659

36.87 B.J. McCarragher, H. Asada: A discrete event approach to the control of robotic assembly tasks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 331–336

36.88 B.J. McCarragher, H. Asada: The discrete event modeling and trajectory planning of robotic assembly tasks, ASME J. Dyn. Syst. Meas. Control **117**, 394–400 (1995)

36.89 R. Suárez, L. Basañez, J. Rosell: Using configuration and force sensing in assembly task planning and execution, Proc. Int. Symp. Assem. Task Plan. (1995) pp. 273–279

36.90 H. Hirukawa, Y. Papegay, T. Matsui: A motion planning algorithm for convex polyhedra in contact under translation and rotation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Diego (1994) pp. 3020–3027

36.91 X. Ji, J. Xiao: Planning motion compliant to complex contact states, Int. J. Robotics Res. **20**(6), 446–465 (2001)

36.92 L.E. Kavraki, P. Svestka, J.C. Latombe, M. Overmars: Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robotics Autom. **12**(4), 566–580 (1996)

36.93 B. Hannaford, P. Lee: Hidden Markov model analysis of force/torque information in telemanipulation, Int. J. Robotics Res. **10**(5), 528–539 (1991)

36.94 G.E. Hovland, B.J. McCarragher: Hidden Markov models as a process monitor in robotic assembly, Int. J. Robotics Res. **17**(2), 153–168 (1998)

36.95 T. Takahashi, H. Ogata, S. Muto: A method for analyzing human assembly operations for use in automatically generating robot commands, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2, Atlanta (1993) pp. 695–700

36.96 P. Sikka, B.J. McCarragher: Rule-based contact monitoring using examples obtained by task demonstration, Proc. 15th Int. Jt. Conf. Artif. Intell., Nagoya (1997) pp. 514–521

36.97 E. Cervera, A. Del Pobil, E. Marta, M. Serna: Perception-based learning for motion in contact in task planning, J. Intell. Robots Syst. **17**(3), 283–308 (1996)

36.98 L.M. Brignone, M. Howarth: A geometrically validated approach to autonomous robotic assembly, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Lausanne (2002) pp. 1626–1631

36.99 M. Nuttin, J. Rosell, R. Suárez, H. Van Brussel, L. Basañez, J. Hao: Learning approaches to contact estimation in assembly tasks with robots, Proc. 3rd Eur. Workshop Learn. Robotics, Heraklion (1995)

36.100 L.J. Everett, R. Ravari, R.A. Volz, M. Skubic: Generalized recognition of single-ended contact formations, IEEE Trans. Robotics Autom. **15**(5), 829–836 (1999)

36.101 M. Skubic, R.A. Volz: Identifying single-ended contact formations from force sensor patterns, IEEE Trans. Robotics Autom. **16**(5), 597–603 (2000)

36.102 S. Hirai, H. Asada: Kinematics and statics of manipulation using the theory of polyhedral convex cones, Int. J. Robotics Res. **12**(5), 434–447 (1993)

36.103 H. Hirukawa, T. Matsui, K. Takase: Automatic determination of possible velocity and applicable force of frictionless objects in contact from a geometric model, IEEE Trans. Robotics Autom. **10**(3), 309–322 (1994)

36.104 B.J. McCarragher, H. Asada: Qualitative template matching using dynamic process models for state transition recognition of robotic assembly, ASME J. Dyn. Syst. Meas. Control **115**(2), 261–269 (1993)

36.105 T.M. Schulteis, P.E. Dupont, P.A. Millman, R.D. Howe: Automatic identification of remote environments, Proc. ASME Dyn. Syst. Control Div., Atlanta (1996) pp. 451–458

36.106 A.O. Farahat, B.S. Graves, J.C. Trinkle: Identifying contact formations in the presence of uncertainty, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Pittsburg (1995) pp. 59–64

36.107 J. Xiao, L. Zhang: Contact constraint analysis and determination of geometrically valid contact formations from possible contact primitives, IEEE Trans. Robotics Autom. **13**(3), 456–466 (1997)

36.108 H. Mosemann, T. Bierwirth, F.M. Wahl, S. Stoeter: Generating polyhedral convex cones from contact graphs for the identification of assembly process states, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 744–749

36.109 J. Xiao, L. Zhang: Towards obtaining all possible contacts – Growing a polyhedron by its location uncertainty, IEEE Trans. Robotics Autom. **12**(4), 553–565 (1996)

36.110 M. Spreng: A probabilistic method to analyze ambiguous contact situations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 543–548

36.111 N. Mimura, Y. Funahashi: Parameter identification of contact conditions by active force sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Diego (1994) pp. 2645–2650

36.112 B. Eberman: A model-based approach to Cartesian manipulation contact sensing, Int. J. Robotics Res. **16**(4), 508–528 (1997)

36.113 T. Debus, P. Dupont, R. Howe: Contact state estimation using multiple model estimation and Hidden Markov model, Int. J. Robotics Res. **23**(4-5), 399–413 (2004)

36.114 J. De Geeter, H. Van Brussel, J. De Schutter, M. Decréton: Recognizing and locating objects with local sensors, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (1996) pp. 3478–3483

36.115 J. De Schutter, H. Bruyninckx, S. Dutré, J. De Geeter, J. Katupitiya, S. Demey, T. Lefebvre: Estimating first-order geometric parameters and monitoring contact transitions during force-controlled compliant motions, Int. J. Robotics Res. **18**(12), 1161–1184 (1999)

36.116 T. Lefebvre, H. Bruyninckx, J. De Schutter: Polyhedral contact formation identification for autonomous compliant motion: Exact nonlinear bayesian filtering, IEEE Trans. Robotics **21**(1), 124–129 (2005)

36.117 T. Lefebvre, H. Bruyninckx, J. De Schutter: Online statistical model recognition and state estimation for autonomous compliant motion systems, IEEE Trans. Syst. Man Cybern. C **35**(1), 16–29 (2005)

36.118 K. Gadeyne, T. Lefebvre, H. Bruyninckx: Bayesian hybrid model-state estimation applied to simultaneous contact formation recognition and geometrical parameter estimation, Int. J. Robotics Res. **24**(8), 615–630 (2005)

36.119 K. Kitagaki, T. Ogasawara, T. Suehiro: Methods to detect contact state by force sensing in an edge mating task, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 701–706

36.120 T. Lefebvre, H. Bruyninckx, J. De Schutter: Task planning with active sensing for autonomous compliant motion, Int. J. Robotics Res. **24**(1), 61–82 (2005)

36.121 T. Debus, P. Dupont, R. Howe: Distinguishability and identifiability testing of contact state models, Adv. Robotics **19**(5), 545–566 (2005)

36.122 W. Meeussen, J. Xiao, J. De Schutter, H. Bruyninckx, E. Staffetti: Automatic verification of contact states taking into account manipulator constraints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 3583–3588

36.123 N.S. Bedrossian: Classification of singular configurations for redundant manipulators, Proc. 1990 IEEE Int. Conf. Robotics Autom. (ICRA), Cincinnati (1990) pp. 818–823

36.124 F.-T. Cheng, T.-L. Hour, Y.-Y. Sun, T.-H. Chen: Study and resolution of singularities for a 6-DOF PUMA manipulator, IEEE Trans. Syst. Man Cybern. B **27**(2), 332–343 (1997)

36.125 A. Sarić, J. Xiao, J. Shi: Robotic surface assembly via contact state transitions, Proc. IEEE Int. Conf. Autom. Sci. Eng. (CASE) (2013) pp. 954–959

36.126 J. Park, R. Cortesao, O. Khatib: Multi-contact compliant motion control for robotic manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 4789–4794

36.127 W. Meeussen, E. Staffetti, H. Bruyninckx, J. Xiao, J. De Schutter: Integration of planning and execution in force controlled compliant motion, J. Robotics Auton. Syst. **56**(5), 437–450 (2008)

36.128 A. Stolt, M. Linderoth, A. Robertsson, R. Johansson: Force controlled robotic assembly without a force sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (2012) pp. 1538–1543

36.129 S. Tachi, T. Sakaki, H. Arai, S. Nishizawa, J.F. Pelaez-Polo: Impedance control of a direct-drive manipulator without using force sensors, Adv. Robotics **5**(2), 183–205 (1990)

36.130 M. Van Damme, B. Beyl, V. Vanderborght, V. Grosu, R. Van Ham, I. Vanderniepen, A. Matthys, D. Lefeber: Estimating robot end-effector force from noisy actuator torque measurements, Proc. Int. Conf. Robotics Autom. (ICRA), Shanghai (2011) pp. 1108–1113

36.131 D.E. Koditschek: Exact robot navigation by means of potential functions: Some topological considerations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Raleigh (1987) pp. 1–6

36.132 E. Rimon, D.E. Koditschek: Exact robot navigation using artificial potential fields, IEEE Trans. Robotics Autom. **8**(5), 501–518 (1992)

36.133 E. Rimon, D.E. Koditschek: The construction of analytic diffeomorphisms for exact robot navigation on star worlds, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Scottsdale (1989) pp. 21–26

36.134 J. Barraquand, J.-C. Latombe: Robot motion planning: A distributed representation approach, Int. J. Robotics Res. **10**(6), 628–649 (1991)

36.135 C.I. Connolly, J.B. Burns, R. Weiss: Path planning using Laplace's equation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Cincinnati (1990) pp. 2102–2106

36.136 C.I. Connolly, R.A. Grupen: One the applications of harmonic functions to robotics, J. Robotics Syst. **10**(7), 931–946 (1993)

36.137 S.H.J. Feder, E.J.-J. Slotine: Real-time path planning using harmonic potentials in dynamic environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Albuquerque (1997) pp. 811–874

36.138 J.-O. Kim, P. Khosla: Real-time obstacle avoidance using harmonic potential functions, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Sacramento (1991) pp. 790–796

36.139 K. Sato: Collision avoidance in multi-dimensional space using Laplace potential, Proc. 15th Conf. Robotics Soc. Jpn. (1987) pp. 155–156

36.140 S.M. LaValle, P. Konkimalla: Algorithms for computing numerical optimal feedback motion strategies, Int. J. Robotics Res. **20**(9), 729–752 (2001)

36.141 A. van der Schaft, H. Schumacher: *An Introduction to Hybrid Dynamical Systems* (Springer, Belin, Heidelberg 2000)

36.142 W. Choi, J.-C. Latombe: A reactive architecture for planning and executing robot motions with incomplete knowledge, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 1, Osaka (1991) pp. 24–29

36.143 D.C. Conner, H. Choset, A.A. Rizzi: Integrated planning and control for convex-bodied nonholonomic systems using local feedback control policies, Proc. Robotics Sci. Syst., Philadelphia (2006)

36.144 D.C. Conner, A.A. Rizzi, H. Choset: Composition of local potential functions for global robot control and navigation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Las Vegas (2003) pp. 3546–3551

36.145 S.R. Lindemann, S.M. LaValle: Smooth feedback for car-like vehicles in polygonal environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Rome (2007)

36.146 L. Yang, S.M. LaValle: The sampling-based neighborhood graph: A framework for planning and executing feedback motion strategies, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Taipei (2003)

36.147 C. Belta, V. Isler, G.J. Pappas: Discrete abstractions for robot motion planning and control in polygonal environments, IEEE Trans. Robotics Autom. **21**(5), 864–871 (2005)

36.148 R.R. Burridge, A.A. Rizzi, D.E. Koditschek: Sequential composition of dynamically dexterous robot behaviors, Int. J. Robotics Res. **18**(6), 534–555 (1999)

36.149 S.R. Lindemann, S.M. LaValle: Computing smooth feedback plans over cylindrical algebraic decompositions, Proc. Robotics Sci. Syst. (RSS), Philadephia (2006)

36.150 Y. Yang, O. Brock: Elastic roadmaps: Globally task-consitent motion for autonomous mobile manipulation, Proc. Robotics Sci. Syst. (RSS), Philadephia (2006)

36.151 S. Quinlan, O. Khatib: Elastic bands: Connecting path planning and control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2, Atlanta (1993) pp. 802–807

36.152 M. Khatib, H. Jaouni, R. Chatila, J.-P. Laumond: How to implement dynamic paths, Proc.

Int. Symp. Exp. Robotics (1997) pp. 225–236, Preprints
36.153 O. Brock, O. Khatib: Elastic strips: A framework for motion generation in human environments, Int. J. Robotics Res. **21**(12), 1031–1052 (2002)
36.154 M.H. Raibert, J.J. Craig: Hybrid position/force control of manipulators, J. Dyn. Syst. Meas. Control **103**(2), 126–133 (1981)
36.155 R. Featherstone, S. Sonck, O. Khatib: A general contact model for dynamically-decoupled force/motion control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Detroit (1999) pp. 3281–3286
36.156 J. Park, O. Khatib: Multi-link multi-contact force control for manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Barcelona (2005)
36.157 D.C. Ruspini, O. Khatib: A framework for multi-contact multi-body dynamic simulation and haptic display, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Takamatsu (2000) pp. 1322–1327
36.158 K.-S. Chang, R. Holmberg, O. Khatib: The augmented object model: Cooperative manipluation and parallel mechanism dynamics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000) pp. 470–475
36.159 O. Khatib: Object Manipulation in a multi-effector robot system. In: *Robotics Research 4*, ed. by R. Bolles, B. Roth (MIT Press, Cambridge 1988) pp. 137–144
36.160 D. Williams, O. Khatib: The virtual linkage: A model for internal forces in multi-grasp manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1, Altanta (1993) pp. 1030–1035
36.161 J.A. Adams, R. Bajcsy, J. Kosecka, V. Kuma, R. Mandelbaum, M. Mintz, R. Paul, C. Wang, Y. Yamamoto, X. Yun: Cooperative material handling by human and robotic agents: Module development and system synthesis, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Pittsburgh (1995) pp. 200–205
36.162 S. Hayati: Hybrid postiion/force control of multi-arm cooperating robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 82–89
36.163 D. Jung, G. Cheng, A. Zelinsky: Experiments in realizing cooperation between autonomous mobile robots, Proc. Int. Symp. Exp. Robotics (1997) pp. 513–524
36.164 T.-J. Tarn, A.K. Bejczy, X. Yun: Design of dynamic control of two cooperating robot arms: Closed chain formulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 7–13
36.165 M. Uchiyama, P. Dauchez: A symmetric hybrid position/force control scheme for the coordination of two robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Philadelphia (1988) pp. 350–356
36.166 X. Yun, V.R. Kumar: An approach to simultaneious control fo trajecotry and integraction forces in dual-arm configurations, IEEE Trans. Robotics Autom. **7**(5), 618–625 (1991)
36.167 Y.F. Zheng, J.Y.S. Luh: Joint torques for control of two coordinated moving robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 1375–1380
36.168 T. Bretl, Z. McCarthy: Quasi-static manipulation of a Kirchhoff elastic rod based on a geometric analysis of equilibrium configurations, Int. J. Robotics Res. **33**(1), 48–68 (2014)
36.169 J. Russakow, O. Khatib, S.M. Rock: Extended operational space formation for serial-to-parallel chain (branching) manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1, Nagoya (1995) pp. 1056–1061
36.170 K.-S. Chang, O. Khatib: Operational space dynamics: Efficient algorithms for modelling and control of branching mechanisms, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000) pp. 850–856
36.171 K. Kreutz-Delgado, A. Jain, G. Rodriguez: Recursive formulation of operational space control, Int. J. Robotics Res. **11**(4), 320–328 (1992)
36.172 M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, R. Siegwart: Quadrupedal locomotion using hierarchical operational space control, Int. J. Robotics Res. **33**(8), 1047–1062 (2014)
36.173 B. Bayle, J.-Y. Fourquet, M. Renaud: A coordination strategy for mobile manipulation, Proc. Int. Conf. Intell. Auton. Syst., Venice (2000) pp. 981–988
36.174 B. Bayle, J.-Y. Fourquet, M. Renaud: Generalized path generation for a mobile manipulator, Proc. Int. Conf. Mech. Des. Prod., Cairo (2000) pp. 57–66
36.175 B. Bayle, J.-Y. Fourquet, M. Renaud: Using manipulability with nonholonomic mobile manipulators, Proc. Int. Conf. Field Serv. Robotics, Helsinki (2001) pp. 343–348
36.176 J. Peters, S. Schaal: Reinforcement learning for operational space control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Rome (2007)
36.177 T. Simeon, J. Cortes, A. Sahbani, J.P. Laumond: A manipulation planner for pick and place operations under continuous grasps and placements, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)
36.178 A. Miller, P. Allen: Examples of 3D grasp quality computations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (1999)
36.179 A.T. Miller: Graspit: A Versatile Simulator for Robotic Grasping, Ph.D. Thesis (New York, Columbia University 2001)
36.180 G. Wilfong: Motion panning in the presence of movable obstacles, Proc. ACM Symp. Comput. Geom. (1988) pp. 279–288
36.181 M. Erdmann, T. Lozano-Perez: On multiple moving objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 1419–1424
36.182 P.C. Chen, Y.K. Hwang: Pracitcal path planning among movable obstacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1991) pp. 444–449
36.183 M. Stilman, J.J. Kuffner: Navigation among movable obstacles: Real-time reasoning in complex environments, Int. J. Hum. Robotics **2**(4), 1–24 (2005)
36.184 M. Stilman, J.-U. Shamburek, J.J. Kuffner, T. Asfour: Manipulation planning among movable ob-

stacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)
36.185 D. Nieuwenhuisen, A.F. van der Stappen, M.H. Overmars: An effective framework for path planning amidst movable obstacles, Proc. Workshop Algorithm. Found. Robotics (2006)
36.186 K.M. Lynch, M.T. Mason: Stable pushing: Mechanics, controllability, and planning, Int. J. Robotics Res. **15**(6), 533–556 (1996)
36.187 P. Tang, J. Xiao: Generation of point-contact state space between strictly curved objects. In: *Robotics Science and Systems II*, ed. by G.S. Sukhatme, S. Schaal, W. Burgard, D. Fox (MIT Press, Cambridge 2007) pp. 239–246
36.188 H. Nakagaki, K. Kitagaki, T. Ogasawara, H. Tsukune: Study of deformation and insertion tasks of a flexible wire, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 2397–2402
36.189 W. Kraus Jr., B.J. McCarragher: Case studies in the manipulation of flexible parts using a hybrid position/force approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 367–372
36.190 J.Y. Kim, D.J. Kang, H.S. Cho: A flexible parts assembly algorithm based on a visual sensing system, Proc. Int. Symp. Assem. Task Plan. (2001) pp. 417–422
36.191 B.J. Unger, A. Nocolaidis, P.J. Berkelman, A. Thompson, R.L. Klatzky, R.L. Hollis: Comparison of 3-D haptic peg-in-hole tasks in real and virtual environments, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 1751–1756

Multimedia Contents

37. Contact Modeling and Manipulation

Imin Kao, Kevin M. Lynch, Joel W. Burdick

Robotic manipulators use contact forces to grasp and manipulate objects in their environments. Fixtures rely on contacts to immobilize workpieces. Mobile robots and humanoids use wheels or feet to generate the contact forces that allow them to locomote. Modeling of the *contact interface*, therefore, is fundamental to analysis, design, planning, and control of many robotic tasks.

This chapter presents an overview of the modeling of contact interfaces, with a particular focus on their use in manipulation tasks, including graspless or *nonprehensile* manipulation modes such as pushing. Analysis and design of grasps and fixtures also depends on contact modeling, and these are discussed in more detail in Chap. 38. Sections 37.2–37.5 focus on rigid-body models of contact. Section 37.2 describes the kinematic constraints caused by contact, and Sect. 37.3 describes the contact forces that may arise with Coulomb friction. Section 37.4 provides examples of analysis of multicontact manipulation tasks with rigid bodies and Coulomb friction. Section 37.5 extends the analysis to manipulation by pushing. Section 37.6 introduces modeling of contact interfaces, kinematic duality, and pressure distribution and soft contact interface. Section 37.7 describes the concept of the friction limit surface and illustrates it with an example demonstrating the construction of a limit surface for a soft contact. Finally, Sect. 37.8 discusses how these more accurate models can be used in fixture analysis and design.

37.1 **Overview** 931
37.1.1 Choosing a Contact Model 932
37.1.2 Grasp/Manipulation Analysis 932
37.2 **Kinematics of Rigid-Body Contact** 932
37.2.1 Contact Constraints 933
37.2.2 Collections of Parts 934
37.2.3 Graphical Planar Methods 935
37.3 **Forces and Friction** 936
37.3.1 Graphical Planar Methods 937
37.3.2 Duality of Contact Wrenches and Twist Freedoms 937
37.4 **Rigid-Body Mechanics with Friction** 939
37.4.1 Complementarity 939
37.4.2 Quasistatic Assumption 940
37.4.3 Examples 940
37.5 **Pushing Manipulation** 942
37.6 **Contact Interfaces and Modeling** 943
37.6.1 Modeling of Contact Interface 943
37.6.2 Pressure Distribution at Contacts 945
37.7 **Friction Limit Surface** 946
37.7.1 The Friction Limit Surface at a Soft Contact Interface 947
37.7.2 Example of Constructing a Friction Limit Surface 948
37.8 **Contacts in Grasping and Fixture Designs** 949
37.8.1 Contact Stiffness of Soft Fingers 949
37.8.2 Application of Soft Contact Theory to Fixture Design 950
37.9 **Conclusions and Further Reading** 950
Video-References 951
References 951

37.1 Overview

A contact model characterizes both the forces that can be transmitted through the contact as well as the allowed relative motions of the contacting bodies. These characteristics are determined by the geometry of the con-

tacting surfaces and the material properties of the parts, which dictate friction and possible contact deformation.

37.1.1 Choosing a Contact Model

The choice of a contact model largely depends upon the application or analysis that must be carried out. When appropriate analytical models are used, one can determine if a manipulation plan or fixture design meets desired functional requirements within the model's limits.

Rigid-Body Models

Many approaches to manipulation, grasp, and fixture analysis are based on rigid-body models. In the rigid-body model, no deformations are allowed at the points or surfaces of contact between two bodies. Instead, contact forces arise from two sources: the constraint of incompressibility and impenetrability between the rigid bodies, and surface frictional forces. Rigid-body models are straightforward to use, lead to computationally efficient planning algorithms, and are compatible with solid-modeling software systems. Rigid-body models are often appropriate for answering qualitative questions such as *will this fixture be able to hold my workpiece*? and for problems involving stiff parts with low to moderate contact forces.

Rigid-body models are not capable of describing the full range of contact phenomena, however; for example, rigid-body models cannot predict the individual contact forces of a multiple-contact fixture (the static indeterminacy problem [37.1, 2]). Furthermore, workpieces held in fixtures experience non-negligible deformations in many high-force manufacturing operations [37.3–6]. These deformations, which crucially impact machining accuracy, cannot be determined from rigid-body models. Also, a rigid-body model augmented with a Coulomb friction model can lead to mechanics problems that have no solution or multiple solutions [37.7–15]. To overcome the limitations inherent in the rigid-body model, one must introduce compliance into the contact model.

Compliant Models

A *compliant* contact deforms under the influence of applied forces. The forces of interaction at the contact are derived from the *compliance* or *stiffness* model. While compliant contact models are typically more complicated, they have several advantages: they overcome the static indeterminacy inherent to rigid-body models and they predict the deformations of grasped or fixtured parts during loading.

A detailed model of the deformations of real materials can be quite complex. Consequently, for analysis we often introduce *lumped-parameter* or *reduced-order* compliance models having a limited number of variables. In this chapter we summarize a reduced-order *quasi-rigid-body* approach to modeling that can model a variety of compliant materials in a way that is consistent with both the solid mechanics literature and conventional robot analysis and planning paradigms.

Finally, three-dimensional finite-element models [37.16–18] or similar ideas [37.19, 20] can be used to analyze workpiece deformations and stresses in fixtures. While accurate, these numerical approaches have some drawbacks, for example, the grasp stiffness matrix can only be found through difficult numerical procedures. Stiffness matrices are often needed to compute quality measures that are the basis for optimal grasping plans or fixturing designs [37.21, 22]. Thus, these numerical approaches are better suited for verifying final fixture designs.

37.1.2 Grasp/Manipulation Analysis

Once a contact model has been chosen, we can use it to analyze tasks involving multiple contacts. If a part is subject to multiple contacts, the kinematic constraints and force freedoms due to the individual contacts must be combined. This combined analysis facilitates manipulation planning – choosing the contact locations, and possibly the motions or forces applied by those contacts, to achieve the desired behavior of the part. A prime example is the grasping or fixturing problem: choosing contact locations, and possibly contact forces, to prevent motion of a part in the face of external disturbances. This well-studied topic is discussed in greater detail in Chap. 38. Other examples include problems of partial constraint, such as pushing a part or inserting a peg into a hole.

37.2 Kinematics of Rigid-Body Contact

Contact kinematics is the study of how two or more parts can move relative to each other while respecting the rigid-body impenetrability constraint. It also classifies motion in contact as either rolling or slipping.

Consider two rigid bodies whose position and orientation (configuration) is given by the local coordinate column vectors $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$, respectively. Writing the composite configuration as $\boldsymbol{q} = (\boldsymbol{q}_1^{\mathrm{T}}, \boldsymbol{q}_2^{\mathrm{T}})^{\mathrm{T}}$, we de-

fine a *distance* function $d(\boldsymbol{q})$ between the parts that is positive when they are separated, zero when they are touching, and negative when they are in penetration. When $d(\boldsymbol{q}) > 0$, there are no constraints on the motions of the parts. When the parts are in contact ($d(\boldsymbol{q}) = 0$), we look at the time derivatives $\dot{d}$, $\ddot{d}$, etc., to determine if the parts stay in contact or break apart as they follow a trajectory $\boldsymbol{q}(t)$. This can be determined by the following table of possibilities

d	$\dot{d}$	$\ddot{d}$	...
> 0			no contact,
< 0			infeasible (penetration),
$= 0$	> 0		breaking contact,
$= 0$	< 0		infeasible (penetration),
$= 0$	$= 0$	> 0	breaking contact,
$= 0$	$= 0$	< 0	infeasible (penetration),
etc.			

The contact is maintained only if all time derivatives are zero.

The first two time derivatives are written

$$\dot{d} = \left(\frac{\partial d}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} \dot{\boldsymbol{q}} \,, \tag{37.1}$$

$$\ddot{d} = \dot{\boldsymbol{q}}^{\mathrm{T}} \frac{\partial^2 d}{\partial \boldsymbol{q}^2} \dot{\boldsymbol{q}} + \left(\frac{\partial d}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} \ddot{\boldsymbol{q}} \,. \tag{37.2}$$

The terms $\partial d/\partial \boldsymbol{q}$ and $\partial^2 d/\partial \boldsymbol{q}^2$ carry information about the local contact geometry. The former corresponds to the contact normal, while the latter corresponds to the relative curvature of the parts at the contact.

If contact is maintained, we can classify the contact as slipping or rolling. Analogous to the table above, the contact is rolling if and only if there is zero relative tangential velocity, acceleration, etc., between the contact points on the parts. If the relative tangential velocity is nonzero, the parts are slipping; if the relative velocity is zero but relative tangential acceleration or (higher-order derivatives) is not, slipping is incipient.

In this section we focus on a *first-order* analysis of contact kinematics. A first-order analysis concludes that contact is maintained if $d(\boldsymbol{q}) = 0$ and $\dot{d} = 0$. This local linearization of contact kinematics focuses on the velocity $\dot{\boldsymbol{q}}$ and the contact normal in $\partial d/\partial \boldsymbol{q}$; higher-order spatial derivatives of the contact geometry (curvature, etc.) are not considered. While this is a good starting point, it may occasionally lead to erroneous conclusions. For example, *Rimon* and *Burdick* [37.23–25] showed that a first-order analysis may incorrectly predict mobility of a part in a fixture when a second-order analysis shows that it is in fact completely constrained.

Analysis of the kinematics of rolling contact of parameterized surfaces can be found in [37.26]; see also [37.27–30].

37.2.1 Contact Constraints

As described in Chap. 2, a rigid body in space has six degrees of freedom, specified by the location of the origin of a coordinate frame P affixed to the part and the orientation of this coordinate frame relative to an inertial frame O fixed in the world. Let ${}^{O}\boldsymbol{p}_P \in \mathbb{R}^3$ be the position of the part center of mass and ${}^{O}\mathbf{R}_P \in SO(3)$ be the rotation matrix describing the orientation of the part relative to O. The spatial velocity of the part can be written as $\boldsymbol{t} \in \mathbb{R}^6$, sometimes called a *twist*,

$$\boldsymbol{t} = \left(\boldsymbol{\omega}^{\mathrm{T}},\ \boldsymbol{v}^{\mathrm{T}}\right)^{\mathrm{T}} \,,$$

where $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)^{\mathrm{T}}$ and $\boldsymbol{v} = (v_x, v_y, v_z)^{\mathrm{T}}$ give the angular velocity and linear velocity of P in the world frame O, respectively, such that $\boldsymbol{\omega}$ satisfies

$${}^{O}\dot{\mathbf{R}}_P = \boldsymbol{\omega} \times {}^{O}\mathbf{R}_P$$

and $\boldsymbol{v}$ satisfies

$$\boldsymbol{v} = {}^{O}\dot{\boldsymbol{p}}_P - \boldsymbol{\omega} \times {}^{O}\boldsymbol{p}_P \,.$$

It is worth taking a moment to really understand the spatial velocity of a body. It consists of the body's angular velocity expressed in the world frame O, along with the linear velocity of a point as if it were rigidly attached to the body but currently at the origin of the world frame. This point need not be physically on the body. In other words, $\boldsymbol{v}$ is not simply ${}^{O}\dot{\boldsymbol{p}}_P$. This notation will simplify the following expressions, where all velocities and forces will be expressed in the common world frame O. (Be aware that twists are sometimes defined in a body frame instead.)

A point contact acting on the part provides a unilateral constraint which prevents the part from locally moving against the contact normal. Let $\boldsymbol{x}$ be the location of the contact in O. The linear velocity of the point on the part in contact is

$$\boldsymbol{v}_{\mathrm{C}} = \boldsymbol{v} + \boldsymbol{\omega} \times \boldsymbol{x} \,.$$

(We drop the pre-superscripts O for simplicity; for example, ${}^{O}\boldsymbol{x}$ is written simply as $\boldsymbol{x}$.) Let $\hat{\boldsymbol{u}}$ be the unit vector normal pointing into the part (Fig. 37.1). The first-order condition that the part not move into the unilateral constraint can be written

$$\boldsymbol{v}_{\mathrm{C}}^{\mathrm{T}} \hat{\boldsymbol{u}} = (\boldsymbol{v} + \boldsymbol{\omega} \times \boldsymbol{x})^{\mathrm{T}} \hat{\boldsymbol{u}} \geq 0 \,. \tag{37.3}$$

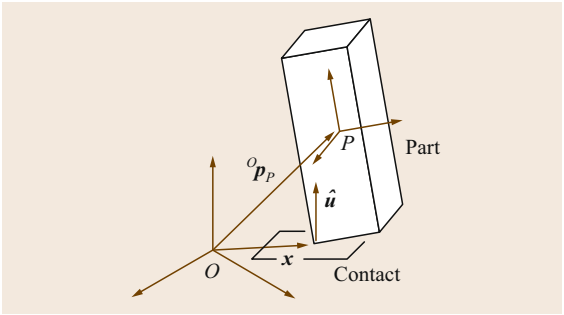


Fig. 37.1 Notation for a part in contact with a manipulator or the environment

In other words, the velocity of the part at C cannot have a component in the opposite direction of the contact normal. To write this another way, define a generalized force or *wrench* $\boldsymbol{w}$ consisting of the torque $\boldsymbol{m}$ and force $\boldsymbol{f}$ acting on the part for a unit force along the contact normal

$$\boldsymbol{w} = \left(\boldsymbol{m}^{\mathrm{T}}, \boldsymbol{f}^{\mathrm{T}}\right)^{\mathrm{T}} = \left[(\boldsymbol{x} \times \hat{\boldsymbol{u}})^{\mathrm{T}}, \hat{\boldsymbol{u}}^{\mathrm{T}}\right]^{\mathrm{T}} .$$

Then (37.3) can be rewritten as

$$\boldsymbol{t}^{\mathrm{T}} \boldsymbol{w} \geq 0 . \tag{37.4}$$

If the external constraint point is moving with linear velocity $\boldsymbol{v}_{\mathrm{ext}}$, (37.4) changes to

$$\boldsymbol{t}^{\mathrm{T}} \boldsymbol{w} \geq \boldsymbol{v}_{\mathrm{ext}}^{\mathrm{T}} \hat{\boldsymbol{u}} , \tag{37.5}$$

which reduces to (37.4) if the constraint is stationary.

Each inequality of the form (37.5) constrains the velocity of the part to a half-space of its six-dimensional velocity space bounded by the hyperplane $\boldsymbol{t}^{\mathrm{T}} \boldsymbol{w} = \boldsymbol{v}_{\mathrm{ext}}^{\mathrm{T}} \hat{\boldsymbol{u}}$. Unioning the set of all constraints, we get a convex polyhedral set of feasible part velocities. A constraint is redundant if its half-space constraint does not change the feasible velocity polyhedron. For a given twist $\boldsymbol{t}$, a constraint is active if

$$\boldsymbol{t}^{\mathrm{T}} \boldsymbol{w} = \boldsymbol{v}_{\mathrm{ext}}^{\mathrm{T}} \hat{\boldsymbol{u}} ; \tag{37.6}$$

otherwise the part is breaking contact at that point. In general, the feasible velocity polyhedron for a part can consist of a six-dimensional interior (where no contact constraint is active), five-dimensional hyperfaces, four-dimensional hyperfaces, and so on, down to one-dimensional edges and zero-dimensional points. A part velocity on an n-dimensional facet of the velocity polyhedron indicates that $6-n$ independent (nonredundant) constraints are active.

If all of the constraints are stationary ($\boldsymbol{v}_{\mathrm{ext}} = 0$), then the boundary of each half-space defined by (37.5) passes through the origin of the velocity space, and the feasible velocity set becomes a cone rooted at the origin. Let $\boldsymbol{w}_i$ be the constraint wrench of stationary contact i. Then the feasible velocity cone is

$$V = \{\boldsymbol{t} \,|\, \boldsymbol{t}^{\mathrm{T}} \boldsymbol{w}_i \geq 0 \quad \forall i\} .$$

If the $\boldsymbol{w}_i$ span the six-dimensional generalized force space, or, equivalently, the convex hull of the $\boldsymbol{w}_i$ contains the origin in the interior, then V is the null set, the stationary contacts completely constrain the motion of the part, and we have form closure, as discussed in Chap. 38.

In the discussion above, each constraint (37.5) divides the part velocity space into three categories: a hyperplane of velocities that maintain contact, a half-space of velocities that separate the parts, and a half-space of velocities that cause the parts to penetrate. Velocities where the contact is maintained can be further broken down into two categories: velocities where the part slips over the contact constraint, and velocities where the part sticks or rolls on the constraint. In the latter case, the part velocity satisfies the three equations

$$\boldsymbol{v} + \boldsymbol{\omega} \times \boldsymbol{x} = \boldsymbol{v}_{\mathrm{ext}} . \tag{37.7}$$

Now we can give each point contact i a label m_i corresponding to the type of contact, called the contact label: `b` if the contact is breaking, `f` if the contact is fixed (including rolling), and `s` if the contact is slipping, i. e., (37.6) is satisfied but (37.7) is not. The *contact mode* for the entire system can be written as the concatenation of the contact labels at the k contacts, $m_1 m_2 \ldots m_k$.

37.2.2 Collections of Parts

The discussion above can be generalized to find the feasible velocities of multiple parts in contact. If parts i and j make contact at a point $\boldsymbol{x}$, where $\hat{\boldsymbol{u}}_i$ points into part i and $\boldsymbol{w}_i = [(\boldsymbol{x} \times \hat{\boldsymbol{u}}_i)^{\mathrm{T}}, \hat{\boldsymbol{u}}_i^{\mathrm{T}}]^{\mathrm{T}}$, then their spatial velocities $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$ must satisfy the constraint

$$(\boldsymbol{t}_i - \boldsymbol{t}_j)^{\mathrm{T}} \boldsymbol{w}_i \geq 0 \tag{37.8}$$

to avoid penetration. This is a homogeneous half-space constraint in the composite $(\boldsymbol{t}_i, \boldsymbol{t}_j)$ velocity space. In an assembly of multiple parts, each pairwise contact contributes another constraint in the composite part velocity space, and the result is a polyhedral convex cone of kinematically feasible velocities rooted at the origin of the composite velocity space. The contact mode for the entire assembly is the concatenation of the contact labels at each contact in the assembly.

If there are moving contacts whose motion is prescribed, e.g., robot fingers, the constraints on the motion of the remaining parts will no longer be homogeneous. As a result, the convex polyhedral feasible velocity space is no longer a cone rooted at the origin.

37.2.3 Graphical Planar Methods

When a part is confined to move in the x–y-plane, the twist $\boldsymbol{t}$ reduces to $\boldsymbol{t} = (\omega_x, \omega_y, \omega_z, v_x, v_y, v_z)^{\mathrm{T}} = (0, 0, \omega_z, v_x, v_y, 0)^{\mathrm{T}}$. The point $(-v_y/\omega_z, v_x/\omega_z)$ is called the center of rotation (COR) in the projective plane, and we can represent any planar twist by its COR and rotational velocity ω_z. (Note that the case $\omega_z = 0$ must be treated with care, as it corresponds to a COR at infinity.) This is sometimes useful for graphical purposes: for a single part constrained by stationary fixtures, at least, we can easily draw the feasible twist cone as CORs [37.14, 31].

As an example, Fig. 37.2a shows a planar part standing on a table and being contacted by a robot finger.

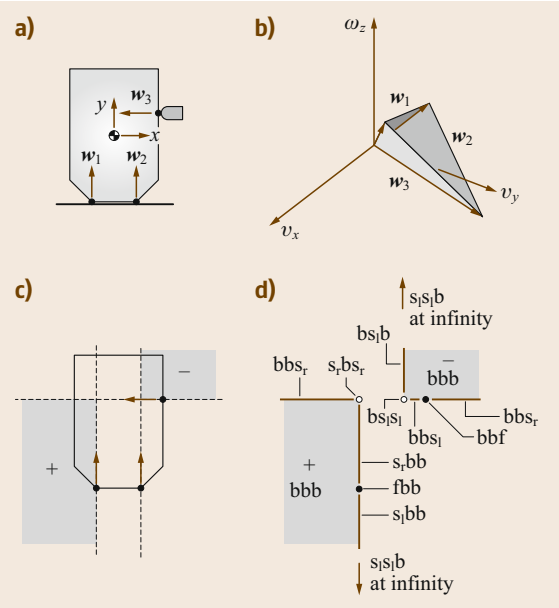


Fig. 37.2 **(a)** A part resting on a table and the three contact constraints. **(b)** The twist cone satisfying the contact constraints. The contact normals are shown with their associated constraint planes. **(c)** The equivalent representation as CORs, shown in *grey*. Note that the lines that extend off to the *left* and to the *bottom wrap around* at infinity and come back in from the *right* and the *top*, respectively, so this COR region should be interpreted as a single connected convex region, equivalent to the twist cone representation in **(b)**. **(d)** The contact modes assigned to each feasible motion. The zero velocity contact mode is fff

This finger is currently stationary, but we will later set it in motion. The finger defines one constraint on the part's motion and the table defines two more. (Note that contact points in the interior of the edge between the part and the table provide redundant kinematic constraints.) The constraint wrenches can be written

$$\boldsymbol{w}_1 = (0, 0, -1, 0, 1, 0)^{\mathrm{T}} ,$$
$$\boldsymbol{w}_2 = (0, 0, 1, 0, 1, 0)^{\mathrm{T}} ,$$
$$\boldsymbol{w}_3 = (0, 0, 1, -1, 0, 0)^{\mathrm{T}} .$$

For a stationary finger, the kinematic constraints yield the feasible twist cone shown in Fig. 37.2b. This region can also be easily visualized in the plane by using the following method: at each contact, draw the contact normal line. Label all points on the normal $\pm$, points to the left of the inward normal $+$, and points to the right $-$. For each contact constraint, all the points labeled $+$ can serve as CORs with positive angular velocity, and all the points labeled $-$ can serve as CORs with negative angular velocity, without violating the contact constraint. After doing this for all the contact normals, keep only the CORs that are consistently labeled. These CORs are a planar representation of the feasible twist cone (Fig. 37.2c).

We can refine this method by assigning contact modes to each feasible COR. For each contact normal, label the COR at the contact point f for fixed, other CORs on the normal line s for slipping, and all other CORs b for breaking contact. The concatenation of these labels gives the part's contact mode for a particular part motion. In the planar case, the label s on the contact normal line can be further refined into $\mathrm{s_r}$ or $\mathrm{s_l}$, indicating whether the part is slipping right or left relative to the constraint. An s COR labeled $+$ above the contact (in the direction of the contact normal) or $-$ below the contact should be relabeled $\mathrm{s_r}$, and an s COR labeled $-$ above the contact or $+$ below should be relabeled $\mathrm{s_l}$ (Fig. 37.2d).

This method can be used readily to determine if the part is in form closure. If there is no COR labeled consistently, then the feasible velocity cone consists of only the zero velocity point, and the part is immobilized by the stationary contacts. This method also makes it clear that at least four contacts are necessary to immobilize the part by the first-order analysis (Chap. 38). This is a weakness of the first-order analysis – curvature effects can be used to immobilize a part with three or even two contacts [37.24]. This weakness can also be seen in Fig. 37.2d. A pure rotation about the COR labeled $\{+, \mathrm{s_r b s_r}\}$ is actually not feasible, but it would be if the part had a small radius of curvature at the contact with the finger. The first-order analysis ignores this curvature.

37.3 Forces and Friction

A commonly used model of friction in robotic manipulation is Coulomb's law [37.32]. This experimental law states that the friction force magnitude f_t in the tangent plane at the contact interface is related to the normal force magnitude f_n by $f_t \le \mu f_n$, where μ is called the friction coefficient. If the contact is sliding, then $f_t = \mu f_n$, and the friction force opposes the direction of motion. The friction force is independent of the speed of sliding.

Often two friction coefficients are defined, a static friction coefficient μ_s and a kinetic (or sliding) friction coefficient μ_k, where $\mu_s \ge \mu_k$. This implies that a larger friction force may be available to resist initial motion, but once motion has begun, the resisting force decreases. Many other friction models have been developed with different functional dependencies on factors such as the speed of sliding and the duration of static contact before sliding. All of these are aggregate models of complex microscopic behavior. For simplicity, we will assume the simplest Coulomb friction model with a single friction coefficient μ. This model is reasonable for hard, dry materials. The friction coefficient depends on the two materials in contact, and typically ranges from 0.1 to 1.

Figure 37.3a shows that this friction law can be interpreted in terms of a *friction cone*. The set of all forces that can be applied to the disk by the supporting line is constrained to be inside this cone. Correspondingly, any force the disk applies to the support is inside the negative of the cone. The half-angle of the cone is $\beta = \tan^{-1}\mu$, as shown in Fig. 37.4. If the disk slips to the left on the support, the force the support applies to it acts on the right edge of the friction cone, with a magnitude determined by the normal force.

If we choose a coordinate frame, the force $\boldsymbol{f}$ applied to the disk by the support can be expressed as a wrench $\boldsymbol{w} = [(\boldsymbol{x} \times \boldsymbol{f})^T, \boldsymbol{f}^T]^T$, where $\boldsymbol{x}$ is the contact location. Thus the friction cone turns into a wrench cone, as shown in Fig. 37.3b. The two edges of the planar friction cone give two half-lines in the wrench space, and the wrenches that can be transmitted to the part through the contact are all nonnegative linear combinations of basis vectors along these edges. If $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ are basis vectors for these wrench cone edges, we write the wrench cone as

$$\mathcal{WC} = \{k_1 \boldsymbol{w}_1 + k_2 \boldsymbol{w}_2 \mid k_1, k_2 \ge 0\} .$$

If there are multiple contacts acting on a part, then the total set of wrenches that can be transmitted to the part through the contacts is the nonnegative linear combination of all the individual wrench cones $\mathcal{WC}_i$,

$$\mathcal{WC} = \text{pos}(\{\mathcal{WC}_i\}) = \left\{ \sum_i k_i \boldsymbol{w}_i | \boldsymbol{w}_i \in \mathcal{WC}_i, k_i \ge 0 \right\} .$$

This composite wrench cone is a convex polyhedral cone rooted at the origin. An example composite

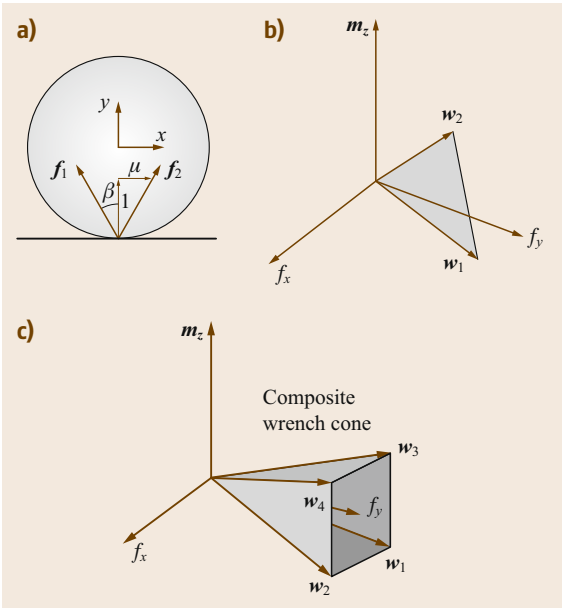


Fig. 37.3 **(a)** A planar friction cone. **(b)** The corresponding wrench cone. **(c)** An example composite wrench cone resulting from two frictional contacts

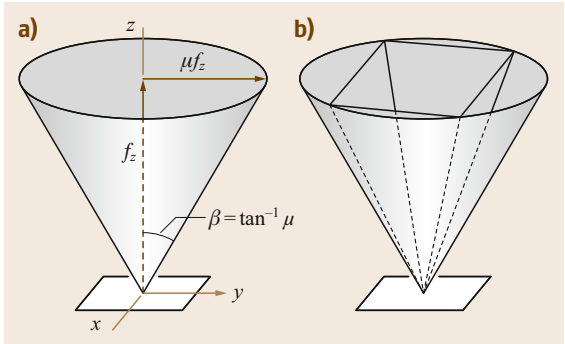


Fig. 37.4 **(a)** A spatial friction cone. The half-angle of the cone is $\beta = \tan^{-1}\mu$. **(b)** An inscribed pyramidal approximation to the friction cone. A more accurate inscribed pyramidal approximation can be used by increasing the number of faces of the pyramid. Depending on the application, a circumscribed pyramid could be used instead of an inscribed pyramid

wrench cone arising from two planar frictional contacts is shown in Fig. 37.3c. If the composite wrench cone is the entire wrench space, then the contacts can provide a force-closure grasp (Chap. 38).

In the spatial case, the friction cone is a circular cone, defined by

$$\sqrt{f_x^2 + f_y^2} \leq \mu f_z \,, \quad f_z \geq 0 \,, \tag{37.9}$$

when the contact normal is in the $+z$-direction (Fig. 37.4). The resulting wrench cone, and composite wrench cone $\text{pos}(\{\mathcal{W}C_i\})$ for multiple contacts, is a convex cone rooted at the origin, but it is not polyhedral. For computational purposes, it is common to approximate circular friction cones as pyramidal cones, as shown in Fig. 37.4. Then individual and composite wrench cones become polyhedral convex cones in the six-dimensional wrench space.

If a contact or set of contacts acting on a part is ideally force controlled, the wrench $\boldsymbol{w}_{\text{ext}}$ specified by the controller must lie within the composite wrench cone corresponding to those contacts. Because these force-controlled contacts choose a subset of wrenches (possibly a single wrench) from this wrench cone, the total composite wrench cone that can act on the part (including other non-force-controlled contacts) may no longer be a homogeneous cone rooted at the origin. This is roughly analogous to the case of velocity-controlled contacts in Sect. 37.2.1, which results in a feasible twist set for the part that is not a cone rooted at the origin. Ideal robot manipulators may be controlled by position control, force control, hybrid position–force control, or some other scheme. The control method must be compatible with the parts' contacts with each other and the environment to prevent excessive forces [37.33].

37.3.1 Graphical Planar Methods

Just as homogeneous twist cones for planar problems can be represented as convex signed (+ or −) COR regions in the plane, homogeneous wrench cones for planar problems can be represented as convex signed regions in the plane. This is called *moment labeling* [37.14, 34]. Given a collection of lines of force in the plane (e.g., the edges of friction cones from a set of point contacts), the set of all nonnegative linear combinations of these can be represented by labeling all the points in the plane with either a + if all resultants make nonnegative moment about that point, a − if all make nonpositive moment about that point, a ± if all make zero moment about that point, and a blank label if there exist resultants making positive moment and resultants making negative moment about that point.

The idea is best illustrated by an example. In Fig. 37.5a, a single line of force is represented by labeling the points to the left of the line with a + and points to the right of the line with a −. Points on the line are labeled ±. In Fig. 37.5b, another line of force is added. Only the points in the plane that are consistently labeled for both lines of force retain their labels; inconsistently labeled points lose their labels. Finally, a third line of force is added in Fig. 37.5c. The result is a single region labeled +. A nonnegative combination of the three lines of force can create any line of force in the plane that passes around this region in a counterclockwise sense. This representation is equivalent to a homogeneous convex wrench cone representation.

37.3.2 Duality of Contact Wrenches and Twist Freedoms

Our discussion of kinematic constraints and friction should make it apparent that, for any point contact and contact label, the number of equality constraints on the part's motion caused by that contact is equal to the number of wrench freedoms it provides. For example, a breaking contact `b` provides zero equality constraints on the part motion and also allows no contact force. A fixed contact `f` provides three motion constraints (the motion of a point on the part is specified) and three freedoms on the contact force: any wrench in the interior of the contact wrench cone is consistent with the contact mode. Finally, a slipping contact `s` provides one equality motion constraint (one equation on the part's motion must be satisfied to maintain the contact), and for a given motion satisfying the constraint, the contact wrench has only one freedom, the magnitude of the contact wrench on the edge of the friction cone and opposite the slipping direction. In the planar case, the motion constraints and wrench freedoms for `b`, `s`, and `f` contacts are zero, one, and two, respectively.

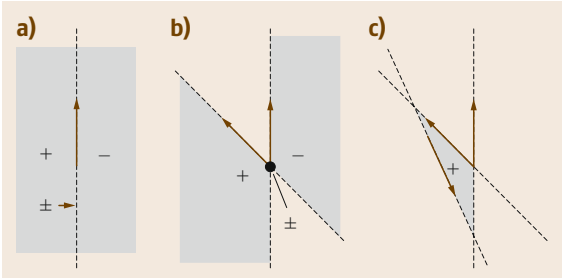


Fig. 37.5 **(a)** Representing a line of force by moment labels. **(b)** Representing the nonnegative linear combinations of two lines of force by moment labels. **(c)** Nonnegative linear combinations of three lines of force

Table 37.1 Duality of the kinematics of contact interface. The force/motion relationships for finger and grasped object are dual to each other in the joint space, contact interface, and the Cartesian space of the object. In this table, the notation of θ, $\boldsymbol{x}_{\mathrm{f}}$, $\boldsymbol{x}_{\mathrm{p}}$, and $\boldsymbol{x}_{\mathrm{b}}$ are the displacement in the joint space, the Cartesian space at the contact point on the finger, the Cartesian space at the contact point on the object, and the reference frame on the object, respectively, with $\boldsymbol{x}_{\mathrm{tr}}$ denoting the transmitted components. The δ prefix denotes infinitesimal changes in the named coordinates. The forces with corresponding subscripts are referenced to the corresponding coordinates as those of the displacement described before

Motion	$\mathbf{J}_\theta \quad \delta\boldsymbol{\theta} \quad = \delta\boldsymbol{x}_{\mathrm{f}}$ $(6\times\boldsymbol{m})(\boldsymbol{m}\times 1) \quad (6\times 1)$	$\mathbf{H} \quad \delta\boldsymbol{x}_{\mathrm{f}} = \delta\boldsymbol{x}_{\mathrm{tr}} = \mathbf{H} \quad \delta\boldsymbol{x}_{\mathrm{p}}$ $(\boldsymbol{n}\times 6)(6\times 1) \quad (\boldsymbol{n}\times 1) \quad (\boldsymbol{n}\times 6)(6\times 1)$	$\mathbf{J}_{\mathrm{c}} \quad \delta\boldsymbol{x}_{\mathrm{b}} = \delta\boldsymbol{x}_{\mathrm{p}}$ $(6\times 6)(6\times 1) \quad (6\times 1)$
	Joints $\rightleftharpoons$	**Contact** $\rightleftharpoons$	**Object**
Force	$\mathbf{J}_\theta^{\mathrm{T}} \quad \boldsymbol{f}_{\mathrm{f}} \quad = \boldsymbol{\tau}$ $(\boldsymbol{m}\times 6)(6\times 1) \quad (\boldsymbol{m}\times 1)$	$\boldsymbol{f}_{\mathrm{f}} \quad = \mathbf{H}^{\mathrm{T}} \quad \boldsymbol{f}_{\mathrm{tr}} \quad = \boldsymbol{f}_{\mathrm{p}}$ $(6\times 1) \quad (6\times n)(n\times 1) \quad (6\times 1)$	$\mathbf{J}_{\mathrm{c}}^{\mathrm{T}} \quad \boldsymbol{f}_{\mathrm{p}} \quad = \boldsymbol{f}_{\mathrm{p}}$ $(6\times 6)(6\times 1) \quad (6\times 1)$

At each contact, the force and velocity constraints can be represented by an $n\times 6$ *constraint* or *selection* matrix, $\mathbf{H}$ [37.3], at the contact interface. This constraint matrix works like a filter to transmit or deny certain components of motion across the contact interface. By the same token, forces/moments applied across the contact interface are also filtered by the same constraint matrix, $\mathbf{H}^{\mathrm{T}}$, in a dual relationship. Three typical contact models are illustrated below. Contact forces and wrenches are measured in a frame at the contact, with the contact normal in the $+z$-direction:

- The *point contact without friction* model. Only a normal force f_z can be exerted between the contacting bodies

$$f_z \geq 0\,, \quad \boldsymbol{w} = \mathbf{H}^{\mathrm{T}} f_z = (0\,0\,1\,0\,0\,0)^{\mathrm{T}} f_z\,. \tag{37.10}$$

- The *point contact with friction* model includes tangential friction forces, f_x and f_y, in addition to normal force f_z,

$$f_z \geq 0\,; |f_t| = \sqrt{f_x^2 + f_y^2} \leq \mu f_z\,,$$

$$\boldsymbol{w} = \mathbf{H}^{\mathrm{T}} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix}, \tag{37.11}$$

where f_x and f_y are the x- and y-components of the tangential contact force, and f_z is the normal force.

- The *soft finger* contact model with a finite contact patch allows, in addition to the friction and normal forces, a torsional moment with respect to the contact normal [37.35–39]. We have

$$f_z \geq 0\,;$$

$$\boldsymbol{w} = \mathbf{H}^{\mathrm{T}} \begin{pmatrix} f_x \\ f_y \\ f_z \\ m_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_x \\ f_y \\ f_z \\ m_z \end{pmatrix}, \tag{37.12}$$

where m_z is the moment with respect to the normal of contact. The finite contact area assumed by the soft contact interface results in the application of a friction moment in addition to the traction forces. If the resultant force on the tangential plane of contact is denoted as

$$f_t = \sqrt{f_x^2 + f_y^2}$$

and the moment with respect to the contact normal is m_z, the following elliptical equation represents the relationship between the force and moment at the onset of sliding

$$\frac{f_t^2}{a^2} + \frac{m_z^2}{b^2} = 1\,, \tag{37.13}$$

where $a = \mu f_z$ is the maximum friction force, and $b = (m_z)_{\max}$ is the maximum moment defined in equation (37.36). Further readings on this subject can be found in [37.35–42].

As can be appreciated from the three cases above, the rows of $\mathbf{H}$ are the directions along which contact forces are supported. Conversely, relative motions of the two objects are constrained along these same directions: $\mathbf{H}\dot{\boldsymbol{q}} = 0$. Thus, the constraint matrix works like a kinematic filter which dictates the components

of motions that can be transmitted through the contact interface. The introduction of this matrix makes it easy to model the contact mechanics in analysis using the dual kinematic relationship between force/moment and motion, as described in Table 37.1. For example, the constraint filter matrix in (37.12) describes that all three components of forces and one component of the moment with respect to the normal of contact can be transmitted through the contact interface of a soft finger.

In Chap. 38, when multiple contacts are considered, such $\mathbf{H}$ matrices associated with each finger can be concatenated in an augmented matrix to work in conjunction with the grasp and Jacobian matrices for the analysis of grasping and manipulation.

Table 37.1 summarizes the dual relationship between the force/moment and displacement – a result that is derived from the principle of virtual work. In Table 37.1, $\mathbf{J}_\theta$ is the joint Jacobian matrix relating joint velocities to fingertip velocities, and $\mathbf{J}_c$ is the Cartesian coordinate transformation matrix relating the contact point with the centroidal coordinates of the grasped object. In addition, the number of degrees of freedom (DOFs) in the task space is n and the number of DOFs in the joint spaces is m.

37.4 Rigid-Body Mechanics with Friction

The manipulation planning problem is to choose the motions or forces applied by manipulator contacts so that the part (or parts) moves as desired. This requires solving the subproblem of determining the motion of parts given a particular manipulator action.

Let $\boldsymbol{q} \in \mathbb{R}^n$ be local coordinates describing the combined configuration of the system consisting of one or more parts and robot manipulators, and let $\boldsymbol{w}_i \in \mathbb{R}^6$ represent a wrench at an active contact i, measured in the common coordinate frame O. Let $\boldsymbol{w}_{\text{all}} \in \mathbb{R}^{6k}$ be the vector obtained by stacking the $\boldsymbol{w}_i$, $\boldsymbol{w}_{\text{all}} = (\boldsymbol{w}_1^{\mathrm{T}}, \boldsymbol{w}_2^{\mathrm{T}}, \ldots, \boldsymbol{w}_k^{\mathrm{T}})^{\mathrm{T}}$ (where there are k contacts), and let $\mathbf{A}(\boldsymbol{q}) \in \mathbb{R}^{n \times 6k}$ be a matrix indicating how (or if) each contact wrench acts on each part. (Note that a contact wrench $\boldsymbol{w}_i$ acting on one part means that there is a contact wrench $-\boldsymbol{w}_i$ acting on the other part in contact.) The problem is to find the contact forces $\boldsymbol{w}_{\text{all}}$ and the system acceleration $\ddot{\boldsymbol{q}}$, given the state of the system $(\boldsymbol{q}, \dot{\boldsymbol{q}})$, the system mass matrix $\mathbf{M}(\boldsymbol{q})$ and resulting Coriolis matrix $\mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, the gravitational forces $\mathbf{g}(\boldsymbol{q})$, the control forces $\boldsymbol{\tau}$, and the matrix $\mathbf{T}(\boldsymbol{q})$ indicating how the control forces $\boldsymbol{\tau}$ act on the system. (Alternatively, if we view the manipulators as position controlled, the manipulator components of $\ddot{\boldsymbol{q}}$ can be directly specified and the corresponding components of the control forces $\boldsymbol{\tau}$ solved for.) One way to solve this problem is to (a) enumerate the set of all possible contact modes for the current active contacts, and (b) for each contact mode, determine if there are wrenches $\boldsymbol{w}_{\text{all}}$ and accelerations $\ddot{\boldsymbol{q}}$ that satisfy the dynamics

$$\mathbf{A}(\boldsymbol{q})\boldsymbol{w}_{\text{all}} + \mathbf{T}(\boldsymbol{q})\boldsymbol{\tau} - \mathbf{g}(\boldsymbol{q}) = \mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \,, \tag{37.14}$$

and that are consistent with the contact mode's kinematic constraints (constraints on $\ddot{\boldsymbol{q}}$) and friction cone force constraints (constraints on $\boldsymbol{w}_{\text{all}}$). This formulation is quite general and applies to multiple parts in contact. Equation (37.14) may be simplified by appropriate representations of the configurations and velocities of rigid-body parts (for example, using angular velocities of the rigid bodies instead of derivatives of local angular coordinates).

This formulation leads to some surprising conclusions: there may be multiple solutions to a particular problem (ambiguity) or there may be no solutions (inconsistency) [37.7–14]. This strange behavior arises from the status of Coulomb's law as an approximate law, and it disappears for zero friction, or friction that is small enough. Despite this weakness of the Coulomb friction law, it is a useful approximation. Nonetheless, if we would like to prove (under the Coulomb model) that a particular desired part motion occurs, we generally must also show that no other motion can occur. Otherwise we have only shown that the desired motion is one of the possible outcomes.

37.4.1 Complementarity

The conditions that each contact provides an equal number of motion constraints and wrench freedoms (Sect. 37.3.2) can be written as *complementarity* conditions. Thus the problem of solving (37.14), subject to contact constraints, can be formulated as a complementarity problem (CP) [37.12, 13, 43–46]. For planar problems, or spatial problems with approximate pyramidal friction cones, the problem is a linear complementarity problem [37.47]. For circular spatial friction cones, the problem is a nonlinear complementarity problem, due to the quadratic constraints describing the cones. In either case, standard algorithms can be used to solve for possible contact modes and part motions.

Alternatively, assuming linearized friction cones, we can formulate a linear constraint satisfaction program (LCSP) for each contact mode (e.g., a linear program with no objective function). The contact mode places linear constraints on the part accelerations, and the solver solves for the part accelerations and the nonnegative coefficients multiplying each of the edges of the friction cones, subject to the dynamics (37.14). Each LCSP with a feasible solution represents a feasible contact mode.

37.4.2 Quasistatic Assumption

A common assumption in robot manipulation planning is the *quasistatic* assumption. This assumption says that parts move slowly enough that inertial effects are negligible. This means that the right- and left-hand sides of (37.14) are zero. With this assumption, we often solve for part *velocities* rather than accelerations. These velocities must be consistent with the kinematic constraints and force constraints, and forces acting on the parts must always sum to zero.

37.4.3 Examples

While rigid-body mechanics problems with friction are usually solved using computational tools for CPs and LCSPs, equivalent graphical methods can be used for some planar problems to assist the intuition. As a simple example, consider the pipe clamp in Fig. 37.6 [37.14]. Under the external wrench $\boldsymbol{w}_{\text{ext}}$, does the clamp slide down the pipe, or does it remain fixed in place? The figure uses moment labeling to represent the composite wrench cone of contact forces that can act on the clamp from the pipe. The wrench $\boldsymbol{w}_{\text{ext}}$ acting on the clamp can be exactly balanced by a wrench in the composite wrench cone. This is evident from the fact that the wrench opposing $\boldsymbol{w}_{\text{ext}}$ passes around the + labeled region in a counterclockwise sense, meaning that it is contained in the contact wrench cone. As a result, static equilibrium (the `ff` contact mode) is a feasible solution for the pipe clamp. To arrive at the same result using an LCSP, label the unit wrenches at the edges of the friction cones $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_4$. Then the clamp can remain at rest if there exist coefficients $a_1, \ldots, a_4$ such that

$$a_1, a_2, a_3, a_4 \geq 0 \,,$$
$$a_1\boldsymbol{w}_1 + a_2\boldsymbol{w}_2 + a_3\boldsymbol{w}_3 + a_4\boldsymbol{w}_4 + \boldsymbol{w}_{\text{ext}} = 0 \,.$$

To show that static equilibrium is the only solution, all other contact modes must be ruled out. Note that, if the friction coefficient at the contacts is too small, the clamp will fall under $\boldsymbol{w}_{\text{ext}}$.

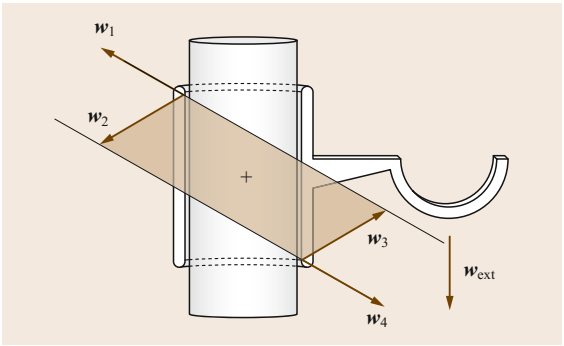


Fig. 37.6 The moment-labeling representation of the composite wrench cone of contact forces the pipe can apply to the pipe clamp. In this figure, the wrench $\boldsymbol{w}_{\text{ext}}$ applied to the pipe clamp can be resisted by forces within the composite wrench cone

Analysis of the classic peg-in-hole problem is similar to that of the pipe clamp. Figure 37.7a shows an angled peg making two-point contact with the hole. If we apply the wrench $\boldsymbol{w}_1$, the forces from the two-point contact cannot resist, so the `ff` mode contact mode is not a possibility, and the peg will continue to move into the hole. If we apply the wrench $\boldsymbol{w}_2$, however, the contacts can resist, and the peg will get stuck. This is called *jamming* [37.48]. If there is higher friction at the contacts, as in Fig. 37.7b, then each friction cone may be able to see the base of the other, and the feasible contact wrenches span the entire wrench space [37.49]. In this case, any wrench we apply may be resisted by the contacts. The peg is said to be *wedged* [37.48]. Whether or not the peg will actually resist our applied wrench depends on how much internal force acts between the contacts. This cannot be answered by our rigid-body model; it requires a compliant contact model, as discussed in Sect. 37.6.

As a slightly more complex quasistatic example, consider a block on a table being pushed by a finger

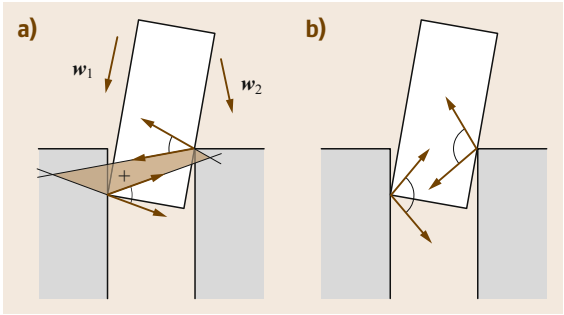


Fig. 37.7 **(a)** The peg will proceed into the hole under the external wrench $\boldsymbol{w}_1$ but will get stuck under the wrench w_2. **(b)** The peg is wedged

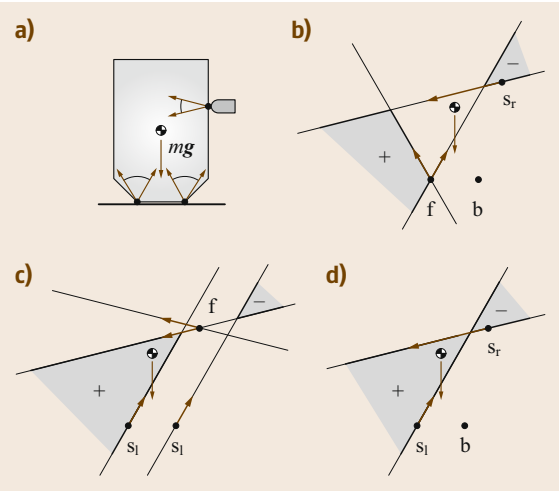


Fig. 37.8 **(a)** A finger moves *left* into a planar block resting on a table in gravity. The contact friction cones are shown. **(b)** Possible contact forces for the contact label f at the leftmost bottom contact, b at the rightmost bottom contact, and s_r at the pushing contact. This contact mode fbr corresponds to the block tipping over the leftmost contact. Note that the contact wrench cone (represented by moment labels) can provide a force that exactly balances the gravitational force. Thus this contact mode is quasistatically possible. **(c)** The contact wrench cone for the $s_l s_l f$ contact mode (block slides *left* on the table) cannot balance the gravitational force. This contact mode is not quasistatically possible. **(d)** The contact wrench cone for the $s_l b s_r$ contact mode (block slides and tips) is not quasistatically possible

that moves horizontally to the left (Fig. 37.8) [37.50, 51]. Does the part begin to tip over, slide, or both tip and slide? The corresponding contact modes are fbs_r, $s_l s_l f$, and $s_l b s_r$, as shown in Fig. 37.8. The figure also shows the composite contact wrench cones using moment labels. It is clear that quasistatic balance between the gravitational force and the contact wrench cone can only occur for the tipping without slipping contact mode fbs_r. Therefore, the only quasistatic solution is that the block begins to tip without sliding, and the speed of this motion is determined by the speed of the finger's motion. The graphical construction can be used to confirm our intuition that tipping of the block occurs if we push high on the block or if the friction coefficient at the block's support is high. Try it by pushing a can or

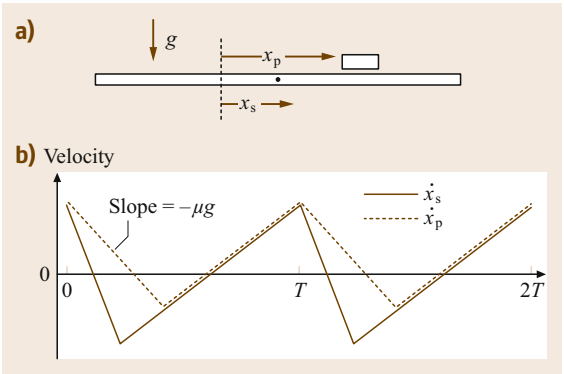


Fig. 37.9 **(a)** A part supported by a horizontally vibrating surface. **(b)** Friction between the part and the surface causes the part to always try to catch up to the surface, but its acceleration is bounded by $\pm\mu g$. The asymmetric motion of the surface gives the part an average positive velocity over a cycle

glass. Quasistatically, the height of the center of mass is immaterial.

A final example is given in Fig. 37.9. A part of mass m and friction coefficient μ is supported by a horizontal surface that moves periodically in the horizontal direction. The periodic motion consists of a large negative acceleration for a short duration and a smaller positive acceleration for a longer duration. The horizontal friction force that can be applied to the part is bounded by $\pm\mu mg$, so the part's horizontal acceleration is bounded by $\pm\mu g$. As a result, the part cannot keep up with the surface as it accelerates backward. Instead, the part slips forward with respect to the surface, which means that the maximal friction force acts in the negative direction, and the part attempts to slow down to the surface's velocity. Eventually the part catches up with the surface as it executes its slow forward acceleration, which is less than μg. Once the part catches up to the surface, it sticks to it, until the next backward acceleration phase. As is evident in Fig. 37.9, the average part velocity over a cycle is positive, so the part moves forward on the surface [37.52, 53]. This idea can be extended to create a wide variety of frictional force fields on a rigid plate vibrated with three degrees of freedom in a horizontal plane [37.54, 55] or with a full six degrees of freedom [37.56].

For further examples of manipulation planning using the rigid-body model with Coulomb friction, see [37.14] and the references therein.

37.5 Pushing Manipulation

The friction limit surface (Sect. 37.7) is useful for analyzing pushing manipulation, as it describes the friction forces that can occur as a part slides over a support surface. When the part is pushed with a wrench contained within the limit surface, friction between the part and the support resists the pushing wrench and the part remains motionless. When the part slides quasistatically, the pushing wrench $\boldsymbol{w}$ lies on the limit surface, and the part's twist $\boldsymbol{t}$ is normal to the limit surface at $\boldsymbol{w}$ (Fig. 37.10). When the part translates without rotating, the friction force magnitude is μmg, where m is the part's mass and g is gravitational acceleration. The force applied by the part to the surface is directed through the part's center of mass in the direction of translation.

If the pushing wrench makes positive moment about the part's center of mass, the part will rotate counterclockwise (CCW), and if it makes negative moment about the part's center of mass, it will rotate clockwise (CW). Similarly, if the contact point on the part moves along a line that passes around the part's center of mass in a CW (respectively, CCW) sense, then the part will rotate CW (CCW). From these two observations and considering all possible contact modes, we can conclude that a part pushed at a point contact will rotate CW (CCW) if either (1) both edges of the contact friction cone pass CW (CCW) about the center of mass, or (2) one edge of the friction cone and the pushing direction at the contact both pass around the center of mass in a CW (CCW) sense [37.14, 57], (Fig. 37.11).

This observation allows pushing to be used to reduce uncertainty in part orientation. A series of pushes with a flat fence can be used to completely eliminate the uncertainty in the orientation of a polygonal part [37.57–59]. Bounds on the rate of rotation of parts [37.14, 60, 61] allow the use of a sequence of stationary fences suspended above a conveyor belt to orient parts by pushing them as they are carried along by the conveyor [37.62, 63]. *Stable* pushing plans (Fig. 37.12) use pushing motions that are guaranteed to keep the part fixed to the pusher as it moves, even

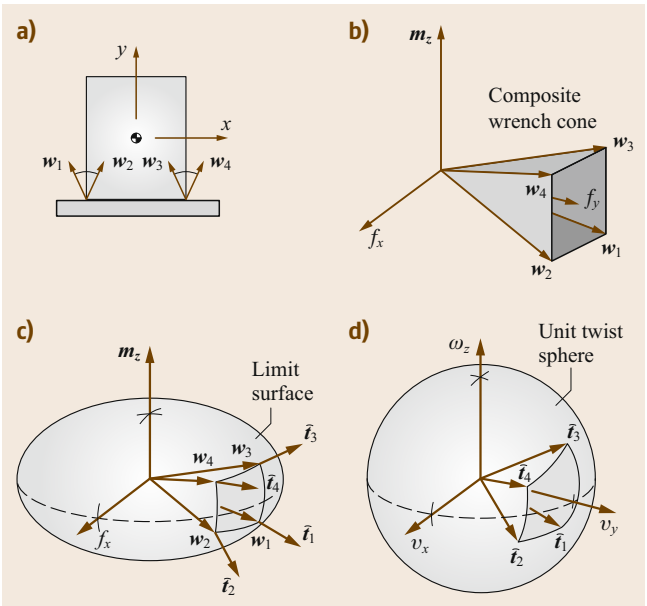


Fig. 37.10 (a) Contact between a pusher and a sliding object. (b) The composite contact wrench cone. (c) Mapping the wrench cone through the limit surface to twists of the object. (d) The unit twists that can result from forces in the composite wrench cone

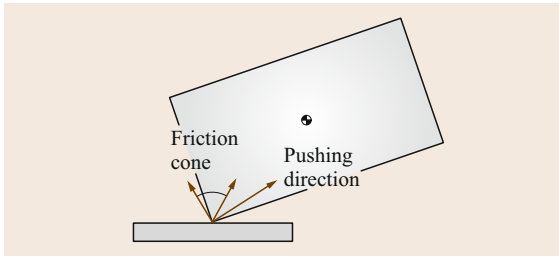


Fig. 37.11 The pusher's motion direction votes for counterclockwise rotation of the part, but it is outvoted by the two edges of the friction cone, which both indicate clockwise rotation

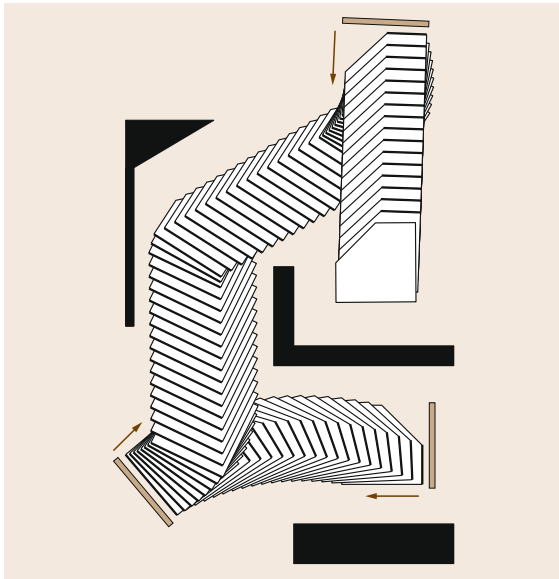


Fig. 37.12 Stable pushes can be used to maneuver a part among obstacles

in the face of uncertainty in the part's pressure distribution [37.14, 64, 65]. Extensions of this work find pushing motions for planar assemblies of parts so that they remain fixed in their relative configurations during motion [37.66–68].

The examples above assume that pushing forces and support friction forces act in the same plane. Other work on pushing has considered three-dimensional effects, where pushing forces are applied above the support plane [37.69].

37.6 Contact Interfaces and Modeling

Contact interface is a general expression to describe the kinematics and kinetics of contacts. Contact applies in various contexts in robotics research; thus, it is more meaningful to refer to generic contact as an *interface*, which is not limited to fingers in grasping and manipulation. The concept of *contact interface* extends the traditional context of physical *contact*. It refers to an interface which imparts kinematic filtering as well as the duality of force/motion transmitted across the contact interface.

Consequently, a contact interface, whether it is rigid or deformable, can be considered as a *kinematic filter* which includes two characteristics: (a) motion and force transmission, and (b) kinematic duality between force/moment and motion. Different contact interfaces will be described in the following sections.

Previous sections of this chapter have assumed rigid bodies in contact. In reality, however, all contacts are accompanied by some deformation of the objects. Often this is by design, as in the case of compliant robot fingertips. When deformation is non-negligible, elastic contact models can be used.

37.6.1 Modeling of Contact Interface

Contact modeling depends on the nature of bodies in contact, including their material properties, applied force, contact deformation, and elastic properties. This section discusses different contact models.

Rigid-Body and Point-Contact Models

With the rigid-body assumption, as discussed in the previous sections of this chapter, two models are often used: (a) *point contact without friction* and (b) *point contact with friction*. In the former case, the contact can only apply force in the direction normal to the contact. In the latter case, a tangential friction force is applied in addition to the normal force. The simplest analytical model for point contact with friction is the Coulomb friction model as presented in (37.9).

Hertzian Contact Model

Elastic contact modeling was first studied and formulated more than a century ago by *Hertz* in 1882 [37.70] based on contact between two linear elastic materials with a normal force which results in very small contact deformation. This is commonly called the *Hertzian* contact and can be found in most mechanics textbooks, such as [37.71, 72]. Hertz made two important explicit assumptions in order for his contact model to be applicable:

1. Objects of linear elastic materials in contact,
2. Small contact deformation compared to the dimension of object.

Hertz also conducted experiments using a spherical glass lens against a planar glass plate to validate the contact theory.

Two relevant results of the Hertzian contact theory applied to robotic contact interfaces are summarized as follows. The first pertains to the radius of the contact area. *Hertz* [37.70] studied the growth of the contact area as a function of the applied normal force N based on the linear elastic model. Based on ten experimental trials, he concluded that the radius of contact is proportional to the normal force raised to the power of $1/3$, which is consistent with the analytical results he derived based on the linear elastic model. That is, the radius of contact, a, is related to the normal force, N, by

$$a \propto N^{\frac{1}{3}} \,. \tag{37.15}$$

The second result pertains to the pressure distribution over the assumed symmetric contact area – a second-order pressure distribution of the nature of an ellipse or circle. For a symmetric and circular contact area, the pressure distribution is

$$p(r) = \frac{N}{\pi a^2}\sqrt{1-\left(\frac{r}{a}\right)^2}, \tag{37.16}$$

where N is the normal force, a is the radius of contact, and r is the distance from the center of contact with $0 \le r \le a$.

Soft Contact Model. A typical contact interface between a soft finger and contact surface is illustrated in Fig. 37.13. In typical robotic contact interfaces, the materials of the fingertips are not linear elastic. A model

that extends linear to nonlinear elastic contact was presented in [37.38] with a power-law equation which subsumes the Hertzian contact theory,

$$a = c\,N^{\gamma}\,, \tag{37.17}$$

where $\gamma = n/(2n+1)$ is the exponent of the normal force, n is the strain-hardening exponent, and c is a constant depending on the size and curvature of the fingertip as well as the material properties. Equation (37.17) is the new power law that relates the growth of the circular contact radius to the applied normal force for soft fingers. Note that the equation is derived assuming circular contact area. For linear elastic materials, the constant n is equal to 1 with $\gamma = 1/3$, resulting in the Hertzian contact model in (37.15). Therefore, the soft contact model in (37.17) subsumes the Hertzian contact model.

Viscoelastic Soft Contact Model. Typical soft materials used in robotic fingertip, e.g., rubber, silicone and polymers, show viscoelastic properties. Viscoelasticity is a physical phenomenon of time-dependent strain and stress [37.73]. Specifically in the context of robotic grasping and manipulation, there are two types of time dependent responses [37.74–76]:

1. *Relaxation*: The evolution of force in grasping while the displacement is held constant;
2. *Creep*: The evolution of displacement in contact and grasping while the external force is held constant.

Such time-dependent responses would approach equilibrium asymptotically. Viscoelastic materials also exhibit the properties of:

1. *Strain history dependence*: The response of the material depends on prior strain history; and
2. *Energy dissipation*: A net energy dissipation associated with a complete cycle of loading and unloading.

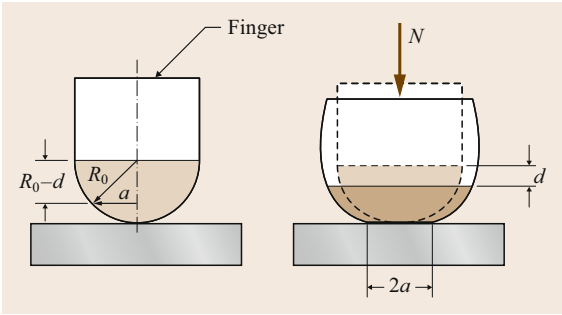


Fig. 37.13 An elastic soft fingertip with a hemispherical tip making contact with a rigid surface

Two popular viscoelastic models include: the Kelvin–Voigt/Maxwell model (the spring-damper model) and the Fung's model, which will be described in the following.

(1) Kelvin–Voigt/Maxwell model. The Kelvin–Voigt's solid model was constructed by a spring and a damper connected in parallel [37.77] as shown in Fig. 37.14. The relation between the stress, σ, and strain, ϵ, can be formulated as

$$\sigma = k\epsilon + c\frac{\mathrm{d}\epsilon}{\mathrm{d}t}\,. \tag{37.18}$$

Maxwell proposed in 1867 the Maxwell fluid model [37.78] as a single set of spring and damper in series as shown in Fig. 37.15. The relationship between the stress, σ, and strain, ϵ, can be expressed as

$$\frac{1}{k}\frac{\mathrm{d}\sigma}{\mathrm{d}t} + \frac{\sigma}{c} = \frac{\mathrm{d}\epsilon}{\mathrm{d}t}\,. \tag{37.19}$$

The *generalized Maxwell model* utilizes multiple serial spring-damper sets and a spring connected in parallel. One can employ the curve-fitting techniques using the experimental data to find the modeling parameters of springs and dampers.

However, the issue of this model is often on the lack of consistency of the parameters (stiffness constants and damping factors) obtained from the model. Such parameters can have large discrepancy in numerical values, often in two orders of magnitude or higher, while representing the same material with supposedly similar values of springs and dampers. These parameters sometimes can also present unrealistic difference in scales under different experimental setup (e.g., in [37.79, 80]).

(2) Fung's model. A popular viscoelastic model in modeling biomedical materials is the Fung's model [37.76], proposed by *Fung* in 1993. The main idea of the model is to represent the reacting force as the product of two independent responses: the temporal response and the elastic response, while incorporating

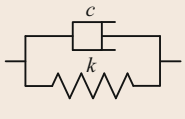


Fig. 37.14 Kelvin–Voigt model

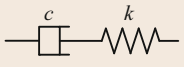


Fig. 37.15 Maxwell model

the history of the stress response. The model may be written as

$$T(t) = \int_{-\infty}^{t} G(t-\tau)\frac{\partial T^{(e)}[\lambda(t)]}{\partial \lambda}\frac{\partial \lambda(\tau)}{\partial \tau}\,\mathrm{d}\tau\,, \quad (37.20)$$

where $T(t)$ is the tensile stress at time t, with a step increase of size λ in elongation on the specimen, the function $T^{(e)}(\lambda)$ is the so-called elastic response, and $G(t)$, a normalized function of time, is the reduced relaxation function.

Tiezzi and *Kao* [37.42, 81, 82] simplified this model to study the soft contact interface by assuming no past stress history, as expressed in (37.21)

$$\mathcal{G}(\delta, t) = N^{(e)}(\delta)\cdot g(t)\,, \quad (37.21)$$

where $\mathcal{G}(\delta, t)$ represents the grasping force as a function of the displacement δ and time t, $N^{(e)}(\delta)$ represents the elastic response of normal force as a function of the displacement (or depression unto the object), and $g(t)$ represents the temporal response of relaxation or creep. The important property of this model is the separation of the spatial response and temporal response as two independent functions. It was shown that the viscoelastic model has an important implication on the stability of grasping which cannot be captured by rigid or linear elastic modeling.

Other Models. In addition to the aforementioned models, other models from different viewpoints were proposed, as follows: From the viewpoint of rheological perspectives [37.83–86]; molecular perspectives [37.73, 87–92]; energy perspectives [37.85]; distributed modeling perspectives [37.93–98] and stress wave propagation perspectives [37.99–102].

37.6.2 Pressure Distribution at Contacts

In Sect. 37.6.1, when the Hertzian contact theory is considered, the assumed pressure distribution for small elastic deformation is given in (37.16). As the radius of curvature of the two asperities increases and the material properties change to hyperelastic, the pressure distribution becomes more uniform [37.38, 103, 104]. Generalizing equation (37.16), the pressure distribution function for circular contact area with radius a is

$$p(r) = C_k\frac{N}{\pi a^2}\left[1-\left(\frac{r}{a}\right)^k\right]^{\frac{1}{k}}\,, \quad (37.22)$$

where N is the normal force, a is the radius of contact, r is the radius with $0 \le r \le a$, k determines the shape of the pressure profile, and C_k is a coefficient that adjusts for the profile of pressure distribution over the contact area to satisfy the equilibrium condition. In (37.22), $p(r)$ is defined for $0 \le r \le a$. By symmetry, $p(r) = p(-r)$ when $-a \le r \le 0$, as shown in Fig. 37.16. When k becomes larger, the pressure distribution approaches uniform distribution, as shown in Fig. 37.16. It is also required that the integral of the pressure over the contact area be equal to the normal force; that is

$$\int_R p(r)\,\mathrm{d}A = \int_{\theta=0}^{2\pi}\int_{r=0}^{a} p(r)\,r\,\mathrm{d}r\,\mathrm{d}\theta = N\,. \quad (37.23)$$

The coefficient C_k can be obtained by substituting (37.22) into (37.23). It is interesting to note that when (37.23) is integrated, both the normal force and radius of contact vanish, leaving only the constant C_k as follows

$$C_k = \frac{3}{2}\frac{k\Gamma\left(\frac{3}{k}\right)}{\Gamma\left(\frac{1}{k}\right)\Gamma\left(\frac{2}{k}\right)}\,, \quad (37.24)$$

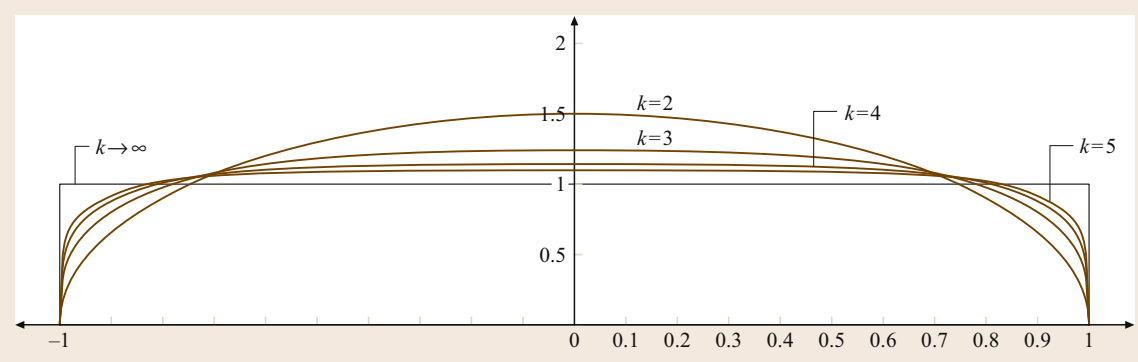


Fig. 37.16 Illustration of pressure distribution with respect to the normalized radius, r/a. The plot shows an axisymmetric pressure distribution with $k = 2, 3, 4, 5$, and ∞. As k increases, the pressure profile becomes more uniform

where $k = 1, 2, 3, \ldots$ are typically integer values (although noninteger k values are also possible) and $\Gamma()$ is the gamma function [37.105]. The numerical values of C_k for a few values of k are listed in Table 37.2 for reference. A normalized pressure distribution with respect to the normalized radius, r/a, is plotted in Fig. 37.16. As can be seen from the figure, when k approaches infinity, the pressure profile will become that of a uniformly distributed load with magnitude of $N/(\pi a^2)$, in which case $C_k = 1.0$.

For linear elastic materials, $k = 2$ can be used, although it is found that $k \cong 1.8$ is also appropriate in some cases [37.106]. For nonlinear elastic and viscoelastic materials, the value of k tends to be higher, depending on the properties of the material.

Table 37.2 Values of the coefficient C_k in the pressure distribution (37.22)

k	Coefficient C_k
$k = 2$ (circular)	$C_2 = 1.5$
$k = 3$ (cubic)	$C_3 = 1.24$
$k = 4$ (quadruple)	$C_4 = 1.1441$
$k \to \infty$ (uniform)	$C_\infty = 1.0$

37.7 Friction Limit Surface

Section 37.3 introduced the notion of a wrench cone, describing the set of wrenches that can be applied through a point contact with friction. Any contact wrench is limited to lie within the surface of the cone. This gives rise to the general notion of a *friction limit surface* – the surface bounding the set of wrenches that can be applied through a given contact or set of contacts.

In this section we study the particular case of limit surfaces arising from planar contact patches [37.7, 107]. A planar contact patch occurs when a flat object slides on the floor, a soft robot finger presses against a face of a polyhedron, or a foot of a humanoid robot pushes on the ground. We would like to know what wrenches can be transmitted through such a contact. In the rest of this section, we focus on the properties of the limit surface of a planar contact patch with a specified pressure distribution over the patch.

For ease of discussion, we will call one of the objects in contact the *part* (e.g., the flat object on the floor or the robot finger) and the other object a stationary *support*. We define a coordinate frame so that the planar contact patch is in the $z = 0$ plane, and let $p(\boldsymbol{r}) \geq 0$ be the contact pressure distribution between the part and support as a function of the location $\boldsymbol{r} = (x, y)^{\mathrm{T}}$. The friction coefficient at the contact patch is μ. If the planar velocity of the part is $\boldsymbol{t} = (\omega_z, v_x, v_y)^{\mathrm{T}}$, then the linear velocity at $\boldsymbol{r}$ is

$$\boldsymbol{v}(\boldsymbol{r}) = (v_x - \omega_z y, v_y + \omega_z x)^{\mathrm{T}} ,$$

and the unit velocity is $\hat{\boldsymbol{v}}(\boldsymbol{r}) = \boldsymbol{v}(\boldsymbol{r})/\|\boldsymbol{v}(\boldsymbol{r})\|$. The infinitesimal force applied by the part to the support at $\boldsymbol{r}$, in the plane of sliding, is

$$\mathrm{d}\boldsymbol{f}(\boldsymbol{r}) = [\mathrm{d}f_x(\boldsymbol{r}), \mathrm{d}f_y(\boldsymbol{r})]^{\mathrm{T}} = \mu p(\boldsymbol{r}) \hat{\boldsymbol{v}}(\boldsymbol{r}) . \qquad (37.25)$$

The total wrench the part applies to the support is

$$\boldsymbol{w} = \begin{pmatrix} m_z \\ f_x \\ f_y \end{pmatrix} = \int_A \begin{pmatrix} x\,\mathrm{d}f_y(\boldsymbol{r}) - y\,\mathrm{d}f_x(\boldsymbol{r}) \\ \mathrm{d}f_x(\boldsymbol{r}) \\ \mathrm{d}f_y(\boldsymbol{r}) \end{pmatrix} \mathrm{d}A , \qquad (37.26)$$

where A is the support area.

As expected, the wrench is independent of the speed of motion: for a given $\boldsymbol{t}_0$ and all $\alpha > 0$, the wrench resulting from velocities $\alpha \boldsymbol{t}_0$ is identical. Viewing (37.26) as a mapping from twists to wrenches, we can map the surface of the unit twist sphere ($\|\hat{\boldsymbol{t}}\| = 1$) to a surface in the wrench space. This surface is the limit surface, and it is closed, convex, and encloses the origin of the wrench space (Fig. 37.17). The portion of the wrench

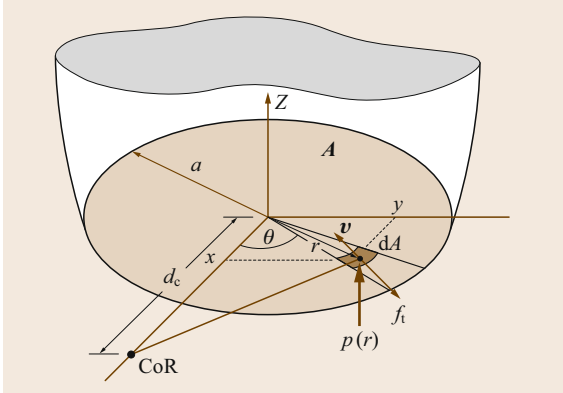


Fig. 37.17 Contact and coordinates for COR and local infinitesimal area $\mathrm{d}A$ for numerical integration to construct the limit surface of soft fingers

space enclosed by this surface is exactly the set of wrenches the part can transmit to the support. When the part slips on the support ($\boldsymbol{t} \neq 0$), the contact wrench $\boldsymbol{w}$ lies on the limit surface, and by the maximum-work inequality, the twist $\boldsymbol{t}$ is normal to the limit surface at $\boldsymbol{w}$. If the pressure distribution $p(\boldsymbol{r})$ is finite everywhere, the limit surface is smooth and strictly convex, and the mapping from unit twists to unit wrenches, and vice versa, is continuous and one-to-one. The limit surface also satisfies the property $\boldsymbol{w}(-\hat{\boldsymbol{t}}) = -\boldsymbol{w}(\hat{\boldsymbol{t}})$.

37.7.1 The Friction Limit Surface at a Soft Contact Interface

Given a pressure distribution in (37.22) we can numerically construct the corresponding friction limit surface by using (37.25) and (37.26). For an infinitesimal contact area, the contact resembles a point contact. Hence, the Coulomb's law of friction can be employed.

In the following, we will exploit the kinematic property of center of rotation (COR) with modeling symmetry to formulate equations for the magnitudes of friction force and moment on a circular contact patch. Figure 37.17 shows a circular contact patch with an instantaneous COR. By moving (or scanning) the COR along the x-axis, different possible combinations of friction force and moment can be obtained to construct the limit surface. More details can be found in [37.35–38, 41, 41, 42, 107–109].

The following derivation is for the total friction force (f_t) and moment (m_z) on the contact interface for the COR at a distance d_c along the x-axis, as shown in Fig. 37.17. By varying the COR distance d_c from $-\infty$ to ∞, all possible combinations of (f_t, m_z) can be found in order to construct the entire friction limit surface.

The tangential force over the entire contact area can be obtained by integrating shear force on each infinitesimal areas, $\mathrm{d}A$, on which the Coulomb's law of friction is observed, over the entire contact area, A. When Hertzian contact pressure distribution [37.70] is considered, use $k = 2$ in (37.22). Setting $\boldsymbol{r} = (x\,, y)^{\mathrm{T}}$ and $r = \|\boldsymbol{r}\|$, the total tangential force can be integrated using (37.25) to obtain

$$\boldsymbol{f}_t = \begin{pmatrix} f_x \\ f_y \end{pmatrix} = -\int_A \mu\, \hat{\boldsymbol{v}}(\boldsymbol{r})\, p(r)\, \mathrm{d}A\,, \tag{37.27}$$

where A denotes the circular contact region in Fig. 37.17, $\boldsymbol{f}_t$ is the tangential force vector with the direction shown in Fig. 37.17, μ is the coefficient of friction, $\hat{\boldsymbol{v}}(\boldsymbol{r})$ is the unit vector in the direction of the velocity vector $\boldsymbol{v}(\boldsymbol{r})$ with respect to the COR on the infinitesimal area $\mathrm{d}A$ at the location $\boldsymbol{r}$, and $p(r)$ is the pressure distribution at distance r from the center of contact. Since the pressure along the annular ring with distance r from the center is the same, we denote it by $p(r)$ instead of $p(\boldsymbol{r})$. The minus sign denotes the opposite directions of $\hat{\boldsymbol{v}}(\boldsymbol{r})$ and $\boldsymbol{f}_t$. Since we are primarily interested in the magnitude of the friction force and moment, we shall omit this sign in the later derivation when magnitudes are concerned.

Similarly, the moment about the z-axis, or the normal to the contact area, is

$$m_z = \int_A \mu \|\boldsymbol{r} \times \hat{\boldsymbol{v}}(\boldsymbol{r})\|\, p(r)\, \mathrm{d}A\,, \tag{37.28}$$

where $\|\boldsymbol{r} \times \hat{\boldsymbol{v}}(\boldsymbol{r})\|$ is the magnitude of the cross product of the vectors $\boldsymbol{r}$ and $\hat{\boldsymbol{v}}(\boldsymbol{r})$, whose direction is normal to the contact surface.

The unit vector $\hat{\boldsymbol{v}}(\boldsymbol{r})$ is related to the distance d_c from the origin to the COR chosen on the x-axis from Fig. 37.17, and can be written as follows

$$\begin{aligned} \hat{\boldsymbol{v}}(\boldsymbol{r}) &= \frac{1}{\sqrt{(x-d_c)^2 + y^2}} \begin{bmatrix} -y \\ (x-d_c) \end{bmatrix} \\ &= \frac{1}{\sqrt{(r\cos\theta - d_c)^2 + (r\sin\theta)^2}} \\ &\quad \times \begin{bmatrix} -r\sin\theta \\ (r\cos\theta - d_c) \end{bmatrix}. \end{aligned} \tag{37.29}$$

Due to symmetry, $f_x = 0$ for all CORs along the x-axis; therefore, the magnitude of the tangential force in the contact tangent plane is $f_t = f_y$. Substituting equations (37.22) and (37.29) into (37.27) and (37.28), we obtain

$$\begin{aligned} f_t &= \int_A \mu \frac{(r\,\cos\theta - d_c)}{\sqrt{r^2 + d_c^2 - 2rd_c\,\cos\theta}} C_k \\ &\quad \times \frac{N}{\pi a^2}\left[1 - \left(\frac{r}{a}\right)^k\right]^{\frac{1}{k}} \mathrm{d}A\,. \end{aligned} \tag{37.30}$$

Similarly, the moment about the axis normal to the plane is

$$\begin{aligned} m_z &= \int_A \mu \frac{r^2 - r\,d_c\cos\theta}{\sqrt{r^2 + d_c^2 - 2rd_c\,\cos\theta}} C_k \\ &\quad \times \frac{N}{\pi a^2}\left[1 - \left(\frac{r}{a}\right)^k\right]^{\frac{1}{k}} \mathrm{d}A. \end{aligned} \tag{37.31}$$

In (37.30) and (37.31), polar coordinates are defined such that $x = r\,\cos\theta$, $y = r\,\sin\theta$, and $\mathrm{d}A = r\,\mathrm{d}r\,\mathrm{d}\theta$. We also introduce a normalized coordinate

$$\tilde{r} = \frac{r}{a}\,. \tag{37.32}$$

From (37.32), we can write $\mathrm{d}r = a\,\mathrm{d}\tilde{r}$. We also denote $\tilde{d}_\mathrm{c} = d_\mathrm{c}/a$, and assume that μ is constant throughout the contact area. Substituting the normalized coordinate into equation (37.30) and dividing both sides by μN, we can derive

$$\frac{f_\mathrm{t}}{\mu N} = \frac{C_k}{\pi}\int_0^{2\pi}\int_0^1 \frac{\left(\tilde{r}^2\cos\theta - \tilde{r}\tilde{d}_\mathrm{c}\right)}{\sqrt{\tilde{r}^2 + \tilde{d}_\mathrm{c}^2 - 2\tilde{r}\tilde{d}_\mathrm{c}\cos\theta}} \times (1-\tilde{r}^k)^{\frac{1}{k}}\,\mathrm{d}\tilde{r}\mathrm{d}\theta\,. \tag{37.33}$$

Substituting again $\tilde{r} = r/a$ and $\mathrm{d}r = a\,\mathrm{d}\tilde{r}$ into (37.31) and normalizing with $a\mu N$, we obtain

$$\frac{m_z}{a\mu N} = \frac{C_k}{\pi}\int_0^{2\pi}\int_0^1 \frac{\left(\tilde{r}^3\cos\theta - \tilde{r}^2\tilde{d}_\mathrm{c}\right)}{\sqrt{\tilde{r}^2 + \tilde{d}_\mathrm{c}^2 - 2\tilde{r}\tilde{d}_\mathrm{c}\cos\theta}} \times (1-\tilde{r}^k)^{\frac{1}{k}}\,\mathrm{d}\tilde{r}\mathrm{d}\theta\,. \tag{37.34}$$

Equations (37.33) and (37.34) can be numerically integrated for a distance d_c or $\tilde{d}_\mathrm{c}$ to yield a point on the limit surface for a prescribed pressure distribution $p(r)$ given by (37.22). Both equations involve elliptic integrals whose closed-form solutions may not exist but can be evaluated numerically. When the COR distance d_c varies from $-\infty$ to ∞, all possible combinations of (f_t, m_z) can be obtained for plotting the friction limit surface.

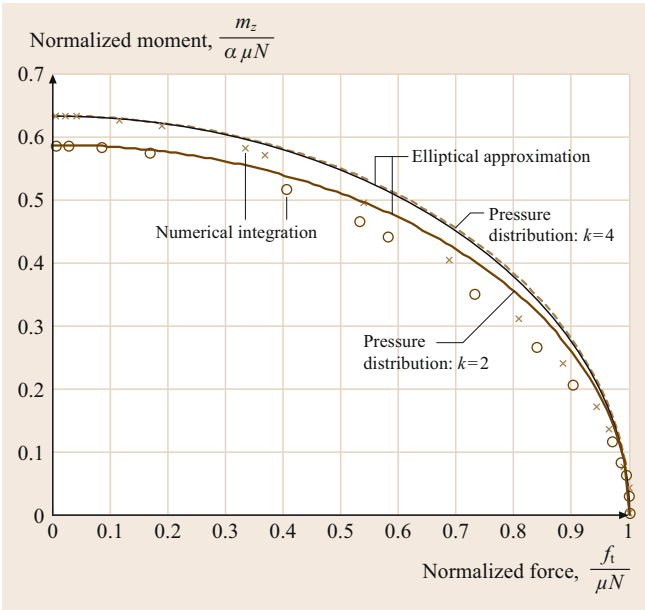


Fig. 37.18 Example limit surface obtained by numerical integration and elliptical approximation. The numerical integration is based on the pressure distribution, $p(r)$, and the coefficient, C_k. In this figure, the pressure distributions in (37.22) of both $k = 2$ and $k = 4$ are used

37.7.2 Example of Constructing a Friction Limit Surface

When the pressure distribution is fourth order with $k = 4$, the coefficient from Table 37.2 is $C_4 = (6/\sqrt{\pi})(\Gamma(3/4)/\Gamma(1/4)) = 1.1441$. Equations (37.33) and (37.34) can be written as

$$\frac{f_\mathrm{t}}{\mu N} = 0.3642\int_0^{2\pi}\int_0^1 \frac{\left(\tilde{r}^2\cos\theta - \tilde{r}\tilde{d}_\mathrm{c}\right)}{\sqrt{\tilde{r}^2 + \tilde{d}_\mathrm{c}^2 - 2\tilde{r}\tilde{d}_\mathrm{c}\cos\theta}} \times (1-\tilde{r}^4)^{\frac{1}{4}}\,\mathrm{d}\tilde{r}\mathrm{d}\theta\,.$$

$$\frac{m_z}{a\mu N} = 0.3642\int_0^{2\pi}\int_0^1 \frac{\left(\tilde{r}^3\cos\theta - \tilde{r}^2\tilde{d}_\mathrm{c}\right)}{\sqrt{\tilde{r}^2 + \tilde{d}_\mathrm{c}^2 - 2\tilde{r}\tilde{d}_\mathrm{c}\cos\theta}} \times (1-\tilde{r}^4)^{\frac{1}{4}}\,\mathrm{d}\tilde{r}\mathrm{d}\theta\,.$$

Numerical integration for different values of $\tilde{d}_\mathrm{c}$ yields pairs of $(f_\mathrm{t}/(\mu N),\ m_z/(a\mu N))$. Plots of these pairs are shown in Fig. 37.18.

A reasonable approximation to these numerical results is given by the following equation of an ellipse

$$\left(\frac{f_\mathrm{t}}{\mu N}\right)^2 + \left(\frac{m_z}{(m_z)_\mathrm{max}}\right)^2 = 1\,, \tag{37.35}$$

where the maximum moment $(m_z)_\mathrm{max}$ is

$$(m_z)_\mathrm{max} = \int_A \mu\,|r| C_k \frac{N}{\pi a^2}\left[1 - \left(\frac{r}{a}\right)^k\right]^{\frac{1}{k}}\mathrm{d}A\,, \tag{37.36}$$

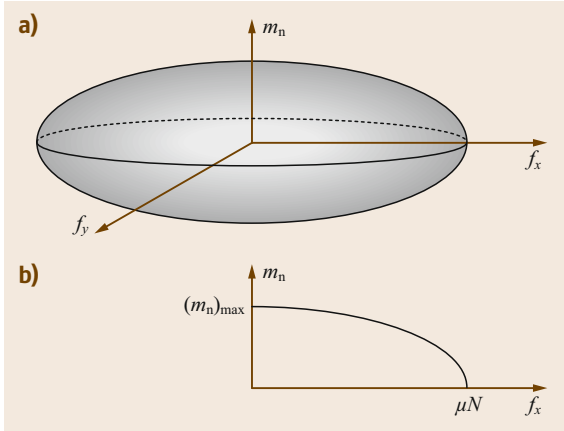


Fig. 37.19a,b Friction limit surface for soft fingers: **(a)** a 3-D ellipsoid representing the limit surface, and **(b)** a section of the ellipsoidal limit surface showing the coupled relationship between the force and moment, as those obtained in Fig. 37.18

obtained by (37.31) with the COR at $d_c = 0$. This defines the quarter-elliptical curves in Fig. 37.18. This approximation is the basis for constructing a three-dimensional (3-D) ellipsoidal limit surface, as illustrated in Fig. 37.19, and is a good model for the soft contact described in Sect. 37.6.1. More details can be found in [37.35–37, 39–42].

37.8 Contacts in Grasping and Fixture Designs

It is often important to relate the force and contact displacement (or deformation of contact interface) in grasping and fixture design in which deformable contacts are concerned. Furthermore, such a force–displacement relationship is typically nonlinear due to the nature of contact. A linear expression such as Hooke's law cannot capture the instantaneous and overall characteristics of force and displacement for these contacts. In this section, we formulate and discuss such a relationship using the elastic contact model.

From Fig. 37.13 and the geometry of contact, (37.17), relating the contact radius with normal force, can be rewritten. Assuming $d \ll R_0$, the following equation can be derived [37.39]

$$N = c_d\, d^{\zeta}\,, \tag{37.37}$$

where c_d is a proportional constant, and ζ is

$$\zeta = \frac{1}{2\gamma}\,. \tag{37.38}$$

Both c_d and ζ can be obtained experimentally. The exponent ζ can also be obtained from γ through (37.38) if γ is already known for the fingertip. The range of the exponent in (37.37) is $3/2 \le \zeta < \infty$. In (37.37), the approach or the vertical depression of the fingertip, d, is proportional to the normal force raised to the power of 2γ (cf. (37.38)) which ranges from 0 to $2/3$.

A plot of normal force versus displacement of contact is shown in Fig. 37.20 using (37.37) with a nonlinear power-law equation. Experimental data of contact between a soft finger and flat surface show consistent results when compared with Fig. 37.20 [37.39].

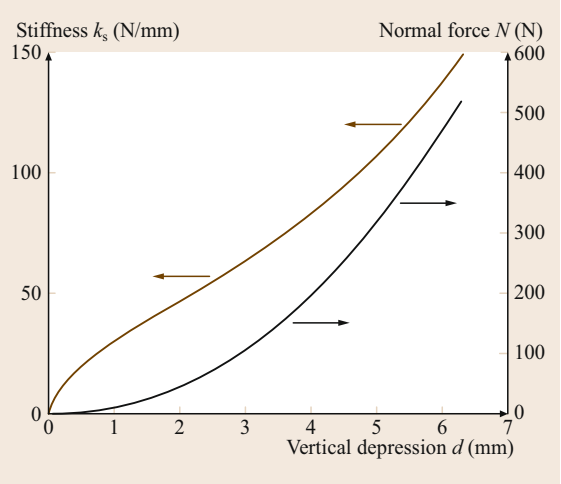


Fig. 37.20 The typical load–depression curve in (37.37) shown with a *scale to the right*, and the contact stiffness, k_s, as given by (37.39) with a *scale to the left*

37.8.1 Contact Stiffness of Soft Fingers

The nonlinear contact stiffness of a soft finger is defined as the ratio of the change in normal force with respect to the change in vertical depression at the contact. The contact stiffness of soft fingers can be obtained by differentiating (37.37) as follows

$$k_s = \frac{\partial N}{\partial d} = c_d\, \zeta\, d^{\zeta-1} = \zeta \left(\frac{N}{d}\right). \tag{37.39}$$

Substitute (37.37) with $d = \left(\frac{N}{c_d}\right)^{1/\zeta}$ into (37.39) to derive

$$k_s = c_d^{\frac{1}{\zeta}}\, \zeta\, N^{\frac{\zeta-1}{\zeta}} = c_d^{2\gamma}\, \zeta\, N^{1-2\gamma}\,. \tag{37.40}$$

Thus, the nonlinear contact stiffness of soft fingers, derived in a succinct form in (37.39), is the product of the exponent ζ and the ratio of the normal force versus the approach, N/d. A typical contact stiffness as a function of vertical depression is plotted in Fig. 37.20. The stiffness shown in the figure increases with the force. The expressions for the stiffness, k_s, as a function of various parameter(s) are summarized in the following Table 37.3.

Table 37.3 Summary of equations of contact stiffness

	$f(d)$	$f(N)$	$f(N,d)$
k_s	$c_d \zeta d^{\zeta-1}$	$c_d^{\frac{1}{\zeta}}\, \zeta N^{\frac{\zeta-1}{\zeta}}$	$\zeta\left(\frac{N}{d}\right)$

Table 37.4 Summary of the contact mechanics equations for linear elastic (when $\gamma = 1/3$ or $\zeta = 3/2$) and nonlinear elastic soft fingers

Description	Equation for soft fingers	Parameters
Power law	$a = c\,N^{\gamma}$	$0 \le \gamma \le \frac{1}{3}$
Pressure distribution	$p(r) = p(0)\left[1 - \left(\frac{r}{a}\right)^{k}\right]^{\frac{1}{k}}$	Typically $k \ge 1.8$
Contact approach	$N = c_{\mathrm{d}}\, d^{\zeta}$	$\frac{3}{2} \le \zeta \le \infty$
Contact stiffness	$k_{\mathrm{s}} = \zeta \frac{N}{d}$	Nonlinear

The change in stiffness can be obtained by differentiating (37.39) to derive the following equations

$$\frac{\partial k_{\mathrm{s}}}{\partial d} = c_{\mathrm{d}}\,\zeta(\zeta - 1)\,d^{\zeta-2}\,, \tag{37.41}$$

$$\frac{\partial k_{\mathrm{s}}}{\partial N} = \frac{\zeta - 1}{d}\,. \tag{37.42}$$

Equation (37.39) suggests that the stiffness of soft contacts always increases because the ratio N/d always increases, with constant ζ for prescribed fingertip material and range of normal force. This is consistent with the observation that the contact stiffness becomes larger (i. e., stiffer) with larger depression and force. Additionally, (37.42) suggests that the contact stiffness k_{s} always increases with the normal force because $\partial k_{\mathrm{s}}/\partial N > 0$ owing to $\zeta \ge 1.5$. Moreover, the change in stiffness with respect to the normal load is inversely proportional to the vertical depression, d, as derived in (37.42). This result asserts that the rate of increase in the contact stiffness will gradually become less and less as the normal load and vertical depression increase.

A summary of the equations for soft fingers is presented in Table 37.4. The range of the exponent of the general power-law equation in (37.17) is $0 \le \gamma \le 1/3$. The Hertzian contact has an exponent of $\gamma = 1/3$; therefore, the Hertzian contact theory for linear elastic materials is a special case of (37.17). Linear elastic materials, such as steel or other metallic fingertips, under small deformation, generally follow the Hertzian contact theory fairly well. The power-law equation in (37.17) should be employed for fingertips made of softer materials, such as soft rubber or silicone or even viscoelastic fingertips.

37.8.2 Application of Soft Contact Theory to Fixture Design

The preceding analysis and results can be applied in fixturing design and other applications that involve contact with finite areas [37.6]. In fixture design with soft contacts (e.g., copper surfaces) under relatively large deformation and load, the power-law equations in Table 37.4 should be considered in place of the Hertzian contact equation. In these cases, the Hertzian contact model is no longer accurate and should be replaced. On the other hand, if linear elastic materials are used in fixture design with relatively small deformation ($d/R_0 \le 5\%$), the exponent should be taken as $\gamma = 1/3$ when applying the contact theory. Furthermore, the exponent γ was found to be material dependent and not geometry dependent in general [37.38]. Once the value of γ is determined for the material (for example, using a tensile testing machine with the experimental procedure in [37.38]), it can be employed for the analysis of fixturing design using the relevant equations presented herewith.

37.9 Conclusions and Further Reading

In this chapter, both rigid-body and elastic models of contact interfaces are discussed, including kinematic constraints and the duality between the contact wrenches and twists. Contact forces that may arise with Coulomb friction are described. Multiple-contact manipulation tasks with rigid-body and Coulomb friction are presented. Soft and viscoelastic contact interfaces are presented and discussed. The friction limit surface is introduced and utilized to analyze pushing problems. An example of constructing a friction limit surface for soft contact is presented based on the formulation of force/moment of soft contact interface. Applications of these contact models in fixture analysis and design are presented with the modeling of contact interface. Many references in the bibliography section of this chapter provide further reading on this subject.

Manipulation modes include grasping, pushing, rolling, batting, throwing, catching, casting, and other kinds of quasistatic and dynamic manipulation [37.110]. This chapter presents an overview of contact modeling interfaces, with a particular focus on their use in manipulation tasks, including graspless or *nonprehensile* manipulation modes such as pushing. Analyses of grasps and fixtures are also presented. *Mason*'s textbook [37.14] expands on a number of themes in this chapter, including the theory of polyhedral convex cones, graphical methods for planar problems, and

applications to manipulation planning. Fundamental material on polyhedral cones can be found in [37.111] and their application to representations of twists and wrenches in [37.10, 112–114]. Twists and wrenches are elements of classical *screw theory*, which is covered in the texts [37.115, 116] and from the point of view of robotics in the texts [37.30, 117, 118].

In addition, several relevant chapters in this handbook provide further reading for theory and applications, for example, Chaps. 2 and 38.

Video-References

VIDEO 802 Pushing, sliding, and toppling
available from http://handbookofrobotics.org/view-chapter/37/videodetails/802

VIDEO 803 Horizontal Transport by 2-DOF Vibration
available from http://handbookofrobotics.org/view-chapter/37/videodetails/803

VIDEO 804 Programmable Velocity Vector Fields by 6-DOF Vibration
available from http://handbookofrobotics.org/view-chapter/37/videodetails/804

References

37.1 A. Bicchi: On the problem of decomposing grasp and manipulation forces in multiple whole-limb manipulation, Int. J. Robotics Auton. Syst. **13**, 127–147 (1994)

37.2 K. Harada, M. Kaneko, T. Tsuji: Rolling based manipulation for multiple objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000) pp. 3888–3895

37.3 M.R. Cutkosky, I. Kao: Computing and controlling the compliance of a robotic hand, IEEE Trans. Robotics Autom. **5**(2), 151–165 (1989)

37.4 M.R. Cutkosky, S.-H. Lee: Fixture planning with friction for concurrent product/process design, Proc. NSF Eng. Des. Res. Conf. (1989)

37.5 S.-H. Lee, M. Cutkosky: Fixture planning with friction, ASME J. Eng. Ind. **113**(3), 320–327 (1991)

37.6 Q. Lin, J.W. Burdick, E. Rimon: A stiffness-based quality measure for compliant grasps and fixtures, IEEE Trans. Robotics Autom. **16**(6), 675–688 (2000)

37.7 P. Lötstedt: Coulomb friction in two-dimensional rigid body systems, Z. Angew. Math. Mech. **61**, 605–615 (1981)

37.8 P. Lötstedt: Mechanical systems of rigid bodies subject to unilateral constraints, SIAM J. Appl. Math. **42**(2), 281–296 (1982)

37.9 P.E. Dupont: The effect of Coulomb friction on the existence and uniqueness of the forward dynamics problem, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Nice (1992) pp. 1442–1447

37.10 M.A. Erdmann: On a representation of friction in configuration space, Int. J. Robotics Res. **13**(3), 240–271 (1994)

37.11 K.M. Lynch, M.T. Mason: Pulling by pushing, slip with infinite friction, and perfectly rough surfaces, Int. J. Robotics Res. **14**(2), 174–183 (1995)

37.12 J.S. Pang, J.C. Trinkle: Complementarity formulations and existence of solutions of dynamic multi-rigid-body contact problems with Coulomb friction, Math. Prog. **73**, 199–226 (1996)

37.13 J.C. Trinkle, J.S. Pang, S. Sudarsky, G. Lo: On dynamic multi-rigid-body contact problems with Coulomb friction, Z. Angew. Math. Mech. **77**(4), 267–279 (1997)

37.14 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT Press, Cambrige 2001)

37.15 Y.-T. Wang, V. Kumar, J. Abel: Dynamics of rigid bodies undergoing multiple frictional contacts, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Nice (1992) pp. 2764–2769

37.16 T.H. Speeter: Three-dimensional finite element analysis of elastic continua for tactile sensing, Int. J. Robotics Res. **11**(1), 1–19 (1992)

37.17 K. Dandekar, A.K. Srinivasan: A 3-dimensional finite element model of the monkey fingertip for predicting responses of slowly adapting mechanoreceptors, ASME Bioeng. Conf., Vol. 29 (1995) pp. 257–258

37.18 N. Xydas, M. Bhagavat, I. Kao: Study of soft-finger contact mechanics using finite element analysis and experiments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (2000)

37.19 K. Komvopoulos, D.-H. Choi: Elastic finite element analysis of multi-asperity contacts, J. Tribol. **114**, 823–831 (1992)

37.20 L.T. Tenek, J. Argyris: *Finite Element Analysis for Composite Structures* (Kluwer, Bosten 1998)

37.21 Y. Nakamura: Contact stability measure and optimal finger force control of multi-fingered robot hands, crossing bridges: Advances in flexible automation and robotics, Proc. U.S.-Jpn. Symp. Flex. Autom. (1988) pp. 523–528

37.22 Y.C. Park, G.P. Starr: Optimal grasping using a multifingered robot hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Cincinnati (1990) pp. 689–694

37.23 E. Rimon, J. Burdick: On force and form closure for multiple finger grasps, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 1795–1800

37.24 E. Rimon, J.W. Burdick: New bounds on the number of frictionless fingers required to immobilize

planar objects, J. Robotics Sys. **12**(6), 433–451 (1995)
37.25 E. Rimon, J.W. Burdick: Mobility of bodies in contact – Part I: A 2nd-order mobility index for multiple-finger grasps, IEEE Trans. Robotics Autom. **14**(5), 696–708 (1998)
37.26 D.J. Montana: The kinematics of contact and grasp, Int. J. Robotics Res. **7**(3), 17–32 (1988)
37.27 C.S. Cai, B. Roth: On the planar motion of rigid bodies with point contact, Mech. Mach. Theory **21**(6), 453–466 (1986)
37.28 C. Cai, B. Roth: On the spatial motion of a rigid body with point contact, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 686–695
37.29 A.B.A. Cole, J.E. Hauser, S.S. Sastry: Kinematics and control of multifingered hands with rolling contact, IEEE Trans. Autom. Control **34**(4), 398–404 (1989)
37.30 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)
37.31 F. Reuleaux: *The Kinematics of Machinery* (Dover, New York 1963), reprint of MacMillan, 1876
37.32 C.A. Coulomb: *Theorie des Machines Simples en Ayant Egard au Frottement de Leurs Parties et a la Roideur des Cordages* (Bachelier, Paris 1821)
37.33 Y. Maeda, T. Arai: Planning of graspless manipulation by a multifingered robot hand, Adv. Robotics **19**(5), 501–521 (2005)
37.34 M.T. Mason: Two graphical methods for planar contact problems, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Osaka (1991) pp. 443–448
37.35 R. Howe, I. Kao, M. Cutkosky: Sliding of robot fingers under combined torsion and shear loading, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1, Philadelphia (1988) pp. 103–105
37.36 I. Kao, M.R. Cutkosky: Dextrous manipulation with compliance and sliding, Int. J. Robotics Res. **11**(1), 20–40 (1992)
37.37 R.D. Howe, M.R. Cutkosky: Practical force-motion models for sliding manipulation, Int. J. Robotics Res. **15**(6), 555–572 (1996)
37.38 N. Xydas, I. Kao: Modeling of contact mechanics and friction limit surface for soft fingers with experimental results, Int. J. Robotics Res. **18**(9), 941–950 (1999)
37.39 I. Kao, F. Yang: Stiffness and contact mechanics for soft fingers in grasping and manipulation, IEEE Trans. Robotics Autom. **20**(1), 132–135 (2004)
37.40 J. Jameson, L. Leifer: Quasi-Static Analysis: A method for predicting grasp stability, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 876–883
37.41 S. Goyal, A. Ruina, J. Papadopoulos: Planar sliding with dry friction: Part 2, Dynamics of motion Wear **143**, 331–352 (1991)
37.42 P. Tiezzi, I. Kao: Modeling of viscoelastic contacts and evolution of limit surface for robotic contact interface, IEEE Trans. Robotics **23**(2), 206–217 (2007)
37.43 M. Anitescu, F. Potra: Formulating multi-rigid-body contact problems with friction as solvable linear complementarity problems, ASME J. Nonlin. Dyn. **14**, 231–247 (1997)
37.44 S. Berard, J. Trinkle, B. Nguyen, B. Roghani, J. Fink, V. Kumar: daVinci code: A multi-model simulation and analysis tool for multi-body systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)
37.45 P. Song, J.-S. Pang, V. Kumar: A semi-implicit time-stepping model for frictional compliant contact problems, Int. J. Numer. Methods Eng. **60**(13), 2231–2261 (2004)
37.46 D. Stewart, J. Trinkle: An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and Coulomb friction, Int. J. Numer. Methods Eng. **39**, 2673–2691 (1996)
37.47 R.W. Cottle, J.-S. Pang, R.E. Stone: *The Linear Complementarity Problem* (Academic, New York 1992)
37.48 S.N. Simunovic: Force information in assembly processes, Int. Symp. Ind. Robots (1975)
37.49 V.-D. Nguyen: Constructing force-closure grasps, Int. J. Robotics Res. **7**(3), 3–16 (1988)
37.50 K.M. Lynch: Toppling manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999)
37.51 M.T. Zhang, K. Goldberg, G. Smith, R.-P. Berretty, M. Overmars: Pin design for part feeding, Robotica **19**(6), 695–702 (2001)
37.52 D. Reznik, J. Canny: The Coulomb pump: A novel parts feeding method using a horizontally-vibrating surface, Proc. IEEE Int. Conf. Robotics Autom. (1998) pp. 869–874
37.53 A.E. Quaid: A miniature mobile parts feeder: Operating principles and simulation results, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 2221–2226
37.54 D. Reznik, J. Canny: A flat rigid plate is a universal planar manipulator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 1471–1477
37.55 D. Reznik, J. Canny: C'mon part, do the local motion!, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 2235–2242
37.56 T. Vose, P. Umbanhowar, K.M. Lynch: Vibration-induced frictional force fields on a rigid plate, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)
37.57 M.T. Mason: Mechanics and planning of manipulator pushing operations, Int. J. Robotics Res. **5**(3), 53–71 (1986)
37.58 K.Y. Goldberg: Orienting polygonal parts without sensors, Algorithmica **10**, 201–225 (1993)
37.59 R.C. Brost: Automatic grasp planning in the presence of uncertainty, Int. J. Robotics Res. **7**(1), 3–17 (1988)
37.60 J.C. Alexander, J.H. Maddocks: Bounds on the friction-dominated motion of a pushed object, Int. J. Robotics Res. **12**(3), 231–248 (1993)
37.61 M.A. Peshkin, A.C. Sanderson: The motion of a pushed, sliding workpiece,, IEEE J. Robotics Autom. **4**(6), 569–598 (1988)
37.62 M.A. Peshkin, A.C. Sanderson: Planning robotic manipulation strategies for workpieces that slide, IEEE J. Robotics Autom. **4**(5), 524–531 (1988)

37.63 M. Brokowski, M. Peshkin, K. Goldberg: Curved fences for part alignment, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 467–473

37.64 K.M. Lynch: The mechanics of fine manipulation by pushing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Nice (1992) pp. 2269–2276

37.65 K.M. Lynch, M.T. Mason: Stable pushing: Mechanics, controllability, and planning, Int. J. Robotics Res. **15**(6), 533–556 (1996)

37.66 K. Harada, J. Nishiyama, Y. Murakami, M. Kaneko: Pushing multiple objects using equivalent friction center, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 2485–2491

37.67 J.D. Bernheisel, K.M. Lynch: Stable transport of assemblies: Pushing stacked parts, IEEE Trans. Autom. Sci. Eng. **1**(2), 163–168 (2004)

37.68 J.D. Bernheisel, K.M. Lynch: Stable transport of assemblies by pushing, IEEE Trans. Robotics **22**(4), 740–750 (2006)

37.69 H. Mayeda, Y. Wakatsuki: Strategies for pushing a 3D block along a wall, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Osaka (1991) pp. 461–466

37.70 H. Hertz: On the Contact of Rigid Elastic Solids and on Hardness. In: *Miscellaneous Papers*, ed. by H. Hertz (MacMillan, London 1882) pp. 146–183

37.71 K.L. Johnson: *Contact Mechanics* (Cambridge Univ. Press, Cambridge 1985)

37.72 S.P. Timoshenko, J.N. Goodier: *Theory of Elasticity*, 3rd edn. (McGraw-Hill, New York 1970)

37.73 M.A. Meyers, K.K. Chawla: *Mechanical Behavior of Materials* (Prentice Hall, Upper Saddle River, 1999)

37.74 C.D. Tsai: Nonlinear Modeling on Viscoelastic Contact Interface: Theoretical Study and Experimental Validation, Ph.D. Thesis (Stony Brook University, Stony Brook 2010)

37.75 C. Tsai, I. Kao, M. Higashimori, M. Kaneko: Modeling, sensing and interpretation of viscoelastic contact interface, J. Adv. Robotics **26**(11/12), 1393–1418 (2012)

37.76 Y.C. Fung: *Biomechanics: Mechanical Properties of Living Tissues* (Springer, Berlin, Heidelberg 1993)

37.77 W. Flugge: *Viscoelasticity* (Blaisdell, Waltham 1967)

37.78 J.C. Maxwell: On the dynamical theory of gases, Philos. Trans. R. Soc. Lond. **157**, 49–88 (1867)

37.79 N. Sakamoto, M. Higashimori, T. Tsuji, M. Kaneko: An optimum design of robotic hand for handling a visco-elastic object based on maxwell model, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 1219–1225

37.80 D.P. Noonan, H. Liu, Y.H. Zweiri, K.A. Althoefer, L.D. Seneviratne: A dual-function wheeled probe for tissue viscoelastic property indentification during minimally invasive surgery, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2629–2634

37.81 P. Tiezzi, I. Kao: Characteristics of contact and limit surface for viscoelastic fingers, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Orlando (2006) pp. 1365–1370

37.82 P. Tiezzi, I. Kao, G. Vassura: Effect of layer compliance on frictional behavior of soft robotic fingers, Adv. Robotics **21**(14), 1653–1670 (2007)

37.83 M. Kimura, Y. Sugiyama, S. Tomokuni, S. Hirai: Constructing rheologically deformable virtual objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3737–3743

37.84 W.N. Findley, J.S.Y. Lay: A modified superposition principle applied to creep of non-linear viscoelastic material under abrupt changes in state of combined stress, Trans. Soc. Rheol. **11**(3), 361–380 (1967)

37.85 D.B. Adolf, R.S. Chambers, J. Flemming: Potential energy clock model: Justification and challenging predictions, J. Rheol. **51**(3), 517–540 (2007)

37.86 A.Z. Golik, Y.F. Zabashta: A molecular model of creep and stress relaxation in crystalline polymers, Polym. Mech. **7**(6), 864–869 (1971)

37.87 B.H. Zimm: Dynamics of polymer molecules in dilute solution: Viscoelasticity, flow birefringence and dielectric loss, J. Chem. Phys. **24**(2), 269–278 (1956)

37.88 T. Alfrey: A molecular theory of the viscoelastic behavior of an amorphous linear polymer, J. Chem. Phys. **12**(9), 374–379 (1944)

37.89 P.E. Rouse Jr.: A theory of the linear viscoelastic properties of dilute solutions of coiling polymers, J. Chem. Phys. **21**(7), 1272–1280 (1953)

37.90 F. Bueche: The viscoelastic properties of plastics, J. Chem. Phys. **22**(4), 603–609 (1954)

37.91 L.R.G. Treloar: *The Physics of Rubber Elasticity* (Clarendon Press, Oxford, 1975)

37.92 T.G. Goktekin, A.W. Bargteil, J.F. O'Brien: A method for animating viscoelastic fluid, ACM Trans. Graph. **23**(3), 463–468 (1977)

37.93 S. Arimoto, P.A.N. Nguyen, H.Y. Han, Z. Doulgeri: Dynamics and control of a set of dual fingers with soft tips, Robotica **18**, 71–80 (2000)

37.94 T. Inoue, S. Hirai: Modeling of soft fingertip for object manipulation using tactile sensig, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Las Vegas, Nevada (2003)

37.95 T. Inoue, S. Hirai: Rotational contact model of soft fingertip for tactile sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 2957–2962

37.96 T. Inoue, S. Hirai: Elastic model of deformable fingertip for soft-fingered manipulation, IEEE Trans. Robotics **22**, 1273–1279 (2006)

37.97 T. Inoue, S. Hirai: Dynamic stable manipulation via soft-fingered hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 586–591

37.98 V.A. Ho, D.V. Dat, S. Sugiyama, S. Hirai: Development and analysis of a sliding tactile soft fingertip embedded with a microforce/moment sensor, IEEE Trans. Robotics **27**(3), 411–424 (2011)

37.99 D. Turhan, Y. Mengi: Propagation of initially plane waves in nonhomogeneous viscoelastic media, Int. J. Solids Struct. **13**(2), 79–92 (1977)

37.100 P. Stucky, W. Lord: Finite element modeling of transient ultrasonic waves in linear viscoelastic media, IEEE Trans. Ultrason. Ferroelectr. Freq. Control **48**(1), 6–16 (2001)

37.101 J.M. Pereira, J.J. Mansour, B.R. Davis: Dynamic measurement of the viscoelastic properties of skin, J. Biomech. **24**(2), 157–162 (1991)

37.102 R. Fowles, R.F. Williams: Plane stress wave propagation in solids, J. Appl. Phys. **41**(1), 360–363 (1970)

37.103 E. Wolf: *Progress in Optics* (North-Holland, Amsterdam 1992)

37.104 E.J. Nicolson, R.S. Fearing: The reliability of curvature estimates from linear elastic tactile sensors, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995)

37.105 M. Abramowitz, I. Stegun: *Handbook of Mathematical Functions with Formulas, Graphs, and mathematical Tables*, 7th edn. (Dover, New York 1972)

37.106 I. Kao, S.-F. Chen, Y. Li, G. Wang: Application of bio-engineering contact interface and MEMS in robotic and human augmented systems, IEEE Robotics Autom, Mag. **10**(1), 47–53 (2003)

37.107 S. Goyal, A. Ruina, J. Papadopoulos: Planar sliding with dry friction: Part 1. Limit surface and moment function, Wear **143**, 307–330 (1991)

37.108 J.W. Jameson: Analytic Techniques for Automated Grasp. Ph.D. Thesis (Department of Mechanical Engineering, Stanford University, Stanford 1985)

37.109 S. Goyal, A. Ruina, J. Papadopoulos: Limit surface and moment function description of planar sliding, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Scottsdale (1989) pp. 794–799

37.110 K.M. Lynch, M.T. Mason: Dynamic nonprehensile manipulation: Controllability, planning, and experiments, Int. J. Robotics Res. **18**(1), 64–92 (1999)

37.111 A.J. Goldman, A.W. Tucker: Polyhedral convex cones. In: *Linear Inequalities and Related Systems*, ed. by H.W. Kuhn, A.W. Tucker (Princeton Univ. Press, Princeton 1956)

37.112 M.A. Erdman: A configuration space friction cone, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Osaka (1991) pp. 455–460

37.113 M.A. Erdmann: Multiple-point contact with friction: Computing forces and motions in configuration space, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Yokohama (1993) pp. 163–170

37.114 S. Hirai, H. Asada: Kinematics and statics of manipulation using the theory of polyhedral convex cones, Int. J. Robotics Res. **12**(5), 434–447 (1993)

37.115 R.S. Ball: *The Theory of Screws* (Cambridge Univ. Press, Cambridge 1900)

37.116 K.H. Hunt: *Kinematic Geometry of Mechanisms* (Oxford Univ. Press, Oxford 1978)

37.117 J.K. Davidson, K.H. Hunt: *Robots and Screw Theory* (Oxford Univ. Press, Oxford 2004)

37.118 J.M. Selig: *Geometric Fundamentals of Robotics*, 2nd edn. (Springer, Berlin, Heidelberg 2005)

Multimedia Contents

38. Grasping

Domenico Prattichizzo, Jeffrey C. Trinkle

This chapter introduces fundamental models of grasp analysis. The overall model is a coupling of models that define contact behavior with widely used models of rigid-body kinematics and dynamics. The contact model essentially boils down to the selection of components of contact force and moment that are transmitted through each contact. Mathematical properties of the complete model naturally give rise to five primary grasp types whose physical interpretations provide insight for grasp and manipulation planning.

After introducing the basic models and types of grasps, this chapter focuses on the most important grasp characteristic: complete restraint. A grasp with complete restraint prevents loss of contact and thus is very secure. Two primary restraint properties are *form closure* and *force closure*. A form closure grasp guarantees maintenance of contact as long as the links of the hand and the object are well-approximated as rigid and as long as the joint actuators are sufficiently strong. As will be seen, the primary difference between form closure and force closure grasps is the latter's reliance on contact friction. This translates into requiring fewer contacts to achieve force closure than form closure.

The goal of this chapter is to give a thorough understanding of the all-important grasp properties of form and force closure. This will be done through detailed derivations of grasp models and discussions of illustrative examples. For an in-depth historical perspective and a treasure-trove bibliography of papers addressing a wide range of topics in grasping, the reader is referred to [38.1].

38.1 **Models and Definitions** 956
38.1.1 Velocity Kinematics 957
38.1.2 Dynamics and Equilibrium 960
38.2 **Controllable Twists and Wrenches** 961
38.2.1 Grasp Classifications 962
38.2.2 Limitations of Rigid-Body Formulation 963
38.2.3 Desirable Properties 963
38.3 **Compliant Grasps** 965
38.4 **Restraint Analysis** 967
38.4.1 Form Closure 968
38.4.2 Force Closure 972
38.5 **Examples** 975
38.5.1 Example 1: Grasped Sphere 975
38.5.2 Example 2: Grasped Polygon in the Plane 978
38.5.3 Example 3: Hyperstatic Grasps 980
38.5.4 Example 4: Duality 983
38.5.5 Example 5: Form Closure 983
38.6 **Conclusion and Further Reading** 985
Video-References 986
References 986

Mechanical hands were developed to give robots the ability to grasp objects of varying geometric and physical properties. The first robotic hand designed for dexterous manipulation was the Salisbury Hand (Fig. 38.1) [38.2]. It has three three-jointed fingers; enough to control all six degrees of freedom of an object and the grip pressure. The fundamental grasp modeling and analysis done by Salisbury provides a basis for grasp synthesis and dexterous manipulation research which continues today. Some of the most ma-

ture analysis techniques are embedded in the widely used software *GraspIt!* [38.3] and *SynGrasp* [38.4, 5]. *GraspIt!* contains models for several robot hands and provides tools for grasp selection, dynamic grasp simulation, and visualization. SynGrasp is a MATLAB Toolbox that can be obtained from [38.6] and provides models and functions for grasp analysis with both fully and underactuated hands. It can be a useful educational tool to get aquainted with the mathematical framework of robotic grasping described in this Chapter. Over the years since the Salisbury Hand was built, many articulated robot hands have been developed. Nearly all of these have one actuator per joint or fewer. A notable exception is the *DLR hand arm system* developed by DLR (Deutsches Zentrum für Luft- und Raumfahrt), which has two actuators per joint that drive each joint independently with two antagonistic tendons (Fig. 38.7) [38.7].

38.1 Models and Definitions

A mathematical model of grasping must be capable of predicting the behavior of the hand and object under various loading conditions that may arise during grasping. Generally, the most desirable behavior is grasp maintenance in the face of unknown disturbing forces and moments applied to the object. Typically these disturbances arise from inertia forces which become appreciable during high-speed manipulation or applied forces such as those due to gravity. Grasp maintenance means that the contact forces applied by the hand are such that they prevent contact separation and unwanted contact sliding. The special class of grasps that can be maintained for every possible disturbing load is known as closure grasps. Figure 38.1 shows the Salisbury Hand [38.2, 8], executing a closure grasp of an object by wrapping its fingers around it and pressing it against its palm. Formal definitions, analysis, and computational tests for closure will be presented in Sect. 38.4.

Figure 38.2 illustrates some of the main quantities that will be used to model grasping systems. Assume that the links of the hand and the object are rigid and that there is a unique, well-defined tangent plane at each contact point. Let $\{\boldsymbol{N}\}$ represent a conveniently chosen inertial frame fixed in the workspace. The frame $\{\boldsymbol{B}\}$ is fixed to the object with its origin defined relative to $\{\boldsymbol{N}\}$ by the vector $\boldsymbol{p} \in \mathbb{R}^3$, where $\mathbb{R}^3$ denotes three-dimensional Euclidean space. A convenient choice for $\boldsymbol{p}$ is the center of mass of the object. The position of contact point i in $\{\boldsymbol{N}\}$ is defined by the vector $\boldsymbol{c}_i \in \mathbb{R}^3$. At contact point i, we define a frame $\{\boldsymbol{C}\}_i$, with axes $\{\hat{\boldsymbol{n}}_i, \hat{\boldsymbol{t}}_i, \hat{\boldsymbol{o}}_i\}$ ($\{\boldsymbol{C}\}_1$ is shown in *exploded* view). The unit vector $\hat{\boldsymbol{n}}_i$ is normal to the contact tangent plane and directed toward the object. The other two unit vectors are orthogonal and lie in the tangent plane of the contact.

Let the joints be numbered from 1 to n_q. Denote by $\boldsymbol{q} = [q_1 \cdots q_{n_q}]^{\mathrm{T}} \in \mathbb{R}^{n_q}$ the vector of joint displacements, where the superscript $^{\mathrm{T}}$ indicates matrix transposition. Also, let $\boldsymbol{\tau} = [\tau_1 \cdots \tau_{n_q}]^{\mathrm{T}} \in \mathbb{R}^{n_q}$ represent joint loads (forces in prismatic joints and torques in revolute joints). These loads can result from actuator actions, other applied forces, and inertia forces. They could also arise from interaction at the contacts between the object and hand. However, it will be convenient to sep-

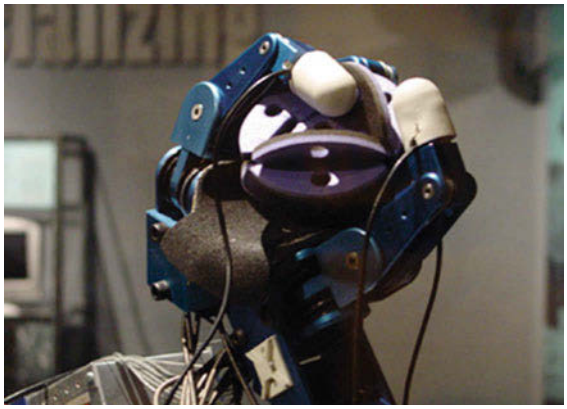


Fig. 38.1 The Salisbury Hand grasping an object

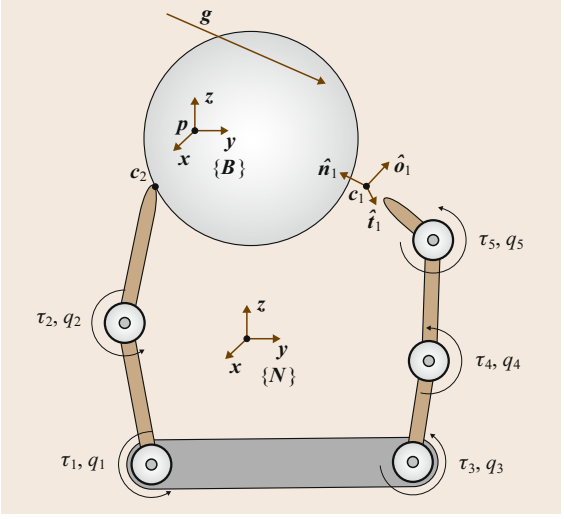


Fig. 38.2 Main quantities for grasp analysis

arate joint loads into two components: those arising from contacts and those arising from all other sources. Throughout this Chapter, non-contact loads will be denoted by $\boldsymbol{\tau}$.

Let $\boldsymbol{u} \in \mathbb{R}^{n_u}$ denote the vector describing the position and orientation of $\{\boldsymbol{B}\}$ relative to $\{\boldsymbol{N}\}$. For planar systems $n_u = 3$. For spatial systems, n_u is three plus the number of parameters used to represent orientation; typically three (for Euler angles) or four (for unit quaternions). Denote by $\boldsymbol{v} = [\boldsymbol{v}^{\mathrm{T}}\ \boldsymbol{\omega}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{n_v}$, the twist of the object described in $\{\boldsymbol{N}\}$. It is composed of the translational velocity $\boldsymbol{v} \in \mathbb{R}^3$ of the point $\boldsymbol{p}$ and the angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$ of the object, both expressed in $\{\boldsymbol{N}\}$. A twist of a rigid body can be referred to any convenient frame fixed to the body. The components of the referred twist represent the velocity of the origin of the new frame and the angular velocity of the body, both expressed in the new frame (Table 38.1). For a rigorous treatment of twists and wrenches see [38.9, 10]. Note that for planar systems, $\boldsymbol{v} \in \mathbb{R}^2$, $\omega \in \mathbb{R}$, and so $n_v = 3$.

Another important point is $\dot{\boldsymbol{u}} \neq \boldsymbol{v}$. Instead, these variables are related by the matrix $\mathbf{V}$ as

$$\dot{\boldsymbol{u}} = \mathbf{V}\boldsymbol{v}\,, \tag{38.1}$$

where the matrix $\mathbf{V} \in \mathbb{R}^{n_u \times n_v}$ is not generally square, but nonetheless satisfies $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}$ [38.11], where $\mathbf{I}$ is the identity matrix, and the dot over the $\boldsymbol{u}$ implies differentiation with respect to time. Note that for planar systems, $\mathbf{V} = \mathbf{I} \in \mathbb{R}^{3\times 3}$.

Let $\boldsymbol{f} \in \mathbb{R}^3$ be the force applied to the object at the point $\boldsymbol{p}$ and let $\boldsymbol{m} \in \mathbb{R}^3$ be the applied moment. These are combined into the object load, or wrench, vector denoted by $\boldsymbol{g} = \left[\boldsymbol{f}^{\mathrm{T}}\ \boldsymbol{m}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{n_v}$, where $\boldsymbol{f}$ and $\boldsymbol{m}$ are expressed in $\{\boldsymbol{N}\}$. Like twists, wrenches can be referred to any convenient frame fixed to the body. One can think of this as translating the line of application of the force until it contains the origin of the new frame, then adjusting the moment component of the wrench to offset the moment induced by moving the line of the force. Last, the force and adjusted moment are expressed in the new frame. As done with the joint loads, the object wrench will be partitioned into two main parts: contact and non-contact wrenches. Throughout this chapter, $\boldsymbol{g}$ will denote the non-contact wrench on the object.

38.1.1 Velocity Kinematics

The material in this chapter is valid for a wide range of robot hands and other grasping mechanisms. The hand is assumed to be composed of a palm that serves as the common base for any number of fingers, each with any number of joints. The formulations given in this chapter are expressed explicitly in terms of only revolute and prismatic joints. However, most other common joints can be modeled by combinations of revolute and prismatic joints (e.g., cylindrical and planar). Any number of contacts may occur between any link and the object.

Table 38.1 Primary notation for grasp analysis

Notation	Definition
n_{c}	Number of contacts
n_q	Number of joints of the hand
n_v	Number of degrees of freedom of object
n_λ	Number of contact wrench components
$\boldsymbol{q} \in \mathbb{R}^{n_q}$	Joint displacements
$\dot{\boldsymbol{q}} \in \mathbb{R}^{n_q}$	Joint velocities
$\boldsymbol{\tau} \in \mathbb{R}^{n_q}$	Non-contact joint loads
$\boldsymbol{u} \in \mathbb{R}^{n_u}$	Position and orientation of object
$\boldsymbol{v} \in \mathbb{R}^{n_v}$	Twist of object
$\boldsymbol{g} \in \mathbb{R}^{n_v}$	Non-contact object wrench
$\boldsymbol{\lambda} \in \mathbb{R}^{n_\lambda}$	Transmitted contact wrenches
$\boldsymbol{v}_{\mathrm{cc}} \in \mathbb{R}^{n_\lambda}$	Transmitted contact twists
$\{\boldsymbol{B}\}$	Frame fixed in object
$\{\boldsymbol{C}\}_i$	Frame at contact i
$\{\boldsymbol{N}\}$	Inertial frame

Grasp Matrix and Hand Jacobian

Two matrices are of the utmost importance in grasp analysis: the *grasp matrix* $\mathbf{G}$ and the *hand Jacobian* $\mathbf{J}$. These matrices define the relevant velocity kinematics and force transmission properties of the contacts. The following derivations of $\mathbf{G}$ and $\mathbf{J}$ will be done under the assumption that the system is three-dimensional ($n_v = 6$). Changes for planar systems will be noted later.

Each contact should be considered as two coincident points; one on the hand and one on the object. The hand Jacobian maps the joint velocities to the twists of the hand to the contact frames, while the transpose of the grasp matrix maps the object twist to the contact frames. Finger joint motions induce a rigid body motion in each link of the hand. It is implicit in the terminology, twists of the hand, that the twist referred to contact i is the twist of the link involved in contact i. Thus these matrices can be derived from the transforms that change the reference frame of a twist.

To derive the grasp matrix, let $\boldsymbol{\omega}^{\mathrm{N}}_{\mathrm{obj}}$ denote the angular velocity of the object expressed in $\{\boldsymbol{N}\}$ and let $\boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{obj}}$, also expressed in $\{\boldsymbol{N}\}$, denote the velocity of the point on the object coincident with the origin of $\{\boldsymbol{C}\}_i$. These velocities can be obtained from the object twist referred to $\{\boldsymbol{N}\}$ as

$$\begin{pmatrix} \boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{obj}} \\ \boldsymbol{\omega}^{\mathrm{N}}_{\mathrm{obj}} \end{pmatrix} = \mathbf{P}_i^{\mathrm{T}}\boldsymbol{v}, \tag{38.2}$$

where

$$\mathbf{P}_i = \begin{pmatrix} \mathbf{I}_{3\times3} & \mathbf{0} \\ \mathbf{S}(\boldsymbol{c}_i - \boldsymbol{p}) & \mathbf{I}_{3\times3} \end{pmatrix} \tag{38.3}$$

$\mathbf{I}_{3\times3} \in \mathbb{R}^{3\times3}$ is the identity matrix, and $\mathbf{S}(\boldsymbol{c}_i - \boldsymbol{p})$ is the cross product matrix, that is, given a three-vector $\boldsymbol{r} = [r_x\, r_y\, r_z]^{\mathrm{T}}$, $\mathbf{S}(\boldsymbol{r})$ is defined as follows

$$\mathbf{S}(\boldsymbol{r}) = \begin{pmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{pmatrix}.$$

The object twist referred to $\{\boldsymbol{C}\}_i$ is simply the vector on the left-hand side of (38.2) expressed in $\{\boldsymbol{C}\}_i$.

Let $\mathbf{R}_i = (\hat{\boldsymbol{n}}_i\, \hat{\boldsymbol{t}}_i\, \hat{\boldsymbol{o}}_i) \in \mathbb{R}^{3\times3}$ represent the orientation of the i-th contact frame $\{\boldsymbol{C}\}_i$ with respect to the inertial frame (the unit vectors $\hat{\boldsymbol{n}}_i$, $\hat{\boldsymbol{t}}_i$, and $\hat{\boldsymbol{o}}_i$ are expressed in $\{\boldsymbol{N}\}$). Then the object twist referred to $\{\boldsymbol{C}\}_i$ is given as

$$\boldsymbol{v}_{i,\mathrm{obj}} = \overline{\mathbf{R}}_i^{\mathrm{T}} \begin{pmatrix} \boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{obj}} \\ \boldsymbol{\omega}^{\mathrm{N}}_{\mathrm{obj}} \end{pmatrix}, \tag{38.4}$$

where $\overline{\mathbf{R}}_i = \mathrm{Blockdiag}(\mathbf{R}_i,\ \mathbf{R}_i) = \begin{pmatrix} \mathbf{R}_i & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_i \end{pmatrix} \in \mathbb{R}^{6\times6}$.

Substituting $\mathbf{P}_i^{\mathrm{T}}\boldsymbol{v}$ from (38.2) into (38.4) yields the partial grasp matrix $\tilde{\mathbf{G}}_i^{\mathrm{T}} \in \mathbb{R}^{6\times6}$, that maps the object twist from $\{\boldsymbol{N}\}$ to $\{\boldsymbol{C}\}_i$

$$\boldsymbol{v}_{i,\mathrm{obj}} = \tilde{\mathbf{G}}_i^{\mathrm{T}}\boldsymbol{v}\,, \tag{38.5}$$

where

$$\tilde{\mathbf{G}}_i^{\mathrm{T}} = \overline{\mathbf{R}}_i^{\mathrm{T}}\mathbf{P}_i^{\mathrm{T}}\,. \tag{38.6}$$

The hand Jacobian can be derived similarly. Let $\boldsymbol{\omega}^{\mathrm{N}}_{i,\mathrm{hnd}}$ be the angular velocity of the link of the hand touching the object at contact i, expressed in $\{\boldsymbol{N}\}$, and define $\boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{hnd}}$ as the translational velocity of contact i on the hand, expressed in $\{\boldsymbol{N}\}$. These velocities are related to the joint velocities through the matrix $\mathbf{Z}_i$ whose columns are the Plücker coordinates of the axes of the joints [38.9, 10]. We have

$$\begin{pmatrix} \boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{hnd}} \\ \boldsymbol{\omega}^{\mathrm{N}}_{i,\mathrm{hnd}} \end{pmatrix} = \mathbf{Z}_i\dot{\boldsymbol{q}}\,, \tag{38.7}$$

where $\mathbf{Z}_i \in \mathbb{R}^{6\times n_q}$ is defined as

$$\mathbf{Z}_i = \begin{pmatrix} \boldsymbol{d}_{i,1} & \cdots & \boldsymbol{d}_{i,n_q} \\ \boldsymbol{\kappa}_{i,1} & \cdots & \boldsymbol{\kappa}_{i,n_q} \end{pmatrix}, \tag{38.8}$$

with the vectors $\boldsymbol{d}_{i,j}, \boldsymbol{\kappa}_{i,j} \in \mathbb{R}^3$ defined as

$$\boldsymbol{d}_{i,j} = \begin{cases} \mathbf{0}_{3\times1} & \text{if contact force } i \text{ does not affect joint } j\,, \\ \hat{\boldsymbol{z}}_j & \text{if joint } j \text{ is prismatic}\,, \\ \mathbf{S}(\boldsymbol{c}_i - \boldsymbol{\zeta}_j)^{\mathrm{T}}\hat{\boldsymbol{z}}_j & \text{if joint } j \text{ is revolute}\,, \end{cases}$$

$$\boldsymbol{\kappa}_{i,j} = \begin{cases} \mathbf{0}_{3\times1} & \text{if contact force } i \text{ does not affect joint } j\,, \\ \mathbf{0}_{3\times1} & \text{if joint } j \text{ is prismatic}\,, \\ \hat{\boldsymbol{z}}_j & \text{if joint } j \text{ is revolute}\,, \end{cases}$$

where $\boldsymbol{\zeta}_j$ is the origin of the coordinate frame associated with the j-th joint and $\hat{\boldsymbol{z}}_j$ is the unit vector in the direction of the z-axis in the same frame, as shown in Fig. 38.12. Both vectors are expressed in $\{\boldsymbol{N}\}$. These frames may be assigned by any convenient method, for example, the Denavit–Hartenberg method [38.12]. The $\hat{\boldsymbol{z}}_j$-axis is the rotational axis for revolute joints and the direction of translation for prismatic joints.

The final step in referring the hand twists to the contact frames is change the frame of expression of $\boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{hnd}}$ and $\boldsymbol{\omega}^{\mathrm{N}}_{i,\mathrm{hnd}}$ to $\{\boldsymbol{C}\}_i$

$$\boldsymbol{v}_{i,\mathrm{hnd}} = \overline{\mathbf{R}}_i^{\mathrm{T}} \begin{pmatrix} \boldsymbol{v}^{\mathrm{N}}_{i,\mathrm{hnd}} \\ \boldsymbol{\omega}^{\mathrm{N}}_{i,\mathrm{hnd}} \end{pmatrix}. \tag{38.9}$$

Combining (38.9) and (38.7) yields the partial hand Jacobian $\tilde{\mathbf{J}}_i \in \mathbb{R}^{6\times n_q}$, which relates the joint velocities to the contact twists on the hand

$$\boldsymbol{v}_{i,\mathrm{hnd}} = \tilde{\mathbf{J}}_i\dot{\boldsymbol{q}}\,, \tag{38.10}$$

where

$$\tilde{\mathbf{J}}_i = \overline{\mathbf{R}}_i^{\mathrm{T}}\mathbf{Z}_i. \tag{38.11}$$

To simplify notation, stack all the twists on the hand and object into the vectors $\boldsymbol{v}_{c,\mathrm{hnd}} \in \mathbb{R}^{6n_c}$ and $\boldsymbol{v}_{c,\mathrm{obj}} \in \mathbb{R}^{6n_c}$ as follows

$$\boldsymbol{v}_{c,\xi} = \begin{pmatrix} \boldsymbol{v}_{1,\xi}^{\mathrm{T}} & \cdots & \boldsymbol{v}_{n_c,\xi}^{\mathrm{T}} \end{pmatrix}^{\mathrm{T}}, \quad \xi = \{\mathrm{obj}, \mathrm{hnd}\}\,.$$

Now the *complete grasp matrix* $\tilde{\mathbf{G}} \in \mathbb{R}^{6\times6n_c}$ and *complete hand Jacobian* $\tilde{\mathbf{J}} \in \mathbb{R}^{6n_c\times n_q}$ relate the various velocity quantities as follows

$$\boldsymbol{v}_{c,\mathrm{obj}} = \tilde{\mathbf{G}}^{\mathrm{T}}\boldsymbol{v}\,, \tag{38.12}$$

$$\boldsymbol{v}_{c,\mathrm{hnd}} = \tilde{\mathbf{J}}\dot{\boldsymbol{q}}\,, \tag{38.13}$$

where

$$\tilde{\mathbf{G}}^{\mathrm{T}} = \begin{pmatrix} \tilde{\mathbf{G}}_1^{\mathrm{T}} \\ \vdots \\ \tilde{\mathbf{G}}_{n_c}^{\mathrm{T}} \end{pmatrix}, \quad \tilde{\mathbf{J}} = \begin{pmatrix} \tilde{\mathbf{J}}_1 \\ \vdots \\ \tilde{\mathbf{J}}_{n_c} \end{pmatrix}. \tag{38.14}$$

The term *complete* is used to emphasize that all $6n_c$ twist components at the contacts are included in the mapping. See Example 1, Part 1 and Example 3, Part 1 at the end of this chapter for clarification.

Contact Modeling

Contacts play a central role in grasping. Contacts allows to impose a given motion to the object or to apply a given force through the object. All grasping actions go through contacts whose model and control is crucial in grasping. Three contact models useful for grasp analysis are reviewed here. For a complete discussion of contact modeling in robotics, readers are referred to Chap. 37.

The three models of greatest interest in grasp analysis are known as *point-contact-without-friction*, *hard-finger*, and *soft-finger* [38.13]. These models select components of the contact twists to transmit between the hand and the object. This is done by equating a subset of the components of the hand and object twist at each contact. The corresponding components of the contact force and moment are also equated, but without regard for the constraints imposed by a friction model (Sect. 38.4.2).

The point-contact-without-friction (PwoF) model is used when the contact patch is very small and the surfaces of the hand and object are slippery. With this model, only the normal component of the translational velocity of the contact point on the hand (i. e., the first component of $\boldsymbol{v}_{i,\mathrm{hnd}}$) is transmitted to the object. The two components of tangential velocity and the three components of angular velocity are not transmitted. Analogously, the normal component of the contact force is transmitted, but the frictional forces and moments are assumed to be negligible.

A hard finger (HF) model is used when there is significant contact friction, but the contact patch is small, so that no appreciable friction moment exists. When this model is applied to a contact, all three translational velocity components of the contact point on the hand (i. e., the first three components of $\boldsymbol{v}_{i,\mathrm{hnd}}$) and all three components of the contact force are transmitted through the contact. None of the angular velocity components or moment components are transmitted.

The soft finger (SF) model is used in situations in which the surface friction and the contact patch are large enough to generate significant friction forces and a friction moment about the contact normal. At a contact where this model is enforced, the three translational velocity components of the contact on the hand and the angular velocity component about the contact normal are transmitted (i. e., the first four components of $\boldsymbol{v}_{i,\mathrm{hnd}}$). Similarly, all three components of contact force and the normal component of the contact moment are transmitted.

Remark 38.1

The reader may see a contradiction between the rigid-body assumption and the soft-finger model. The rigid-body assumption is an approximation that simplifies all aspects of the analysis of grasping, but nonetheless is sufficiently accurate in many real situations. Without it, grasp analysis would be impractical. On the other hand, the need for a soft-finger model is a clear admission that the finger links and object are not rigid. However, it can be usefully applied in situations in which the amount of deformation required to obtain a large contact patch is small. Such situations occur when the local surface geometries are similar. If large finger or body deformations exist in the real system, the rigid-body approach presented in this chapter should be used with caution.

To develop the PwoF, HF, and SF models, define the relative twist at contact i as follows

$$\begin{pmatrix}\tilde{\mathbf{J}}_i & -\tilde{\mathbf{G}}_i^{\mathrm{T}}\end{pmatrix}\begin{pmatrix}\dot{\boldsymbol{q}}\\ \boldsymbol{\nu}\end{pmatrix} = \boldsymbol{\nu}_{i,\mathrm{hnd}} - \boldsymbol{\nu}_{i,\mathrm{obj}}\,.$$

A particular contact model is defined through the matrix $\mathbf{H}_i \in \mathbb{R}^{n_{\lambda i}\times 6}$, which selects $n_{\lambda i}$ components of the relative contact twist and sets them to zero

$$\mathbf{H}_i(\boldsymbol{\nu}_{i,\mathrm{hnd}} - \boldsymbol{\nu}_{i,\mathrm{obj}}) = \mathbf{0}\,.$$

These components are referred to as transmitted degrees of freedom (DOF). Define $\mathbf{H}_i$ as follows

$$\mathbf{H}_i = \left[\begin{array}{c|c}\mathbf{H}_{i\mathrm{F}} & \mathbf{0}\\ \hline \mathbf{0} & \mathbf{H}_{i\mathrm{M}}\end{array}\right], \tag{38.15}$$

where $\mathbf{H}_{i\mathrm{F}}$ and $\mathbf{H}_{i\mathrm{M}}$ are the translational and rotational component selection matrices. Table 38.2 gives the definitions of the selection matrices for the three contact models, where *vacuous* for $\mathbf{H}_{i\mathrm{M}}$ means that the corresponding block row matrix in (38.15) is void (i. e., it has zero rows and columns). Notice that for the SF model, $\mathbf{H}_{i\mathrm{M}}$ selects rotation about the contact normal.

After choosing a transmission model for each contact, the kinematic contact constraint equations for all n_c contacts can be written in compact form as

$$\mathbf{H}(\boldsymbol{\nu}_{c,\mathrm{hnd}} - \boldsymbol{\nu}_{c,\mathrm{obj}}) = \mathbf{0}\,, \tag{38.16}$$

Table 38.2 Selection matrices for three contact models

Model	$n_{\lambda i}$	$\mathbf{H}_{i\mathrm{F}}$	$\mathbf{H}_{i\mathrm{M}}$
PwoF	1	(1 0 0)	Vacuous
HF	3	$\mathbf{I}_{3\times 3}$	Vacuous
SF	4	$\mathbf{I}_{3\times 3}$	(1 0 0)

where

$$\mathbf{H} = \text{Blockdiag}(\mathbf{H}_1, \ldots, \mathbf{H}_{n_c}) \in \mathbb{R}^{n_\lambda \times 6n_c},$$

and the number of twist components n_λ transmitted through the n_c contacts is given by $n_\lambda = \sum_{i=1}^{n_c} n_{\lambda i}$.

Finally, by substituting (38.12) and (38.13) into (38.16) one gets the compact form of the velocity kinematic contact constraints

$$\begin{pmatrix} \mathbf{J} & -\mathbf{G}^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{\nu} \end{pmatrix} = \mathbf{0}\,, \tag{38.17}$$

where the grasp matrix and hand Jacobian are finally defined as

$$\begin{aligned} \mathbf{G}^{\mathrm{T}} &= \mathbf{H}\tilde{\mathbf{G}}^{\mathrm{T}} \in \mathbb{R}^{n_\lambda \times 6}\,, \\ \mathbf{J} &= \mathbf{H}\tilde{\mathbf{J}} \in \mathbb{R}^{n_\lambda \times n_q}\,. \end{aligned} \tag{38.18}$$

For more details on the construction of $\mathbf{H}$, the grasp matrix, and the hand Jacobian, readers are referred to [38.14–16] and the references therein. Also, see Example 1, Part 2 and Example 3, Part 2.

It is worth noting that (38.17) can be written in the following form

$$\mathbf{J}\dot{\boldsymbol{q}} = \boldsymbol{\nu}_{\mathrm{cc,hnd}} = \boldsymbol{\nu}_{\mathrm{cc,obj}} = \mathbf{G}^{\mathrm{T}}\boldsymbol{\nu}\,, \tag{38.19}$$

where $\boldsymbol{\nu}_{\mathrm{cc,hnd}}$ and $\boldsymbol{\nu}_{\mathrm{cc,obj}}$ contain only the components of the twists that are transmitted by the contacts. To underline the central role of contact constraints in grasping, it is worth noting that *grasp maintenance* is defined as the situation in which constraints (38.19) are maintained over time. The kinematic contact constraint holds only if the contact force satisfies the friction constraints for the contact models with friction or the unilateral constraint for the contact model without friction (Sect. 38.4.2).

Thus, when a contact is frictionless, contact maintenance implies continued contact, but sliding is allowed. However, when a contact is of the type HF, contact maintenance implies sticking contact, since sliding would violate the HF model. Similarly, for a SF contact, there may be no sliding or relative rotation about the contact normal.

For the remainder of this chapter, it will be assumed that $\boldsymbol{\nu}_{\mathrm{cc,hnd}} = \boldsymbol{\nu}_{\mathrm{cc,obj}}$, so the notation will be shortened to $\boldsymbol{\nu}_{\mathrm{cc}}$.

Planar Simplifications

Assume that the plane of motion is the (x, y)-plane of $\{N\}$. The vectors $\boldsymbol{\nu}$ and $\boldsymbol{g}$ reduce in dimension from six to three by dropping components three, four, and five. The dimensions of vectors $\boldsymbol{c}_i$ and $\boldsymbol{p}$ reduce from three to two. The i-th rotation matrix becomes $\mathbf{R}_i = [\hat{\boldsymbol{n}}_i\, \hat{\boldsymbol{t}}_i] \in \mathbb{R}^{2\times 2}$ (where the third component of $\hat{\boldsymbol{n}}_i$ and $\hat{\boldsymbol{t}}_i$ is dropped) and (38.4) holds with $\overline{\mathbf{R}}_i = \text{Blockdiag}(\mathbf{R}_i, 1) \in \mathbb{R}^{3\times 3}$. Equation (38.2) holds with

$$\mathbf{P}_i = \begin{pmatrix} \mathbf{I}_{2\times 2} & \mathbf{0} \\ \mathbf{S}_2(\boldsymbol{c}_i - \boldsymbol{p}) & 1 \end{pmatrix},$$

where $\mathbf{S}_2$ is the analog of the cross product matrix for two-dimensional vectors, given as

$$\mathbf{S}_2(\boldsymbol{r}) = \begin{pmatrix} -r_y & r_x \end{pmatrix}.$$

Equation (38.7) holds with $\boldsymbol{d}_{i,j} \in \mathbb{R}^2$ and $\boldsymbol{\kappa}_{i,j} \in \mathbb{R}$ defined as

$$\boldsymbol{d}_{i,j} = \begin{cases} \mathbf{0}_{2\times 1} & \text{if contact force } i \text{ does not affect the joint } j\,, \\ \hat{\boldsymbol{z}}_j & \text{if joint } j \text{ is prismatic}\,, \\ \mathbf{S}(\boldsymbol{c}_i - \boldsymbol{\zeta}_j)^{\mathrm{T}} & \text{if joint } j \text{ is revolute}\,, \end{cases}$$

$$\kappa_{i,j} = \begin{cases} 0 & \text{if contact force } i \text{ does not affect joint } j\,, \\ 0 & \text{if joint } j \text{ is prismatic}\,, \\ 1 & \text{if joint } j \text{ is revolute}\,. \end{cases}$$

The complete grasp matrix and hand Jacobian have reduced sizes: $\tilde{\mathbf{G}}^{\mathrm{T}} \in \mathbb{R}^{3n_c\times 3}$ and $\tilde{\mathbf{J}} \in \mathbb{R}^{3n_c\times n_q}$. As far as contact constraint is concerned, (38.15) holds with $\mathbf{H}_{i\mathrm{F}}$ and $\mathbf{H}_{i\mathrm{M}}$ defined in Table 38.3.

In the planar case, the SF and HF models are equivalent, because the object and the hand lie in a plane. Rotations about the contact normals would cause out-of-plane motions. Finally, the dimensions of the grasp matrix and hand Jacobian are reduced to the following sizes: $\mathbf{G}^{\mathrm{T}} \in \mathbb{R}^{n_\lambda\times 3}$ and $\mathbf{J} \in \mathbb{R}^{n_\lambda\times n_q}$. See Example 1, Part 3 and Example 2, Part 1.

38.1.2 Dynamics and Equilibrium

Dynamic equations of the system can be written as

$$\begin{aligned} &\mathbf{M}_{\mathrm{hnd}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{b}_{\mathrm{hnd}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \mathbf{J}^{\mathrm{T}}\boldsymbol{\lambda} = \boldsymbol{\tau}_{\mathrm{app}} \\ &\mathbf{M}_{\mathrm{obj}}(\boldsymbol{u})\dot{\boldsymbol{\nu}} + \boldsymbol{b}_{\mathrm{obj}}(\boldsymbol{u}, \boldsymbol{\nu}) - \mathbf{G}\boldsymbol{\lambda} = \boldsymbol{g}_{\mathrm{app}} \\ &\text{subject to constraint (38.17)}\,, \end{aligned} \tag{38.20}$$

Table 38.3 Definitions

Model	$n_{\lambda i}$	$\mathbf{H}_{i\mathrm{F}}$	$\mathbf{H}_{i\mathrm{M}}$
PwoF	1	$\begin{pmatrix} 1 & 0 \end{pmatrix}$	Vacuous
HF/SF	2	$\mathbf{I}_{2\times 2}$	Vacuous

Table 38.4 Vectors of contact force and moment components, also known as the wrench intensity vector, transmitted through contact i

Model	$\boldsymbol{\lambda}_i$
PwoF	(f_{in})
HF	$(f_{in}\, f_{it}\, f_{io})^{\mathrm{T}}$
SF	$(f_{in}\, f_{it}\, f_{io}\, m_{in})^{\mathrm{T}}$

where $\mathbf{M}_{\mathrm{hnd}}(\cdot)$ and $\mathbf{M}_{\mathrm{obj}}(\cdot)$ are symmetric, positive definite inertia matrices, $\boldsymbol{b}_{\mathrm{hnd}}(\cdot,\cdot)$ and $\boldsymbol{b}_{\mathrm{obj}}(\cdot,\cdot)$ are the velocity-product terms, $\boldsymbol{g}_{\mathrm{app}}$ is the force and moment applied to the object by gravity and other external sources, $\boldsymbol{\tau}_{\mathrm{app}}$ is the vector of external loads and actuator actions, and the vector $\mathbf{G}\boldsymbol{\lambda}$ is the total wrench applied to the object by the hand. The vector $\boldsymbol{\lambda}$ contains the contact force and moment components transmitted through the contacts and expressed in the contact frames. Specifically, $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1^{\mathrm{T}} \cdots \boldsymbol{\lambda}_{n_c}^{\mathrm{T}}]^{\mathrm{T}}$, where

$$\boldsymbol{\lambda}_i = \mathbf{H}_i [f_{in}\, f_{it}\, f_{io}\, m_{in}\, m_{it}\, m_{io}]^{\mathrm{T}} \,.$$

The subscripts indicate one normal (n) and two tangential (t,o) components of contact force $\boldsymbol{f}$ and moment $\boldsymbol{m}$. For a SF, HF, or PwoF contact, $\boldsymbol{\lambda}_i$ is defined as in Table 38.4. Finally, it is worth noting that $\mathbf{G}_i\boldsymbol{\lambda}_i = \tilde{\mathbf{G}}_i\mathbf{H}_i^{\mathrm{T}}\boldsymbol{\lambda}_i$ is the wrench applied through contact i, where $\tilde{\mathbf{G}}_i$ and $\mathbf{H}_i$ are defined in (38.6) and (38.15). The vector $\boldsymbol{\lambda}_i$ is known as the wrench intensity vector for contact i.

Equation (38.20) represents the dynamics of the hand and object without regard for the kinematic constraints imposed by the contact models. Enforcing them, the dynamic model of the system can be written as follows

$$\begin{pmatrix} \mathbf{J}^{\mathrm{T}} \\ -\mathbf{G} \end{pmatrix} \boldsymbol{\lambda} = \begin{pmatrix} \boldsymbol{\tau} \\ \boldsymbol{g} \end{pmatrix} \tag{38.21}$$

subject to $\mathbf{J}\dot{\boldsymbol{q}} = \mathbf{G}^{\mathrm{T}}\boldsymbol{v} = \boldsymbol{v}_{\mathrm{cc}}$, where

$$\begin{aligned} \boldsymbol{\tau} &= \boldsymbol{\tau}_{\mathrm{app}} - \mathbf{M}_{\mathrm{hnd}}(\boldsymbol{q})\ddot{\boldsymbol{q}} - \boldsymbol{b}_{\mathrm{hnd}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ \boldsymbol{g} &= \boldsymbol{g}_{\mathrm{app}} - \mathbf{M}_{\mathrm{obj}}(\boldsymbol{u})\dot{\boldsymbol{v}} - \boldsymbol{b}_{\mathrm{obj}}(\boldsymbol{u}, \boldsymbol{v}) \,. \end{aligned} \tag{38.22}$$

One should notice that the dynamic equations are closely related to the velocity kinematic model in (38.17). Specifically, just as $\mathbf{J}$ and $\mathbf{G}^{\mathrm{T}}$ transmit only selected components of contact twists, $\mathbf{J}^{\mathrm{T}}$ and $\mathbf{G}$ in (38.20) serve to transmit only the corresponding components of the contact wrenches.

When the inertia terms are negligible, as occurs during slow motion, the system is said to be quasi-static. In this case, (38.22) becomes

$$\begin{aligned} \boldsymbol{\tau} &= \boldsymbol{\tau}_{\mathrm{app}} \,, \\ \boldsymbol{g} &= \boldsymbol{g}_{\mathrm{app}} \,, \end{aligned} \tag{38.23}$$

and does not depend on joint and object velocities. Consequently, when the grasp is in static equilibrium or moves quasi-statically, one can solve the first equation and the constraint in (38.21) independently to compute $\boldsymbol{\lambda}$, $\dot{\boldsymbol{q}}$, and $\boldsymbol{v}$. It is worth noting that such a force/velocity decoupled solution is not possible when dynamic effects are appreciable, since the first equation in (38.21) depends on the third one through (38.22).

Remark 38.2

Equation (38.21) highlights an important alternative view of the grasp matrix and the hand Jacobian. $\mathbf{G}$ can be thought of as a mapping from the transmitted contact forces and moments to the set of wrenches that the hand can apply to the object, while $\mathbf{J}^{\mathrm{T}}$ can be thought of as a mapping from the transmitted contact forces and moments to the vector of joint loads. Notice that these interpretations hold for both dynamic and quasi-static conditions.

38.2 Controllable Twists and Wrenches

In hand design and in grasp and manipulation planning, it is important to know the set of twists that can be imparted to the object by movements of the fingers, and conversely, the conditions under which the hand can prevent all possible motions of the object. The dual view is that one needs to know the set of wrenches that the hand can apply to the object and under what conditions any wrench in $\mathbb{R}^6$ can be applied through the contacts. This knowledge will be gained by studying the various subspaces associated with $\mathbf{G}$ and $\mathbf{J}$ [38.17].

The spaces, shown in Fig. 38.3, are the column spaces and null spaces of $\mathbf{G}$, $\mathbf{G}^{\mathrm{T}}$, $\mathbf{J}$, and $\mathbf{J}^{\mathrm{T}}$. Column space (also known as range) and null space will be denoted by $\mathcal{R}(\cdot)$ and $\mathcal{N}(\cdot)$, respectively. The arrows show the propagation of the various velocity and load quantities through the grasping system. For example, in the left part of Fig. 38.3 it is shown how any vector $\dot{\boldsymbol{q}} \in \mathbb{R}^{n_q}$ can be decomposed into a sum of two orthogonal vectors in $\mathcal{R}(\mathbf{J}^{\mathrm{T}})$ and in $\mathcal{N}(\mathbf{J})$ and how $\dot{\boldsymbol{q}}$ is mapped to $\mathcal{R}(\mathbf{J})$ by multiplication by $\mathbf{J}$.

It is important to recall two facts from linear algebra. First, a matrix $\mathbf{A}$ maps vectors from $\mathcal{R}(\mathbf{A}^{\mathrm{T}})$ to $\mathcal{R}(\mathbf{A})$ in a one-to-one and onto fashion, that is, the map $\mathbf{A}$ is a bijection. The generalized inverse $\mathbf{A}^{+}$ of

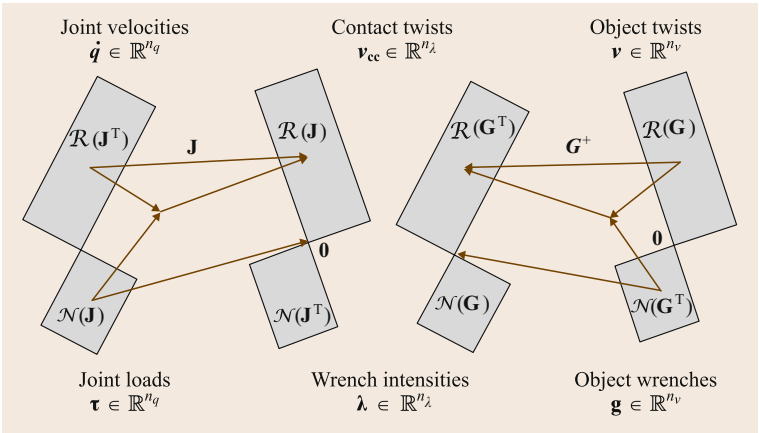


Fig. 38.3 Linear maps relating twists and wrenches of a grasping system

$\mathbf{A}$ is a bijection that maps vectors in the opposite direction [38.18]. Also, $\mathbf{A}$ maps all vectors in $\mathcal{N}(\mathbf{A})$ to zero. Finally, there is no non-trivial vector that $\mathbf{A}$ can map into $\mathcal{N}(\mathbf{A}^T)$. This implies that if $\mathcal{N}(\mathbf{G}^T)$ is non-trivial, then the hand will not be able to control all degrees of freedom of the object's motion. This is certainly true for quasi-static grasping, but when dynamics are important, they may cause the object to move along the directions in $\mathcal{N}(\mathbf{G}^T)$.

38.2.1 Grasp Classifications

The four null spaces motivate a basic classification of grasping systems defined in Table 38.5. Assuming solutions to (38.21) exist, the following force and velocity equations provide insight into the physical meaning of the various null spaces

$$\dot{q} = \mathbf{J}^+ v_{cc} + \mathbf{N}(\mathbf{J})\gamma \,, \tag{38.24}$$

$$v = (\mathbf{G}^T)^+ v_{cc} + \mathbf{N}(\mathbf{G}^T)\gamma \,, \tag{38.25}$$

$$\lambda = -\mathbf{G}^+ g + \mathbf{N}(\mathbf{G})\gamma \,, \tag{38.26}$$

$$\lambda = (\mathbf{J}^T)^+ \tau + \mathbf{N}(\mathbf{J}^T)\gamma \,. \tag{38.27}$$

In these equations, $\mathbf{A}^+$ denotes the generalized inverse, henceforth pseudoinverse, of a matrix $\mathbf{A}$, $\mathbf{N}(\mathbf{A})$ denotes a matrix whose columns form a basis for $\mathcal{N}(\mathbf{A})$, and γ is an arbitrary vector of appropriate dimension that parameterizes the solution sets. If not otherwise specified, the context will make clear if the generalized inverse is left or right.

Table 38.5 Basic grasp classes

Condition	Class	Many-to-one
$\mathcal{N}(\mathbf{J}) \neq \mathbf{0}$	Redundant	$\dot{q} \to v_{cc}$ $\tau \to \lambda$
$\mathcal{N}(\mathbf{G}^T) \neq \mathbf{0}$	Indeterminate	$v \to v_{cc}$ $g \to \lambda$
$\mathcal{N}(\mathbf{G}) \neq \mathbf{0}$	Graspable	$\lambda \to g$ $v_{cc} \to v$
$\mathcal{N}(\mathbf{J}^T) \neq \mathbf{0}$	Defective	$\lambda \to \tau$ $v_{cc} \to \dot{q}$

If the null spaces represented in the equations are non-trivial, then one immediately sees the first many-to-one mapping in Table 38.5. To see the other many-to-one mappings, and in particular the defective class, consider (38.24). It can be rewritten with v_{cc} decomposed into orthogonal components v_{rs} and v_{lns} in $\mathcal{R}(\mathbf{J})$ and $\mathcal{N}(\mathbf{J}^T)$, respectively

$$\dot{q} = \mathbf{J}^+ (v_{rs} + v_{lns}) + \mathbf{N}(\mathbf{J})\gamma \,. \tag{38.28}$$

Recall that every vector in $\mathcal{N}(\mathbf{A}^T)$ is orthogonal to every row of $\mathbf{A}^+$. Therefore $\mathbf{J}^+ v_{lns} = \mathbf{0}$. If γ and v_{rs} are fixed in (38.28), then $\dot{q}$ is unique. Thus it is clear that if $\mathcal{N}(\mathbf{J}^T)$ is non-trivial, then a subspace of twists of the hand at the contacts will map to a single joint velocity vector. Applying the same approach to the other three equations (38.25)–(38.27) yields the other many-to-one mappings listed in Table 38.5.

Equations (38.21) and (38.24)–(38.27), motivate the following definitions.

Definition 38.1 Redundant
A grasping system is said to be *redundant* if $\mathcal{N}(\mathbf{J})$ is non-trivial.

Joint velocities $\dot{q}$ in $\mathcal{N}(\mathbf{J})$ are referred to as *internal hand velocities*, since they correspond to finger motions, but do not generate motion of the hand in the constrained directions at the contact points. If the quasi-static model applies, it can be shown that these motions are not influenced by the motion of the object and vice versa.

Definition 38.2 Indeterminate
A grasping system is said to be *indeterminate* if $\mathcal{N}(\mathbf{G}^{\mathrm{T}})$ is non-trivial.

Object twists $\boldsymbol{v}$ in $\mathcal{N}(\mathbf{G}^{\mathrm{T}})$ are called *internal object twists*, since they correspond to motions of the object, but do not cause motion of the object in the constrained directions at the contacts. If the static model applies, it can be shown that these twists cannot be controlled by finger motions.

Definition 38.3 Graspable
A grasping system is said to be *graspable* if $\mathcal{N}(\mathbf{G})$ is non-trivial.

Wrench intensities $\boldsymbol{\lambda}$ in $\mathcal{N}(\mathbf{G})$ are referred to as *internal object forces*. These wrenches are *internal* because they do not contribute to the acceleration of the object, i. e., $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$. Instead, these wrench intensities affect the tightness of the grasp. Thus, internal wrench intensities play a fundamental role in maintaining grasps that rely on friction (Sect. 38.4.2).

Definition 38.4 Defective
A grasping system is said to be *defective* if $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ is non-trivial.

Wrench intensities $\boldsymbol{\lambda}$ in $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ are called *internal hand forces*. These forces do not influence the hand joint dynamics given in (38.20). If the static model is considered, it can be easily shown that wrench intensities belonging to $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ cannot be generated by joint actions, but can be resisted by the structure of the hand.

See Example 1, Part 4, Example 2, Part 2 and Example 3, Part 3.

38.2.2 Limitations of Rigid-Body Formulation

The rigid-body dynamics equation (38.20) can be rewritten with Lagrange multipliers associated with the contact constraints as

$$\mathbf{M}_{\mathrm{dyn}} \begin{pmatrix} \ddot{\boldsymbol{q}} \\ \dot{\boldsymbol{\nu}} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\tau} - \boldsymbol{b}_{\mathrm{hnd}} \\ \boldsymbol{\nu} - \boldsymbol{b}_{\mathrm{obj}} \\ \boldsymbol{b}_{\mathrm{c}} \end{pmatrix}, \tag{38.29}$$

where $\boldsymbol{b}_{\mathrm{c}} = [\partial(\mathbf{J}\dot{\boldsymbol{q}})/\partial \boldsymbol{q}]\dot{\boldsymbol{q}} - [\partial(\mathbf{G}\boldsymbol{\nu})/\partial \boldsymbol{u}]\dot{\boldsymbol{u}}$ and

$$\mathbf{M}_{\mathrm{dyn}} = \begin{pmatrix} \mathbf{M}_{\mathrm{hnd}} & \mathbf{0} & \mathbf{J}^{\mathrm{T}} \\ \mathbf{0} & \mathbf{M}_{\mathrm{obj}} & -\mathbf{G} \\ \mathbf{J} & -\mathbf{G}^{\mathrm{T}} & \mathbf{0} \end{pmatrix}.$$

In order for this equation to completely determine the motion of the system, it is necessary that matrix $\mathbf{M}_{\mathrm{dyn}}$ be invertible. This case is considered in detail in [38.19], where the dynamics of multi-finger manipulation is studied under the hypothesis that the hand Jacobian is full row rank, i. e., $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$. For all manipulation systems with non-invertible $\mathbf{M}_{\mathrm{dyn}}$, rigid-body dynamics fails to determine the motion and the wrench intensity vector. By observing that

$$\mathcal{N}(\mathbf{M}_{\mathrm{dyn}}) = \{(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{\nu}}, \boldsymbol{\lambda})^{\mathrm{T}} | \ddot{\boldsymbol{q}} = \mathbf{0}, \dot{\boldsymbol{\nu}} = \mathbf{0}, \boldsymbol{\lambda} \in \mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G})\},$$

the same arguments apply under quasi-static conditions defined by (38.21) and (38.23). When $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G}) \neq \mathbf{0}$, the rigid-body approach fails to solve the first equation in (38.21), thus leaving $\boldsymbol{\lambda}$ indeterminate.

Definition 38.5 Hyperstatic
A grasping system is said to be *hyperstatic* if

$$\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G})$$

is non-trivial.

In such systems there are internal forces (Definition 38.3) belonging to $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ that are not controllable as discussed for defective grasps. Rigid-body dynamics is not satisfactory for hyperstatic grasps, since the rigid-body assumption leads to undetermined contact wrenches [38.20].

See Example 3, Part 3.

38.2.3 Desirable Properties

For a general purpose grasping system, there are three main desirable properties: control of the object twist $\boldsymbol{\nu}$, control of object wrench $\boldsymbol{g}$, and control of the internal forces. Control of these quantities implies that the hand can deliver the desired $\boldsymbol{\nu}$ and $\boldsymbol{g}$ with specified grip pressure by the appropriate choice of joint velocities and actions. The conditions on $\mathbf{J}$ and $\mathbf{G}$ equivalent to these properties are given in Table 38.6.

We derive the conditions in two steps. First, we ignore the structure and configuration of the hand (captured in $\mathbf{J}$) by assuming that each contact point on each

Table 38.6 Desirable properties of a grasp

Task requirement	Required conditions
All wrenches possible, $\boldsymbol{g}$ All twists possible, $\boldsymbol{\nu}$	$\mathrm{rank}(\mathbf{G}) = n_{\nu}$
Control all wrenches, $\boldsymbol{g}$ Control all twists, $\boldsymbol{\nu}$	$\mathrm{rank}(\mathbf{GJ}) = \mathrm{rank}(\mathbf{G}) = n_{\nu}$
Control all internal forces	$\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$

Fig. 38.4 The Salisbury Hand

finger can be commanded to move in any direction transmitted by the chosen contact model. An important perspective here is that $\boldsymbol{\nu}_{cc}$ is seen as the independent input variable and $\boldsymbol{\nu}$ is seen as the output. The dual interpretation is that the actuators can generate any contact force and moment in the constrained directions at each contact. Similarly, $\boldsymbol{\lambda}$ is seen as the input and $\boldsymbol{g}$ is seen as the output. The primary property of interest under this assumption is whether or not the arrangement and types of contacts on the object (captured in $\mathbf{G}$) are such that a sufficiently dexterous hand could control its fingers so as to impart any twist $\boldsymbol{\nu} \in \mathbb{R}^6$ to the object and, similarly, to apply any wrench $\boldsymbol{g} \in \mathbb{R}^6$ to the object.

All Object Twists Possible

Given a set of contact locations and types, by solving (38.19) for $\boldsymbol{\nu}$ or observing the map $\mathbf{G}$ on the right side of Fig. 38.3, one sees that the achievable object twists are those in $\mathcal{R}(\mathbf{G})$. Those in $\mathcal{N}(\mathbf{G}^{\mathrm{T}})$ cannot be achieved by any hand using the given grasp. Therefore, to achieve any object twist, we must have: $\mathcal{N}(\mathbf{G}^{\mathrm{T}}) = \mathbf{0}$, or equivalently, $\mathrm{rank}(\mathbf{G}) = n_\nu$. Any grasp with three non-collinear hard contacts or two distinct soft contacts satisfies this condition.

All Object Wrenches Possible

This case is the dual of the previous case, so we expect the same condition. From (38.21), one immediately obtains the condition $\mathcal{N}(\mathbf{G}^{\mathrm{T}}) = \mathbf{0}$, so again we have $\mathrm{rank}(\mathbf{G}) = n_\nu$.

To obtain the conditions needed to control the various quantities of interest, the structure of the hand cannot be ignored. Recall that the only achievable contact twists on the hand are in $\mathcal{R}(\mathbf{J})$, which is not necessarily equal to $\mathbb{R}^{n_\lambda}$.

Control All Object Twists

By solving (38.17) for $\boldsymbol{\nu}$, one sees that in order to cause any object twist $\boldsymbol{\nu}$ by choice of joint velocities $\dot{\boldsymbol{q}}$, we must have $\mathcal{R}(\mathbf{GJ}) = \mathbb{R}^{n_\nu}$ and $\mathcal{N}(\mathbf{G}^{\mathrm{T}}) = 0$. These conditions are equivalent to $\mathrm{rank}(\mathbf{GJ}) = \mathrm{rank}(\mathbf{G}) = n_\nu$.

Control All Object Wrenches

This property is dual to the previous one. Analysis of (38.21) yields the same conditions:

$$\mathrm{rank}(\mathbf{GJ}) = \mathrm{rank}(\mathbf{G}) = n_\nu \ .$$

Control All Internal Forces

Equation (38.20) shows that wrench intensities with no effect on object motion are those in $\mathcal{N}(\mathbf{G})$. In general, not all the internal forces may be actively controlled by joint actions. In [38.16, 21] it has been shown that all internal forces in $\mathcal{N}(\mathbf{G})$ are controllable if and only if $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$.

See Example 1, Part 5 and Example 2, Part 3.

Design Considerations of the Salisbury Hand

The Salisbury Hand in Fig. 38.4 was designed to have the smallest number of joints that would meet all the task requirements in Table 38.6. Assuming HF contacts, three non-collinear contacts is the minimum number such that $\mathrm{rank}(\mathbf{G}) = n_\nu = 6$. In this case, $\mathbf{G}$ has six rows and nine columns and the dimension of $\mathcal{N}(\mathbf{G})$ is three [38.2, 8]. The ability to control all internal forces and apply an arbitrary wrench to the object requires that $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$, so the minimum dimension of the column space of $\mathbf{J}$ is nine. To achieve this, the hand must have at least nine joints, which Salisbury implemented as three fingers, each with three revolute joints.

The intended way to execute a dexterous manipulation task with the Salisbury Hand is to grasp the object at three non-collinear points with the fingertips, forming a grasp triangle. To secure the grasp, the internal forces are controlled so that the contact points are maintained without sliding. Dexterous manipulation can be thought of as moving the fingertips to control the positions of the vertices of the grasp triangle.

38.3 Compliant Grasps

In this section we extend the rigid-body model to include compliance. This is needed to design controllers that can implement desired compliance behaviors of a grasped object when it contacts the environment, which can increase the robustness of static grasps and dexterous manipulation tasks. It also facilitates the analysis of hands designed with flexible mechanical elements [38.22, 23] and grasp control strategies that exploit grasp *synergies* [38.24, 25]. Thanks to compliance, hands can be designed to maintain a secure grasp with fewer joints, which provides greater mechanical robustness and reduces the complexity of planning grasps.

On the other hand, reducing the number of DOFs in robotic hands demands a compliant design of the whole structure to adapt the shape of the hand to different objects and to improve robustness with respect to uncertainties [38.26]. Compliance can be passive or active. Passive compliance is due to the structural deformation of robot components, including joints, while active compliance refers to virtual elasticity of actuators, e.g., due to the proportional action of a PD joint controller, that can be actively set by changing the control parameters [38.26–29].

In the following, we extend the grasp analysis relaxing the rigid-body contact constraints to take into account both the compliance and the low number of DOFs in robotic hands.

If the hand structure is not perfectly stiff, as shown in Fig. 38.5, the actual vector of joint variables $\boldsymbol{q}$ can be different from the reference one $\boldsymbol{q}_\mathrm{r}$, given to the joint actuator controllers, and their difference is related to the joint effort $\boldsymbol{\tau}$ through a compliance matrix $\mathbf{C}_q \in \mathbb{R}^{n_q \times n_q}$ by the constitutive equation

$$\boldsymbol{q}_\mathrm{r} - \boldsymbol{q} = \mathbf{C}_q \boldsymbol{\tau} \ . \tag{38.30}$$

Note that if the hand structure is perfectly rigid, $\mathbf{C}_q = 0$ and the hand stiffness $\mathbf{K}_q = \mathbf{C}_q^{-1}$ is not defined.

It will be clear in a moment that to deal with a low number of DOFs of the hand, a compliant model of the contact must be considered. According to Definition 38.4, in case of a low number of DOFs, it is very likely that the grasp will be defective, i. e., with a non-trivial $\mathcal{N}(\mathbf{J}^\mathrm{T})$. It is worth noting that this typically happens also in power grasps [38.30] where the hand envelops the object establishing contacts even with inner limbs. In this case the hand Jacobian is a tall matrix with a non-trivial null space of its transpose.

If the system is very defective, it is very likely that the grasp will be hyperstatic according to Definition 38.5 and consequently the rigid-body model in (38.21) is under-determined and does not admit a unique solution for the contact force vector $\boldsymbol{\lambda}$ as discussed in Sect. 38.2.2. Note that computing the force distribution $\boldsymbol{\lambda}$ is crucial in grasp analysis since it allows one to evaluate if the contact constraints are fulfilled, and consequently, if the grasp will be maintained.

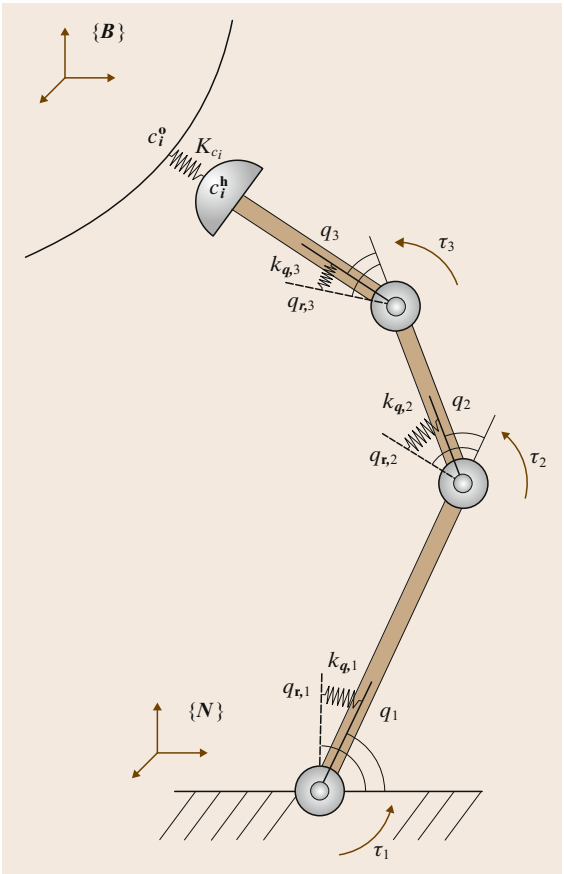


Fig. 38.5 Compliant joints and compliant contacts, main definitions

The force distribution problem in hyperstatic grasps is an under-determined problem of statics under the assumption of rigid contacts of (38.17). To solve the problem, we need to enrich the model with more information on the contact forces. A possibile solution is to substitute the rigid-body kinematic constraint (38.19) with a compliant model of the contact interaction [38.21], obtained introducing a set of springs between the contact points on the hand $\boldsymbol{c}^\mathrm{h}$ and on the object $\boldsymbol{c}^\mathrm{o}$ as shown in Fig. 38.5

$$\mathbf{C}_\mathrm{c} \boldsymbol{\lambda} = \boldsymbol{c}^\mathrm{h} - \boldsymbol{c}^\mathrm{o} \ , \tag{38.31}$$

where $\mathbf{C}_c \in \mathbb{R}^{n_\lambda \times n_\lambda}$ is the contact compliance matrix which is symmetric and positive definite. The contact stiffness matrix is defined as the reciprocal of the contact compliance matrix $\mathbf{K}_c = \mathbf{C}_c^{-1}$.

The analysis is presented in a quasi-static framework: from an equilibrium reference configuration $\boldsymbol{q}_{r,0}$, $\boldsymbol{q}_0$, $\boldsymbol{\tau}_0$, $\boldsymbol{\lambda}_0$, $\boldsymbol{u}_0$ and $\boldsymbol{g}_0$, a small input perturbation is applied to the joint references $\boldsymbol{q}_{r,0} + \Delta\boldsymbol{q}_r$ and to the external load $\boldsymbol{g}_0 + \Delta\boldsymbol{g}$ which moves the grasp system to a new equilibrium configuration whose linear approximation is represented by $\boldsymbol{q}_0 + \Delta\boldsymbol{q}$, $\boldsymbol{\tau}_0 + \Delta\boldsymbol{\tau}$, $\boldsymbol{\lambda}_0 + \Delta\boldsymbol{\lambda}$, $\boldsymbol{u}_0 + \Delta\boldsymbol{u}$.

In the following, for the sake of simplicity, it is convenient to refer $\mathbf{G}$ and $\mathbf{J}$ matrices to the object reference frame $\{\boldsymbol{B}\}$ and not to the inertial frame $\{\boldsymbol{N}\}$ as previously described. Note that this only applies to this section on compliance analysis, while the rest of this chapter still refers to matrices $\mathbf{G}$ and $\mathbf{J}$ as described in Sect. 38.1.1.

Let $\mathbf{R}_b \in \mathbb{R}^{3\times3}$ represent the orientation of the object frame $\{\boldsymbol{B}\}$ with respect to the inertial frame. Then the object twist referred to $\{\boldsymbol{B}\}$ is given as

$$\boldsymbol{v}_{i,\mathrm{obj}} = \overline{\mathbf{R}}_b^{\mathrm{T}} \begin{pmatrix} \boldsymbol{v}_{i,\mathrm{obj}}^{\mathrm{N}} \\ \boldsymbol{\omega}_{\mathrm{obj}}^{\mathrm{N}} \end{pmatrix}, \tag{38.32}$$

where

$$\overline{\mathbf{R}}_b = \mathrm{Blockdiag}(\mathbf{R}_b, \mathbf{R}_b) = \begin{pmatrix} \mathbf{R}_b & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_b \end{pmatrix} \in \mathbb{R}^{6\times6}.$$

Then, substituting $\mathbf{P}_i^{\mathrm{T}}\boldsymbol{v}$ from (38.2) into (38.32) yields the partial grasp matrix $\tilde{\mathbf{G}}_i^{\mathrm{T}} \in \mathbb{R}^{6\times6}$, that maps the object twist from $\{\boldsymbol{N}\}$ to contact frame $\{\boldsymbol{C}\}_i$ referred to $\{\boldsymbol{B}\}$.

Similarly, the hand twists can be expressed to the object frame $\{\boldsymbol{B}\}$

$$\boldsymbol{v}_{i,\mathrm{hnd}} = \overline{\mathbf{R}}_b^{\mathrm{T}} \begin{pmatrix} \boldsymbol{v}_{i,\mathrm{hnd}}^{\mathrm{N}} \\ \boldsymbol{\omega}_{i,\mathrm{hnd}}^{\mathrm{N}} \end{pmatrix} \tag{38.33}$$

and combining (38.33) and (38.7) yields the partial hand Jacobian $\tilde{\mathbf{J}}_i \in \mathbb{R}^{6\times n_q}$, which relates the joint velocities to the contact twists on the hand expressed with respect to $\{\boldsymbol{B}\}$.

Table 38.7 gives the definitions of the selection matrices for the three contact models when object and hand twists are referred to frame $\{\boldsymbol{B}\}$. In the planar case, as far as contact constraint is concerned, equation (38.15) holds with $\mathbf{H}_{i\mathrm{F}}$ and $\mathbf{H}_{i\mathrm{M}}$ defined in Table 38.8.

The contact force variation $\Delta\boldsymbol{\lambda}$ from the equilibrium configuration is then related to the relative displacement between the object and the fingers at the contact points as

$$\mathbf{C}_c\Delta\boldsymbol{\lambda} = (\mathbf{J}\Delta\boldsymbol{q} - \mathbf{G}^{\mathrm{T}}\Delta\boldsymbol{u}). \tag{38.34}$$

Table 38.7 Selection matrices for three contact models, when the object twists are expressed with respect to $\{\boldsymbol{B}\}$

Model	$n_{\lambda i}$	$\mathbf{H}_{i\mathrm{F}}$	$\mathbf{H}_{i\mathrm{M}}$
PwoF	1	$\hat{\boldsymbol{n}}_i^{b\mathrm{T}}$	Vacuous
HF	3	$\left[\hat{\boldsymbol{n}}_i^b, \hat{\boldsymbol{t}}_i^b, \hat{\boldsymbol{o}}_i^b\right]^{\mathrm{T}}$	Vacuous
SF	4	$\left[\hat{\boldsymbol{n}}_i^b, \hat{\boldsymbol{t}}_i^b, \hat{\boldsymbol{o}}_i^b\right]^{\mathrm{T}}$	$\hat{\boldsymbol{n}}_i^{b\mathrm{T}}$

$\hat{\boldsymbol{n}}_i^b$ is the normal unit vector at the contact i expressed with respect to $\{\boldsymbol{B}\}$ reference system.

Table 38.8 Selection matrices for the planar simplified case, when the object twists are expressed with respect to $\{\boldsymbol{B}\}$

Model	$n_{\lambda i}$	$\mathbf{H}_{i\mathrm{F}}$	$\mathbf{H}_{i\mathrm{M}}$
PwoF	1	$\hat{\boldsymbol{n}}_i^{b\mathrm{T}}$	Vacuous
HF/SF	2	$\left[\hat{\boldsymbol{n}}_i^b, \hat{\boldsymbol{t}}_i^b\right]^{\mathrm{T}}$	Vacuous

By assuming that the perturbed configuration is sufficiently near to the reference one, the following linearized relationships can be found for $\boldsymbol{g}$, and $\boldsymbol{\tau}$

$$\Delta\boldsymbol{g} = -\mathbf{G}\Delta\boldsymbol{\lambda} \tag{38.35}$$

$$\Delta\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\Delta\boldsymbol{\lambda} + \mathbf{K}_{J,q}\Delta\boldsymbol{q} + \mathbf{K}_{J,u}\Delta\boldsymbol{u} \tag{38.36}$$

where

$$\mathbf{K}_{J,q} = \frac{\partial\mathbf{J}\boldsymbol{\lambda}_0}{\partial\boldsymbol{q}} \quad \text{and} \quad \mathbf{K}_{J,u} = \frac{\partial\mathbf{J}\boldsymbol{\lambda}_0}{\partial\boldsymbol{u}}$$

are the variation of the hand Jacobian with respect to $\boldsymbol{q}$ and $\boldsymbol{u}$ variations, respectively.

Remark 38.3

In (38.35) and (38.36) both the grasp matrix and hand Jacobian are expressed with respect to the object reference frame, and by neglecting rolling between the fingers and the object at the contact points, $\mathbf{G}$ becomes constant, while $\mathbf{J}(\boldsymbol{q},\boldsymbol{u})$, in general, depends on both hand and object configurations. Matrix $\mathbf{K}_{J,q}$ represent the variability of $\mathbf{J}(\boldsymbol{q},\boldsymbol{u})$ matrix with respect to hand and object configurations.

Matrices $\mathbf{K}_{J,q} \in \mathbb{R}^{n_q\times n_q}$ and $\mathbf{K}_{J,u} \in \mathbb{R}^{n_q\times n_v}$ are usually referred to as *geometric stiffness matrices* [38.28]. Furthermore, it can be verified that the matrix $\mathbf{K}_{J,q}$ is symmetric [38.27].

By substituting (38.36) in (38.30) we get

$$\mathbf{J}_{\mathrm{R}}\mathbf{C}_q\mathbf{J}^{\mathrm{T}}\Delta\boldsymbol{\lambda} = \mathbf{J}_{\mathrm{R}}\Delta\boldsymbol{q}_r - \mathbf{J}\Delta\boldsymbol{q} - \mathbf{J}_{\mathrm{R}}\mathbf{C}_q\mathbf{K}_{J,u}\Delta\boldsymbol{u}, \tag{38.37}$$

where

$$\mathbf{J}_{\mathrm{R}} = \mathbf{J}\left(\mathbf{I} + \mathbf{C}_q\mathbf{K}_{J,q}\right)^{-1}.$$

Summing (38.34) and (38.37) the following expression can be found for the contact forces displacement $\Delta\boldsymbol{\lambda}$

$$\Delta\boldsymbol{\lambda} = \mathbf{K}_{c,e}\left(\mathbf{J}_{\mathrm{R}}\Delta\boldsymbol{q}_{\mathrm{r}} - \mathbf{G}_{\mathrm{R}}^{\mathrm{T}}\Delta\boldsymbol{u}\right) \tag{38.38}$$

where

$$\mathbf{G}_{\mathrm{R}}^{\mathrm{T}} = \mathbf{G}^{\mathrm{T}} + \mathbf{J}_{\mathrm{R}}\mathbf{C}_q\mathbf{K}_{J,u}$$

and

$$\mathbf{K}_{c,e} = \left(\mathbf{C}_{\mathrm{c}} + \mathbf{J}_{\mathrm{R}}\mathbf{C}_q\mathbf{J}^{\mathrm{T}}\right)^{-1} . \tag{38.39}$$

Matrix $\mathbf{K}_{c,e}$ represents the equivalent contact stiffness, that takes into account both the joint and the contact compliance. If the geometrical terms are neglected, i. e., $\mathbf{K}_{J,q} = 0$ and $\mathbf{K}_{J,u} = 0$, the classical expression $\mathbf{K}_{c,e} = \left(\mathbf{C}_{\mathrm{c}} + \mathbf{J}\mathbf{C}_q\mathbf{J}^{\mathrm{T}}\right)$ for the equivalent contact stiffness matrix can be found [38.28].

By substituting (38.38) in (38.35), the object displacement can be evaluated as a function of the small input perturbations $\Delta\boldsymbol{q}_{\mathrm{r}}$ and $\Delta\boldsymbol{g}$

$$\Delta\boldsymbol{u} = \left(\mathbf{G}\mathbf{K}_{c,e}\mathbf{G}_{\mathrm{R}}^{\mathrm{T}}\right)^{-1}\left(\mathbf{G}\mathbf{K}_{c,e}\mathbf{J}_{\mathrm{R}}\Delta\boldsymbol{q}_{\mathrm{r}} + \Delta\boldsymbol{g}\right) . \tag{38.40}$$

When $\Delta\boldsymbol{q}_{\mathrm{r}} = 0$, (38.40) can be rewritten as

$$\Delta\boldsymbol{g} = \mathbf{K}\Delta\boldsymbol{u} ,$$
$$\text{with } \mathbf{K} = \mathbf{G}\mathbf{K}_{c,e}\mathbf{G}_{\mathrm{R}}^{\mathrm{T}} ,$$

where the term that multiplies the external wrench variation $\Delta\boldsymbol{g}$ represents the reciprocal of the *grasp stiffness matrix* $\mathbf{K}$. The grasp stiffness evaluates ability of the robotic grasp to resist to external load variations applied to the object.

Regarding the force distribution $\Delta\boldsymbol{\lambda}$, by substituting (38.40) into (38.38), the variation of the contact forces is evaluated as

$$\Delta\boldsymbol{\lambda} = \mathbf{G}_g^{+}\Delta\boldsymbol{g} + \mathbf{P}\Delta\boldsymbol{q}_{\mathrm{r}} , \tag{38.41}$$

where

$$\mathbf{G}_g^{+} = \mathbf{K}_{c,e}\mathbf{G}_{\mathrm{R}}^{\mathrm{T}}\left(\mathbf{G}\mathbf{K}_{c,e}\mathbf{G}_{\mathrm{R}}^{\mathrm{T}}\right)^{-1} ,$$
$$\mathbf{P} = \left(\mathbf{I} - \mathbf{G}_g^{+}\mathbf{G}\right)\mathbf{K}_{c,e}\mathbf{J}_{\mathrm{R}} .$$

Matrix $\mathbf{G}_g^{+}$ is a right pseudo-inverse of grasp matrix $\mathbf{G}$ that takes into account both the geometrical effects and the hand and contact stiffness. Matrix $\mathbf{P}$ maps the reference joint variables $\Delta\boldsymbol{q}_{\mathrm{r}}$ to the contact force variation $\Delta\boldsymbol{\lambda}$.

It is worth noting that $\left(\mathbf{I} - \mathbf{G}_g^{+}\mathbf{G}\right)$ is a projector onto the null space of $\mathbf{G}$ and then each contact force variation $\Delta\boldsymbol{\lambda}_h = \mathbf{P}\Delta\boldsymbol{q}_{\mathrm{r}}$, produced by modifying joint reference values, satisfies the equation

$$\mathbf{G}\Delta\boldsymbol{\lambda}_h = 0 ,$$

and then belongs to the internal force subspace.

Summarizing, compliant grasps have been analyzed with a linearized quasi-static model and the main relationships mapping joint references, the controlled inputs, and external disturbances, the wrenches, onto object motions and contact forces have been evaluated in (38.40) and (38.41), respectively.

38.4 Restraint Analysis

The most fundamental requirements in grasping and dexterous manipulation are the abilities to hold an object and control its position and orientation relative to the palm of the hand. The two most useful characterizations of grasp restraint are *force closure* and *form closure*. These names were in use as early as 1876 in the field of machine design to distinguish between joints that required an external force to maintain contact, and those that did not [38.31]. For example, some water wheels had a cylindrical axle that was laid in a horizontal semi-cylindrical groove split on either side of the wheel. During operation, the weight of the wheel acted to *close* the groove-axle contacts, hence the term *force closure*. By contrast, if the grooves were replaced by cylindrical holes just long enough to accept the axle, then the contacts would be closed by the geometry (even if the direction of the gravitational force was reversed), hence the term *form closure*.

When applied to grasping, form and force closure have the following interpretations. Assume that a hand grasping an object has its joint angles locked and its palm fixed in space. Then the grasp has *form closure*, or the object is *form-closed*, if it is impossible to move the object, even infinitesimally. Under the same conditions, the grasp has *force closure*, or the object is *force-closed*, if for any non-contact wrench experienced by the object, contact wrench intensities exist that satisfy (38.20) and are consistent with the constraints imposed by the friction models applicable at the contact points. Notice that all form closure grasps are also force closure grasps. When under form closure, the object cannot move even infinitesimally relative to the hand, regardless of the non-contact wrench. Therefore, the hand maintains the object in equilibrium for any external wrench, which is the force closure requirement.

Roughly speaking, form closure occurs when the palm and fingers wrap around the object forming a cage with no *wiggle room* such as the grasp shown in Fig. 38.6. This kind of grasp is also called a *power grasp* [38.32] or *enveloping grasp* [38.33]. However, force closure is possible with fewer contacts, as shown in Fig. 38.7, but in this case, force closure requires the ability to control internal forces. It is also possible for a grasp to have partial form closure, indicating that only a subset of the possible degrees of freedom are restrained [38.34]. An example of such a grasp is shown in Fig. 38.8. In this grasp, fingertip placement between the ridges around the periphery of the bottle cap provides form closure against relative rotation about the axis of the helix of the threads and also against translation perpendicular to that axis, but the other three degrees of freedom are restrained through force closure. Strictly speaking, given a grasp of a real object by a human hand it is impossible to prevent relative motion of the object with respect to the palm due to the compliance of the hand and object. Preventing all motion is possible only if the contacting bodies are rigid, as is assumed in most mathematical models employed in grasp analysis.

Fig. 38.6 The palm and fingers combine to create a very secure form closure grasp of a router

Fig. 38.7 Hand and wrist of the tendon driven DLR Hand Arm System grasping a tool [38.7]. This grasp has a force closure grasp appropriate for dexterous manipulation. (DLR Hand Arm System, Photo courtesy of DLR 2011)

38.4.1 Form Closure

To make the notion of form closure precise, introduce a signed gap function denoted by $\psi_i(\boldsymbol{u}, \boldsymbol{q})$ at each of the n_c contact points between the object and the hand. The gap function is zero at each contact, becomes positive if contact breaks, and negative if penetration occurs. The gap function can be thought of as distance between the contact points. In general, this function is dependent on the shapes of the contacting bodies. Let $\bar{\boldsymbol{u}}$ and $\bar{\boldsymbol{q}}$ represent the configurations of the object and hand for a given grasp; then

$$\psi_i(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}}) = \boldsymbol{0} \quad \forall\, i = 1, \ldots, n_c\,. \tag{38.42}$$

The form closure condition can now be stated in terms of a differential change $\mathrm{d}\boldsymbol{u}$ of $\bar{\boldsymbol{u}}$:

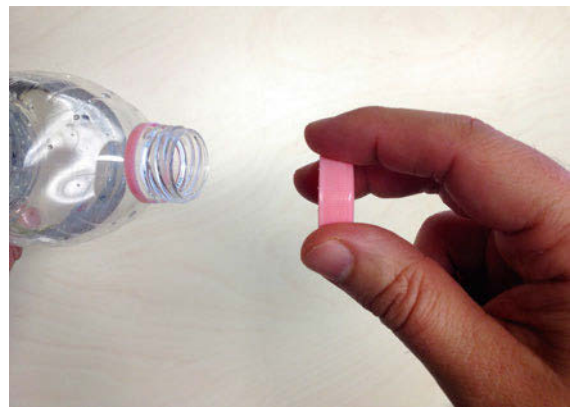


Fig. 38.8 In the grasp depicted, contact with the ridges on the bottle cap create partial form closure in the direction of cap rotation (when screwing it in) and also in the directions of translation perpendicular to the axis of rotation. To achieve complete control over the cap, the grasp achieves force closure over the other three degrees of freedom

Definition 38.6

A grasp $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})$ has *form closure* if and only if the following implication holds

$$\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}}) \geq \mathbf{0} \Rightarrow \mathrm{d}\boldsymbol{u} = \mathbf{0}\,, \tag{38.43}$$

where $\boldsymbol{\psi}$ is the n_c-dimensional vector of gap functions with i-th component equal to $\psi_i(\boldsymbol{u}, \boldsymbol{q})$. By definition, inequalities between vectors imply that the inequality is applied between corresponding components of the vectors.

Expanding the gap function vector in a Taylor series about $\bar{\boldsymbol{u}}$, yields infinitesimal form closure tests of various orders. Let ${}^{\beta}\boldsymbol{\psi}(\boldsymbol{u}, \boldsymbol{q}), \beta = 1, 2, 3, \ldots,$ denote the Taylor series approximation truncated after the terms of order β in $\mathrm{d}\boldsymbol{u}$. From (38.42), it follows that the first-order approximation is

$${}^{1}\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}}) = \frac{\partial \boldsymbol{\psi}(\boldsymbol{u}, \boldsymbol{q})}{\partial \boldsymbol{u}}|_{(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})} \mathrm{d}\boldsymbol{u}\,,$$

where $\frac{\partial \boldsymbol{\psi}(\boldsymbol{u}, \boldsymbol{q})}{\partial \boldsymbol{u}}|_{(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})}$ denotes the partial derivative of $\boldsymbol{\psi}$ with respect to $\boldsymbol{u}$ evaluated at $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})$. Replacing $\boldsymbol{\psi}$ with its approximation of order β in (38.43) implies three relevant cases of order β:

1. If there exists $\mathrm{d}\boldsymbol{u}$ such that ${}^{\beta}\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}})$ has at least one strictly positive component, then the grasp does not have form closure of order β.
2. If for every non-zero $\mathrm{d}\boldsymbol{u}$, ${}^{\beta}\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}})$ has at least one strictly negative component, then the grasp has form closure of order β.
3. If neither case i) nor case ii) applies for all ${}^{\eta}\boldsymbol{\psi}(\bar{\boldsymbol{u}} + \mathrm{d}\boldsymbol{u}, \bar{\boldsymbol{q}})\ \forall\ \eta \leq \beta$, then higher-order analysis is required to determine the existence of form closure.

Figure 38.9 illustrates form closure concepts using several planar grasps of gray objects by fingers shown as black disks. The concepts are identical for grasps of three-dimensional objects, but are more clearly illustrated in the plane. The grasp on the left has first-order form closure. Note that first-order form closure only involves the first derivatives of the distance functions. This implies that the only relevant geometry in first-order form closure are the locations of the contacts and the directions of the contact normals. The grasp in the center has form closure of higher order, with the specific order depending on the degrees of the curves defining the surfaces of the object and fingers in the neighborhoods of the contacts [38.35]. Second-order form closure analysis depends on the curvatures of the two contacting bodies in addition to the geometric information used to analyze first-order form closure. The grasp on the right does not have form closure of any order, because the object can translate horizontally and rotate about its center.

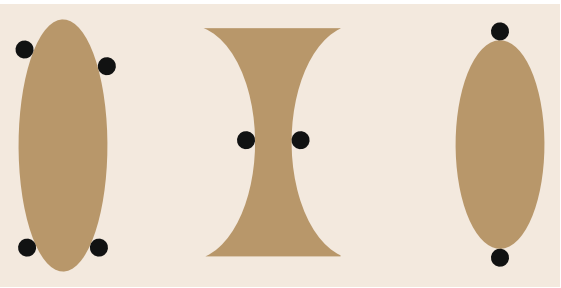


Fig. 38.9 Three planar grasps; two with form closure of different orders and one without form closure

First-Order Form Closure

First-order form closure exists if and only if the following implication holds

$$\frac{\partial \boldsymbol{\psi}(\boldsymbol{u}, \boldsymbol{q})}{\partial \boldsymbol{u}}|_{(\bar{\boldsymbol{u}}, \bar{\boldsymbol{q}})} \mathrm{d}\boldsymbol{u} \geq \mathbf{0} \Rightarrow \mathrm{d}\boldsymbol{u} = \mathbf{0}\,.$$

The first-order form closure condition can be written in terms of the object twist $\boldsymbol{\nu}$

$$\mathbf{G}_{\mathrm{n}}^{\mathrm{T}} \boldsymbol{\nu} \geq \mathbf{0} \Rightarrow \boldsymbol{\nu} = \mathbf{0}\,, \tag{38.44}$$

where $\mathbf{G}_{\mathrm{n}}^{\mathrm{T}} = \partial \boldsymbol{\psi} / \partial \boldsymbol{u} \mathbf{V} \in \mathbb{R}^{n_c \times 6}$. Recall that $\mathbf{V}$ is the kinematic map defined in (38.1). Also notice that $\mathbf{G}_{\mathrm{n}}$ is the grasp matrix, when all contact points are frictionless.

Because the gap functions only quantify distances, the product $\mathbf{G}_{\mathrm{n}}^{\mathrm{T}} \boldsymbol{\nu}$ is the vector of normal components of the instantaneous velocities of the object at the contact points (which must be non-negative to prevent interpenetration). This in turn implies that the grasp matrix is the one that would result from the assumption that all contacts are of the type PwoF.

An equivalent condition in terms of the contact wrench intensity vector $\boldsymbol{\lambda}_{\mathrm{n}} \in \mathbb{R}^{n_c}$ can be stated as follows. A grasp has first-order form closure if and only if

$$\left.\begin{array}{l} \mathbf{G}_{\mathrm{n}} \boldsymbol{\lambda}_{\mathrm{n}} = -\boldsymbol{g} \\ \boldsymbol{\lambda}_{\mathrm{n}} \geq \mathbf{0} \end{array}\right\} \forall\ \boldsymbol{g} \in \mathbb{R}^6\,. \tag{38.45}$$

The physical interpretation of this condition is that equilibrium can be maintained under the assumption that the contacts are frictionless. Note that the components of $\boldsymbol{\lambda}_{\mathrm{n}}$ are the magnitudes of the normal components of the contact forces. The subscript $(.)_{\mathrm{n}}$ is used to emphasize that $\boldsymbol{\lambda}_{\mathrm{n}}$ contains no other force or moment components.

Since $\boldsymbol{g}$ must be in the range of $\mathbf{G}_n$ for equilibrium to be satisfied, and since $\boldsymbol{g}$ is an arbitrary element of $\mathbb{R}^6$, then in order for condition (38.45) to be satisfied, the rank of $\mathbf{G}_n$ must be six. Assuming $\text{rank}(\mathbf{G}_n) = 6$, another equivalent mathematical statement of first-order form closure is: there exists $\boldsymbol{\lambda}_n$ such that the following two conditions hold [38.36]

$$\mathbf{G}_n\boldsymbol{\lambda}_n = \mathbf{0}\,,$$
$$\boldsymbol{\lambda}_n > \mathbf{0}\,. \tag{38.46}$$

This means that there exists a set of strictly compressive normal contact forces in the null space of $\mathbf{G}_n$. In other words, one can squeeze the object as tightly as desired while maintaining equilibrium. A second interpretation of this condition is that the non-negative span of the columns of $\mathbf{G}_n$ must equal $\mathbb{R}^6$. Equivalently, the convex hull of the columns of $\mathbf{G}_n$ must strictly contain the origin of $\mathbb{R}^6$. As will be seen, the span and hull interpretations will provide a conceptual link called *frictional form closure* that lies between form closure and force closure.

The duality of conditions (38.44) and (38.45) can be seen clearly by examining the set of wrenches that can be applied by frictionless contacts and the corresponding set of possible object twists. For this discussion, it is useful to give definitions of cones and their duals.

Definition 38.7

A *cone* C is a set of vectors $\boldsymbol{\varsigma}$ such that for every $\boldsymbol{\varsigma}$ in C, every non-negative scalar multiple of $\boldsymbol{\varsigma}$ is also in C.

Equivalently, a cone is a set of vectors closed under addition and non-negative scalar multiplication.

Definition 38.8

Given a cone C with elements $\boldsymbol{\varsigma}$, the *dual cone* C^* with elements $\boldsymbol{\varsigma}^*$, is the set of vectors such that the dot product of $\boldsymbol{\varsigma}^*$ with each vector in C is non-negative. Mathematically

$$C^* = \{\boldsymbol{\varsigma}^* | \boldsymbol{\varsigma}^{\mathrm{T}}\boldsymbol{\varsigma}^* \geq 0, \forall \boldsymbol{\varsigma} \in C\}\,. \tag{38.47}$$

See Example 4.

First-Order Form Closure Requirements

Several useful necessary conditions for form closure are known. In 1897 *Somov* proved that at least seven contacts are necessary to form close a rigid object with six degrees of freedom [38.37, 38]. *Lakshminarayana* generalized this to prove that $n_\nu + 1$ contacts are necessary to form close an object with n_ν degrees of freedom [38.34] (based on *Goldman* and *Tucker* 1956 [38.39]) (Table 38.9). This led to the definition of partial form closure that was mentioned above in the discussion of the hand grasping the bottle cap. *Markenscoff* and *Papadimitriou* determined a tight upper bound, showing that for all objects whose surfaces are not surfaces of revolution, at most $n_\nu + 1$ contacts are necessary [38.40]. Form closure is impossible to achieve for surfaces of revolution.

To emphasize the fact that $n_\nu + 1$ contacts are necessary and *not* sufficient, consider grasping a cube with seven or more points of contact. If all contacts are on one face, then clearly, the grasp does not have form closure.

First-Order Form Closure Tests

Because form closure grasps are very secure, it is desirable to design or synthesize such grasps. To do this, one needs a way to test candidate grasps for form closure, and rank them, so that the *best* grasp can be chosen. One reasonable measure of form closure can be derived from the geometric interpretation of the condition (38.46). The null space constraint and the positivity of $\boldsymbol{\lambda}_n$ represent the addition of the columns of $\mathbf{G}_n$ scaled by the components of $\boldsymbol{\lambda}_n$. Any choice of $\boldsymbol{\lambda}_n$ closing this loop is in $\mathcal{N}(\mathbf{G}_n)$. For a given loop, if the magnitude of the smallest component of $\boldsymbol{\lambda}_n$ is positive, then the grasp has form closure, otherwise it does not. Let us denote this smallest component by d. Since such a loop, and hence d, can be scaled arbitrarily, $\boldsymbol{\lambda}_n$ must be bounded.

After verifying that $\mathbf{G}_n$ has full row rank, a quantitative form closure test based on the above observations can be formulated as a linear program (LP) in the unknowns d and $\boldsymbol{\lambda}_n$ as follows

LP1: maximize: d (38.48)

subject to: $\mathbf{G}_n\boldsymbol{\lambda}_n = \mathbf{0}$ (38.49)

$\mathbf{I}\boldsymbol{\lambda}_n - \mathbf{1}d \geq \mathbf{0}$ (38.50)

$d \geq 0$ (38.51)

$\mathbf{1}^{\mathrm{T}}\boldsymbol{\lambda}_n \leq n_c\,,$ (38.52)

where $\mathbf{I} \in \mathbb{R}^{n_c \times n_c}$ is the identity matrix and $\mathbf{1} \in \mathbb{R}^{n_c}$ is a vector with all components equal to 1. The last inequality is designed to prevent this LP from becoming unbounded. A typical LP solution algorithm determines

Table 38.9 Minimum number of contacts n_c required to form close an object with n_ν degrees of freedom

n_ν	n_c
3 (planar grasp)	4
6 (spatial grasp)	7
n_ν (general)	$n_\nu + 1$

infeasibility or unboundedness of the constraints in Phase I of the algorithm, and considers the result before attempting to calculate an optimal value [38.41]. If LP1 is infeasible, or if the optimal value d^* is zero, then the grasp is not form closed.

The quantitative form closure test (38.48)–(38.52) has $n_c + 8$ constraints and $n_c + 1$ unknowns. For a typical grasp with $n_c < 10$, this is a small linear program that can be solved very quickly using the simplex method. However, one should note that d^* is dependent on the choice of units used when forming $\mathbf{G}_n$. It would be advisable to non-dimensionalize the components of the wrenches to avoid dependence of the optimal d on one's choice of units. This could be done by dividing the first three rows of $\mathbf{G}_n$ by a characteristic force and the last three rows by a characteristic moment. If one desires a binary test, LP1 can be converted into one by dropping the last constraint (38.52) and applying only Phase I of the simplex algorithm.

In summary, quantitative form closure testing is a two-step process:

Form Closure Test.

1. Compute rank($\mathbf{G}_n$).
   a) If rank($\mathbf{G}_n$) $\neq n_v$, then form closure does not exist. Stop.
   b) If rank($\mathbf{G}_n$) $= n_v$, continue.
2. Solve LP1.
   a) If $d^* = 0$, then form closure does not exist.
   b) If $d^* > 0$, then form closure exists and $d^* n_c$ is a crude measure of how far the grasp is from losing form closure.

See Example 5, Part 1.

Variations of the Test. If the rank test fails, then the grasp could have partial form closure over as many as rank($\mathbf{G}_n$) degrees of freedom. If one desires to test this, then LP1 must be solved using a new $\mathbf{G}_n$ formed by retaining only the rows corresponding to the degrees of freedom for which partial form closure is to be tested. If $d^* > 0$, then partial form closure exists.

A second variation arises when one knows in advance that the object is already partially constrained. For example, in the case of a steering wheel, the driver knows, that relative to her, the steering wheel has only one degree of freedom. A form closure grasp suitable for driving would be required only to restraint the wheel's rotation about the steering column. In general, assume that the object is constrained by a set of bilateral constraints, which can be written as

$$\mathbf{B}^{\mathrm{T}}\boldsymbol{v} = \mathbf{0}\,. \tag{38.53}$$

With these additional constraints, the form closure property can be expressed as follows

$$\left.\begin{aligned}\mathbf{G}_n^{\mathrm{T}}\boldsymbol{v} &\geq \mathbf{0}\\ \mathbf{B}^{\mathrm{T}}\boldsymbol{v} &= \mathbf{0}\end{aligned}\right\} \Rightarrow \boldsymbol{v} = \mathbf{0}\,, \tag{38.54}$$

which can be shown to be equivalent to

$$\overline{\mathbf{G}}_n^{\mathrm{T}}\boldsymbol{\rho} \geq \mathbf{0} \Rightarrow \boldsymbol{\rho} = \mathbf{0}\,, \tag{38.55}$$

where $\boldsymbol{\rho}$ is an arbitrary vector with length equal to the dimension of the null space of $\mathbf{B}^{\mathrm{T}}$ and $\overline{\mathbf{G}}_n = \mathbf{A}_B\mathbf{G}_n$, where $\mathbf{A}_B$ is an annihilator of the column space of $\mathbf{B}$. One possible way to construct this annihilator is $\mathbf{A}_B = \left(\mathbf{N}\left(\mathbf{B}^{\mathrm{T}}\right)\right)^{\mathrm{T}}$, where $\mathbf{N}\left(\mathbf{B}^{\mathrm{T}}\right)$ is a matrix whose columns form a basis for the null space of $\mathbf{B}^{\mathrm{T}}$. Also note that $N(\mathbf{B}^{\mathrm{T}})\boldsymbol{\rho}$ is a possible twist of the object consistent with the bilateral constraints (i. e., the twists that must be eliminated by the unilateral constraints).

The form closure condition for an object partially constrained by bilateral constraints can also be stated in terms of wrenches

$$\begin{aligned}\overline{\mathbf{G}}_n\boldsymbol{\lambda}_n &= \mathbf{0}\,,\\ \boldsymbol{\lambda}_n &> \mathbf{0}\,.\end{aligned} \tag{38.56}$$

Notice that twist conditions (38.44) and (38.55) and wrench conditions (38.46) and (38.56) are formally analogous, and therefore, the quantitative form closure test can be applied by substituting $\overline{\mathbf{G}}_n$ for $\mathbf{G}_n$ and $(n_v - \mathrm{rank}(\mathbf{B}))$ for n_v. For detailed information on the derivations of the constrained form closure conditions, please refer to [38.42].

See Example 5, Part 2.

Monotonicity. In grasp synthesis, sometimes it is desirable for the grasp metric to increase monotonically with the number of contact points, the intuition being, that adding contacts to a stable grasp will make it more stable. The solution d^* does not have this property. However `formClosure.m`, a Matlab function which is available at [38.43], returns $d^* n_c$ as the metric, because it nearly maintains monotonicity when adding random contacts to random grasps. This can be demonstrated by running `test_monotonicity.m`.

One can design metrics that are monotonic with norms on the set of wrenches that the hand can apply. Let the polytope $\mathcal{G} = \{\boldsymbol{g} | \boldsymbol{g} = \mathbf{G}_n\boldsymbol{\lambda}_n, \mathbf{0} \leq \boldsymbol{\lambda}_n \leq \mathbf{1}\}$ denote the wrench space, which is the set of wrenches that can be applied by the hand if the contact forces magnitudes are limited to one. A metric proposed by *Ferrari* and *Canny* [38.44] is the radius of the largest sphere in $\mathcal{G}$ with origin at the origin of the wrench space. The radius is greater than 0 if and only if the grasp has form

closure. In addition, adding a contact point will always produce a new $\mathcal{G}$ that is a superset of the original $\mathcal{G}$. Thus the radius of the sphere cannot reduce as a result. Hence the monotonicity property holds.

See Example 5, Part 3.

Planar Simplifications

In the planar case, *Nguyen* [38.45] developed a graphical qualitative test for form closure. Figure 38.10 shows a form closure grasp with four contacts. To test form closure one partitions the normals into two groups of two. Let C_1 be the non-negative span of two normals in one pair and C_2 be the non-negative span of the other pair. A grasp has form closure if and only if C_1 and C_2 or $-C_1$ and $-C_2$ *see each other* for any pairings. Two cones *see each other* if the open line segment defined by the vertices of the cones lies in the interior of both cones. In the presence of more than four contacts, if any set of four contacts satisfies this condition, then the grasp has form closure.

Notice that this graphical test can be difficult to execute for grasps with more than four contacts. Also, it does not extend to grasps of three-dimensional (3-D) objects and does not provide a closure measure.

38.4.2 Force Closure

A grasp has force closure, or is force-closed, if the grasp can be maintained in the face of any object wrench. Force closure is similar to form closure, but relaxed to allow friction forces to help balance the object wrench. A benefit of including friction in the analysis is the reduction in the number of contact points needed for closure. A three-dimensional object with six degrees of freedom requires seven contacts for form closure, but for force closure, only two contacts are needed if they are modeled as soft fingers, and only three (non-collinear) contacts are needed if they are modeled as hard fingers.

Force closure relies on the ability of the hand to squeeze arbitrarily tightly in order to compensate for large applied wrenches that can only be resisted by friction. Figure 38.15 shows a grasped polygon. Consider applying a wrench to the object that is a pure force acting upward along the $\boldsymbol{y}$-axis of the inertial frame. It seems intuitive that if there is enough friction, the hand will be able to squeeze the object with friction forces preventing the object's upward escape. Also, as the applied force increases in magnitude, the magnitude of the squeezing force will have to increase accordingly.

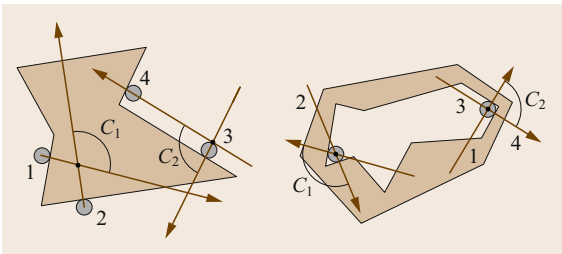


Fig. 38.10 Planar grasps with first-order form closure

Since force closure is dependent on the friction models, common models will be introduced before giving formal definitions of force closure.

Friction Models

Recall the components of force and moment transmitted through contact i under the various contact models given earlier in Table 38.4. At contact point i, the friction law imposes constraints on the components of the contact force and moment. Specifically, the frictional components of $\boldsymbol{\lambda}_i$ are constrained to lie inside a limit surface, denoted by $\mathcal{L}_i$, that scales linearly with the product $\mu_i f_{in}$, where μ_i is the coefficient of friction at contact i. In the case of Coulomb friction, the limit surface is a circle of radius $\mu_i f_{in}$. The Coulomb friction cone $\mathcal{F}_i$ is a subset of $\mathbb{R}^3$

$$\mathcal{F}_i = \{(f_{in}, f_{it}, f_{io}) | \sqrt{f_{it}^2 + f_{io}^2} \leq \mu_i f_{in}\} \,. \tag{38.57}$$

More generally, the friction laws of interest have limit surfaces defined in the space of friction components, $\mathbb{R}^{n_{\lambda i}-1}$ and friction cones $\mathcal{F}_i$ defined in the space of $\boldsymbol{\lambda}_i$, $\mathbb{R}^{n_{\lambda i}}$. They can be written as follows

$$\mathcal{F}_i = \{\boldsymbol{\lambda}_i \in \mathbb{R}^{n_{\lambda i}} \mid ||\boldsymbol{\lambda}_i||_w \leq f_{in}\} \,, \tag{38.58}$$

where $||\boldsymbol{\lambda}_i||_w$ denotes a weighted quadratic norm of the friction components at contact i. The limit surface is defined by $||\boldsymbol{\lambda}_i||_w = f_{in}$.

Table 38.10 defines useful weighted quadratic norms for the three contact models: PwoF, HF, and SF. The parameter μ_i is the friction coefficient for the tangential forces, υ_i is the torsional friction coefficient, and a is the characteristic length of the object that is used to ensure consistent units for the norm of the SF model.

Table 38.10 Norms for the three main contact models

Model	$\|\|\boldsymbol{\lambda}_i\|\|_w$
PwoF	0
HF	$\frac{1}{\mu_i}\sqrt{f_{it}^2 + f_{io}^2}$
SF	$\frac{1}{\mu_i}\sqrt{f_{it}^2 + f_{io}^2} + \frac{1}{a\,\upsilon_i}\lvert m_{in}\rvert$

Remark 38.4

There are several noteworthy points to be made about the friction cones. First, all of them implicitly or explicitly constrain the normal component of the contact force to be non-negative. The cone for SF contacts has a cylindrical limit surface with circular cross section in the (f_{it}, f_{io})-plane and rectangular cross section in the (f_{it}, m_{in})-plane. With this model, the amount of torsional friction that can be transmitted is independent of the lateral friction load. An improved model that couples the torsional friction limit with the tangential limit was studied by *Howe* and *Cutkosky* [38.46].

A Force Closure Definition

One common definition of force closure can be stated simply by modifying condition (38.45) to allow each contact force to lie in its friction cone rather than along the contact normal. Because this definition does not consider the hand's ability to control contact forces, this definition will be referred to as *frictional form closure*. A grasp will be said to have frictional form closure if and only if the following condition is satisfied

$$\left.\begin{array}{l}\mathbf{G}\boldsymbol{\lambda} = -\boldsymbol{g} \\ \boldsymbol{\lambda} \in \mathcal{F}\end{array}\right\} \forall\, \boldsymbol{g} \in \mathbb{R}^{n_v}\,,$$

where $\mathcal{F}$ is the composite friction cone defined as:

$$\begin{aligned}\mathcal{F} &= \mathcal{F}_1 \times \cdots \times \mathcal{F}_{n_c} \\ &= \{\boldsymbol{\lambda} \in \mathbb{R}^m | \boldsymbol{\lambda}_i \in \mathcal{F}_i;\ i = 1, \ldots, n_c\}\,,\end{aligned}$$

and each $\mathcal{F}_i$ is defined by (38.58) and one of the models listed in Table 38.10.

Letting $\mathrm{Int}(\mathcal{F})$ denote the interior of the composite friction cone, *Murray* et al. give the following equivalent definition [38.19]:

Definition 38.9

(Proposition 5.2, Murray et al.) A grasp has *frictional form closure* if and only if the following conditions are satisfied:

1. $\mathrm{rank}(\mathbf{G}) = n_v$
2. $\exists\, \boldsymbol{\lambda}$ such that $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$ and $\boldsymbol{\lambda} \in \mathrm{Int}(\mathcal{F})$.

These conditions define what Murray et al. call *force closure*. The force closure definition adopted here is stricter than frictional form closure; it additionally requires that the hand be able to control the internal object forces.

Definition 38.10

A grasp has *force closure* if and only if

$$\mathrm{rank}(\mathbf{G}) = n_v,\ \mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}\,,$$

and there exists $\boldsymbol{\lambda}$ such that $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$ and $\boldsymbol{\lambda} \in \mathrm{Int}(\mathcal{F})$.

The full row rank condition on the matrix $\mathbf{G}$ is the same condition required for form closure, although $\mathbf{G}$ is different from $\mathbf{G}_n$ used to determine form closure. If the rank test passes, then one must still find $\boldsymbol{\lambda}$ satisfying the remaining three conditions. Of these, the null space intersection test can be performed easily by linear programming techniques, but the friction cone constraint is quadratic, and thus forces one to use non-linear programming techniques. While exact non-linear tests have been developed [38.47], only approximate tests will be presented here.

Approximate Force Closure Tests

Any of the friction cones discussed can be approximated as the non-negative span of a finite number n_g of generators $\boldsymbol{s}_{ij}$ of the friction cone. Given this, one can represent the set of applicable contact wrenches at contact i as follows

$$\mathbf{G}_i\boldsymbol{\lambda}_i = \mathbf{S}_i\boldsymbol{\sigma}_i, \quad \boldsymbol{\sigma}_i \geq 0\,,$$

where $\mathbf{S}_i = [\boldsymbol{s}_{i1} \cdots \boldsymbol{s}_{in_g}]$ and $\boldsymbol{\sigma}_i$ is a vector of non-negative generator weights. If contact i is frictionless, then $n_g = 1$ and $\mathbf{S}_i = [\hat{\boldsymbol{n}}_i^{\mathrm{T}}((\boldsymbol{c}_i - \boldsymbol{p}) \times \hat{\boldsymbol{n}}_i)^{\mathrm{T}}]^{\mathrm{T}}$.

If contact i is of type HF, we represent the friction cone by the non-negative sum of uniformly spaced contact force generators (Fig. 38.11) whose non-negative span approximates the Coulomb cone with an inscribed regular polyhedral cone. This leads to the following def-

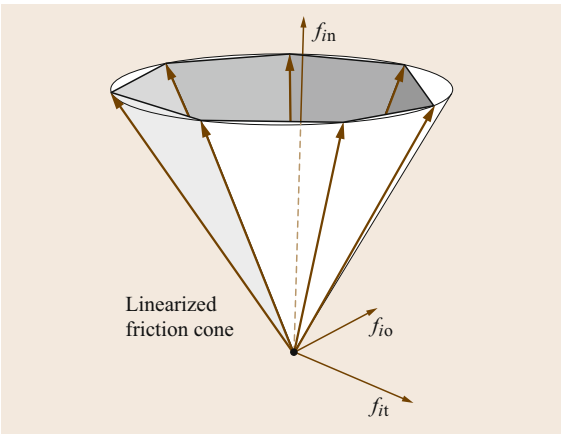


Fig. 38.11 Quadratic cone approximated as a polyhedral cone with seven generators

inition of $\mathbf{S}_i$

$$\mathbf{S}_i = \begin{pmatrix} \cdots & 1 & \cdots \\ \cdots & \mu_i \cos(2k\pi/n_g) & \cdots \\ \cdots & \mu_i \sin(2k\pi/n_g) & \cdots \end{pmatrix}, \tag{38.59}$$

where the index k varies from one to n_g. If one prefers to approximate the quadratic friction cone by a circumscribing polyhedral cone, one simply replaces μ_i in the above definition with $\mu_i/\cos(\pi/n_g)$.

The adjustment needed for the SF model is quite simple. Since the torsional friction in this model is decoupled from the tangential friction, its generators are given by $[1\ 0\ 0\ \pm b\upsilon_i]^T$. Thus $\mathbf{S}_i$ for the SF model is

$$\mathbf{S}_i = \left(\begin{array}{ccc|cc} \cdots & 1 & \cdots & 1 & 1 \\ \cdots & \mu_i \cos(2k\pi/n_g) & \cdots & 0 & 0 \\ \cdots & \mu_i \sin(2k\pi/n_g) & \cdots & 0 & 0 \\ \hline \cdots & 0 & \cdots & b\upsilon_i & -b\upsilon_i \end{array}\right), \tag{38.60}$$

where b is the characteristic length used to unify units. The set of total contact wrenches that may be applied by the hand without violating the contact friction law at any contact can be written as

$$\mathbf{G}\boldsymbol{\lambda} = \mathbf{S}\boldsymbol{\sigma}, \quad \boldsymbol{\sigma} \geq \mathbf{0},$$

where

$$\mathbf{S} = (\mathbf{S}_1, \cdots, \mathbf{S}_{n_g})$$

and

$$\boldsymbol{\sigma} = (\boldsymbol{\sigma}_1^T \cdots \boldsymbol{\sigma}_{n_g}^T)^T.$$

It is convenient to reformulate the friction constraints in a dual form

$$\mathbf{F}_i \boldsymbol{\lambda}_i \geq \mathbf{0}. \tag{38.61}$$

In this form, each row of $\mathbf{F}_i$ is normal to a face formed by two adjacent generators of the approximate cone. For an HF contact, row i of $\mathbf{F}_i$ can be computed as the cross product of $\boldsymbol{s}_i$ and $\boldsymbol{s}_{i+1}$. In the case of an SF contact, the generators are of dimension four, so simple cross products will not suffice. However, general methods exist to perform the conversion from the generator form to the face normal form [38.39].

The face normal constraints for all contacts can be combined into the following compact form

$$\mathbf{F}\boldsymbol{\lambda} \geq \mathbf{0}, \tag{38.62}$$

where $\mathbf{F} = \text{Blockdiag}(\mathbf{F}_1, \ldots, \mathbf{F}_{n_c})$.

Let $\boldsymbol{e}_i \in \mathbb{R}^{n_{\lambda i}}$ be the first row of $\mathbf{H}_i$. Further let

$$\boldsymbol{e} = [\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{n_c}] \in \mathbb{R}^{n_\lambda}$$

and let

$$\mathbf{E} = \text{Blockdiag}(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{n_c}) \in \mathbb{R}^{n_\lambda \times n_c}.$$

The following linear program is a quantitative test for frictional form closure. The optimal objective function value d^* is a measure of the distance the contact forces are from the boundaries of their friction cones, and hence a crude measure of how far a grasp is from losing frictional form closure.

LP2: maximize: d
subject to: $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$
$\mathbf{F}\boldsymbol{\lambda} - \mathbf{1}d \geq \mathbf{0}$
$d \geq 0$
$\boldsymbol{e}\boldsymbol{\lambda} \leq n_c$.

The last inequality in LP2 is simply the sum of the magnitudes of the normal components of the contact forces. After solving LP2, if $d^* = 0$ frictional form closure does not exist, but if $d^* > 0$, then it does.

If the grasp has frictional form closure, the last step to determine the existence of force closure is to verify the condition $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^T) = \mathbf{0}$. If it holds, then the grasp has force closure. This condition is easy to verify with another linear program LP3.

LP3: maximize: d
subject to: $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$
$\mathbf{J}^T\boldsymbol{\lambda} = \mathbf{0}$
$\mathbf{E}\boldsymbol{\lambda} - \mathbf{1}d \geq \mathbf{0}$
$d \geq 0$
$\boldsymbol{e}\boldsymbol{\lambda} \leq n_c$.

In summary, force closure testing is a three-step process.

Approximate Force Closure Test.

1. Compute rank($\mathbf{G}$).
   a) If rank($\mathbf{G}$) $\neq n_\upsilon$, then force closure does not exist. Stop.
   b) If rank($\mathbf{G}$) $= n_\upsilon$, continue.
2. Solve LP2: *Test frictional form closure.*
   a) If $d^* = 0$, then frictional form closure does not exist. Stop.
   b) If $d^* > 0$, then frictional form closure exists and d^* is a crude measure of how far the grasp is from losing frictional form closure.

3. Solve LP3: *Test control of internal force.*
   a) If $d^* > 0$, then force closure does not exist.
   b) If $d^* = 0$, then force closure exists.

Variation of the Test. A variation of the approximate force closure test arises when the object is partially constrained with the bilateral constraints described by (38.53), in such a cases the definitions of frictional form closure becomes

$$\begin{aligned} \overline{\mathbf{G}}\boldsymbol{\lambda} &= \mathbf{0}\,, \\ \boldsymbol{\lambda} &\in \operatorname{int}(\mathcal{F})\,, \end{aligned} \tag{38.63}$$

where $\overline{\mathbf{G}} = \mathbf{A}_B\mathbf{G}$, $\mathbf{A}_B$ is an annihilator of the column space of $\mathbf{B}$. Similarly, the force closure condition can be stated as

$$\begin{aligned} \overline{\mathbf{G}}\boldsymbol{\lambda} &= \mathbf{0}\,, \\ \boldsymbol{\lambda} &\in \operatorname{int}(\mathcal{F})\,, \\ \mathcal{N}(\overline{\mathbf{G}}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}}) &= \mathbf{0}\,, \end{aligned} \tag{38.64}$$

Since the frictional form closure definition (38.63) is analogous to Definition 38.9 and the force closure definition (38.64) is analogous to Definition 38.10, the force closure test can be applied by substituting $\overline{\mathbf{G}}$ for $\mathbf{G}$, provided that, $(n_v - \operatorname{rank}(\mathbf{B}))$ is substituted for n_v. For detailed information on the derivation of these conditions, please refer to [38.42].

See Example 1, Part 6.

Planar Simplifications

In planar grasping systems, the approximate method described above is exact. This is because the SF models are meaningless, since rotations about the contact normal would cause motions out of the plane. With regard to the HF model, for planar problems, the quadratic friction cone becomes linear, with its cone represented exactly as

$$\mathbf{F}_i = \frac{1}{\sqrt{1+\mu_i^2}} \begin{pmatrix} \mu_i & 1 \\ \mu_i & -1 \end{pmatrix}. \tag{38.65}$$

Nguyen's graphical form closure test can be applied to planar grasps with two frictional contacts [38.45]. The only change is that the four contact normals are replaced by the four generators of the two friction cones. However, the test can only determine frictional form closure, since it does not incorporate the additional information needed to determine force closure.

38.5 Examples

38.5.1 Example 1: Grasped Sphere

Part 1: $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{J}}$

Figure 38.12 shows a planar projection of a three-dimensional sphere of radius r grasped by two fingers, which make two contacts at angles θ_1 and θ_2. The frames $\{\boldsymbol{C}\}_1$ and $\{\boldsymbol{C}\}_2$ are oriented so that their $\hat{\boldsymbol{o}}$-directions point out of the plane of the figure (as indicated by the small filled circles at the contact points). The axes of the frames $\{\boldsymbol{N}\}$ and $\{\boldsymbol{B}\}$ were chosen to be axis-aligned with coincident origins located at the center of the sphere. The z-axes are pointing out of the page. Observe that since the two joint axes of the left finger are perpendicular to the (x, y)-plane, it operates in that plane for all time. The other finger has three revolute joints. Because its first and second axes, $\hat{z}_3$ and $\hat{z}_4$, currently lie in the plane, rotation about $\hat{z}_3$ will cause $\hat{z}_4$ to attain an out-of-plane component and would cause the finger tip at contact 2 to leave the plane.

In the current configuration, the rotation matrix for the i-th contact frame is defined as follows

$$\mathbf{R}_i = \begin{pmatrix} -\cos(\theta_i) & \sin(\theta_i) & 0 \\ -\sin(\theta_i) & -\cos(\theta_i) & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{38.66}$$

The vector from the origin of $\{\boldsymbol{N}\}$ to the i-th contact point is given by

$$\boldsymbol{c}_i - \boldsymbol{p} = r \begin{pmatrix} \cos(\theta_i) & \sin(\theta_i) & 0 \end{pmatrix}^{\mathrm{T}}. \tag{38.67}$$

Substituting into (38.3) and (38.6) yields the complete grasp matrix for contact i

$$\tilde{\mathbf{G}}_i = \left(\begin{array}{ccc|ccc} -c_i & s_i & 0 & & & \\ -s_i & -c_i & 0 & & \mathbf{0} & \\ 0 & 0 & 1 & & & \\ \hline 0 & 0 & rs_i & -c_i & s_i & 0 \\ 0 & 0 & -rc_i & -s_i & -c_i & 0 \\ 0 & -r & 0 & 0 & 0 & 1 \end{array}\right), \tag{38.68}$$

where $\mathbf{0} \in \mathbb{R}^{3\times 3}$ is the zero matrix and c_i and s_i are abbreviations for $\cos(\theta_i)$ and $\sin(\theta_i)$, respectively. The complete grasp matrix is defined as: $\tilde{\mathbf{G}} = (\tilde{\mathbf{G}}_1\ \tilde{\mathbf{G}}_2) \in \mathbb{R}^{6\times 12}$.

The accuracy of this matrix can be verified by inspection, according to Remark 38.2. For example, the first column is the unit wrench of the unit contact normal; the first three components are the direction cosines

of $\hat{\boldsymbol{n}}_i$ and the last three are $(\boldsymbol{c}_i - \boldsymbol{p}) \times \hat{\boldsymbol{n}}_i$. Since $\hat{\boldsymbol{n}}_i$ is collinear with $(\boldsymbol{c}_i - \boldsymbol{p})$, the cross products (the last three components of the column) are zero. The last three components of the second column represent the moment of $\hat{\boldsymbol{t}}_i$ about the $\boldsymbol{x}$-, $\boldsymbol{y}$-, and $\boldsymbol{z}$-axes of $\{N\}$. Since $\hat{\boldsymbol{t}}_i$ lies in the (x, y)-plane, the moments with the x- and $\boldsymbol{y}$-axes are zero. Clearly $\hat{\boldsymbol{t}}_i$ produces a moment of $-r$ about the $\boldsymbol{z}$-axis.

Construction of the hand Jacobian $\tilde{\mathbf{J}}_i$ for contact i requires knowledge of the joint axis directions and the origins of the frames fixed to the links of each finger. Figure 38.13 shows the hand in the same configuration as in Fig. 38.12, but with some additional data needed to construct the hand Jacobian. Assume that the origins of the joint frames lie in the plane of the figure. In the current configuration, the quantities of interest for contact 1, expressed in $\{\boldsymbol{C}\}_1$ are

$$\boldsymbol{c}_1 - \boldsymbol{\zeta}_1 = \begin{pmatrix} l_2 & l_1 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.69}$$

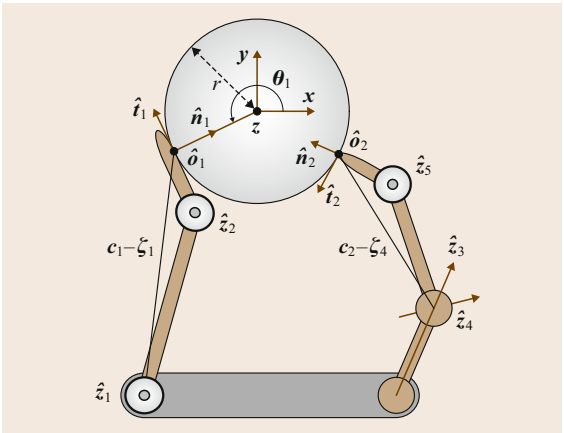


Fig. 38.12 A sphere grasped by a two-fingered hand with 5 revolute joints

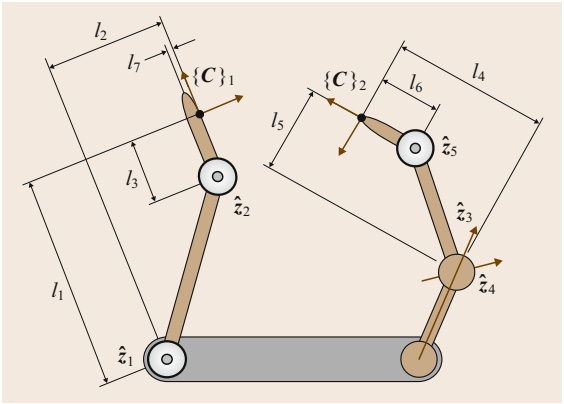


Fig. 38.13 Relevant data for the hand Jacobian in Fig. 38.12

$$\boldsymbol{c}_1 - \boldsymbol{\zeta}_2 = \begin{pmatrix} l_7 & l_3 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.70}$$

$$\hat{\boldsymbol{z}}_1 = \hat{\boldsymbol{z}}_2 = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \tag{38.71}$$

The quantities of interest for contact 2, in $\{\boldsymbol{C}\}_2$ are

$$\boldsymbol{c}_2 - \boldsymbol{\zeta}_3 = \boldsymbol{c}_2 - \boldsymbol{\zeta}_4 = \begin{pmatrix} l_4 & l_5 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.72}$$

$$\boldsymbol{c}_2 - \boldsymbol{\zeta}_5 = \begin{pmatrix} l_6 & 0 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.73}$$

$$\hat{\boldsymbol{z}}_3 = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.74}$$

$$\hat{\boldsymbol{z}}_4(q_3) = \frac{\sqrt{2}}{2} \begin{pmatrix} -1 & 1 & 0 \end{pmatrix}^{\mathrm{T}} , \tag{38.75}$$

$$\hat{\boldsymbol{z}}_5(q_3, q_4) = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \tag{38.76}$$

Generally all of the components of the $\boldsymbol{c} - \boldsymbol{\zeta}$ and $\hat{\boldsymbol{z}}$ vectors (including the components that are zero in the current configuration), are functions of $\boldsymbol{q}$ and $\boldsymbol{u}$. The dependencies of the $\hat{\boldsymbol{z}}$ vectors are shown explicitly.

Substituting into (38.14), (38.11) and (38.8) yields the complete hand Jacobian $\tilde{\mathbf{J}} \in \mathbb{R}^{12\times 5}$

$$\tilde{\mathbf{J}} = \left(\begin{array}{cc|ccc} -l_1 & -l_3 & & & \\ l_2 & l_7 & & & \\ 0 & 0 & & & \\ 0 & 0 & & \mathbf{0} & \\ 0 & 0 & & & \\ 1 & 1 & & & \\ \hline & & 0 & 0 & 0 \\ & & 0 & 0 & l_6 \\ & & l_4 & \frac{\sqrt{2}}{2}(l_4 + l_5) & 0 \\ & \mathbf{0} & 0 & -\frac{\sqrt{2}}{2} & 0 \\ & & -1 & -\frac{\sqrt{2}}{2} & 0 \\ & & 0 & 0 & 1 \end{array}\right) .$$

The horizontal dividing line partitions $\tilde{\mathbf{J}}$ into $\tilde{\mathbf{J}}_1$ (on top) and $\tilde{\mathbf{J}}_2$ (on the bottom). The columns correspond to joints 1 through 5. The block diagonal structure is a result of the fact that finger i directly affects only contact i.

Example 1, Part 2: G and J

Assume that the contacts in Fig. 38.12 are both of type SF. Then the selection matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \left(\begin{array}{cccccc|cccccc} 1 & 0 & 0 & 0 & 0 & 0 & & & & & & \\ 0 & 1 & 0 & 0 & 0 & 0 & & & & \mathbf{0} & & \\ 0 & 0 & 1 & 0 & 0 & 0 & & & & & & \\ 0 & 0 & 0 & 1 & 0 & 0 & & & & & & \\ \hline & & & & & & 1 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & 0 & 1 & 0 & 0 & 0 & 0 \\ & & & \mathbf{0} & & & 0 & 0 & 1 & 0 & 0 & 0 \\ & & & & & & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right) ,$$

thus the matrices $\mathbf{G}^{\mathrm{T}} \in \mathbb{R}^{8\times 6}$ and $\mathbf{J} \in \mathbb{R}^{8\times 5}$ are constructed by removing rows 5, 6, 11, and 12 from $\tilde{\mathbf{G}}^{\mathrm{T}}$ and $\tilde{\mathbf{J}}$

$$\mathbf{G}^{\mathrm{T}} = \left(\begin{array}{cccccc} -c_1 & -s_1 & 0 & 0 & 0 & 0 \\ s_1 & -c_1 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & r\,s_1 & -r\,c_1 & 0 \\ 0 & 0 & 0 & -c_1 & -s_1 & 0 \\ \hline -c_2 & -s_2 & 0 & 0 & 0 & 0 \\ s_2 & -c_2 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & r\,s_2 & -r\,c_2 & 0 \\ 0 & 0 & 0 & -c_2 & -s_2 & 0 \end{array}\right), \quad (38.77)$$

$$\mathbf{J} = \left(\begin{array}{cc|ccc} -l_1 & -l_3 & & & \\ l_2 & l_7 & & \mathbf{0} & \\ 0 & 0 & & & \\ 0 & 0 & & & \\ \hline & & 0 & 0 & 0 \\ & & 0 & 0 & l_6 \\ \mathbf{0} & & l_4 & \frac{\sqrt{2}}{2}(l_4+l_5) & 0 \\ & & 0 & -\frac{\sqrt{2}}{2} & 0 \end{array}\right). \quad (38.78)$$

Notice that changing the contact models is easily accomplished by removing more rows. Changing contact 1 to HF would eliminate the fourth rows from $\mathbf{G}^{\mathrm{T}}$ and $\mathbf{J}$. Changing it to PwoF would eliminate the second, third and fourth rows of $\mathbf{G}^{\mathrm{T}}$ and $\mathbf{J}$. Changing the model at contact 2 would remove either just the eighth row or the sixth, seventh, and eighth rows.

Example 1, Part 3: Reduce to Planar Case

The grasp shown in Fig. 38.12 can be reduced to a planar problem by following the explicit formulas given above, but it can also be done by understanding the physical interpretations of the various rows and columns of the matrices. Proceed by eliminating velocities and forces that are out of the plane. This can be done by removing the z-axes from $\{N\}$ and $\{B\}$, and the $\hat{\boldsymbol{o}}$-directions at the contacts. Further, joints 3 and 4 must be locked. The resulting $\mathbf{G}^{\mathrm{T}}$ and $\mathbf{J}$ are constructed eliminating certain rows and columns. $\mathbf{G}^{\mathrm{T}}$ is formed by removing rows 3, 4, 7, and 8 and columns 3, 4, and 5. $\mathbf{J}$ is formed by removing rows 3, 4, 7, and 8 and columns 3 and 4 yielding

$$\mathbf{G}^{\mathrm{T}} = \left(\begin{array}{ccc} -c_1 & -s_1 & 0 \\ s_1 & -c_1 & -r \\ \hline -c_2 & -s_2 & 0 \\ s_2 & -c_2 & -r \end{array}\right), \quad (38.79)$$

$$\mathbf{J} = \left(\begin{array}{cc|c} -l_1 & -l_3 & 0 \\ l_2 & l_7 & 0 \\ \hline 0 & 0 & 0 \\ 0 & 0 & l_6 \end{array}\right). \quad (38.80)$$

Example 1, Part 4: Grasp Classes

The first column of Table 38.11 reports the dimensions of the main subspaces of $\mathbf{J}$ and $\mathbf{G}$ for the sphere grasping example with different contact models. Only non-trivial null spaces are listed.

In the case of two HF contact models, all four null spaces are non-trivial, so the system satisfies the conditions for all four grasp classes. The system is graspable because there is an internal force along the line segment connecting the two contact points. Indeterminacy is manifested in the fact that the hand cannot resist a moment acting about that line. Redundancy is seen to exist since joint 3 can be used to move contact 2 out of the plane of the figure, but joint 4 can be rotated in the opposite direction to cancel this motion. Finally, the grasp is defective, because the contact forces and the instantaneous velocities along the $\hat{\boldsymbol{o}}_1$ and $\hat{\boldsymbol{n}}_2$ directions of contact 1 and 2, respectively, cannot be controlled through the joint torques and velocities. These interpretations are borne out in the null space basis matrices below, computed using $r = 1$, $\cos(\theta_1) = -0.8 = -\cos(\theta_2)$, $\sin(\theta_1) = \cos(\theta_2) = -0.6$, $l_1 = 3$, $l_2 = 2$, $l_3 = 1$, $l_4 = 2$, $l_5 = 1$, $l_6 = 1$, and $l_7 = 0$

$$\mathbf{N}(\mathbf{J}) \approx \begin{pmatrix} 0 \\ 0 \\ -0.73 \\ 0.69 \\ 0 \end{pmatrix}, \quad \mathbf{N}(\mathbf{G}^{\mathrm{T}}) \approx \begin{pmatrix} 0 \\ 0 \\ 0.51 \\ 0.86 \\ 0 \\ 0 \end{pmatrix}, \quad (38.81)$$

$$\mathbf{N}(\mathbf{G}) \approx \begin{pmatrix} 0.57 \\ -0.42 \\ 0 \\ 0.57 \\ 0.42 \\ 0 \end{pmatrix}, \quad \mathbf{N}(\mathbf{J}^{\mathrm{T}}) = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}. \quad (38.82)$$

Table 38.11 Dimensions of main subspaces and classifications of grasp studied in Example 1

Models	Dimension	Class
HF,HF	$\dim \mathcal{N}(\mathbf{J}) = 1$	Redundant
	$\dim \mathcal{N}(\mathbf{G}^{\mathrm{T}}) = 1$	Indeterminate
	$\dim \mathcal{N}(\mathbf{G}) = 1$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 2$	Defective
SF,HF	$\dim \mathcal{N}(\mathbf{J}) = 1$	Redundant
	$\dim \mathcal{N}(\mathbf{G}) = 1$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 3$	Defective
HF,SF	$\dim \mathcal{N}(\mathbf{G}) = 1$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 2$	Defective
SF,SF	$\dim \mathcal{N}(\mathbf{G}) = 2$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 3$	Defective

Notice that changing either contact to SF makes it possible for the hand to resist external moments applied about the line containing the contacts, so the grasp loses indeterminacy, but retains graspability (with squeezing still possible along the line of the contacts). However, if contact 2 is the SF contact, the grasp loses its redundancy. While the second contact point can still be moved out of the plane by joint 3 and back in by joint 4, this canceled translation of the contact point yields a net rotation about $\hat{\boldsymbol{n}}_2$ (this also implies that the hand can control the moment applied to the object along the line containing the contacts). Changing to SF at contact 2 does *not* affect the hand's inability to move contact 1 and contact 2 in the $\hat{\boldsymbol{o}}_1$ and $\hat{\boldsymbol{n}}_2$ directions, so the defectivity property is retained.

Example 1, Part 5: Desirable Properties

Assuming contact model types of SF and HF at contacts 1 and 2, respectively, $\mathbf{G}$ is full row rank and so $\mathcal{N}(\mathbf{G}^{\mathrm{T}}) = \mathbf{0}$ (see Table in part 4 of this example). Therefore, as long as the hand is sufficiently dexterous, it can apply any wrench in $\mathbb{R}^6$ to the object. Also, if the joints are locked, object motion will be prevented. Assuming the same problem values used in the previous part of this problem, the matrix $\mathbf{G}^{\mathrm{T}}$ is

$$\mathbf{G}^{\mathrm{T}} = \left(\begin{array}{cccccc} -c_1 & -s_1 & 0 & 0 & 0 & 0 \\ s_1 & -c_1 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & rs_1 & -rc_1 & 0 \\ 0 & 0 & 0 & -c_1 & -s_1 & 0 \\ \hline -c_2 & -s_2 & 0 & 0 & 0 & 0 \\ s_2 & -c_2 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & rs_2 & -rc_2 & 0 \end{array}\right) \tag{38.83}$$

Bases for the three non-trivial null spaces are

$$\mathbf{N}(\mathbf{J}^{\mathrm{T}}) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \tag{38.84}$$

$$\mathbf{N}(\mathbf{J}) \approx \begin{pmatrix} 0 \\ 0 \\ -0.73 \\ 0.69 \\ 0 \end{pmatrix}, \quad \mathbf{N}(\mathbf{G}) \approx \begin{pmatrix} 0.57 \\ -0.42 \\ 0 \\ 0 \\ 0.57 \\ 0.42 \\ 0 \end{pmatrix}. \tag{38.85}$$

Since $\mathcal{R}(\mathbf{J})$ is four-dimensional and $\mathcal{N}(\mathbf{G})$ is one-dimensional, the maximum dimension of $\mathcal{R}(\mathbf{J}) + \mathcal{N}(\mathbf{G})$ cannot be more than 5, and therefore, the hand cannot control all possible object velocities. For example, the contact velocity $\boldsymbol{v}_{\mathrm{cc}} = (0\,0\,0\,0.8\,0\,0\,0)^{\mathrm{T}}$ is in $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$, and so cannot be controlled by the fingers. It is also equal to 0.6 times the third column of $\mathbf{G}^{\mathrm{T}}$ plus the 4th column of $\mathbf{G}^{\mathrm{T}}$ and therefore is in $\mathcal{R}(\mathbf{G}^{\mathrm{T}})$. Since the mapping between $\mathcal{R}(\mathbf{G})$ and $\mathcal{R}(\mathbf{G}^{\mathrm{T}})$ is one-to-one and onto, this uncontrollable contact velocity corresponds to a unique uncontrollable object velocity, $\boldsymbol{v} = (0\,0\,0.6\,1\,0\,0)$. In other words, the hand cannot cause the center of the sphere to translate in the z-direction, while also rotating about the $\boldsymbol{x}$-axis (and not other axes simultaneously).

On the question of controlling all internal object forces, the answer is *yes*, since $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G}) = 0$. This conclusion is clear from the fact that $\mathcal{N}(\mathbf{G})$ has non-zero values in the first, second, and sixth positions, while all columns of $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ have zeros in those positions.

Example 1, Part 6: Force Closure

Again assume that contacts 1 and 2 on the grasped sphere were modeled as SF and HF contacts, respectively. Under this assumption, $\mathbf{G}$ is full row rank, and the internal force corresponds to equal and opposite contact forces. For frictional form closure to exist, the internal force must lie within the friction cones. Choosing r and the sines and cosines of θ_1 and θ_2 as in example 1, part 4, frictional form closure can be shown to exist if both friction coefficients are greater than 0.75. For this grasp, since $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G}) = \mathbf{0}$, frictional form closure is equivalent to force closure.

The plot in Fig. 38.14 was generated by fixing μ_2 at a specific value and varying μ_1 from 0.5 to 2.0. Notice that for $\mu_1 < 0.75$, force closure does not exist regardless of the value of μ_2. The metric increases smoothly until a specific value of μ_1. From that point on, the friction coefficient at contact 2 is the limiting factor. To increase the metric further, μ_2 must be increased.

38.5.2 Example 2: Grasped Polygon in the Plane

Part 1: G and J

Figure 38.15 shows a planar hand grasping a polygon. Finger 1 (on the right) contains two joints numbered 1 and 2. Finger 2 contains joints 3–7, which are numbered in increasing order moving from the palm distally. The inertial frame has been chosen to lie inside the object, with its $\boldsymbol{x}$-axis passing through contacts 1 and 2, and collinear with the normal vector of contact 2.

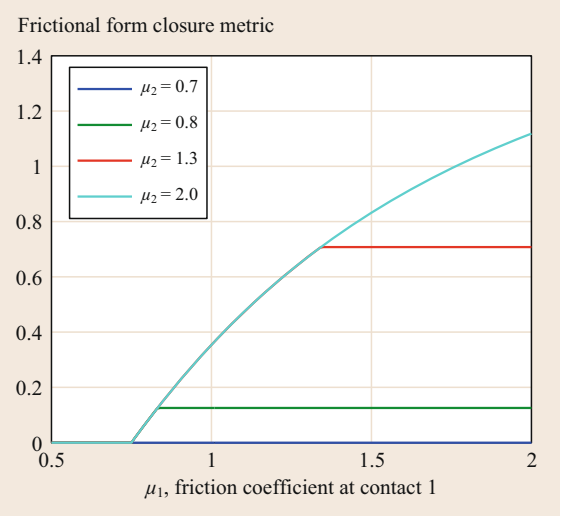


Fig. 38.14 Plot of force closure metric versus friction coefficient on contact 1

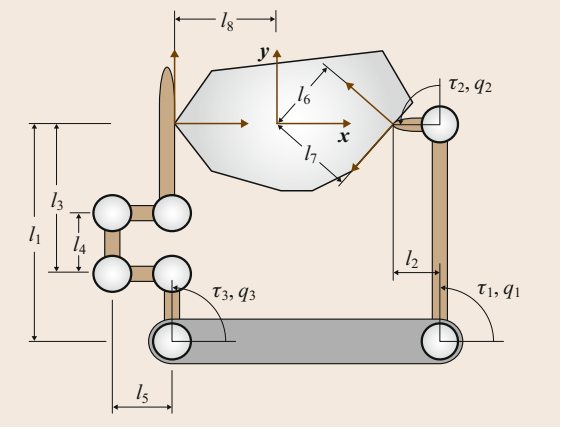


Fig. 38.15 Planar hand with 2 fingers and 7 joints grasping a polygonal object

The rotation matrices are given by

$$\mathbf{R}_1 = \begin{pmatrix} -0.8 & -0.6 \\ 0.6 & -0.8 \end{pmatrix}, \quad \mathbf{R}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{38.86}$$

Assuming HF contacts, $\mathbf{G}$ is given as follows

$$\mathbf{G} = \left(\begin{array}{cc|cc} -0.8 & -0.6 & 1 & 0 \\ 0.6 & -0.8 & 0 & 1 \\ l_6 & -l_7 & 0 & -l_8 \end{array}\right). \tag{38.87}$$

Notice that the first two columns of $\mathbf{G}$ correspond to the normal and tangential unit vectors at contact 1. The third and fourth columns correspond to contact 2.

Assuming HF contacts and all joints are active (i. e., not locked), $\mathbf{J}$ is

$$\mathbf{J}^{\mathrm{T}} = \left(\begin{array}{cc|cc} 0.8l_1 & 0.6l_1 & \multicolumn{2}{c}{\multirow{2}{*}{$\mathbf{0}$}} \\ -0.6l_2 & 0.8l_2 & & \\ \hline & & -l_1 & 0 \\ & & -l_3 & 0 \\ \multicolumn{2}{c|}{\mathbf{0}} & -l_3 & l_5 \\ & & -l_3 + l_4 & l_5 \\ & & -l_3 + l_4 & 0 \end{array}\right). \tag{38.88}$$

The first two columns of $\mathbf{J}^{\mathrm{T}}$ are the torques required to produce a unit force in the $\hat{\boldsymbol{n}}_1$ and $\hat{\boldsymbol{t}}_1$ directions at contact 1. The horizontal line through the matrix partitions the contributions for the first finger (the upper part) and second finger. Notice that both $\mathbf{J}^{\mathrm{T}}$ and $\mathbf{G}$ are full column rank.

Example 2, Part 2: Grasp Classes

This example clearly illustrates the physical qualities of the various grasp classes without introducing features that can cloud the descriptions.

We now discuss the details of the four grasp classes using the previous planar example. During these discussions it is useful to choose non-dimensional values for the parameters in the grasping system. The lengths were assumed to have the following values (the results are the same, regardless of a particular choice of units, so units are not specified)

$$l_1 = 2.89\,, \quad l_2 = 0.75\,, \quad l_3 = 1.97\,, \tag{38.89}$$
$$l_4 = 0.80\,, \quad l_5 = 0.80\,, \quad l_6 = 0.90\,, \tag{38.90}$$
$$l_7 = 1.20\,, \quad l_8 = 1.35\,. \tag{38.91}$$

Redundant. Redundancy exists if $\mathcal{N}(\mathbf{J})$ is non-trivial. Assuming that both contacts are hard contacts and all the joints are active, $\operatorname{rank}(\mathbf{J}) = 4$, so $\mathcal{N}(\mathbf{J})$ is three-dimensional. A basis for $\mathcal{N}(\mathbf{J})$ was obtained using Matlab's `null()` function

$$\mathbf{N}(\mathbf{J}) \approx \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline -0.49 & -0.31 & -0.27 \\ 0.53 & 0.64 & -0.17 \\ 0.49 & -0.50 & -0.02 \\ -0.49 & 0.50 & 0.02 \\ -0.02 & 0.01 & 0.95 \end{array}\right). \tag{38.92}$$

Since the first two rows are zero, $\mathcal{N}(\mathbf{J})$ does not include motions of the first finger (on the right of the palm). To understand this, assume the object is fixed in the plane. Then the first finger cannot maintain sticking contact at contact 1 unless its joints are also fixed.

The three non-zero columns corresponding to finger 2 show that there are 3 basis motions of its joints that allow the finger contact to stick to the object contact. For example, the first column shows that if joint 3 moves roughly as much as joints 4, 5, and 6, but in the opposite direction as joints 4 and 5 and in the same directionas joint 6, while joint 7 is more or less fixed, then contact 2 will be maintained.

Notice that finger 2 contains a parallelogram. Because of this geometry, one can see that the vector $(0\,0\,0\;-1\,1\;-1\,1)^{\mathrm{T}}$ is an element of $\mathcal{N}(\mathbf{J})$. The velocity interpretation of this vector is that the link of the finger connected to the palm, and the link touching the object remain fixed in space, while the parallelogram moves as a simple four-bar mechanism. Similarly, joint actions in $\mathcal{N}(\mathbf{J})$ do not affect the contact forces, but cause internal hand velocities. Also, notice that since $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) = \mathbf{0}$, the entire space of possible generalized velocities and forces at the contacts can be generated by the joints.

Indeterminate. As noted above, with HF contact models, the system is graspable. However, replacing the HF models with PwoF models removes the tangent force components in the $\hat{\boldsymbol{t}}_1$ and $\hat{\boldsymbol{t}}_2$ directions. This effectively removes columns 2 and 4 from $\mathbf{G}$, which guarantees that the system will be indeterminate. The reduced matrix is denoted by $\mathbf{G}_{(1,3)}$. In this case $\mathcal{N}\left(\mathbf{G}^{\mathrm{T}}_{(1,3)}\right)$ is

$$\mathbf{N}\left(\mathbf{G}^{\mathrm{T}}_{(1,3)}\right) \approx \begin{pmatrix} 0 \\ -0.83 \\ 0.55 \end{pmatrix}. \tag{38.93}$$

Physically, this basis vector corresponds to moving the object such that the point coincident with the origin of $\{\boldsymbol{N}\}$ moves directly downward, while the object rotates counter clockwise. Also, if the analogous force and moment were applied to the object, the frictionless contacts could not maintain equilibrium.

Graspable. With two HF contact models in force, $\mathrm{rank}(\mathbf{G}) = 3$, so $\mathcal{N}(\mathbf{G})$ is one–dimensional and the system is graspable. The null space basis vector of the grasp matrix is

$$\mathbf{N}(\mathbf{G}) \approx \begin{pmatrix} 0.57 \\ 0.42 \\ \hline 0.71 \\ 0 \end{pmatrix}. \tag{38.94}$$

The physical interpretation of this basis vector is two opposing forces acting through the two contact points. Recall that because the contact model is kinematic, there is no consideration of contact friction. However, given the direction of the contact normal relative to the line of the internal force, one can see that if the coefficient of friction is not greater than 0.75, squeezing tightly will cause sliding at contact 1, thus violating the kinematic contact model.

Defective. In a defective grasp, $\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \neq \mathbf{0}$. Given that the original $\mathbf{J}$ is full row rank, the grasp is *not* defective. However, it can be made defective by locking a number of joints and/or changing the hand's configuration so that $\mathbf{J}$ is no longer full rank. For example, locking joints 4, 5, 6, and 7 makes finger 2 a single-link finger with only joint 3 active. In this new grasping system, $\mathbf{J}^{\mathrm{T}}_{(1,2,3)}$ is simply the first three rows of the original $\mathbf{J}^{\mathrm{T}}$ given in (38.88), where the subscript is the list of indices of active joints. The null space basis vector is

$$\mathbf{N}(\mathbf{J}^{\mathrm{T}}_{(1,2,3)}) = \begin{pmatrix} 0 \\ 0 \\ \hline 0 \\ 1 \end{pmatrix}. \tag{38.95}$$

This grasp is defective, since there is a subspace of contact velocities and forces that cannot be controlled by joint generalized velocities and forces. Since only the last component of $\mathbf{N}(\mathbf{J}^{\mathrm{T}}_{(1,2,3)})$ is non-zero, it would be impossible for the hand to give the contact point 2 on the object a velocity in the $\hat{\boldsymbol{t}}_2$-direction while maintaining the contact. This is also clear from the arrangement of joint 3, contact 2, and the direction of the contact normal. The dual interpretation is that forces in $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ are resisted by the structure and the corresponding joint loads is zero, or equivalently that those forces are not controllable by the hand. Notice that if the model of contact 2 were changed to point-without-friction, then $\mathbf{N}(\mathbf{J}^{\mathrm{T}}_{(1,2,3)}) = \mathbf{0}$ and the system would no longer be defective.

38.5.3 Example 3: Hyperstatic Grasps

Part 1: $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{J}}$

Figure 38.16 shows a planar projection of a three-dimensional sphere of radius l grasped by one finger only, with 3 revolute joints, through 3 contacts. The frames $\{\boldsymbol{C}\}_1$, $\{\boldsymbol{C}\}_2$ and $\{\boldsymbol{C}\}_3$ are oriented so that their $\hat{\boldsymbol{o}}$-directions point out of the plane of the figure (as indicated by the small filled circle). The axes of the frames $\{\boldsymbol{N}\}$ and $\{\boldsymbol{B}\}$ were chosen to be axis-aligned with coincident origins located at the center of the sphere. The z-axes are pointing out of the page. Observe that since

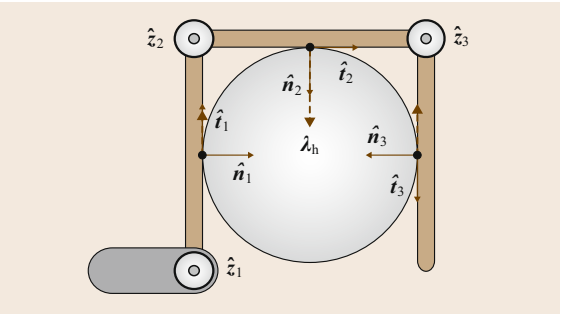


Fig. 38.16 A sphere grasped by a finger with three revolute joints. The force direction $\boldsymbol{\lambda}_h$ (*dashed line*) is a force that belongs to both $\mathcal{N}(\mathbf{G})$ and $\mathcal{N}(\mathbf{J}^{\mathrm{T}})$ and causes hyperstaticity

the three joint axes of the finger are perpendicular to the (x, y)-plane, the grasp operates in that plane for all time.

Assume that the width of all the links of the robotic hand is zero. Rotation matrices $\mathbf{R}_i$ and vectors $\boldsymbol{c}_i - \boldsymbol{p}$ for $i = 1, \ldots, 3$, can be computed as in (38.66) and (38.67) considering that $\theta_1 = \pi$, for contact 1, $\theta_2 = \pi/2$ and $\theta_3 = 0$, for contact 2 and 3, respectively. Finally, the complete grasp matrix is

$$\tilde{\mathbf{G}}^{\mathrm{T}} = \begin{pmatrix} \tilde{\mathbf{G}}_1 & \tilde{\mathbf{G}}_2 & \tilde{\mathbf{G}}_2 \end{pmatrix}^{\mathrm{T}} \in \mathbb{R}^{18\times 6} ,$$

where $\tilde{\mathbf{G}}_i$ is as defined in (38.68)

$$\tilde{\mathbf{G}}^{\mathrm{T}} = \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & l & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & l & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & -l & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right)$$

Construction of the complete hand Jacobian $\tilde{\mathbf{J}}_i$ for contact i requires knowledge of the joint axis directions and the origins of the frames fixed to the links of each finger. Assume that the origins of the Denavit–Hartenberg (DH) frames lie in the plane of the figure. In the current configuration, the quantities of interest for contact 1, expressed directly in $\{\boldsymbol{N}\}$ are

$$\begin{aligned} \boldsymbol{c}_1 - \boldsymbol{\zeta}_1 &= \begin{pmatrix} 0 & l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_1 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \end{aligned}$$

The quantities of interest for contact 2, in $\{\boldsymbol{N}\}$ are

$$\begin{aligned} \boldsymbol{c}_2 - \boldsymbol{\zeta}_1 &= \begin{pmatrix} l & 2l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \boldsymbol{c}_2 - \boldsymbol{\zeta}_2 &= \begin{pmatrix} l & 0 & 0 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_1 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_2 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \end{aligned}$$

The quantities of interest for contact 3, in $\{\boldsymbol{N}\}$ are

$$\begin{aligned} \boldsymbol{c}_3 - \boldsymbol{\zeta}_1 &= \begin{pmatrix} 2l & l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \boldsymbol{c}_3 - \boldsymbol{\zeta}_2 &= \begin{pmatrix} 2l & -l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \boldsymbol{c}_3 - \boldsymbol{\zeta}_3 &= \begin{pmatrix} 0 & -l & 0 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_1 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_2 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} , \\ \hat{\boldsymbol{z}}_3 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} . \end{aligned}$$

The complete hand Jacobian $\tilde{\mathbf{J}} \in \mathbb{R}^{18\times 3}$ (contact velocities are expressed in $\{\boldsymbol{C}\}_i$) is

$$\tilde{\mathbf{J}} = \left(\begin{array}{ccc} -l & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ \hline -l & -l & 0 \\ -2l & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline l & -l & -l \\ -2l & -2l & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{array}\right) .$$

The horizontal dividing lines partition $\tilde{\mathbf{J}}$ into $\tilde{\mathbf{J}}_1$ (on top), $\tilde{\mathbf{J}}_2$, and $\tilde{\mathbf{J}}_3$ (on the bottom). The columns correspond to joints 1 through 3.

Example 3, Part 2: G and J

Assume that the 3 contacts in Fig. 38.16 are of type HF. Then the selection matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{pmatrix}, \tag{38.96}$$

where $\mathbf{I}$ and $\mathbf{0}$ are in $\mathbb{R}^{3\times3}$, thus matrices $\mathbf{G}^{\mathrm{T}} \in \mathbb{R}^{9\times6}$ and $\mathbf{J} \in \mathbb{R}^{9\times3}$ are obtained by removing rows related to rotations from $\tilde{\mathbf{G}}^{\mathrm{T}}$ and $\tilde{\mathbf{J}}$

$$\mathbf{G}^{\mathrm{T}} = \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & l & 0 \\ \hline 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & l & 0 & 0 \\ \hline -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & -l & 0 \end{array}\right),$$

$$\mathbf{J} = \left(\begin{array}{ccc} l & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline l & l & 0 \\ 2l & 0 & 0 \\ 0 & 0 & 0 \\ \hline l & l & l \\ 2l & 2l & 0 \\ 0 & 0 & 0 \end{array}\right).$$

Example 3, Part 3: Grasp Classes

The first column of Table 38.12 reports the dimensions of the main subspaces of $\mathbf{J}^{\mathrm{T}}$ and $\mathbf{G}$ for the sphere grasping example with three hard finger contacts. Only non-trivial null spaces are listed.

The system is defective because there are generalized contact forces belonging to the subspace that are resisted by the structure, which correspond to zero joint actions

$$\mathbf{N}(\mathbf{J}^{\mathrm{T}}) = \left(\begin{array}{cccccc} 0 & 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right).$$

The first three columns represent generalized forces acting at the three contact points in a direction perpendicular to the plane of the Fig. 38.16. The fourth column corresponds to a contact force applied only along the $\hat{\boldsymbol{t}}_1$ direction.

System is graspable because the subspace of internal forces is three-dimensional; a possible basis matrix is

$$\mathbf{N}(\mathbf{G}) = \left(\begin{array}{ccc} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \\ \hline 1 & 0 & 2 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{array}\right).$$

The three force vectors of subspace $\mathbf{N}(\mathbf{G})$ are easily identified from Fig. 38.16. Note that all forces are expressed in local contact frames. The first column vector of $\mathbf{N}(\mathbf{G})$ represents opposed forces at contacts 1 and 2 along the line joining contacts 1 and 2. The second column vector parameterizes opposed forces at contacts 1 and 3 along the line joining contacts 1 and 3. The last vector represents forces along direction $\boldsymbol{\lambda}_h$, shown as the (dashed lines) in Fig. 38.16. Note that this direction (in wrench intensity space) corresponds to two upward friction forces at the left and right contacts and one downward with double the magnitude from the center of the top link in the work space.

Finally, the grasp is hyperstatic because

$$\mathbf{N}(\mathbf{G}) \cap \mathbf{N}(\mathbf{J}^{\mathrm{T}}) = \left(\begin{array}{c} 0 \\ 1 \\ 0 \\ \hline 2 \\ 0 \\ 0 \\ \hline 0 \\ -1 \\ 0 \end{array}\right) \neq 0 .$$

Table 38.12 Dimensions of main subspaces and classification of grasp given in Example 3

Dimension	Class
$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) = 6$	Defective
$\dim \mathcal{N}(\mathbf{G}) = 3$	Graspable
$\dim \mathcal{N}(\mathbf{J}^{\mathrm{T}}) \cap \mathcal{N}(\mathbf{G}) = 1$	Hyperstatic

Hyperstatic forces in this subspace, are internal forces that cannot be controlled through the hand joints. In Fig. 38.16 the internal force $\boldsymbol{\lambda}_h$ that is also in $\mathbf{N}(\mathbf{J}^{\mathrm{T}})$ is reported.

The grasp in Fig. 38.16 is an example of *power grasp*, style of grasp mentioned earlier that uses many contact points not only on the fingertips but also on the links of fingers and the palm [38.8, 28, 33].

All power grasps are kinematically defective ($\mathcal{N}(\mathbf{J}^{\mathrm{T}}) \neq \mathbf{0}$) and usually are hyperstatic. According to Sect. 38.2.2, rigid-body modeling is not sufficient to capture the overall system behavior, because the generalized contact forces in $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^{\mathrm{T}})$ leave the dynamics indeterminate.

Many approaches have been used to overcome rigid-body limitation in hyperstatic grasps such as those proposed in [38.16, 20, 21] where visco–elastic contact models have been used to solve the force indeterminacy. In [38.48], authors found that a sufficient condition for hyperstaticity is $m > q + 6$ where m is the dimension of contact force vector.

38.5.4 Example 4: Duality

Consider a frictionless disc constrained to translate in the plane; (Fig. 38.17). In this problem $n_v = 2$, so the space of applied contact forces and object velocities is the plane $\mathbb{R}^2$. In the top pair of pictures, a single (fixed) contact point imposes a half space constraint on the instantaneous velocity and limits the force at a frictionless contact to the ray. Both the ray and the (dark gray) half space are defined by the contact normal pointing into the object. Notice that the ray and half space are dual cones. When two contacts are present, the (light gray) force cone becomes the non-negative span of the two contact normals and the velocity cone is its dual. With the addition of the third contact, the grasp has form closure as indicated by the degeneration of the velocity cone to the origin and the expansion of the force cone to become equal to the plane.

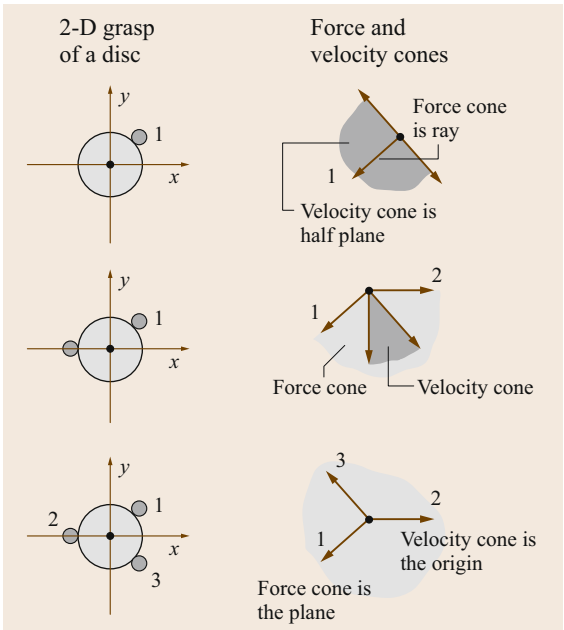


Fig. 38.17 Case of a translating disc in the plane: Relationship between frictionless contacts and possible disc velocities and net contact forces

It is important to point out that the discussion of the dual cones applies to three-dimensional bodies after replacing the contact normals with the columns of $\mathbf{G}$.

38.5.5 Example 5: Form Closure

Part 1: Unilateral Constraints

Form closure of a spatial object requires seven unilateral contacts, which is difficult to illustrate. Therefore, the only form closure example analyzed in this chapter is the planar problem shown in Fig. 38.18. In the plane only four unilateral contacts are needed. In this problem, even though the fourth contact is at a vertex of the object, the contact normal is still well-defined. The angle α of the finger is allowed to vary, and it can be shown that form closure exists if α lies in the interval: $1.0518 < \alpha < \frac{\pi}{2}$. Notice that a critical value of α occurs when the lower edge of C_2 contains contact point 3 ($\alpha \approx 1.0518$) and contact point 2 ($\alpha = \frac{\pi}{2}$). Beyond these angles, the cone C_1 and C_2 can no longer *see* each other.

Choosing the frame for analysis with origin at the fourth contact point, the grasp matrix for this example

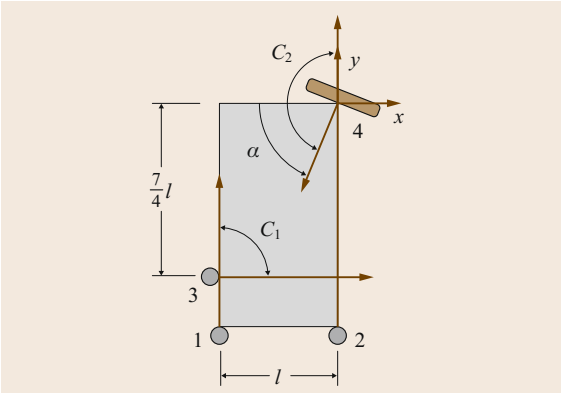


Fig. 38.18 Plot of closure metrics versus angle of contact for $1.04 < \alpha < 1.59$

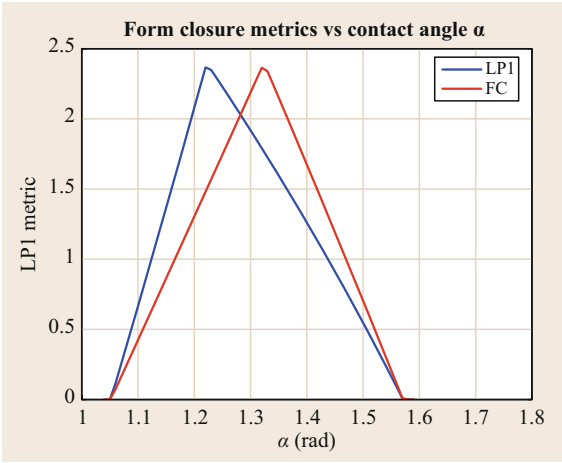


Fig. 38.19 Planar grasp with first-order form closure if $1.052 < \alpha < \frac{\pi}{2}$

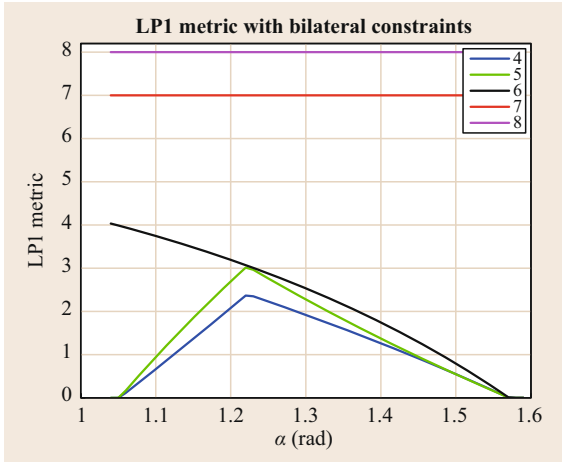


Fig. 38.20 First-order form closure metric for Fig. 38.18 with progressively more contacts converted to bilateral

is

$$\mathbf{G} = \begin{pmatrix} 0 & 0 & 1 & -\cos(\alpha) \\ 1 & 1 & 0 & -\sin(\alpha) \\ -l & 0 & \frac{7}{4}l & 0 \end{pmatrix}. \tag{38.97}$$

Form closure was tested for a range of angles. The blue curve in Fig. 38.19 is the LP1 metric ($d^* * n_c$ returned by the function `formClosure.m`, available from [38.43]), which indicates that the grasp farthest from losing form closure has $\alpha \approx 1.222$ radians, which is the configuration shown in Fig. 38.18. For comparison purposes, the Ferrari–Canny metric (scaled to have the same maximum as the LP1 metric) is plotted in red. They agree on the angles for which form closure exists, but differ on the optimal α.

Example 5, Part 2: Bilateral Constraints

Let us next consider form closure with a mix of bilateral and unilateral constraints using (38.54). For example, if contact 1 is treated as a bilateral constraint and the remaining are considered to be unilateral, $\mathbf{G}_n$ and $\mathbf{B}$ are

$$\mathbf{B} = \begin{pmatrix} 0 \\ 1 \\ -l \end{pmatrix}, \quad \mathbf{G}_n = \begin{pmatrix} 0 & 1 & -\cos(\alpha) \\ 1 & 0 & -\sin(\alpha) \\ 0 & \frac{7}{4}l & 0 \end{pmatrix}. \tag{38.98}$$

Additional contacts can be converted to bilateral by moving their corresponding columns from $\mathbf{G}_n$ to $\mathbf{B}$. Figure 38.20 shows five plots of the form closure metric LP1. In the legend, *4* refers to the grasp with all four contacts treated as unilateral, *5* means that contact 1 has been converted to bilateral as defined by the previous equation, *6* means that contacts 1 and 2 have been converted to bilateral, etc. The plots show an important property of the LP1 metric; the maximum value is equal to the number of unilateral contacts (bilateral contacts count as two unilateral contacts), which is attained for curves *7* and *8*. For this particular problem, these maxima are achieved for all α.

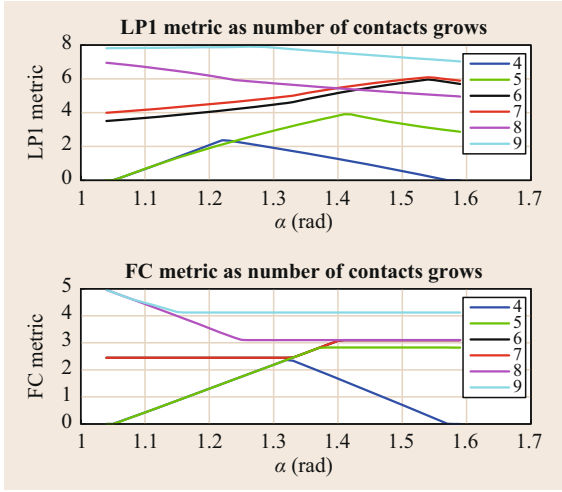


Fig. 38.21 Comparison of the monotonicity properties of the LP1 and Ferrari–Canny form closure metrics

Example 5, Part 3: Monotonicity

If one were to zoom in on curves *4* and *5* near $\alpha = 1.5$, one would see that LP1 is not monotonic as constraints are added. That is, when contact *1* is converted from unilateral (curve *4*) to bilateral (curve *5*), for some values of α, the LP1 metric is smaller despite the additional constraint. To demonstrate monotonicity, the monotonic Ferrari–Canny metric and the non-monotonic LP1

metric were computed as a function of α for a sequence of grasps. Starting with the original four contacts, five unilateral contacts were added: $\{(0, -1.3l), (-0.5l, 0), (0, 1.75l), (-l, -0.6l), (-0.7l, 0)\}$. Figure 38.21 shows the metrics with 4, 5, 6, 7, 8, and 9 contacts, as indicated by the legend.

The LP1 metric does not increase monotonically as contacts are added for every α. Near $\alpha = 1.2$ adding the fifth contact reduces the metric slightly. Also, near $\alpha = 1.55$, adding the eighth contact reduces the metric. By contrast, the Ferrari–Canny metric is monotonic, although not strictly monotonic; the plots labeled *6* and *7* are identical for all α and there are other intervals of α where the metric is constant as contacts are added. For example, near $\alpha = 1.35$, the grasps with 5, 6, and 7 contacts all have the same metric value.

38.6 Conclusion and Further Reading

A great deal of understanding of grasping systems can be derived from the simple linear kinematic, dynamic, and contact models presented in this chapter. The most widely used grasp classifications and closure properties can all be derived from these models under the rigid-body assumption. Linearizing these models leads to metrics and tests that can be computed efficiently using computational linear algebra and linear programming techniques. Grasp synthesis tools built on these tests take object and hand models as inputs and return a set of possible grasp configurations as outputs (see for example [38.3]). In-depth discussions of grasp kinematics and grasp classifications can be found in [38.15, 20, 20, 34, 48–52].

One has to wonder what insights have been lost as a result of the simplifying assumptions made in this chapter. For the interested reader, there are a host of papers that analyze grasping systems under more sophisticated assumptions. In general, bodies are curved and compliant [38.17, 28, 35, 53–55] and contact friction models are not quite as simple as the linearized ones so widely adopted. For example, if a contact has to resist a moment about its normal, its effective tangential friction coefficient is reduced [38.46, 56]. In this chapter, the quadratic Coulomb friction cone was approximated by a polyhedral cone. The analysis problems are more difficult when using the quadratic cone, but they are quite tractable [38.47, 57].

In principle, a properly designed grasping system could be controlled to maintain all contacts, but worldly realities can lead to unwanted slipping or twisting. This leads us back to the topic of grasp stability, which is too often equated to grasp closure. However, grasp closure is really equivalent to the existence of equilibrium, which is a necessary, but not sufficient, condition for stability. The common definition of stability outside of the field of robotic grasping requires that when a system is deflected from an equilibrium point, the system returns to this point. From this perspective, and under the assumption of no slipping at the contact points, it is known that all closure grasps are stable, but the converse is not necessarily true [38.53, 54, 58, 59]. However, stability analysis when contacts slide is still an open question.

Given a stable grasp, another important consideration not discussed in this chapter is that of grasp force distribution problem, i. e., finding *good* actuator torques and contact forces to balance a given external load applied to the object. This problem was studied in the context of walking machines first by *McGhee* and *Orin* [38.60] and later by several others [38.61, 62]. *Kumar* and *Waldron* applied similar techniques to force distribution problems in grasping [38.63]. Work by *Han* et al. and *Buss* et al. solved the force distribution problem with non-linear friction cone constraints by taking advantage of convex optimization techniques [38.47, 57, 64]. In power grasps, this problem of finding a good distribution is difficult, because the space of controllable contact wrenches is severely restricted by the large number of contacts [38.21, 32, 33, 65].

Grasp synthesis largely depends on the structure of the robotic hands that are often very complex systems with many degrees of freedom, sensors and actuators, which are necessary to adapt to many different objects and tasks. An important research area, not discussed in this chapter, is that of designing simplified hands coupling some of the degrees of freedom, reducing the number of effective inputs, and leading to more efficient, simpler and reliable designs [38.23, 66]. A reduction of independent inputs is observed also in human hand movement data, where few variables, defined as postural synergies, explain most of the variance in the configurations of the hand while grasping different objects [38.25, 67]. The reduction of the number of independently controlled inputs in the hand affects grasp properties, and in particular the ability of the hand to dexterously controlling grasp forces and in-hand manipulation as discussed in [38.68–71]

All of the above considerations implicitly assume that a grasp has been achieved, which is no easy task. The bulk of today's research in robotic grasping can be fairly characterized as focused on *grasp acquisition*.

In other words, the problem is to move the hand from a state of no contact with the object to one in which a satisfactory grasp has been achieved. When a robot identifies an object to grasp, its knowledge of the object's pose and geometry are not perfect. Even if they were, the robot's control system could not move the hand perfectly to the desired grasping points. The hand will bump the object accidentally, altering its pose, possibly leading to grasp failure.

Areas of current research cover methods that exploit some detailed aspects of planned or sensed contact interactions that occur before achieving the final grasp and those that try to be robust to pre-grasp contact interactions. In the first category are quasistatic push-grasping, dynamic grasp acquisition, and perception. Push-grasping seeks out contact prior to wrapping the fingers in order to allow the object to settle into a good position against the palm. It has, so far, been applied to objects that can slide stably across a horizontal surface cluttered with object that are not to be grasped [38.72]. Dynamic planning combines a dynamic model of grasping that includes intermittent contact to design optimal controllers and grasping actions simultaneously [38.73]. The performance of both of the above in real-time implementations could be improved by new grasp perception methods, which estimate the pose of the object relative to the hand [38.74–76].

Impedance controllers are being developed to reduce the negative effects of unexpected contacts between the hand and object as the grasp is being formed [38.77] and between the object and environment during object transport tasks [38.78]. Independent contact regions are surface patches on the object which have the property: if a contact point is stabilized anywhere inside each region, then the grasp will have force closure [38.79, 80]. With this approach, a small amount of jostling will not cause grasp failure. Perhaps caging takes the independent regions idea to the extreme. Here the goal is to find a configuration of the hand that loosely surrounds the object with the additional condition that the object cannot escape without deforming itself or the hand [38.81, 82]. The challenges in caging are in finding a pre-cage configuration and a finger motion plan that impose the least restrictive accuracy requirements on the robots perception and control systems.

Video-References

VIDEO 551 Grasp analysis using the MATLAB toolbox SynGrasp
available from http://handbookofrobotics.org/view-chapter/04/videodetails/551

References

38.1 A. Bicchi: Hands for dextrous manipulation and powerful grasping: A difficult road towards simplicity, IEEE Trans. Robotics Autom. **16**, 652–662 (2000)

38.2 J.K. Salisbury: Kinematic and Force Analysis of Articulated Hands, Ph.D. Thesis (Stanford University, Stanford 1982)

38.3 A.T. Miller, P.K. Allen: GraspIt! A versatile simulator for robotic grasping, IEEE Robotics Autom. Mag. **11**(4), 110–122 (2004)

38.4 M. Malvezzi, G. Gioioso, G. Salvietti, D. Prattichizzo: SynGrasp: A matlab toolbox for underactuated and compliant hands, IEEE Robotics Autom. Mag. **22**(4), 52–68 (2015)

38.5 M. Malvezzi, G. Gioioso, G. Salvietti, D. Prattichizzo, A. Bicchi: Syngrasp: A matlab toolbox for grasp analysis of human and robotic hands, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 1088–1093

38.6 SynGrasp: A MATLAB Toolbox for Grasp Analysis of Human and Robotic Hands, http://syngrasp.dii.unisi.it/

38.7 M. Grebenstein, A. Albu-Schäffer, T. Bahls, M. Chalon, O. Eiberger, W. Friedl, R. Gruber, S. Haddadin, U. Hagn, R. Haslinger, H. Hoppner, S. Jorg, M. Nickl, A. Nothhelfer, F. Petit, J. Reill, N. Seitz, T. Wimbock, S. Wolf, T. Wusthoff, G. Hirzinger: The DLR hand arm system, IEEE Conf. Robotics Autom. (2011) pp. 3175–3182

38.8 K. Salisbury, W. Townsend, B. Ebrman, D. DiPietro: Preliminary design of a whole-arm manipulation system (WAMS), Proc. IEEE Int. Conf. Robotics Autom. (1988) pp. 254–260

38.9 M.S. Ohwovoriole, B. Roth: An extension of screw theory, J. Mech. Des. **103**, 725–735 (1981)

38.10 K.H. Hunt: *Kinematic Geometry of Mechanisms* (Oxford Univ. Press, Oxford 1978)

38.11 T.R. Kane, D.A. Levinson, P.W. Likins: *Spacecraft Dynamics* (McGraw Hill, New York 1980)

38.12 J.J. Craig: *Introduction to Robotics: Mechanics and Control*, 2nd edn. (Addison-Wesley, Reading 1989)

38.13 J.K. Salisbury, B. Roth: Kinematic and force analysis of articulated mechanical hands, J. Mech. Transm. Autom. Des. **105**(1), 35–41 (1983)

38.14 M.T. Mason, J.K. Salisbury Jr: *Robot Hands and the Mechanics of Manipulation* (MIT Press, Cambridge 1985)
38.15 A. Bicchi: On the closure properties of robotic grasping, Int. J. Robotics Res. **14**(4), 319–334 (1995)
38.16 D. Prattichizzo, A. Bicchi: Consistent task specification for manipulation systems with general kinematics, ASME J. Dyn. Syst. Meas. Control **119**(4), 760–767 (1997)
38.17 J. Kerr, B. Roth: Analysis of multifingered hands, Int. J. Robotics Res. **4**(4), 3–17 (1986)
38.18 G. Strang: *Introduction to Linear Algebra* (Wellesley-Cambridge Press, Wellesley 1993)
38.19 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robot Manipulation* (CRC, Boca Raton 1993)
38.20 D. Prattichizzo, A. Bicchi: Dynamic analysis of mobility and graspability of general manipulation systems, IEEE Trans. Robotics Autom. **14**(2), 241–258 (1998)
38.21 A. Bicchi: On the problem of decomposing grasp and manipulation forces in multiple whole-limb manipulation, Int. J. Robotics Auton. Syst. **13**, 127–147 (1994)
38.22 A.M. Dollar, R.D. Howe: Joint coupling and actuation design of underactuated hands for unstructured environments, Int. J. Robotics Res. **30**, 1157–1169 (2011)
38.23 L. Birglen, T. Lalibertè, C. Gosselin: *Underactuated robotic hands*, Springer Tracts in Advanced Robotics (Springer, Berlin, Heidelberg 2008)
38.24 M.G. Catalano, G. Grioli, A. Serio, E. Farnioli, C. Piazza, A. Bicchi: Adaptive synergies for a humanoid robot hand, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012) pp. 7–14
38.25 M. Gabiccini, A. Bicchi, D. Prattichizzo, M. Malvezzi: On the role of hand synergies in the optimal choice of grasping forces, Auton. Robots **31**, 235–252 (2011)
38.26 M. Malvezzi, D. Prattichizzo: Evaluation of grasp stiffness in underactuated compliant hands, Proc. IEEE Int. Conf. Robotics Autom. (2013) pp. 2074–2079
38.27 S.F. Chen, I. Kao: Conservative congruence transformation for joint and cartesian stiffness matrices of robotic hands and fingers, Int. J. Robotics Res. **19**(9), 835–847 (2000)
38.28 M.R. Cutkosky, I. Kao: Computing and controlling the compliance of a robotic hand, IEEE Trans. Robotics Autom. **5**(2), 151–165 (1989)
38.29 A. Albu-Schaffer, O. Eiberger, M. Grebenstein, S. Haddadin, C. Ott, T. Wimbock, S. Wolf, G. Hirzinger: Soft robotics, IEEE Robotics Autom, Mag. **15**(3), 20–30 (2008)
38.30 A. Bicchi: Force distribution in multiple whole-limb manipulation, Proc. IEEE Int. Conf. Robotics Autom. (1993)
38.31 F. Reuleaux: *The Kinematics of Machinery* (Macmillan, New York 1876), Republished by Dover, New York, 1963
38.32 T. Omata, K. Nagata: Rigid body analysis of the indeterminate grasp force in power grasps, IEEE Trans. Robotics Autom. **16**(1), 46–54 (2000)
38.33 J.C. Trinkle: The Mechanics and Planning of Enveloping Grasps, Ph.D. Thesis (University of Pennsylvania, Department of Systems Engineering, 1987)
38.34 K. Lakshminarayana: *Mechanics of Form Closure*, Tech. Rep., Vol. 78-DET-32 (ASME, New York 1978)
38.35 E. Rimon, J. Burdick: Mobility of bodies in contact i: A 2nd order mobility index for multiple-finger grasps, IEEE Trans. Robotics Autom. **14**(5), 696–708 (1998)
38.36 B. Mishra, J.T. Schwartz, M. Sharir: On the existence and synthesis of multifinger positive grips, Algorithmica **2**(4), 541–558 (1987)
38.37 P. Somov: Über Schraubengeschwindigkeiten eines festen Körpers bei verschiedener Zahl von Stützflächen, Z. Math. Phys. **42**, 133–153 (1897)
38.38 P. Somov: Über Schraubengeschwindigkeiten eines festen Körpers bei verschiedener Zahl von Stützflächen, Z. Math. Phys. **42**, 161–182 (1897)
38.39 A.J. Goldman, A.W. Tucker: Polyhedral convex cones. In: *Linear Inequalities and Related Systems*, ed. by H.W. Kuhn, A.W. Tucker (Princeton Univ., York 1956) pp. 19–40
38.40 X. Markenscoff, L. Ni, C.H. Papadimitriou: The geometry of grasping, Int. J. Robotics Res. **9**(1), 61–74 (1990)
38.41 D.G. Luenberger: *Linear and Nonlinear Programming*, 2nd edn. (Addison-Wesley, Reading 1984)
38.42 G. Muscio. J.C. Trinkle: *Grasp Closure Analysis of Bilaterally Constrained Objects*, Tech. Rep. Ser., Vol. 13-01 (Department of Computer Science, Rensselear Polytechnic Institute, Troy 2013)
38.43 Rensselaer Computer Science: http://www.cs.rpi.edu/twiki/bin/view/RoboticsWeb/LabSoftware
38.44 C. Ferrari, J. Canny: Planning optimal grasps, Proc. IEEE Int. Conf. Robotics Autom. (1986) pp. 2290–2295
38.45 V.D. Nguyen: *The Synthesis of Force Closure Grasps in the Plane, M.S. Thesis Ser.* (MIT Department of Mechanical Engineering, Cambridge 1985), AI-TR861
38.46 R.D. Howe, M.R. Cutkosky: Practical force-motion models for sliding manipulation, Int. J. Robotics Res. **15**(6), 557–572 (1996)
38.47 L. Han, J.C. Trinkle, Z. Li: Grasp analysis as linear matrix inequality problems, IEEE Trans. Robotics Autom. **16**(6), 663–674 (2000)
38.48 J.C. Trinkle: On the stability and instantaneous velocity of grasped frictionless objects, IEEE Trans. Robotics Autom. **8**(5), 560–572 (1992)
38.49 K.H. Hunt, A.E. Samuel, P.R. McAree: Special configurations of multi-finger multi-freedom grippers – A kinematic study, Int. J. Robotics Res. **10**(2), 123–134 (1991)
38.50 D.J. Montana: The kinematics of multi-fingered manipulation, IEEE Trans. Robotics Autom. **11**(4), 491–503 (1995)
38.51 Y. Nakamura, K. Nagai, T. Yoshikawa: Dynamics and stability in coordination of multiple robotic mechanisms, Int. J. Robotics Res. **8**, 44–61 (1989)
38.52 J.S. Pang, J.C. Trinkle: Stability characterizations of rigid body contact problems with coulomb friction, Z. Angew. Math. Mech. **80**(10), 643–663 (2000)

38.53 M.R. Cutkosky: *Robotic Grasping and Fine Manipulation* (Kluwer, Norwell 1985)
38.54 W.S. Howard, V. Kumar: On the stability of grasped objects, IEEE Trans. Robotics Autom. **12**(6), 904–917 (1996)
38.55 A.B.A. Cole, J.E. Hauser, S.S. Sastry: Kinematics and control of multifingered hands with rolling contacts, IEEE Trans. Autom. Control **34**, 398–404 (1989)
38.56 R.I. Leine, C. Glocker: A set-valued force law for spatial Coulomb–Contensou friction, Eur. J. Mech. A **22**(2), 193–216 (2003)
38.57 M. Buss, H. Hashimoto, J. Moore: Dexterous hand grasping force optimization, IEEE Trans. Robotics Autom. **12**(3), 406–418 (1996)
38.58 V. Nguyen: Constructing force-closure grasps, Int. J. Robotics Res. **7**(3), 3–16 (1988)
38.59 E. Rimon, J.W. Burdick: Mobility of bodies in contact II: How forces are generated by curvature effects, Proc. IEEE Int. Conf. Robotics Autom. (1998) pp. 2336–2341
38.60 R.B. McGhee, D.E. Orin: A mathematical programming approach to control of positions and torques in legged locomotion systems, Proc. ROMANCY (1976)
38.61 K. Waldron: Force and motion management in legged locomotion, IEEE J. Robotics Autom. **2**(4), 214–220 (1986)
38.62 T. Yoshikawa, K. Nagai: Manipulating and grasping forces in manipulation by multi-fingered grippers, Proc. IEEE Int. Conf. Robotics Autom. (1987) pp. 1998–2007
38.63 V. Kumar, K. Waldron: Force distribution in closed kinematic chains, IEEE J. Robotics Autom. **4**(6), 657–664 (1988)
38.64 M. Buss, L. Faybusovich, J. Moore: Dikin-type algortihms for dexterous grasping force optimization, Int. J. Robotics Res. **17**(8), 831–839 (1998)
38.65 D. Prattichizzo, J.K. Salisbury, A. Bicchi: Contact and grasp robustness measures: Analysis and experiments. In: *Experimental Robotics-IV, Lecture Notes in Control and Information Sciences*, Vol. 223, ed. by O. Khatib, K. Salisbury (Springer, Berlin, Heidelberg 1997) pp. 83–90
38.66 A.M. Dollar, R.D. Howe: The highly adaptive sdm hand: Design and performance evaluation, Int. J. Robotics Res. **29**(5), 585–597 (2010)
38.67 M.G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, A. Bicchi: Adaptive synergies for the design and control of the Pisa/IIT SoftHand, Int. J. Robotics Res. **33**(5), 768–782 (2014)
38.68 M.T. Ciocarlie, P.K. Allen: Hand posture subspaces for dexterous robotic grasping, Int. J. Robotics Res. **28**(7), 851–867 (2009)
38.69 T. Wimbock, B. Jahn, G. Hirzinger: Synergy level impedance control for multifingered hands, IEEE/RSJ Int Conf Intell. Robots Syst. (IROS) (2011) pp. 973–979
38.70 D. Prattichizzo, M. Malvezzi, M. Gabiccini, A. Bicchi: On the manipulability ellipsoids of underactuated robotic hands with compliance, Robotics Auton. Syst. **60**(3), 337–346 (2012)
38.71 D. Prattichizzo, M. Malvezzi, M. Gabiccini, A. Bicchi: On motion and force controllability of precision grasps with hands actuated by soft synergies, IEEE Trans. Robotics **29**(6), 1440–1456 (2013)
38.72 M.R. Dogar, S.S. Srinivasa: A framework for push-grasping in clutter, Robotics Sci. Syst. (2011)
38.73 M. Posa, R. Tedrake: Direct trajectory optimization of rigid body dynamical systems through contact, Proc. Workshop Algorithm. Found. Robotics (2012)
38.74 L. Zhang, J.C. Trinkle: The application of particle filtering to grasp acquistion with visual occlusion and tactile sensing, Proc. IEEE Int. Conf Robotics Autom. (2012)
38.75 P. Hebert, N. Hudson, J. Ma, J. Burdick: Fusion of stereo vision, force-torque, and joint sensors for estimation of in-hand object location, Proc. IEEE Int. Conf. Robotics Autom. (2011) pp. 5935–5941
38.76 S. Haidacher, G. Hirzinger: Estimating finger contact location and object pose from contact measurements in 3nd grasping, Proc. IEEE Int. Conf. Robotics Autom. (2003) pp. 1805–1810
38.77 T. Schlegl, M. Buss, T. Omata, G. Schmidt: Fast dextrous re-grasping with optimal contact forces and contact sensor-based impedance control, Proc. IEEE Int. Conf. Robotics Autom. (2001) pp. 103–108
38.78 G. Muscio, F. Pierri, J.C. Trinkle: A hand/arm controller that simultaneously regulates internal grasp forces and the impedance of contacts with the environment, IEEE Conf. Robotics Autom. (2014)
38.79 M.A. Roa, R. Suarez: Computation of independent contact regions for grasping 3-D objects, IEEE Trans. Robotics **25**(4), 839–850 (2009)
38.80 M.A. Roa, R. Suarez: Influence of contact types and uncertainties in the computation of independent contact regions, Proc. IEEE Int. Conf. Robotics Autom. (2011) pp. 3317–3323
38.81 A. Rodriguez, M.T. Mason, S. Ferry: From caging to grasping, Int. J. Robotics Res. **31**(7), 886–900 (2012)
38.82 C. Davidson, A. Blake: Error-tolerant visual planning of planar grasp, 6th Int. Conf. Comput. Vis. (1998) pp. 911–916

Multimedia Contents

39. Cooperative Manipulation

Fabrizio Caccavale, Masaru Uchiyama

This chapter is devoted to cooperative manipulation of a common object by means of two or more robotic arms. The chapter opens with a historical overview of the research on cooperative manipulation, ranging from early 1970s to very recent years. Kinematics and dynamics of robotic arms cooperatively manipulating a tightly grasped rigid object are presented in depth. As for the kinematics and statics, the chosen approach is based on the so-called symmetric formulation; fundamentals of dynamics and reduced-order models for closed kinematic chains are discussed as well. A few special topics, such as the definition of geometrically meaningful cooperative task space variables, the problem of load distribution, and the definition of manipulability ellipsoids, are included to give the reader a complete picture of modeling and evaluation methodologies for cooperative manipulators. Then, the chapter presents the main strategies for controlling both the motion of the cooperative system and the interaction forces between the manipulators and the grasped object; in detail, fundamentals of hybrid force/position control, proportional–derivative (PD)-type force/position control schemes, feedback linearization techniques, and impedance control approaches are given. In the last section further reading on advanced topics related to control of cooperative robots is suggested; in detail, advanced nonlinear control strategies are briefly discussed (i. e., intelligent control approaches, synchronization control, decentralized control); also, fundamental results on modeling and control of cooperative systems possessing some degree of flexibility are briefly outlined.

39.1 **Historical Overview** 990
39.2 **Kinematics and Statics** 991
39.2.1 Symmetric Formulation 992
39.2.2 Multifingered Manipulation 994
39.3 **Cooperative Task Space** 995
39.4 **Dynamics and Load Distribution** 996
39.4.1 Reduced-Order Models 997
39.4.2 Load Distribution 997
39.5 **Task-Space Analysis** 998
39.6 **Control** 999
39.6.1 Hybrid Control 999
39.6.2 PD Force/Motion Control 1000
39.6.3 Feedback Linearization Approaches 1001
39.6.4 Impedance Control 1002
39.7 **Conclusions and Further Reading** 1003
Video-References 1004
References 1004

It was not long after the emergence of robotics technologies that multi-arm robot systems began to be investigated by robotics researchers. In the early 1970s, research on this topic had already started. This interest was mainly due to typical limitations in applications of single-arm robots. It has been recognized, in fact, that many tasks that are difficult or impossible to execute by a single robot become feasible when two or more manipulators are employed in a cooperative way. Such tasks include, for instance, carrying heavy or large payloads, the assembly of multiple parts without using special fixtures, and handling of objects that are flexible or possess extra degrees of freedom. Research on this subject has been aimed at solving existing problems and opening up a new stream of applications in flexible manufacturing systems as well as in poorly structured environments (e.g., outer space and undersea).

39.1 Historical Overview

Examples of research work in the early days include that by *Fujii* and *Kurono* [39.1], *Nakano* et al. [39.2], and *Takase* et al. [39.3]. Already in those pieces of work important key issues in the control of multi-arm robots were investigated: master/slave control, force/compliance control, and task-space control. In [39.1] *Fujii* and *Kurono* proposed to adopt a compliance control concept for the coordination of multiple manipulators; they defined a task vector with respect to the object frame and controlled the compliance expressed in the same coordinate frame. An interesting feature of the work by *Fujii* and *Kurono* [39.1] and by *Takase* et al. [39.3] is that force/compliance control was implemented by exploiting back-drivability of the actuators, without using force/torque sensors. The importance of this technique in practical applications, however, was not recognized at that time, and more complex approaches using force/torque sensors lured people in robotics. *Nakano* et al. [39.2, 4] proposed a master/slave force control approach for the coordination of two arms carrying an object cooperatively and pointed out the necessity of force control for cooperative multiple robots.

In the 1980s, based on several fundamental theoretical results for single-arm robots, strong research on multi-arm robotic systems was renewed [39.5]. Definition of task vectors with respect to the object to be handled [39.6], dynamics and control of the closed kinematic chain formed by the multi-arm robot and the object [39.7, 8], and force control issues, such as hybrid position/force control [39.9–12], were explored. Through this research work, a strong theoretical background for the control of multi-arm robots has been formed, providing the basis for research on more advanced topics from the 1990s to today.

How to parameterize the constraint forces/moments on the object based on the dynamic model of the whole cooperative system has been recognized as a critical issue; in fact, this parametrization leads to the definition of task variables for the control and hence to an answer to one of the most frequently asked questions in the field of multi-arm robotics: how to control simultaneously the trajectory of the object, the mechanical stresses (internal forces/moments) acting on the object, load sharing among the arms, and even the external forces/moments on the object. Force decomposition may be a key to solving these problems and has been studied by *Uchiyama* and *Dauchez* [39.11, 12] and *Walker* et al. [39.13] as well as *Bonitz* and *Hsia* [39.14]. In detail, devising a geometrically clear parametrization of the internal forces/moments acting on the object has been recognized as an important problem, solved in [39.15, 16]. Several cooperative control schemes based on the sought parameterizations have been designed, including control of motion and force [39.11, 12, 17–19] and impedance/compliance control [39.20–22]. Other approaches include adaptive control [39.23, 24], kinematic control [39.25], task-space regulation [39.26], joint-space control [39.27, 28], and coordinated control [39.29].

Also, the definition of user-oriented task-space variables for coordinated control [39.26, 30] and the development of meaningful performance measures [39.31–34] have been fruitfully investigated in 1990s.

Load sharing among the arms is also an interesting issue on which many papers have been published [39.35–40]. Load sharing may be exploited both for optimal load distribution among arms and for robust holding of the object, when the object is held by the arms without being grasped rigidly. In both cases, anyhow, this becomes a problem of optimization and can be solved by either heuristic [39.41] or mathematical methods [39.42].

Other research efforts have been focused on cooperative handling of multibodied or even flexible objects [39.43–45]. Control of multi-flexible-arm robots has been investigated [39.46–48], since the merits of flexible-arm robots can be exploited in cooperative systems [39.49], i. e., lightweight structure, intrinsic compliance and hence safety, etc.

Robust holding of an object in the presence of slippage of end-effectors on the object may be achieved as well, if the slippage is detected correctly [39.50].

A more recent control framework for cooperative systems is so-called synchronization control [39.51, 52]; in this class of approaches the control problem is formulated in terms of suitably defined errors accounting for the motion synchronization between the manipulators involved in the cooperative task. As for the nonlinear control of cooperative manipulation systems, efforts have been spent on intelligent control (e.g., [39.53–55]) as well as on the investigation of control strategies in the presence of partial state feedback [39.56]. The problem of controlling underactuated cooperative manipulators is tackled, e.g., in [39.57].

The problem of the implementation of cooperative control strategies on conventional industrial robots has attracted the increasing interest of the research community. In fact, the control units of industrial robots do not present all the features needed to implement nonlinear torque control schemes, while the integration of force/torque sensing in standard industrial robot control units is often cumbersome and tends to be avoided in in-

dustry for many reasons: unreliability, cost, etc. Hence, the rebirth of the early methods, where force sensors were not used (*Fujii* and *Kurono* [39.1], *Inoue* [39.58]), has become attractive for industrial settings. Hybrid position/force control without using force/torque sensors has been successfully implemented [39.59]. Interesting results on the implementation of effective cooperative control strategies for industrial robots are presented, e.g., in [39.60], where a design approach that makes use of tool-based coordinate systems, trajectory generation, and distributed control of multiple robots is presented. Also, a task planning approach for multi-arm industrial robotic work-cells has been recently proposed in [39.30], together with a language for cooperative task programming.

Another interesting aspect, related to the reliability and safety of cooperative manipulation systems, is investigated in [39.61], where the use of nonrigid grippers is considered to avoid large internal forces and, at the same time, to achieve safe manipulation of the object, even in the presence of failures and unpredicted contacts with the external environment.

It is worth mentioning the strict relationship between issues related to grasping of objects by fingers/hands (widely described in Chap. 38) and those related to cooperative manipulation. In fact, in both cases, multiple manipulation structures grasp a commonly manipulated object. In multifingered hands only some motion components are transmitted through the contact point to the manipulated object (unilateral constraints), while cooperative manipulation via robotic arms is achieved by rigid (or near-rigid) grasp points and interaction takes place by transmitting all the motion components through the grasping points (bilateral constraints). While many common problems between the two fields can be tackled in a conceptually similar way (e.g., kinetostatic modeling, force control), many other are specific of each of the two application fields (e.g., form and force closure for multifingered hands). Interestingly, a common frame for both cooperative and multifingered manipulation has been proposed [39.25]; namely rolling/sliding of the grasp points on the object is taken into account by modeling the contacts via rotational/prismatic joints; then, the desired manipulator/finger joints trajectories are derived from the desired object motion by using numerical inverse kinematics algorithms. In [39.62–64], contributions to modeling and control of cooperative manipulation systems in the presence of different types of manipulators/object contact can be found. Moreover, since cooperative manipulators can be often viewed as closed-chain mechanisms, a strong link with research related to parallel manipulators (Chap. 18) exists, especially from a modeling point of view.

Finally, it is worth mentioning the important issue of cooperative transportation and manipulation of objects via multiple mobile robotic system [39.65, 66], which is currently subject of investigation.

39.2 Kinematics and Statics

Consider a system composed by M manipulators, each equipped with N_i joints ($i = 1, \ldots, M$). Let $\boldsymbol{p}_i$ denote the (3×1) vector of the position of the i-th end-effector coordinate frame, $\mathcal{T}_i$, with respect to a common base frame, $\mathcal{T}$; let $\mathbf{R}_i$ denote the (3×3) matrix which expresses the orientation of $\mathcal{T}_i$ with respect to the base frame $\mathcal{T}$.

Both $\boldsymbol{p}_i$ and $\mathbf{R}_i$ can be expressed as a function of the $(N_i \times 1)$ vector of the joints variables of each manipulator $\boldsymbol{q}_i$ through the direct kinematics equations

$$\begin{cases} \boldsymbol{p}_i = \boldsymbol{p}_i(\boldsymbol{q}_i) \,, \\ \mathbf{R}_i = \mathbf{R}_i(\boldsymbol{q}_i) \,. \end{cases} \tag{39.1}$$

Of course, the orientation of the end-effector frame may be expresses in terms of a minimal set of angles, i. e., a set of three Euler angles $\boldsymbol{\phi}_i$. In this case, direct kinematics can be expressed in terms of an operational space (pseudo-)vector $\boldsymbol{x}_i$ as follows

$$\boldsymbol{x}_i = \boldsymbol{k}_i(\boldsymbol{q}_i) = \begin{pmatrix} \boldsymbol{p}_i(\boldsymbol{q}_i) \\ \boldsymbol{\phi}_i(\boldsymbol{q}_i) \end{pmatrix} \,. \tag{39.2}$$

The linear $\dot{\boldsymbol{p}}_i$ and angular $\boldsymbol{\omega}_i$ velocity of the i-th end-effector can be collected in the (6×1) generalized velocities vector $\boldsymbol{v}_i = \left(\dot{\boldsymbol{p}}_i^{\mathrm{T}} \; \boldsymbol{\omega}_i^{\mathrm{T}}\right)^{\mathrm{T}}$. Then, the differential direct kinematics can be expressed as

$$\boldsymbol{v}_i = \mathbf{J}_i(\boldsymbol{q}_i)\dot{\boldsymbol{q}}_i \,, \tag{39.3}$$

where the $(6 \times N_i)$ matrix $\mathbf{J}_i$ is the so-called *geometric Jacobian* of the i-th manipulator (Chap. 2). When the velocity is expressed as the derivative of the operational space vector, differential kinematics takes a formally similar form

$$\dot{\boldsymbol{x}}_i = \frac{\partial \boldsymbol{k}_i(\boldsymbol{q}_i)}{\partial \boldsymbol{q}_i}\dot{\boldsymbol{q}}_i = \mathbf{J}_{\mathrm{A}i}(\boldsymbol{q}_i)\dot{\boldsymbol{q}}_i \,, \tag{39.4}$$

where the $(6 \times N_i)$ matrix $\mathbf{J}_{\mathrm{A}i}$ is the so-called *analytical Jacobian* of the i-th manipulator (Chap. 2).

Let us consider the (6×1) vector of the generalized forces acting at the i-th end-effector

$$\boldsymbol{h}_i = \begin{pmatrix} \boldsymbol{f}_i \\ \boldsymbol{n}_i \end{pmatrix} \,, \tag{39.5}$$

where $\boldsymbol{f}_i$ and $\boldsymbol{n}_i$ denote the force and moment, respectively. By invoking the principle of virtual work, a relation dual to (39.3) can be derived

$$\boldsymbol{\tau}_i = \mathbf{J}_i^{\mathrm{T}}(\boldsymbol{q}_i)\boldsymbol{h}_i \,, \tag{39.6}$$

where $\boldsymbol{\tau}_i$ is the $(N_i \times 1)$ vector of the forces/torques acting at the joints of the i-th manipulator.

For the sake of simplicity, consider a system of two cooperative robots (Fig. 39.1) manipulating a common object. Let C be a fixed point of the object (e.g., the center of mass of the object), whose position in the base frame is given by the vector $\boldsymbol{p}_{\mathrm{C}}$; moreover, let $\mathcal{T}_{\mathrm{C}}$ be a coordinate frame attached to object with origin in C. The *virtual stick* [39.11, 12] is defined as the vector $\boldsymbol{r}_i$ $(i = 1, 2)$ which determines the position of $\mathcal{T}_{\mathrm{C}}$ with respect to $\mathcal{T}_i$ $(i = 1, 2)$; it is assumed that $\boldsymbol{r}_i$ behaves as a rigid stick fixed to the i-th end-effector. Hence, each virtual stick, expressed in the frame $\mathcal{T}_i$ (or $\mathcal{T}_{\mathrm{C}}$), is a constant vector if the commonly grasped object is rigid and tightly grasped by each manipulator. In this case, the direct kinematics of each manipulator can be expressed in terms of a virtual end-effector frame $\mathcal{T}_{\mathrm{S},i} = \mathcal{T}_{\mathrm{C}}$, having the same orientation of $\mathcal{T}_{\mathrm{C}}$ and origin located in $\boldsymbol{p}_{\mathrm{S},i} = \boldsymbol{p}_i + \boldsymbol{r}_i = \boldsymbol{p}_{\mathrm{C}}$. Hence, the position and orientation at the tip of each virtual stick are given by $(i = 1, 2)$

$$\boldsymbol{p}_{\mathrm{S},i} = \boldsymbol{p}_{\mathrm{C}} \,, \quad \mathbf{R}_{\mathrm{S},i} = \mathbf{R}_{\mathrm{C}} \,.$$

The set of Euler angles corresponding to $\mathbf{R}_{\mathrm{S},i}$ will be denoted by the (3×1) vector $\boldsymbol{\phi}_{\mathrm{S},i}$. Hereafter, it is assumed that the object grasped by the two manipulators can be considered rigid (or nearly rigid) and tightly (or nearly tightly) attached to each end-effector. Hence, the displacements between the above defined frames can be considered null or negligible. Otherwise, if the manipulated object is subject to deformation (e.g., in the case of a flexible object) and/or the grasp is not tight (e.g., in the case of compliant grippers) the above defined coordinate frames, will be subject to nonnegligible displacements with respect to each other.

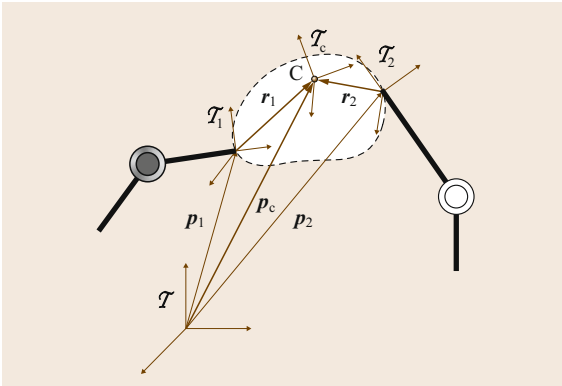


Fig. 39.1 Grasp geometry for a two-manipulator cooperative system manipulating a common object

Let $\boldsymbol{h}_{\mathrm{S},i}$ denote the vector of generalized forces acting at the tip of the i-th virtual stick; it can be easily verified that the following equation holds

$$\boldsymbol{h}_{\mathrm{S},i} = \begin{pmatrix} \mathbf{I}_3 & \mathbf{O}_3 \\ -\mathbf{S}(\boldsymbol{r}_i) & \mathbf{I}_3 \end{pmatrix} \boldsymbol{h}_i = \mathbf{W}_i \boldsymbol{h}_i \,, \tag{39.7}$$

where $\mathbf{O}_l$ and $\mathbf{I}_l$ denote, respectively, the null matrix and the identity matrix of $(l \times l)$ dimensions, and $\mathbf{S}(\boldsymbol{r}_i)$ is the (3×3) skew-symmetric matrix operator performing the cross product. It is worth noticing that $\mathbf{W}_i$ is always full rank.

By invoking the virtual works principle, a relation dual to (39.7) can easily be derived

$$\boldsymbol{v}_i = \begin{pmatrix} \mathbf{I}_3 & \mathbf{S}(\boldsymbol{r}_i) \\ \mathbf{O}_3 & \mathbf{I}_3 \end{pmatrix} \boldsymbol{v}_{\mathrm{S},i} = \mathbf{W}_i^{\mathrm{T}} \boldsymbol{v}_{\mathrm{S},i} \,, \tag{39.8}$$

where $\boldsymbol{v}_{\mathrm{S},i}$ is the generalized velocity vector of the virtual stick endpoint. When $\boldsymbol{r}_i = \mathbf{0}$, it is $\mathbf{W}_i = \mathbf{I}_6$; in other words, if the end-effector kinematics of each manipulator is referred to the corresponding virtual stick (or the object reduces to point), the forces and velocities at the two end-effectors coincide with their counterparts referred at the virtual sticks.

39.2.1 Symmetric Formulation

The kinetostatic formulation proposed by *Uchiyama* and *Dauchez* [39.12], i. e., the so-called *symmetric formulation*, is based on kinematic and static relationships between generalized forces/velocities acting at the object and their counterparts acting at the manipulators end-effectors (or at the virtual sticks endpoints).

Let us define first the *external forces* as the (6×1) vector of generalized forces given by

$$\boldsymbol{h}_{\mathrm{E}} = \boldsymbol{h}_{\mathrm{S},1} + \boldsymbol{h}_{\mathrm{S},2} = \mathbf{W}_{\mathrm{S}} \boldsymbol{h}_{\mathrm{S}} \,, \tag{39.9}$$

with $\mathbf{W}_{\mathrm{S}} = (\mathbf{I}_6 \; \mathbf{I}_6)$ and $\boldsymbol{h}_{\mathrm{S}} = \left(\boldsymbol{h}_{\mathrm{S},1}^{\mathrm{T}} \; \boldsymbol{h}_{\mathrm{S},2}^{\mathrm{T}}\right)^{\mathrm{T}}$; in other words, $\boldsymbol{h}_{\mathrm{E}}$ represents the vector of generalized forces causing the object's motion. From (39.7) and (39.9) it follows that $\boldsymbol{h}_{\mathrm{E}}$ can be expressed in terms of end-effector forces as well

$$\boldsymbol{h}_{\mathrm{E}} = \mathbf{W}_1 \boldsymbol{h}_1 + \mathbf{W}_2 \boldsymbol{h}_2 = \mathbf{W}\boldsymbol{h} \,, \tag{39.10}$$

with $\mathbf{W} = (\mathbf{W}_1 \; \mathbf{W}_2)$ and $\boldsymbol{h} = \left(\boldsymbol{h}_1^{\mathrm{T}} \; \boldsymbol{h}_2^{\mathrm{T}}\right)^{\mathrm{T}}$. It can be recognized that $\mathbf{W}_{\mathrm{S}}$ $(\mathbf{W})$ is a (6×12) matrix, having a six-dimensional (6-D) range space and a six-dimensional

null space, describing the geometry of the grasp, and thus is usually termed the *grasp matrix*.

The inverse solution to (39.9) yields

$$\boldsymbol{h}_{\mathrm{S}} = \mathbf{W}_{\mathrm{S}}^{\dagger}\,\boldsymbol{h}_{\mathrm{E}} + \mathbf{V}_{\mathrm{S}}\,\boldsymbol{h}_{\mathrm{I}} = \mathbf{U}_{\mathrm{S}}\,\boldsymbol{h}_{\mathrm{O}}\,, \tag{39.11}$$

where $\mathbf{W}_{\mathrm{S}}^{\dagger}$ denotes the Moore–Penrose pseudoinverse of $\mathbf{W}_{\mathrm{S}}$

$$\mathbf{W}_{\mathrm{S}}^{\dagger} = \frac{1}{2}\begin{pmatrix}\mathbf{I}_6\\ \mathbf{I}_6\end{pmatrix}. \tag{39.12}$$

$\mathbf{V}_{\mathrm{S}}$ is a matrix whose columns are a basis of the null space of $\mathbf{W}_{\mathrm{S}}$, e.g.,

$$\mathbf{V}_{\mathrm{S}} = \begin{pmatrix}-\mathbf{I}_6\\ \mathbf{I}_6\end{pmatrix}, \tag{39.13}$$

$\boldsymbol{h}_{\mathrm{O}} = \left(\boldsymbol{h}_{\mathrm{E}}^{\mathrm{T}}\,\boldsymbol{h}_{\mathrm{I}}^{\mathrm{T}}\right)^{\mathrm{T}}$, and

$$\mathbf{U}_{\mathrm{S}} = \left(\mathbf{W}_{\mathrm{S}}^{\dagger}\,\mathbf{V}_{\mathrm{S}}\right). \tag{39.14}$$

The second term on the right-hand side of (39.11), i. e., $\mathbf{V}_{\mathrm{S}}\boldsymbol{h}_{\mathrm{I}}$, represents a vector of generalized forces, referred at the tip of the virtual sticks, which lies in the null space of $\mathbf{W}_{\mathrm{S}}$; thus, such forces do not contribute to external forces. Hence, the (6×1) vector $\boldsymbol{h}_{\mathrm{I}}$ is a generalized force which does not contribute to the object's motion. Therefore, it represents internal loading of the object (i. e., mechanical stresses) and is termed the *internal forces* vector [39.12]. A similar argument can be used for equation (39.10), whose inverse solution is

$$\boldsymbol{h} = \mathbf{W}^{\dagger}\boldsymbol{h}_{\mathrm{E}} + \mathbf{V}\boldsymbol{h}_{\mathrm{I}} = \mathbf{U}\boldsymbol{h}_{\mathrm{O}}\,, \tag{39.15}$$

where

$$\mathbf{U} = (\mathbf{W}^{\dagger}\,\mathbf{V}). \tag{39.16}$$

In [39.13] it has been shown that the first term on the right-hand side of (39.15) represents only contributions to external forces if the pseudoinverse of the grasp matrix is properly defined, i. e.,

$$\mathbf{W}^{\dagger} = \begin{pmatrix}\frac{1}{2}\mathbf{I}_3 & \mathbf{O}_3\\ \frac{1}{2}\mathbf{S}(\boldsymbol{r}_1) & \frac{1}{2}\mathbf{I}_3\\ \frac{1}{2}\mathbf{I}_3 & \mathbf{O}_3\\ \frac{1}{2}\mathbf{S}(\boldsymbol{r}_2) & \frac{1}{2}\mathbf{I}_3\end{pmatrix}. \tag{39.17}$$

As $\mathbf{V}_{\mathrm{S}}$ in (39.11), the columns of $\mathbf{V}$ span the null space of $\mathbf{W}$, and can be chosen as [39.31]

$$\mathbf{V} = \begin{pmatrix}-\mathbf{I}_3 & \mathbf{O}_3\\ -\mathbf{S}(\boldsymbol{r}_1) & -\mathbf{I}_3\\ \mathbf{I}_3 & \mathbf{O}_3\\ \mathbf{S}(\boldsymbol{r}_2) & \mathbf{I}_3\end{pmatrix}. \tag{39.18}$$

A different parametrization of the inverse solutions to (39.9) and (39.10) can be expressed, respectively, as

$$\boldsymbol{h}_{\mathrm{S}} = \mathbf{W}_{\mathrm{S}}^{\dagger}\boldsymbol{h}_{\mathrm{E}} + \left(\mathbf{I}_{12} - \mathbf{W}_{\mathrm{S}}^{\dagger}\mathbf{W}_{\mathrm{S}}\right)\boldsymbol{h}_{\mathrm{S}}^{\star} \tag{39.19}$$

and

$$\boldsymbol{h} = \mathbf{W}^{\dagger}\boldsymbol{h}_{\mathrm{E}} + \left(\mathbf{I}_{12} - \mathbf{W}^{\dagger}\mathbf{W}\right)\boldsymbol{h}^{\star}\,, \tag{39.20}$$

where $\boldsymbol{h}_{\mathrm{S}}^{\star}$ ($\boldsymbol{h}^{\star}$) is an arbitrary (12×1) vector of generalized forces acting at the tip of the i-th virtual stick (i-th end-effector) projected onto the null space of $\mathbf{W}_{\mathrm{S}}$ ($\mathbf{W}$) via $\mathbf{I}_{12} - \mathbf{W}_{\mathrm{S}}^{\dagger}\mathbf{W}_{\mathrm{S}}$ ($\mathbf{I}_{12} - \mathbf{W}^{\dagger}\mathbf{W}$).

By exploiting the principle of virtual work, the mappings between generalized velocities, dual to those derived above, can be established. In detail, the mapping dual to (39.11) is

$$\boldsymbol{v}_{\mathrm{O}} = \mathbf{U}_{\mathrm{S}}^{\mathrm{T}}\,\boldsymbol{v}_{\mathrm{S}}\,, \tag{39.21}$$

where $\boldsymbol{v}_{\mathrm{S}} = \left(\boldsymbol{v}_{\mathrm{S},1}^{\mathrm{T}}\,\boldsymbol{v}_{\mathrm{S},2}^{\mathrm{T}}\right)^{\mathrm{T}}$, $\boldsymbol{v}_{\mathrm{O}} = \left(\boldsymbol{v}_{\mathrm{E}}^{\mathrm{T}}\,\boldsymbol{v}_{\mathrm{I}}^{\mathrm{T}}\right)^{\mathrm{T}}$. The vector $\boldsymbol{v}_{\mathrm{E}}$ can be interpreted as the absolute velocity of the object, while $\boldsymbol{v}_{\mathrm{I}}$ represents the relative velocity between the two coordinate frames $\mathcal{T}_{\mathrm{S},1}$ and $\mathcal{T}_{\mathrm{S},2}$ attached to the tips of the virtual sticks [39.12]; this vector is null when the manipulated object is rigid and rigidly grasped. In a similar way, from (39.15), the following mapping can be devised

$$\boldsymbol{v}_{\mathrm{O}} = \mathbf{U}^{\mathrm{T}}\,\boldsymbol{v}\,, \tag{39.22}$$

where $\boldsymbol{v} = \left(\boldsymbol{v}_1^{\mathrm{T}}\,\boldsymbol{v}_2^{\mathrm{T}}\right)^{\mathrm{T}}$.

Moreover, a set of position and orientation variables corresponding to $\boldsymbol{v}_{\mathrm{E}}$ and $\boldsymbol{v}_{\mathrm{I}}$ can be defined as follows [39.12, 25]

$$\boldsymbol{p}_{\mathrm{E}} = \frac{1}{2}(\boldsymbol{p}_{\mathrm{S},1} + \boldsymbol{p}_{\mathrm{S},2})\,, \quad \boldsymbol{p}_{\mathrm{I}} = \boldsymbol{p}_{\mathrm{S},2} - \boldsymbol{p}_{\mathrm{S},1}\,, \tag{39.23}$$

$$\mathbf{R}_{\mathrm{E}} = \mathbf{R}_1\mathbf{R}^1(\boldsymbol{k}_{21}^1, \vartheta_{21}/2)\,, \quad \mathbf{R}_{\mathrm{I}}^1 = \mathbf{R}_2^1\,, \tag{39.24}$$

where $\mathbf{R}_2^1 = \mathbf{R}_1^{\mathrm{T}}\mathbf{R}_2$ is the matrix expressing the orientation of $\mathcal{T}_2$ with respect to the axes of $\mathcal{T}_1$, while $\boldsymbol{k}_{21}^1$ and ϑ_{21} are, respectively, the equivalent unit vector (expressed with respect to $\mathcal{T}_1$) and rotation angle that determine the mutual orientation expressed by $\mathbf{R}_2^1$. Hence, $\mathbf{R}_{\mathrm{E}}$ expresses a rotation about $\boldsymbol{k}_{21}^1$ by an angle which is half the angle needed to align $\mathcal{T}_2$ with $\mathcal{T}_1$.

In turn, if the orientation variables are expressed in terms of Euler angles, a set of operational space variables can be defined as

$$\boldsymbol{x}_{\mathrm{E}} = \begin{pmatrix}\boldsymbol{p}_{\mathrm{E}}\\ \boldsymbol{\phi}_{\mathrm{E}}\end{pmatrix}, \quad \boldsymbol{x}_{\mathrm{I}} = \begin{pmatrix}\boldsymbol{p}_{\mathrm{I}}\\ \boldsymbol{\phi}_{\mathrm{I}}\end{pmatrix}, \tag{39.25}$$

where

$$\boldsymbol{\phi}_{\mathrm{E}} = \frac{1}{2}(\boldsymbol{\phi}_{\mathrm{S},1} + \boldsymbol{\phi}_{\mathrm{S},2}) \,, \quad \boldsymbol{\phi}_{\mathrm{I}} = \boldsymbol{\phi}_{\mathrm{S},2} - \boldsymbol{\phi}_{\mathrm{S},1} \,. \tag{39.26}$$

It must be remarked, however, that the definitions in (39.26) keep a clear geometric meaning only if the orientation displacements between the virtual stick frames are small. In this case, as shown in [39.11, 12], the corresponding operational space velocities, $\dot{\boldsymbol{x}}_{\mathrm{E}}$ and $\dot{\boldsymbol{x}}_{\mathrm{I}}$, correspond to $\boldsymbol{v}_{\mathrm{E}}$ and $\boldsymbol{v}_{\mathrm{I}}$, respectively, with good approximation. Otherwise, if the orientation displacements become large, the variables defined in (39.26) do not represent any meaningful quantities, and other orientation representations have to be adopted, e.g., the unit quaternion (see Chap. 2 on kinematics for a general introduction to quaternions and Sect. 39.3 for an application to cooperative robots kinematics).

Finally, by using (39.15) and (39.22), together with equations (39.3) and (39.6), the kinetostatic mappings between the force/velocities at the object and their counterparts in the joint space of the manipulators can be given

$$\boldsymbol{\tau} = \mathbf{J}_{\mathrm{O}}^{\mathrm{T}} \boldsymbol{h}_{\mathrm{O}} \,, \tag{39.27}$$

$$\boldsymbol{v}_{\mathrm{O}} = \mathbf{J}_{\mathrm{O}} \dot{\boldsymbol{q}} \,, \tag{39.28}$$

where $\boldsymbol{\tau} = \left(\boldsymbol{\tau}_1^{\mathrm{T}} \; \boldsymbol{\tau}_2^{\mathrm{T}}\right)^{\mathrm{T}}$, $\boldsymbol{q} = \left(\boldsymbol{q}_1^{\mathrm{T}} \; \boldsymbol{q}_2^{\mathrm{T}}\right)^{\mathrm{T}}$, and

$$\mathbf{J}_{\mathrm{O}} = \mathbf{U}^{\mathrm{T}} \mathbf{J} \,, \quad \mathbf{J} = \begin{pmatrix} \mathbf{J}_1 & \mathbf{O}_6 \\ \mathbf{O}_6 & \mathbf{J}_2 \end{pmatrix} \,. \tag{39.29}$$

Formally similar mappings can be established in terms of operational space velocities $\dot{\boldsymbol{x}}_{\mathrm{E}}$ and $\dot{\boldsymbol{x}}_{\mathrm{I}}$ [39.11, 12] and the corresponding operational space forces/moments.

39.2.2 Multifingered Manipulation

Hereafter some connections between the field of multi-arm cooperative manipulation, described in this chapter, and multifingered manipulation, described in Chap. 38, are briefly outlined, focusing on the kinetostatics of the two classes of manipulation systems.

Both in the case of multi-arm cooperative systems and of multifingered manipulation, two or more manipulation structures grasp a commonly manipulated object.

Cooperative manipulation via multiple robotic arms is achieved by rigidly grasping the object (e.g., via rigid fixtures), and thus interaction takes place by exchanging both forces and moments at the grasp points. In other words, all the translational and rotational motion components are transmitted through the grasp points.

On the other side, when a multifingered hand manipulates an object, only some components of the motion are transmitted through the contact points. This is effectively modeled via properly defined constraint matrices, whose expression depends on the type of contact. In other words, constraint matrices act as filters selecting the components of the motion transmitted through each contact. In fact, as shown in Chap. 38, it is convenient to think of the object–finger contact point as twofold: a point on the tip of the finger and a point on the object. Hence, two generalized velocity vectors are defined for the i-th contact point (both expressed with respect to frame $\mathcal{T}_i$): the velocity of the contact point on the hand, $\boldsymbol{v}_{\mathrm{h},i}^{i}$, and the velocity of the contact point on the object $\boldsymbol{v}_{\mathrm{o},i}^{i}$. The corresponding dual generalized force vectors are $\boldsymbol{h}_{\mathrm{h},i}^{i}$ and $\boldsymbol{h}_{\mathrm{o},i}^{i}$, respectively. A contact model is then defined via the $(m_i \times 6)$ constraint matrix $\mathbf{H}_i$, selecting m_i velocity components transmitted through the contact ($\boldsymbol{v}_{\mathrm{t},i}^{i}$), i. e.,

$$\boldsymbol{v}_{\mathrm{t},i}^{i} = \mathbf{H}_i \boldsymbol{v}_{\mathrm{h},i}^{i} = \mathbf{H}_i \boldsymbol{v}_{\mathrm{o},i}^{i} \,. \tag{39.30}$$

The relation dual to (39.30) is

$$\mathbf{H}_i^{\mathrm{T}} \boldsymbol{h}_{\mathrm{t},i}^{i} = \boldsymbol{h}_{\mathrm{h},i}^{i} = \boldsymbol{h}_{\mathrm{o},i}^{i} \,, \tag{39.31}$$

where $\boldsymbol{h}_{\mathrm{t},i}^{i}$ is the vector of the transmitted generalized forces. Hence, (39.10) can be rewritten as

$$\boldsymbol{h}_{\mathrm{E}} = \mathbf{W}_1 \overline{\mathbf{R}}_1 \mathbf{H}_1^{\mathrm{T}} \boldsymbol{h}_{\mathrm{t},1}^{1} + \mathbf{W}_2 \overline{\mathbf{R}}_2 \mathbf{H}_2^{\mathrm{T}} \boldsymbol{h}_{\mathrm{t},2}^{2} \,, \tag{39.32}$$

where $\overline{\mathbf{R}}_i = \mathrm{diag}\{\mathbf{R}_i, \mathbf{R}_i\}$. Hence, a conceptually similar kinetostatic analysis can be developed, leading to the concept of external and internal forces (and related kinematic quantities) as well.

However, it must be remarked that, while in the case of cooperative manipulators, tightly grasping the object, internal stresses are usually undesirable effects (unless controlled squeezing of a deformable object has to be achieved), in multifingered hands suitably controlled internal forces are useful to guarantee a firm grasp even in the presence of external loading applied to the object (see the problem of form and force closure discussed in Chap. 38).

39.3 Cooperative Task Space

The symmetric formulation reviewed in Sect. 39.2.1 defines the fundamental variables and the primary relationships needed to describe the kinematics and the statics of the cooperative manipulators. However, the symmetric formulation was not originally conceived for task and motion planning of cooperative robotic systems, since this formulation effectively describes a cooperative system in terms of forces and velocities at the virtual sticks, while the user is expected to plan the task in terms of geometrically meaningful variables describing the position and the orientation of a set of relevant coordinate frames. Thus, starting from equations (39.23) and (39.24), further research efforts have been devoted to define alternative kinematic formulations, explictly aimed at planning the coordinated motion of general multi-arm systems. A notable example of such alternative formulations is the task-oriented formulation for the definition of the *cooperative task space* originally proposed in [39.25, 26]. By starting from (39.23) and (39.24), this formulation defines directly the task variables in terms of the *absolute* and *relative* motion of the cooperative system, which can be directly computed from the position/orientation of the end-effector coordinate frames.

Let us define as *absolute coordinate frame*, $\mathcal{T}_a$, the frame whose position in the base frame is given by the position vector $\boldsymbol{p}_a$ (*absolute position*)

$$\boldsymbol{p}_a = \frac{1}{2}(\boldsymbol{p}_1 + \boldsymbol{p}_2)\,. \tag{39.33}$$

The orientation of $\mathcal{T}_a$ with respect to the base frame (*absolute orientation*) is defined by means of the rotation matrix $\mathbf{R}_a$

$$\mathbf{R}_a = \mathbf{R}_1\mathbf{R}^1\left(\boldsymbol{k}_{21}^1, \vartheta_{21}/2\right). \tag{39.34}$$

Of course, the sole absolute variables cannot uniquely specify the cooperative motion, since 12 variables are required, e.g., for a two-arm system. Hence, in order to devise a complete description of the system's configuration, the position/orientation of each manipulator relative to the other manipulators in the system must be considered. In detail, for a two-arm system, the *relative position* between the manipulators is defined as

$$\boldsymbol{p}_r = \boldsymbol{p}_2 - \boldsymbol{p}_1, \tag{39.35}$$

while the *relative orientation* between the two end-effector frames is expressed via the following rotation matrix

$$\mathbf{R}_r^1 = \mathbf{R}_1^{\mathrm{T}}\mathbf{R}_2 = \mathbf{R}_2^1\,. \tag{39.36}$$

The variables $\boldsymbol{p}_a$, $\mathbf{R}_a$, $\boldsymbol{p}_r$, and $\mathbf{R}_r^1$ define the *cooperative task space*. Remarkably, $\mathbf{R}_a$ and $\mathbf{R}_r^1$ coincide with $\mathbf{R}_E$ and $\mathbf{R}_I^1$, respectively.

It is worth pointing out a useful feature of the above defined cooperative task space formulation. In fact, it can be recognized that definitions (39.33)–(39.36) are not based on any special assumption on the nature of the manipulated object and/or the grasp. In other words, the cooperative task space variables can be effectively used to describe cooperative systems manipulating non-rigid objects and/or characterized by a non-rigid grasp. Also, the same task-space variables can be used to describe a pure motion coordination task, i. e., when the arms are required to perform a coordinated motion without physically interacting via a commonly manipulated object. When the manipulators hold a rigid object (or a deformable object for which deformations are not commanded), then relative position and orientation are to be kept constant. Otherwise, if a relative motion between the end-effectors is allowed, then $\boldsymbol{p}_r^1$ and $\mathbf{R}_r^1$ may vary according to the actual relative motion.

As shown in [39.25, 26], absolute

$$\dot{\boldsymbol{p}}_a = \frac{1}{2}(\dot{\boldsymbol{p}}_1 + \dot{\boldsymbol{p}}_2)\,, \quad \boldsymbol{\omega}_a = \frac{1}{2}(\boldsymbol{\omega}_1 + \boldsymbol{\omega}_2)\,, \tag{39.37}$$

and relative

$$\dot{\boldsymbol{p}}_r = (\dot{\boldsymbol{p}}_2 - \dot{\boldsymbol{p}}_1)\,, \quad \boldsymbol{\omega}_r = \boldsymbol{\omega}_2 - \boldsymbol{\omega}_1\,, \tag{39.38}$$

linear and angular velocities can be readily derived, as well as their dual variables (i. e., absolute/relative forces and moments)

$$\boldsymbol{f}_a = \boldsymbol{f}_1 + \boldsymbol{f}_2\,, \quad \boldsymbol{n}_a = \boldsymbol{n}_1 + \boldsymbol{n}_2\,, \tag{39.39}$$

$$\boldsymbol{f}_r = \frac{1}{2}(\boldsymbol{f}_2 - \boldsymbol{f}_1)\,, \quad \boldsymbol{n}_r = \frac{1}{2}(\boldsymbol{n}_2 - \boldsymbol{n}_1)\,. \tag{39.40}$$

Kinetostatic mappings, analogous to those derived in the previous section, can be established between linear/angular velocities (force/moments) and their counterparts defined at the end-effectors (or joints) level of each manipulator [39.25, 26].

Remarkably, the variables defined by the symmetric formulation and those used in the task-oriented formulation are related via simple mappings. In fact, forces (angular velocities, orientation variables) always coincide in the two formulations, while moments (linear velocities, position variables) coincide only when the object reduces to a point, or when the kinematics of each manipulator is referred to the tip of the corresponding virtual stick.

As shown in the following example, in the case of planar cooperative systems, the definition of the cooperative task-space variables is straightforward.

Example 39.1 (Cooperative Task-Space Variables for a Planar Two-Arm System)
For a planar two-arm system the i-th end-effector coordinates can be defined via the (3×1) vector

$$\boldsymbol{x}_i = \begin{pmatrix} \boldsymbol{p}_i \\ \varphi_i \end{pmatrix}, \quad i = 1, 2\,,$$

where $\boldsymbol{p}_i$ is the (2×1) vector of the i-th end-effector position in the plane and φ_i is the angle describing its orientation (i. e., the rotation of the end-effector frame about an axis orthogonal to the plane). Hence, the task-space variables can be readily defined as

$$\boldsymbol{x}_\mathrm{a} = \frac{1}{2}(\boldsymbol{x}_1 + \boldsymbol{x}_2)\,, \tag{39.41}$$

$$\boldsymbol{x}_\mathrm{r} = \boldsymbol{x}_2 - \boldsymbol{x}_1\,, \tag{39.42}$$

since the orientation of each end-effector is simply represented by an angle.

In the spatial case, the definition of the orientation variables in terms of Euler angles, as in (39.34) and (39.36), is somewhat critical; in fact, since $\mathcal{T}_1$ and $\mathcal{T}_2$ do not coincide in general, orientation displacements are likely to be large and definitions analogous to (39.26) are not correct. Hence, geometrically meaningful orientation representations have to be adopted, e.g., the unit quaternion (Chap. 2). In detail, by following the approach in [39.26], the orientation variables can be defined as follows. Let

$$\mathcal{Q}^1_{\mathrm{k}>} = \{\eta_\mathrm{k},\, \boldsymbol{\epsilon}^1_\mathrm{k}\} = \left\{\cos\frac{\vartheta_{21}}{4},\, \boldsymbol{k}^1_{21}\sin\frac{\vartheta_{21}}{4}\right\} \tag{39.43}$$

denote the unit quaternion extracted from $\mathbf{R}^1(\boldsymbol{k}^1_{21}, \vartheta_{21}/2)$; let, also, $\mathcal{Q}_1 = \{\eta_1,\, \boldsymbol{\epsilon}_1\}$ and $\mathcal{Q}_2 = \{\eta_2,\, \boldsymbol{\epsilon}_2\}$ denote the unit quaternions extracted from $\mathbf{R}_1$ and $\mathbf{R}_2$, respectively. Then, the absolute orientation can be expressed in terms of quaternion product as follows

$$\mathcal{Q}_\mathrm{a} = \{\eta_\mathrm{a},\, \boldsymbol{\epsilon}_\mathrm{a}\} = \mathcal{Q}_1 * \mathcal{Q}^1_\mathrm{k}\,, \tag{39.44}$$

while the relative orientation can be expressed as the quaternion product

$$\mathcal{Q}^1_\mathrm{r} = \{\eta_\mathrm{r},\, \boldsymbol{\epsilon}^1_\mathrm{r}\} = \mathcal{Q}_1^{-1} * \mathcal{Q}_2\,, \tag{39.45}$$

where $\mathcal{Q}_1^{-1} = \{\eta_1,\, -\boldsymbol{\epsilon}_1\}$ (i. e., the conjugate of $\mathcal{Q}_1$) represents the unit quaternion extracted from $\mathbf{R}_1^\mathrm{T}$.

Finally, it is worth mentioning that a more general cooperative task-space formulation is developed for different classes of multi-arm robotic systems in [39.30].

39.4 Dynamics and Load Distribution

The equations of motion of the i-th manipulator in a cooperative manipulation system are given by

$$\mathbf{M}_i(\boldsymbol{q}_i)\,\ddot{\boldsymbol{q}}_i + \boldsymbol{c}_i(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i) = \boldsymbol{\tau}_i - \mathbf{J}_i^\mathrm{T}(\boldsymbol{q}_i)\boldsymbol{h}_i\,, \tag{39.46}$$

where $\mathbf{M}_i(\boldsymbol{q}_i)$ is the symmetric positive-definite inertia matrix and $\boldsymbol{c}_i(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i)$ is the vector of the forces/torques due to the centrifugal, Coriolis, gravity, and friction effects. The model can be expressed in compact form as

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} - \mathbf{J}^\mathrm{T}(\boldsymbol{q})\boldsymbol{h}\,, \tag{39.47}$$

where the matrices are block-diagonal (e.g., $\mathbf{M} = \mathrm{blockdiag}\{\mathbf{M}_1, \mathbf{M}_2\}$) and the vectors are stacked (e.g., $\boldsymbol{q} = \left(\boldsymbol{q}_1^\mathrm{T}\ \boldsymbol{q}_2^\mathrm{T}\right)^\mathrm{T}$).

The object's motion is described by the classical Newton–Euler equations of rigid body

$$\mathbf{M}_\mathrm{E}(\mathbf{R}_\mathrm{E})\,\dot{\boldsymbol{v}}_\mathrm{E} + \boldsymbol{c}_\mathrm{E}(\mathbf{R}_\mathrm{E}, \boldsymbol{\omega}_\mathrm{E})\boldsymbol{v}_\mathrm{E} = \boldsymbol{h}_\mathrm{E} = \mathbf{W}\boldsymbol{h}\,, \tag{39.48}$$

where $\mathbf{M}_\mathrm{E}$ is the inertia matrix of the object and $\boldsymbol{c}_\mathrm{E}$ collects the nonlinear components of the inertial forces/moments (i. e., gravity, centrifugal and Coriolis forces/moments).

The above equations must be completed by imposing the closed-chain constraints arising from the kinematic coupling between the two manipulators through the commonly manipulated rigid object. The constraints can be expressed by imposing a null internal velocity vector in the mapping (39.21)

$$\boldsymbol{v}_\mathrm{I} = \mathbf{V}_\mathrm{S}^\mathrm{T}\boldsymbol{v}_\mathrm{S} = \boldsymbol{v}_{\mathrm{S},1} - \boldsymbol{v}_{\mathrm{S},2} = \mathbf{0}\,, \tag{39.49}$$

which can be expressed, by using (39.8) and (39.22), in terms of end-effector velocities (where the notation $\mathbf{W}_i^{-\mathrm{T}}$ stands for $(\mathbf{W}_i^\mathrm{T})^{-1}$)

$$\mathbf{V}^\mathrm{T}\boldsymbol{v} = \mathbf{W}_1^{-\mathrm{T}}\boldsymbol{v}_1 - \mathbf{W}_2^{-\mathrm{T}}\boldsymbol{v}_2 = \mathbf{0}\,, \tag{39.50}$$

and, finally, in terms of joint velocities

$$\mathbf{V}^\mathrm{T}\mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} = \mathbf{W}_1^{-\mathrm{T}}\mathbf{J}_1(\boldsymbol{q}_1)\dot{\boldsymbol{q}}_1 - \mathbf{W}_2^{-\mathrm{T}}\mathbf{J}_2(\boldsymbol{q}_2)\dot{\boldsymbol{q}}_2 = \mathbf{0}\,. \tag{39.51}$$

Equations (39.47), (39.48), and (39.51) represent a constrained dynamical model of the cooperative system in the joint space; $N_1 + N_2$ generalized coordinates (i. e., $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$) are related to each other by the six algebraic closed-chain constraints (39.51). This implies that the total number of degrees of freedom is $N_1 + N_2 - 6$ and the model has the form of a set of differential-algebraic equations.

39.4.1 Reduced-Order Models

The above derived dynamic model incorporates a set of closed-chain constraints, which reduces the number of independent generalized coordinates to $N_1 + N_2 - 6$. Hence, it is expected that, by eliminating six equations via the constraints (39.51), a reduced-order model can be obtained. Early work on reduced-order modeling of closed chains can be, e.g., found in [39.67]. Later, in [39.68, 69], this problem has been tackled by first deriving, from (39.47), (39.48), and (39.51), a joint space model of the whole closed chain in the form

$$\mathbf{M}_{\mathrm{C}}(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + \boldsymbol{c}_{\mathrm{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathbf{D}_{\mathrm{C}}(\boldsymbol{q})\,\boldsymbol{\tau}\,, \tag{39.52}$$

where $\mathbf{M}_{\mathrm{C}}$, $\mathbf{D}_{\mathrm{C}}$, and $\boldsymbol{c}_{\mathrm{C}}$ depend on the dynamics of the manipulators and the object, as well as on the geometry of the grasp. The above model can be integrated, once the joint torques vector $\boldsymbol{\tau}$ is specified over an assigned time interval, to solve for the joint variables $\boldsymbol{q}$ (*forward dynamics*). However, the model cannot be used to find $\boldsymbol{\tau}$ from assigned $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, and $\ddot{\boldsymbol{q}}$ (*inverse dynamics*), since the $((N_1 + N_2) \times (N_1 + N_2))$ matrix $\mathbf{D}_{\mathrm{C}}$ is not full rank [39.69], and thus the inverse dynamics problem turns out to be underspecified.

In order to find a reduced-order model, composed by $N_1 + N_2 - 6$ equations, a $((N_1 + N_2 - 6) \times 1)$ pseudovelocity vector has to be considered

$$\boldsymbol{\nu} = \mathbf{B}(\boldsymbol{q})\dot{\boldsymbol{q}}\,, \tag{39.53}$$

where the $((N_1 + N_2 - 6) \times (N_1 + N_2))$ matrix $\mathbf{B}(\boldsymbol{q})$ is selected so that $(\mathbf{A}^{\mathrm{T}}(\boldsymbol{q})\ \mathbf{B}^{\mathrm{T}}(\boldsymbol{q}))^{\mathrm{T}}$ is nonsingular and

$$\mathbf{A}(\boldsymbol{q}) = \mathbf{W}_2^{\mathrm{T}}\,\mathbf{V}^{\mathrm{T}}(\boldsymbol{q})\,.$$

Then, the reduced-order model can be written in terms of the variables $\boldsymbol{q}$, $\boldsymbol{\nu}$, and $\dot{\boldsymbol{\nu}}$ as

$$\boldsymbol{\Sigma}^{\mathrm{T}}(\boldsymbol{q})\mathbf{M}_{\mathrm{C}}(\boldsymbol{q})\boldsymbol{\Sigma}(\boldsymbol{q})\,\dot{\boldsymbol{\nu}} + \boldsymbol{\Sigma}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{c}_{\mathrm{R}}(\boldsymbol{q}, \boldsymbol{\nu}) = \boldsymbol{\Sigma}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{\tau}\,, \tag{39.54}$$

where $\boldsymbol{\Sigma}$ is an $((N_1 + N_2) \times (N_1 + N_2 - 6))$ matrix such that

$$\begin{pmatrix}\mathbf{A}\\ \mathbf{B}\end{pmatrix}^{-1} = (\boldsymbol{\Pi}\ \boldsymbol{\Sigma})\,,$$

and $\boldsymbol{c}_{\mathrm{R}}$ depends on $\boldsymbol{c}_{\mathrm{C}}$, $\boldsymbol{\Sigma}$, and $\boldsymbol{\Pi}$ [39.69]. The reduced-order model can be used for computing the forward dynamics; in this case, however, the problem of expressing a reduced set of *pseudocoordinates* related to $\boldsymbol{\nu}$ in the numerical integration has to be considered. Since $\boldsymbol{\Sigma}^{\mathrm{T}}$ is nonsquare, the inverse dynamics problem still admits infinitely many solutions in terms of $\boldsymbol{\tau}$. However, this does not prevent the application model (39.54) to cooperative manipulators control (e.g., the decoupled control architecture proposed in [39.68, 69]).

39.4.2 Load Distribution

The problem of load sharing in multi-arm robotic systems is that of distributing the load among the arms composing the system (e.g., a strong arm may share the load more than a weak one). This is possible because a multi-arm system has redundant actuators; in fact, if a robotic arm is equipped with the number of actuators strictly needed for supporting the load, no optimization of load distribution is possible. In this section, this problem is presented according to the results in [39.36–42].

We can introduce a load-sharing matrix in the framework adopted for presenting the kinematics of the cooperative manipulators. By replacing the Moore–Penrose inverse in equation (39.11) with a suitably defined generalized inverse, $\mathbf{W}_{\mathrm{S}}^{-}$, the following expression can be devised for the generalized forces at the tip of the virtual sticks

$$\boldsymbol{h}_{\mathrm{S}} = \mathbf{W}_{\mathrm{S}}^{-}\,\boldsymbol{h}_{\mathrm{E}} + \mathbf{V}_{\mathrm{S}}\,\boldsymbol{h}_{\mathrm{I}}'\,, \tag{39.55}$$

where

$$\mathbf{W}_{\mathrm{S}}^{-} = \begin{pmatrix}\mathbf{L}\\ \mathbf{I}_6 - \mathbf{L}\end{pmatrix}^{\mathrm{T}}\,, \tag{39.56}$$

and the matrix $\mathbf{L}$ is the so-called load sharing matrix. It can be easily proved that the nondiagonal elements of $\mathbf{L}$ only yield a $\boldsymbol{h}_{\mathrm{S}}$ vector in the null space of $\mathbf{W}_{\mathrm{S}}$, that is, the space of internal forces/moments. Therefore, without losing generality, let us choose $\mathbf{L}$ such that

$$\mathbf{L} = \mathrm{diag}\,\{\boldsymbol{\lambda}\}\,, \tag{39.57}$$

where the vector $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_6)^{\mathrm{T}}$ collects a set of constants λ_i representing the load-sharing coefficients.

The problem is that of properly tuning the load-sharing coefficients to ensure correct cooperative manipulation of the object by the arms. In order to answer this question, it must be noticed that, by combining equations (39.11) and (39.55), the following relation can be obtained

$$\boldsymbol{h}_{\mathrm{I}} = \mathbf{V}_{\mathrm{S}}^{\dagger}\left(\mathbf{W}_{\mathrm{S}}^{-} - \mathbf{W}_{\mathrm{S}}^{\dagger}\right)\boldsymbol{h}_{\mathrm{E}} + \boldsymbol{h}_{\mathrm{I}}'\,, \tag{39.58}$$

which, bearing in mind that only $\boldsymbol{h}_E$ and $\boldsymbol{h}_S$ are really existing forces/moments, indicates that:

- $\boldsymbol{h}_I$, $\boldsymbol{h}'_I$, and λ_i can be thought of as *artificial* parameters, introduced for better understanding of the manipulation process.
- $\boldsymbol{h}'_I$ and λ_i are not independent; the concept of internal forces/moments and the concept of load sharing are mathematically mixed with each other.

Therefore, tuning the load-sharing coefficients or choosing suitable internal forces/moments is strictly equivalent from the mathematical (and, also, from the performance) point of view. Only one variable among $\boldsymbol{h}_I$, $\boldsymbol{h}'_I$, and $\boldsymbol{\lambda}$ is independent; hence, any of those redundant parameters, to be optimized for load sharing, can be exploited. This is more generally stated in [39.39, 40]. In [39.42], internal forces/moments $\boldsymbol{h}_I$ are tuned, both for the sake of simplicity and for consistency with the adopted control laws.

A very relevant problem related to load sharing is that of robust holding, i. e., the problem of determining forces/moments applied to object by the arms, $\boldsymbol{h}_S$, required to keep the grasp even in the presence of disturbing external forces/moments. This problem can be solved by tuning the internal forces/moments (or the load-sharing coefficients, of course) and is addressed, e.g., in [39.41], where conditions to keep the grasp are expressed via end-effector forces/moments. Namely, by replacing $\boldsymbol{h}_S$ in (39.55) into the equations expressing the grasp conditions, a set of linear inequalities for both $\boldsymbol{h}'_I$ and $\boldsymbol{\lambda}$ are obtained

$$\mathbf{A}_L \boldsymbol{h}'_I + \mathbf{B}_L \boldsymbol{\lambda} < \boldsymbol{c}_L \,, \tag{39.59}$$

where $\mathbf{A}_L$ and $\mathbf{B}_L$ are 6×6 matrices and $\boldsymbol{c}_L$ is a (6×1) vector. In [39.41], a solution $\boldsymbol{\lambda}$ for the inequality is obtained heuristically. The above inequality can be transformed into another inequality expressed with respect to $\boldsymbol{h}_I$, of course; however, $\boldsymbol{\lambda}$ is fitter to such heuristic algorithm because it can be understood intuitively. The same problem may be solved mathematically, by introducing an objective function to be optimized; in this way, the problem is recast in the framework of mathematical programming. To this purpose, let us choose a quadratic cost function of $\boldsymbol{h}_I$ to be minimized

$$f = \boldsymbol{h}_I^T \mathbf{Q} \boldsymbol{h}_I \,, \tag{39.60}$$

where $\mathbf{Q}$ is a (6×6) positive-definite matrix. The above cost function can be seen as a (pseudo)energy to be dissipated by the joint actuators: i. e., the arms dissipate electrical energy in the actuators to yield the internal forces/moments $\boldsymbol{h}_I$. A solution to the quadratic programming problem (39.60) can be found, e.g., in [39.42].

For further insights on the problem of robust holding in the framework of multifingered manipulation, the reader is referred to Chap. 38.

Interestingly, in [39.37] the problem of load distribution is formulated in such a way to take into account manipulators dynamics at the joint level. This approach allows the expression of the load-sharing problem directly in terms of joints actuators torques; at this level, different subtask performance indexes can be used, in a similar way as was done to solve the inverse kinematics of redundant manipulators, to achieve load distribution solutions.

39.5 Task-Space Analysis

As for single-arm robotic systems, a major issue in cooperative manipulation is that of task-space performance evaluation via the definition of suitable manipulability ellipsoids. These concepts have been extended to multi-arm robotic systems in [39.31]. Namely, by exploiting the kinetostatic formulation in Sect. 39.3, velocity and force manipulability ellipsoids are defined by regarding the whole cooperative system as a mechanical transformer from the joint space to the cooperative task space. Since the construction of force/velocity ellipsoids involves nonhomogeneous quantities (i. e., forces and moments, linear and angular velocities), special attention must be paid to the definition of such concepts [39.14, 70, 71]. Also, as for the ellipsoids involving internal forces, a major issue is that of physically meaningful parametrization of the internal forces (see, e.g., the work in [39.15, 16]).

In detail, by following the approach in [39.31], the *external force manipulability ellipsoid* can be defined by the following scalar equation

$$\boldsymbol{h}_E^T \left(\mathbf{J}_E \mathbf{J}_E^T \right) \boldsymbol{h}_E = 1 \,, \tag{39.61}$$

where $\mathbf{J}_E = \mathbf{W}^{\dagger T} \mathbf{J}$. The *external velocity manipulability ellipsoid* is defined dually by the following scalar equation

$$\boldsymbol{v}_E^T \left(\mathbf{J}_E \mathbf{J}_E^T \right)^{-1} \boldsymbol{v}_E = 1 \,. \tag{39.62}$$

In the case of dual-arm systems the *internal force manipulability ellipsoid* can be defined as

$$\boldsymbol{h}_I^T \left(\mathbf{J}_I \mathbf{J}_I^T \right) \boldsymbol{h}_I = 1 \,, \tag{39.63}$$

where $\mathbf{J}_\mathrm{I} = \mathbf{V}^\mathrm{T}\mathbf{J}$. Also, the *internal velocity ellipsoid* can be defined, via kinetostatic duality, as

$$\boldsymbol{v}_\mathrm{I}^\mathrm{T}\left(\mathbf{J}_\mathrm{I}\mathbf{J}_\mathrm{I}^\mathrm{T}\right)^{-1}\boldsymbol{v}_\mathrm{I} = 1\,. \tag{39.64}$$

The internal force/velocity ellipsoids can be defined in the case of cooperative systems composed of more than two manipulators by considering one pair of interacting end-effectors at a time [39.31].

The manipulability ellipsoids can be seen as performance measures aimed at determining the arms' attitude to cooperate in a given system's configuration. Also, as for the single-arm systems, the manipulability ellipsoids can be used to determine optimal postures for redundant multi-arm systems.

Besides the above described approach, two other main approaches have been proposed to analyze the manipulability of cooperative multi-arm systems: the task-oriented manipulability measure [39.32] and polytopes [39.33]. Moreover, in [39.34] a systematic approach to perform dynamic analysis of multi-arm systems is presented. Namely, the concept of dynamic manipulability ellipsoid is extended to multi-arm systems, and indexes of the system's capability of generating object accelerations along assigned task-space directions are devised.

39.6 Control

When a cooperative multi-arm system is employed for the manipulation of a common object, it is important to control both the absolute motion of the held object and the internal stresses applied to it. Hence, most of the control approaches to cooperative robotic systems can be classified as force/motion control schemes, in that they decompose the control action in a motion control loop, aimed at tracking of the desired object motion, and a force control loop, aimed at controlling the internal loading of the object.

Early approaches to the control of cooperative systems are based on the *master/slave* concept [39.2]. Namely, the cooperative system is decomposed into:

- A master arm, which is in charge of imposing the absolute motion of the object; hence, the master arm is position controlled so as to achieve accurate and robust tracking of position/orientation reference trajectories, in the face of external disturbances (e.g., forces due to the interaction with the other cooperating arms); in other words, the master arm is controlled so as to have a *stiff* behavior.
- The slave arms, which are force controlled so as to achieve a *compliant* behavior with respect to the interaction forces; hence, it is expected that the slave arms are capable of following (as smoothly as possible) the motion imposed by the master arm.

A natural evolution of the above approach is the so-called *leader–follower* [39.67], where the follower arms reference motion is computed via closed-chain constraints.

However, such approaches suffered from implementation issues, mainly due to the fact that the compliance of the slave arms has to be very large, so as to follow the motion imposed by the master arm smoothly. Also, a difficulty arises when the roles of master and slave have to be assigned to the arms for a given cooperative task, since the master/slave modes may need to be dynamically changed during the task execution.

Hence, a more natural non-master/slave approach has been pursued later, where the cooperative system is seen as a whole. Namely, the reference motion of the object is used to determine the motion of all the arms in the system and the interaction forces acting at each end-effector are fed back so as to be directly controlled. To this aim, the mappings between the forces and velocities at the end-effector of each manipulator and their counterparts at the manipulated object are to be considered in the design of the control laws.

39.6.1 Hybrid Control

In [39.11, 12] a non-master/slave approach has been proposed, based on the well-known scheme proposed by Raibert and Craig for the robot/environment interaction control of single-arm systems (Chap. 9). The operational space vector for the hybrid position/force control is defined via the following variables

$$\boldsymbol{x}_\mathrm{O} = \begin{pmatrix}\boldsymbol{x}_\mathrm{E}\\ \boldsymbol{x}_\mathrm{I}\end{pmatrix}\,, \tag{39.65}$$

where $\boldsymbol{x}_\mathrm{E}$, $\boldsymbol{x}_\mathrm{I}$ are the operational space vectors, defined by specifying the orientation via a minimal set of orientation angles (e.g., Euler angles). The generalized forces vector to be considered is

$$\boldsymbol{h}_\mathrm{O} = \begin{pmatrix}\boldsymbol{h}_\mathrm{E}\\ \boldsymbol{h}_\mathrm{I}\end{pmatrix}\,. \tag{39.66}$$

The organization of the control scheme is shown diagrammatically in Fig. 39.2. The suffixes d and m

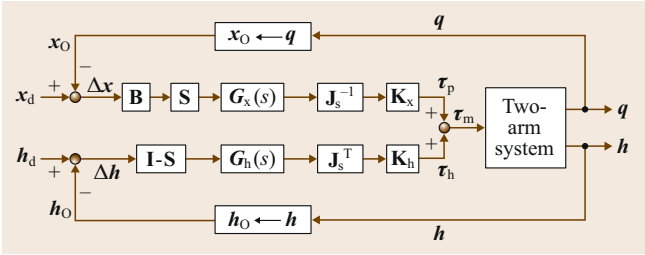


Fig. 39.2 A hybrid position/force control scheme

represent the desired value and the control command, respectively. The command vector $\boldsymbol{\tau}_m$ to the actuators of the two arms is given by two contributions

$$\boldsymbol{\tau}_m = \boldsymbol{\tau}_p + \boldsymbol{\tau}_h \,. \tag{39.67}$$

The first term, $\boldsymbol{\tau}_p$, is the command vector for the position control and is given by

$$\boldsymbol{\tau}_p = \mathbf{K}_x \mathbf{J}_s^{-1} \boldsymbol{G}_x(s) \mathbf{S} \mathbf{B} (\boldsymbol{x}_{O,d} - \boldsymbol{x}_O) \,, \tag{39.68}$$

while $\boldsymbol{\tau}_h$ is the command vector for the force control

$$\boldsymbol{\tau}_h = \mathbf{K}_h \mathbf{J}_s^T \boldsymbol{G}_h(s) (\mathbf{I} - \mathbf{S}) (\boldsymbol{h}_{O,d} - \boldsymbol{h}_O) \,. \tag{39.69}$$

The matrix $\mathbf{B}$ transforms the errors on the orientation angles into equivalent rotation vectors. The matrix $\mathbf{J}_s$ is the Jacobian matrix that transforms the joint velocity $\dot{\boldsymbol{q}}$ into the task-space velocity $\boldsymbol{v}_O$. The matrix operators $\boldsymbol{G}_x(s)$ and $\boldsymbol{G}_h(s)$ represent position and force control laws, respectively. The gain matrices $\mathbf{K}_x$ and $\mathbf{K}_h$ are assumed to be diagonal; their diagonal elements convert velocity and force commands into actuator commands, respectively. The matrix $\mathbf{S}$ selects the position-controlled variables; it is diagonal and its diagonal entries take the values of 1 or 0; namely, the i-th workspace coordinate is position controlled if the i-th diagonal element of $\mathbf{S}$ is 1, while it is force controlled if it is 0. Finally, $\mathbf{I}$ is the identity matrix having the same dimensions as $\mathbf{S}$, while $\boldsymbol{q}$ and $\boldsymbol{h}$ are the vectors of measured joint variables and measured end-effector generalized forces, respectively.

39.6.2 PD Force/Motion Control

In [39.17] a Lyapunov-based approach is pursued to devise force/position PD-type control laws. Namely, the joints torques inputs to each arm are computed as the combination of two contributions

$$\boldsymbol{\tau}_m = \boldsymbol{\tau}_p + \boldsymbol{\tau}_h \,, \tag{39.70}$$

where $\boldsymbol{\tau}_p$ is a PD-type term (eventually including a feedback/feedforward model-based compensation term), taking care of position control, while $\boldsymbol{\tau}_h$ is in charge of internal forces/moments control.

Namely, the PD and model-based terms can be computed at the joint level

$$\boldsymbol{\tau}_p = \mathbf{K}_p \boldsymbol{e}_q - \mathbf{K}_d \dot{\boldsymbol{q}} + \boldsymbol{g} + \mathbf{J}^T \mathbf{W}^\dagger \boldsymbol{g}_E \,, \tag{39.71}$$

where $\boldsymbol{e}_q = \boldsymbol{q}_d - \boldsymbol{q}$, $\boldsymbol{q}_d$ is the vector of desired joint variables, $\mathbf{K}_p$ and $\mathbf{K}_d$ are positive-definite matrix gains, $\boldsymbol{g}$ is the vector of the gravitational forces/torques acting at the manipulators joints, $\boldsymbol{g}_E$ is the vector of gravity forces/moments at the manipulated object.

Since the cooperative task is usually assigned in terms of absolute and relative motion, the equivalent end-effector desired trajectories are to be computed by using the closed-chain constraints, as in the following example.

Example 39.2 (Computation of Desired Trajectories for a Planar Two-Arm System)

For the planar two-arm system, the desired trajectories defining the cooperative task are assigned by specifying the absolute, $\boldsymbol{x}_{a,d}(t)$, and relative, $\boldsymbol{x}_{r,d}(t)$, desired motion. Then, the corresponding end-effectors desired trajectories can be computed as

$$\boldsymbol{x}_{1,d}(t) = \boldsymbol{x}_{a,d}(t) - \frac{1}{2}\boldsymbol{x}_{r,d}(t) \,, \tag{39.72}$$

$$\boldsymbol{x}_{2,d}(t) = \boldsymbol{x}_{a,d}(t) + \frac{1}{2}\boldsymbol{x}_{r,d}(t) \,, \tag{39.73}$$

where (39.41) and (39.42) have been exploited.

In the spatial case (39.72) and (39.73) do not hold, since the absolute and relative orientation must be expressed in terms of geometrically meaningful quantities, e.g., via (39.44) and (39.45). However, the same approach may be pursued at the expense of a slightly more complex expressions of the above formulas [39.26].

Once, the desired position/orientation for each end-effector has been obtained, the inverse kinematics of each arm has to be computed to provide the desired joint trajectories, $\boldsymbol{q}_d$, to the control loop; to this aim, e.g., numerical inverse kinematics algorithms may be employed (Chaps. 2 and 10).

Also, a PD-type control law can be expressed in terms of end-effector variables, i. e.,

$$\boldsymbol{\tau}_p = \mathbf{J}^T (\mathbf{K}_p \boldsymbol{e} - \mathbf{K}_v \boldsymbol{v}) - \mathbf{K}_d \dot{\boldsymbol{q}} + \boldsymbol{g} + \mathbf{J}^T \mathbf{W}^\dagger \boldsymbol{g}_E \,, \tag{39.74}$$

where $\boldsymbol{e}$ is the tracking error computed in terms of end-effector position/orientation variables, $\boldsymbol{v}$ is the vector

collecting the end-effector velocities, and $\mathbf{K}_p$, $\mathbf{K}_v$, and $\mathbf{K}_d$ are positive-definite matrix gains.

Finally, the same concept may be exploited to design a PD control law directly in the object space

$$\boldsymbol{\tau}_p = \mathbf{J}^T\mathbf{W}^\dagger(\mathbf{K}_p\boldsymbol{e}_E - \mathbf{K}_v\boldsymbol{v}_E) - \mathbf{K}_d\dot{\boldsymbol{q}} + \boldsymbol{g} + \mathbf{J}^T\mathbf{W}^\dagger\boldsymbol{g}_E \,, \tag{39.75}$$

where $\boldsymbol{e}_E$ is the tracking error computed in terms of object absolute position/orientation variables, $\boldsymbol{v}_E$ is the object's generalized velocity vector, and $\mathbf{K}_p$, $\mathbf{K}_v$, and $\mathbf{K}_d$ are positive-definite matrix gains.

The internal force control term can instead be designed as follows

$$\boldsymbol{\tau}_h = \mathbf{J}^T\mathbf{V}\boldsymbol{h}_{I,c} \,, \tag{39.76}$$

where

$$\boldsymbol{h}_{I,c} = \boldsymbol{h}_{I,d} + \boldsymbol{G}_h(s)(\boldsymbol{h}_{I,d} - \boldsymbol{h}_I) \,, \tag{39.77}$$

$\boldsymbol{G}_h(s)$ is a matrix operator representing a strictly proper linear filter, such that $\mathbf{I} - \boldsymbol{G}_h(s)$ has zeros only in the left half plane, and $\boldsymbol{h}_{I,d}$ is the vector of desired internal forces; the internal force vector can be computed from the vector of measured end-effector forces as $\boldsymbol{h}_I = \mathbf{V}^\dagger\boldsymbol{h}$. A particularly simple choice, ensuring a null error at steady state, is given by

$$\boldsymbol{G}_h(s) = \frac{1}{s}\mathbf{K}_h \,,$$

where $\mathbf{K}_h$ is a positive-definite matrix. Remarkably, when an infinitely rigid object/grasp is considered, preprocessing of the force error via a strictly proper filter is needed to ensure closed-loop stability [39.17]; e.g., if a simple proportional feedback is adopted (i. e., $\boldsymbol{G}_h(s) = \mathbf{K}_h$), the closed loop will be unstable in the presence of an arbitrary small time delay, unless a small (namely, smaller than 1) force gain is adopted. In practice, the closed kinematic chain will be characterized by some elasticity (e.g., due to grippers, end-effector force/torques sensors, joints); in this case, the product between the control gain $\mathbf{K}_h$ and the stiffness of the flexible components has to be chosen sufficiently small to ensure stability.

An interesting extension of the above approach has been given in [39.26, 28], where kinetostatic filtering of the control action is performed so as to filter all the components of the control input which contribute to internal stresses at the object. Namely, the control law (39.71) can be modified by weighting the proportional term ($\mathbf{K}_p\,\boldsymbol{e}_q$) via the filtering matrix

$$\boldsymbol{\phi} = \mathbf{J}^T(\mathbf{W}^\dagger\mathbf{W} + \mathbf{V}\boldsymbol{\Sigma}\mathbf{V}^\dagger)\mathbf{J}^{-T} \,,$$

where the (6×6) diagonal matrix $\boldsymbol{\Sigma} = \mathrm{diag}\{\sigma_i\}$ weights the components of $\mathbf{J}^{-T}\mathbf{K}_p\boldsymbol{e}_q$ in each direction of the subspace of the internal forces via the constant values $0 \le \sigma_i \le 1$. In detail, if $\boldsymbol{\Sigma} = \mathbf{O}_6$ then all these components are completely canceled from the control action, while the choice $\boldsymbol{\Sigma} = \mathbf{I}_6$ leads to the control law (39.71) without kinetostatic filtering. In a similar way, the control laws (39.74) and (39.75) can be modified so as to introduce proper kinetostatic filtering of $\mathbf{K}_p\,\boldsymbol{e}$ and $\mathbf{K}_p\,\boldsymbol{e}_E$, respectively.

39.6.3 Feedback Linearization Approaches

A further improvement of the PD-plus-gravity compensation control approach has been achieved by introducing a full model compensation, so as to achieve feedback/feedforward linearization of the closed-loop system. The feedback linearization approach formulated at the operational space level is the base for the so-called *augmented object* approach [39.72, 73]. In this approach the system is modeled in the operational space as a whole, by suitably expressing its inertial properties via a single augmented inertia matrix $\mathbf{M}_O$. Hence, the dynamics of the cooperative system in the operational space can be written as

$$\mathbf{M}_O(\boldsymbol{x}_E)\ddot{\boldsymbol{x}}_E + \boldsymbol{c}_O(\boldsymbol{x}_E, \dot{\boldsymbol{x}}_E) = \boldsymbol{h}_E \,. \tag{39.78}$$

In (39.78), $\mathbf{M}_O$ and $\boldsymbol{c}_O$ are the operational space terms modeling, respectively, the inertial properties of the whole system (manipulators and object) and the Coriolis, centrifugal, friction, and gravity terms.

In the framework of feedback linearization (formulated in the operational space), the problem of controlling the internal forces can be solved, e.g., by resorting to the *virtual linkage* model [39.16] or according to the scheme proposed in [39.29], i. e.,

$$\begin{aligned}\boldsymbol{\tau} = {} & \mathbf{J}^T\mathbf{W}^\dagger\left[\mathbf{M}_O\left(\ddot{\boldsymbol{x}}_{E,d} + \mathbf{K}_v\dot{\boldsymbol{e}}_E + \mathbf{K}_p\boldsymbol{e}_E\right) + \boldsymbol{c}_O\right] \\ & + \mathbf{J}^T\mathbf{V}\left[\boldsymbol{h}_{I,d} + \mathbf{K}_h\int(\boldsymbol{h}_{I,d} - \boldsymbol{h}_I)\right].\end{aligned} \tag{39.79}$$

The above control law yields a linear and decoupled closed-loop dynamics

$$\begin{aligned}\ddot{\boldsymbol{e}}_E + \mathbf{K}_v\dot{\boldsymbol{e}}_E + \mathbf{K}_p\boldsymbol{e}_E = \mathbf{0} \,, \\ \tilde{\boldsymbol{h}}_I + \mathbf{K}_h\int\tilde{\boldsymbol{h}}_I\,\mathrm{d}t = \mathbf{0} \,,\end{aligned} \tag{39.80}$$

where $\tilde{\boldsymbol{h}}_I = \boldsymbol{h}_{I,d} - \boldsymbol{h}_I$. Hence, the closed-loop dynamics guarantees asymptotically vanishing motion and force errors.

39.6.4 Impedance Control

An alternative control approach can be pursued based on the well-known impedance concept (Chap. 9). In fact, when a manipulation system interacts with an external environment and/or other manipulators, large values of the contact forces and moments can be avoided by enforcing a compliant behavior, with suitable dynamic features, of the robotic system. Impedance control schemes have been proposed in the case of cooperative manipulation to control object/environment interaction forces [39.21] or internal forces [39.22]. More recently, an impedance scheme for the control of both external forces and internal forces has been proposed [39.74] (in VIDEO 67 some experimental results of impedance control ar a dual-arm cooperative system are documented).

In detail, the impedance scheme in [39.21] enforces the following mechanical impedance behavior between the object displacements and the forces acting on object due to interaction with the environment

$$\mathbf{M}_{\mathrm{E}}\tilde{\boldsymbol{a}}_{\mathrm{E}} + \mathbf{D}_{\mathrm{E}}\tilde{\boldsymbol{v}}_{\mathrm{E}} + \mathbf{K}_{\mathrm{e}}\boldsymbol{e}_{\mathrm{E}} = \boldsymbol{h}_{\mathrm{env}} \,, \tag{39.81}$$

where:

- $\boldsymbol{e}_{\mathrm{E}}$ represents the vector of the object's displacements between the desired and actual pose
- $\tilde{\boldsymbol{v}}_{\mathrm{E}}$ is the difference between the object's desired and actual generalized velocities
- $\tilde{\boldsymbol{a}}_{\mathrm{E}}$ is the difference between the object's desired and actual generalized accelerations,

and $\boldsymbol{h}_{\mathrm{env}}$ is the generalized force acting on object due to the interaction with the environment. The impedance dynamics is characterized in terms of given positive-definite mass ($\mathbf{M}_{\mathrm{E}}$), damping ($\mathbf{D}_{\mathrm{E}}$), and stiffness ($\mathbf{K}_{\mathrm{E}}$) matrices to be properly chosen so as to achieve the desired compliant behavior of the object.

As for the pose displacements used in (39.81), special attention has to be paid to the orientation variables.

Example 39.3 (External Impedance for a Planar Two-Arm System)

For the planar two-arm system the quantities in (39.81) can be defined in a straightforward way. Namely,

$$\boldsymbol{e}_{\mathrm{E}} = \boldsymbol{x}_{\mathrm{E,d}} - \boldsymbol{x}_{\mathrm{E}} \,, \tag{39.82}$$

$$\tilde{\boldsymbol{v}}_{\mathrm{E}} = \dot{\boldsymbol{x}}_{\mathrm{E,d}} - \dot{\boldsymbol{x}}_{\mathrm{E}} \,, \tag{39.83}$$

$$\tilde{\boldsymbol{a}}_{\mathrm{E}} = \ddot{\boldsymbol{x}}_{\mathrm{E,d}} - \ddot{\boldsymbol{x}}_{\mathrm{E}} \,, \tag{39.84}$$

where $\boldsymbol{x}_{\mathrm{E}}$ is the (3×1) vector collecting the object's position and the orientation, while $\boldsymbol{x}_{\mathrm{E,d}}$ represents its desired counterpart. Hence, $\mathbf{M}_{\mathrm{E}}$, $\mathbf{D}_{\mathrm{E}}$, and $\mathbf{K}_{\mathrm{E}}$ are (3×3) matrices.

In the spatial case, orientation displacements cannot be defined as in (39.82), and geometrically meaningful orientation representations (i. e., rotation matrices and/or angle/axis representations) or in terms of operational space variables (i. e., differences between operational space vectors) have to be adopted [39.74].

The impedance scheme in [39.22], enforces a mechanical impedance behavior between the i-th end-effector displacements and the internal forces, i. e.,

$$\mathbf{M}_{\mathrm{I},i}\,\tilde{\boldsymbol{a}}_i + \mathbf{D}_{\mathrm{I},i}\,\tilde{\boldsymbol{v}}_i + \mathbf{K}_{\mathrm{I},i}\,\boldsymbol{e}_i = \boldsymbol{h}_{\mathrm{I},i} \,, \tag{39.85}$$

where:

- $\boldsymbol{e}_i$ is the vector expressing the displacement of the i-th end-effector between the desired and actual pose
- $\tilde{\boldsymbol{v}}_i$ is the vector expressing the difference between the desired and actual velocities of the i-th end-effector
- $\tilde{\boldsymbol{a}}_i$ is the vector expressing the difference between the desired and actual accelerations of the i-th end-effector,

and $\boldsymbol{h}_{\mathrm{I},i}$ is the contribution of the i-th end-effector to the internal force, i. e., the i-th component of the vector $\mathbf{V}\mathbf{V}^{\dagger}\boldsymbol{h}$. Again, the impedance dynamics is characterized in terms of given positive-definite mass ($\mathbf{M}_{\mathrm{I},i}$), damping ($\mathbf{D}_{\mathrm{I},i}$), and stiffness ($\mathbf{K}_{\mathrm{I},i}$) matrices, to be properly chosen so as to achieve a suitable compliant behavior of the end-effectors with respect to internal forces.

Example 39.4 (Internal Impedance for a Planar Two-Arm System)

For the planar two-arm system the quantities in (39.85) can be defined in a straightforward way as well

$$\boldsymbol{e}_i = \boldsymbol{x}_{i,d} - \boldsymbol{x}_i \,, \tag{39.86}$$

$$\tilde{\boldsymbol{v}}_i = \dot{\boldsymbol{x}}_{i,d} - \dot{\boldsymbol{x}}_i \,, \tag{39.87}$$

$$\tilde{\boldsymbol{a}}_i = \ddot{\boldsymbol{x}}_{i,d} - \ddot{\boldsymbol{x}}_i \,, \tag{39.88}$$

where $\boldsymbol{x}_i$ is the (3×1) vector collecting the position in the plane and the orientation angle of the i-th end-effector, while $\boldsymbol{x}_{i,d}$ represents its desired counterpart. Again, $\mathbf{M}_{\mathrm{I},i}$, $\mathbf{D}_{\mathrm{I},i}$, and $\mathbf{K}_{\mathrm{I},i}$ are (3×3) matrices.

As for the spatial case, orientation displacements in (39.85) cannot be defined as in (39.86), and geometrically meaningful orientation representations (i. e., rotation matrices and/or angle/axis representations) or in terms of operational space variables (i. e., differences between operational space vectors) have to be adopted [39.74].

The above two approaches have been combined in [39.74], where two control loops are designed to enforce an impedance behavior both at the object level (external forces) and at the end-effector level (internal forces).

39.7 Conclusions and Further Reading

In this chapter, the fundamentals of cooperative manipulation have been presented. A historical overview of the research on cooperative manipulation has been provided. The kinematics and dynamics of robotic arms cooperatively manipulating a tightly grasped rigid object are considered. Special topics, such as the definition of a cooperative task space and the problem of load distribution, have also been touched on. Then, the main control approaches for cooperative systems have been discussed. A few advanced topics related to control of cooperative robots, i. e., advanced nonlinear control as well as the modeling and control of cooperative systems including flexible elements, will be briefly outlined in the following.

Some of the first attempts to cope with uncertainties and disturbances in cooperative robots control focused on adaptive strategies [39.23, 24], in which the unknown parameters are estimated online, based on a suitable linear-in-the-parameters model of the uncertainties. In detail, the approach in [39.23] is aimed at controlling the object motion, the interaction force due to the contact object/environment, and the internal forces; the adaptive control law estimates the unknown model parameters, both of the manipulators and of the object, on the basis of suitable error equations. In [39.24], the adaptive control concept is used to design a decentralized scheme, i. e., a centralized coordinator is not used, for redundant cooperative manipulators.

A recently developed control framework for cooperative systems is the so-called synchronization control [39.51, 52]; in this approach the control problem is formulated in terms of suitably defined errors accounting for motion synchronization between the manipulators involved in the cooperative task. Namely, the key idea in [39.51] is to ensure tracking of the desired trajectory assigned to each manipulator, while synchronizing its motion with other manipulators motion. In [39.52], the problem of synchronizing the motion of multiple manipulators systems is solved by using only position measurements; the synchronization controller consists of a feedback control law and a set of nonlinear observers, and synchronization is ensured by suitably defining the coupling errors between the manipulators' motions.

Recently, research efforts have been spent on intelligent control [39.53–55]. Namely, in [39.53], a semi-decentralized adaptive fuzzy control scheme, with $\mathcal{H}_\infty$ performance in motion and internal force tracking, is proposed; the controller of each robot consists of two parts: a model-based adaptive controller and an adaptive fuzzy logic controller; the model-based adaptive controller handles the nominal dynamics, including a purely parametric uncertainties model, while the fuzzy logic controller is aimed at counteracting the effect of unstructured uncertainties and external disturbances. In [39.54], a decentralized adaptive fuzzy control scheme is proposed, where the control law makes use of a multi-input multi-output fuzzy logic engine and a systematic online adaptation mechanism. The approach has been further extended in [39.55].

Finally it is worth mentioning the efforts invested in the investigation of control strategies using partial state feedback, i. e., only joints position and end-effector forces are fed back to the controller. A recent contribution was provided by the work in [39.56], in which a decentralized control algorithm that achieves asymptotic tracking of desired positions and forces by using a nonlinear observer for velocities was proposed.

Flexibility in a cooperative system may arise, e.g., due to the use of nonrigid grippers to grasp the manipulated object. In fact, the adoption of compliant grippers allows large internal forces to be avoided and, at the same time, achieves safe manipulation of the object, even in the presence of failures and unpredicted contacts with the external environment. In detail, the work in [39.61] develops a non-model-based decentralized control scheme. Namely, a proportional–derivative (PD) position feedback control scheme with gravity compensation, is designed; the PD scheme is capable of regulating the position/orientation of the manipulated object and, simultaneously, achieves damping of the vibrations induced by the compliant grippers; also, a hybrid scheme is adopted to control internal forces along the directions in which the compliance of grippers is too low to ensure limited internal stresses at the manipulated object.

Other research efforts have been focused on handling multibodied objects, or even flexible objects [39.43–45]. Those objects are difficult to handle and, therefore, assembly of those objects in manufacturing industry is not automated. Also, cooperative control of multi-flexible-arm robots has been investigated [39.46–48]. Once the modeling and control problem is solved (Chap. 11), the flexible-arm robot is a robot with many merits [39.49]: it is lightweight, compliant, and hence safe, etc. Combining control methods for flexible-arm robots, such as vibration suppression, with the cooperative control methods presented in this chapter is straightforward [39.46]. Automated object-retrieval operation with a two-flexible-arm robot has been demonstrated in [39.47, 48] and in VIDEO 68.

Finally, cooperative transportation and manipulation of objects via multiple mobile manipulators

can be considered still an open research subject. In fact, although notable research results have been already devised [39.65, 75–77], the foreseen use of robotic teams in industrial settings (hyperflexible robotic workcells) and/or in collaboration with humans (robotic co-worker concept) raises new challenges related to autonomy and safety of such systems. A recently emerged application scenario is the cooperative transportation of objects via multiple aerial robots [39.66] (see VIDEO 66 for an example of cooperative transportation via multiple unmanned aerial vehicles).

Video-References

VIDEO 66 Cooperative grasping and transportation of objects using multiple UAVs
available from http://handbookofrobotics.org/view-chapter/39/videodetails/66

VIDEO 67 Impedance control for cooperative manipulators
available from http://handbookofrobotics.org/view-chapter/39/videodetails/67

VIDEO 68 Cooperative capturing via flexible manipulators
available from http://handbookofrobotics.org/view-chapter/39/videodetails/68

VIDEO 69 Cooperative grasping and transportation of an object using two industrial manipulators
available from http://handbookofrobotics.org/view-chapter/39/videodetails/69

VIDEO 70 Control of cooperative manipulators in the operational space
available from http://handbookofrobotics.org/view-chapter/39/videodetails/70

References

39.1 S. Fujii, S. Kurono: Coordinated computer control of a pair of manipulators, Proc. 4th IFToMM World Congr. (1975) pp. 411–417

39.2 E. Nakano, S. Ozaki, T. Ishida, I. Kato: Cooperational control of the anthropomorphous manipulator *MELARM*, Proc. 4th Int. Symp. Ind. Robots, Tokyo (1974) pp. 251–260

39.3 K. Takase, H. Inoue, K. Sato, S. Hagiwara: The design of an articulated manipulator with torque control ability, Proc. 4th Int. Symp. Ind. Robots, Tokyo (1974) pp. 261–270

39.4 S. Kurono: Cooperative control of two artificial hands by a mini-computer, Prepr. 15th Jt. Conf. Autom. Control (1972) pp. 365–366, (in Japanese)

39.5 A.J. Koivo, G.A. Bekey: Report of workshop on coordinated multiple robot manipulators: planning, control, and applications, IEEE J. Robotics Autom. **4**(1), 91–93 (1988)

39.6 P. Dauchez, R. Zapata: Co-ordinated control of two cooperative manipulators: The use of a kinematic model, Proc. 15th Int. Symp. Ind. Robots, Tokyo (1985) pp. 641–648

39.7 N.H. McClamroch: Singular systems of differential equations as dynamic models for constrained robot systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 21–28

39.8 T.J. Tarn, A.K. Bejczy, X. Yun: New nonlinear control algorithms for multiple robot arms, IEEE Trans. Aerosp. Electron. Syst. **24**(5), 571–583 (1988)

39.9 S. Hayati: Hybrid position/force control of multi-arm cooperating robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1986) pp. 82–89

39.10 M. Uchiyama, N. Iwasawa, K. Hakomori: Hybrid position/force control for coordination of a two-arm robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Raleigh (1987) pp. 1242–1247

39.11 M. Uchiyama, P. Dauchez: A symmetric hybrid position/force control scheme for the coordination of two robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Philadelphia (1988) pp. 350–356

39.12 M. Uchiyama, P. Dauchez: Symmetric kinematic formulation and non-master/slave coordinated control of two-arm robots, Adv. Robotics **7**(4), 361–383 (1993)

39.13 I.D. Walker, R.A. Freeman, S.I. Marcus: Analysis of motion and internal force loading of objects grasped by multiple cooperating manipulators, Int. J. Robotics Res. **10**(4), 396–409 (1991)

39.14 R.G. Bonitz, T.C. Hsia: Force decomposition in cooperating manipulators using the theory of metric spaces and generalized inverses, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), San Diego (1994) pp. 1521–1527

39.15 D. Williams, O. Khatib: The virtual linkage: A model for internal forces in multi-grasp manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 1025–1030

39.16 K.S. Sang, R. Holmberg, O. Khatib: The augmented object model: cooperative manipulation and parallel mechanisms dynmaics, Proc. 2000 IEEE Int. Conf. Robotics Autom. (ICRA), San Francisco (1995) pp. 470–475

39.17 J.T. Wen, K. Kreutz-Delgado: Motion and force control of multiple robotic manipulators, Automatica **28**(4), 729–743 (1992)

39.18 T. Yoshikawa, X.Z. Zheng: Coordinated dynamic hybrid position/force control for multiple robot manipulators handling one constrained object, Int. J. Robotics Res. **12**, 219–230 (1993)

39.19 V. Perdereau, M. Drouin: Hybrid external control for two robot coordinated motion, Robotica **14**, 141–153 (1996)

39.20 H. Bruhm, J. Deisenroth, P. Schadler: On the design and simulation-based validation of an active compliance law for multi-arm robots, Robotics Auton. Syst. **5**, 307–321 (1989)

39.21 S.A. Schneider, R.H. Cannon Jr.: Object impedance control for cooperative manipulation: Theory and experimental results, IEEE Trans. Robotics Autom. **8**, 383–394 (1992)

39.22 R.G. Bonitz, T.C. Hsia: Internal force-based impedance control for cooperating manipulators, IEEE Trans. Robotics Autom. **12**, 78–89 (1996)

39.23 Y.-R. Hu, A.A. Goldenberg, C. Zhou: Motion and force control of coordinated robots during constrained motion tasks, Int. J. Robotics Res. **14**, 351–365 (1995)

39.24 Y.-H. Liu, S. Arimoto: Decentralized adaptive and nonadaptive position/force controllers for redundant manipulators in cooperation, Int. J. Robotics Res. **17**, 232–247 (1998)

39.25 P. Chiacchio, S. Chiaverini, B. Siciliano: Direct and inverse kinematics for coordinated motion tasks of a two-manipulator system, ASME J. Dyn. Syst. Meas. Control **118**, 691–697 (1996)

39.26 F. Caccavale, P. Chiacchio, S. Chiaverini: Task-space regulation of cooperative manipulators, Automatica **36**, 879–887 (2000)

39.27 G.R. Luecke, K.W. Lai: A joint error-feedback approach to internal force regulation in cooperating manipulator systems, J. Robotics Syst. **14**, 631–648 (1997)

39.28 F. Caccavale, P. Chiacchio, S. Chiaverini: Stability analysis of a joint space control law for a two-manipulator system, IEEE Trans. Autom. Control **44**, 85–88 (1999)

39.29 P. Hsu: Coordinated control of multiple manipulator systems, IEEE Trans. Robotics Autom. **9**, 400–410 (1993)

39.30 F. Basile, F. Caccavale, P. Chiacchio, J. Coppola, C. Curatella: Task-oriented motion planning for multi-arm robotic systems, Robotics Comp.-Integr. Manuf. **28**, 569–582 (2012)

39.31 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Global task space manipulability ellipsoids for multiple arm systems, IEEE Trans. Robotics Autom. **7**, 678–685 (1991)

39.32 S. Lee: Dual redundant arm configuration optimization with task-oriented dual arm manipulability, IEEE Trans. Robotics Autom. **5**, 78–97 (1989)

39.33 T. Kokkinis, B. Paden: Kinetostatic performance limits of cooperating robot manipulators using force-velocity polytopes, Proc. ASME Winter Annu. Meet. Robotics Res., San Francisco (1989)

39.34 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Task space dynamic analysis of multiarm system configurations, Int. J. Robotics Res. **10**, 708–715 (1991)

39.35 D.E. Orin, S.Y. Oh: Control of force distribution in robotic mechanisms containing closed kinematic chains, Trans. ASME J. Dyn. Syst. Meas. Control **102**, 134–141 (1981)

39.36 Y.F. Zheng, J.Y.S. Luh: Optimal load distribution for two industrial robots handling a single object, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Phila. (1988) pp. 344–349

39.37 I.D. Walker, S.I. Marcus, R.A. Freeman: Distribution of dynamic loads for multiple cooperating robot manipulators, J. Robotics Syst. **6**, 35–47 (1989)

39.38 M. Uchiyama: A unified approach to load sharing, motion decomposing, and force sensing of dual arm robots, 5th Int. Symp. Robotics Res., ed. by H. Miura, S. Arimoto (1990) pp. 225–232

39.39 M.A. Unseren: A new technique for dynamic load distribution when two manipulators mutually lift a rigid object. Part 1: The proposed technique, Proc. 1st World Autom. Congr. (WAC), Maui, Vol. 2 (1994) pp. 359–365

39.40 M.A. Unseren: A new technique for dynamic load distribution when two manipulators mutually lift a rigid object. Part 2: Derivation of entire system model and control architecture, Proc. 1st World Autom. Congr. (WAC), Maui, Vol. 2 (1994) pp. 367–372

39.41 M. Uchiyama, T. Yamashita: Adaptive load sharing for hybrid controlled two cooperative manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) Sacramento (1991) pp. 986–991

39.42 M. Uchiyama, Y. Kanamori: Quadratic programming for dextrous dual-arm manipulation, Trans. IMACS/SICE Int. Symp. Robotics Mechatron. Manuf. Syst., Kobe (1993) pp. 367–372

39.43 Y.F. Zheng, M.Z. Chen: Trajectory planning for two manipulators to deform flexible beams, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Atlanta (1993) pp. 1019–1024

39.44 M.M. Svinin, M. Uchiyama: Coordinated dynamic control of a system of manipulators coupled via a flexible object, Prepr. 4th IFAC Symp. Robot Control, Capri (1994) pp. 1005–1010

39.45 T. Yukawa, M. Uchiyama, D.N. Nenchev, H. Inooka: Stability of control system in handling of a flexible object by rigid arm robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Minneapolis (1996) pp. 2332–2339

39.46 M. Yamano, J.-S. Kim, A. Konno, M. Uchiyama: Cooperative control of a 3D dual-flexible-arm robot, J. Intell. Robotics Syst. **39**, 1–15 (2004)

39.47 T. Miyabe, M. Yamano, A. Konno, M. Uchiyama: An approach toward a robust object recovery with flexible manipulators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Maui (2001) pp. 907–912

39.48 T. Miyabe, A. Konno, M. Uchiyama, M. Yamano: An approach toward an automated object retrieval operation with a two-arm flexible manipulator, Int. J. Robotics Res. **23**, 275–291 (2004)

39.49 M. Uchiyama, A. Konno: Modeling, controllability and vibration suppression of 3D flexible robots. In:

Robotics Research, The 7th Int. Symp, ed. by G. Giralt, G. Hirzinger (Springer, London 1996) pp. 90–99
39.50 K. Munawar, M. Uchiyama: Slip compensated manipulation with cooperating multiple robots, 36th IEEE Conf. Decis. Control, San Diego (1997)
39.51 D. Sun, J.K. Mills: Adaptive synchronized control for coordination of multirobot assembly tasks, IEEE Trans. Robotics Autom. **18**, 498–510 (2002)
39.52 A. Rodriguez-Angeles, H. Nijmeijer: Mutual synchronization of robots via estimated state feedback: a cooperative approach, IEEE Trans. Control Syst. Technol. **12**, 542–554 (2004)
39.53 K.-Y. Lian, C.-S. Chiu, P. Liu: Semi-decentralized adaptive fuzzy control for cooperative multirobot systems with H-inf motion/internal force tracking performance, IEEE Trans. Syst. Man Cybern. **32**, 269–280 (2002)
39.54 W. Gueaieb, F. Karray, S. Al-Sharhan: A robust adaptive fuzzy position/force control scheme for cooperative manipulators, IEEE Trans. Control Syst. Technol. **11**, 516–528 (2003)
39.55 W. Gueaieb, F. Karray, S. Al-Sharhan: A robust hybrid intelligent position/force control scheme for cooperative manipulators, IEEE/ASME Trans. Mechatron. **12**, 109–125 (2007)
39.56 J. Gudiño-Lau, M.A. Arteaga, L.A. Muñoz, V. Parra-Vega: On the control of cooperative robots without velocity measurements, IEEE Trans. Control Syst. Technol. **12**, 600–608 (2004)
39.57 R. Tinos, M.H. Terra, J.Y. Ishihara: Motion and force control of cooperative robotic manipulators with passive joints, IEEE Trans. Control Syst. Technol. **14**, 725–734 (2006)
39.58 H. Inoue: Computer controlled bilateral manipulator, Bulletin JSME **14**(69), 199–207 (1971)
39.59 M. Uchiyama, T. Kitano, Y. Tanno, K. Miyawaki: Cooperative multiple robots to be applied to industries, Proc. World Autom. Congr. (WAC), Montpellier (1996) pp. 759–764
39.60 B.M. Braun, G.P. Starr, J.E. Wood, R. Lumia: A framework for implementing cooperative motion on industrial controllers, IEEE Trans. Robotics Autom. **20**, 583–589 (2004)
39.61 D. Sun, J.K. Mills: Manipulating rigid payloads with multiple robots using compliant grippers, IEEE/ASME Trans. Mechatron. **7**, 23–34 (2002)
39.62 M.R. Cutkosky, I. Kao: Computing and controlling the compliance of a robot hand, IEEE Trans. Robotics Autom. **5**, 151–165 (1989)
39.63 A. Jazidie, T. Tsuji, M. Nagamachi, K. Ito: Multipoint compliance control for dual-arm robots utilizing kinematic redundancy, Trans. Soc. Instr. Control Eng. **29**, 637–646 (1993)
39.64 T. Tsuji, A. Jazidie, M. Kaneko: Distributed trajectory generation for multi-arm robots via virtual force interactions, IEEE Trans. Syst. Man Cybern. **27**, 862–867 (1997)
39.65 O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal: Coordination and decentralized cooperation of multiple mobile manipulators, J. Robotics Syst. **13**, 755–764 (1996)
39.66 J. Fink, N. Michael, S. Kim, V. Kumar: Planning and control for cooperative manipulation and transportation with aerial robots, Int. J. Robotics Res. **30**, 324–334 (2011)
39.67 J.Y.S. Luh, Y.F. Zheng: Constrained relations between two coordinated industrial robots for motion control, Int. J. Robotics Res. **6**, 60–70 (1987)
39.68 A.J. Koivo, M.A. Unseren: Reduced order model and decoupled control architecture for two manipulators holding a rigid object, ASME J. Dyn. Syst. Meas. Control **113**, 646–654 (1991)
39.69 M.A. Unseren: Rigid body dynamics and decoupled control architecture for two strongly interacting manipulators manipulators, Robotica **9**, 421–430 (1991)
39.70 J. Duffy: The fallacy of modern hybrid control theory that is based on *Orthogonal Complements* of twist and wrench spaces, J. Robotics Syst. **7**, 139–144 (1990)
39.71 K.L. Doty, C. Melchiorri, C. Bonivento: A theory of generalized inverses applied to robotics, Int. J. Robotics Res. **12**, 1–19 (1993)
39.72 O. Khatib: Object manipulation in a multi-effector robot system. In: *Robotics Research*, Vol. 4, ed. by R. Bolles, B. Roth (MIT Press, Cambridge 1988) pp. 137–144
39.73 O. Khatib: Inertial properties in robotic manipulation: An object level framework, Int. J. Robotics Res. **13**, 19–36 (1995)
39.74 F. Caccavale, P. Chiacchio, A. Marino, L. Villani: Six-DOF impedance control of dual-arm cooperative manipulators, IEEE/ASME Trans. Mechatron. **13**, 576–586 (2008)
39.75 T.G. Sugar, V. Kumar: Control of cooperating mobile manipulators, IEEE Trans. Robotics Autom. **18**, 94–103 (2002)
39.76 C.P. Tang, R.M. Bhatt, M. Abou-Samah, V. Krovi: Screw-theoretic analysis framework for cooperative payload transport by mobile manipulator collectives, IEEE/ASME Trans. Mechatron. **11**, 169–178 (2006)
39.77 H. Bai, J.T. Wen: Cooperative load transport: A formation-control perspective, IEEE Trans. Robotics **26**, 742–750 (2010)

Multimedia Contents

40. Mobility and Manipulation

Oliver Brock, Jaeheung Park, Marc Toussaint

Mobile manipulation requires the integration of methodologies from all aspects of robotics. Instead of tackling each aspect in isolation, mobile manipulation research exploits their interdependence to solve challenging problems. As a result, novel views of long-standing problems emerge. In this chapter, we present these emerging views in the areas of grasping, control, motion generation, learning, and perception. All of these areas must address the shared challenges of high-dimensionality, uncertainty, and task variability. The section on grasping and manipulation describes a trend towards actively leveraging contact and physical and dynamic interactions between hand, object, and environment. Research in control addresses the challenges of appropriately coupling mobility and manipulation. The field of motion generation increasingly blurs the boundaries between control and planning, leading to task-consistent motion in high-dimensional configuration spaces, even in dynamic and partially unknown environments. A key challenge of learning for mobile manipulation consists of identifying the appropriate priors, and we survey recent learning approaches to perception, grasping, motion, and manipulation. Finally, a discussion of promising methods in perception shows how concepts and methods from navigation and active perception are applied.

40.1 **Grasping and Manipulation** 1009
40.1.1 Problem Description 1009
40.1.2 Evaluation of the State of the Art 1010
40.1.3 Toward Robust Grasping and Manipulation 1013
40.2 **Control** 1013
40.2.1 Problem Description 1014
40.2.2 Evaluation of the State of the Art 1014
40.2.3 Toward Control in Mobile Manipulation 1016
40.3 **Motion Generation** 1017
40.3.1 Problem Description 1017
40.3.2 Evaluation of the State of the Art 1018
40.3.3 Toward Motion Generation for Mobile Manipulation 1020
40.4 **Learning** 1021
40.4.1 Problem Description 1021
40.4.2 Evaluation of the State of the Art 1023
40.4.3 Toward Learning in Mobile Manipulation 1025
40.5 **Perception** 1025
40.5.1 Problem Description 1026
40.5.2 Evaluation of the State of the Art 1026
40.5.3 Toward Robust Perception 1028
40.6 **Conclusions and Further Reading** 1029
Video-References 1029
References 1030

What exactly is mobile manipulation? To find an answer to this question, we must consider the historical development of this research area. The term mobile manipulation was coined in the late 1980s to early 1990s, when research labs began to mount robot manipulators on mobile platforms [40.1, 2]. Back then, the term was intended to capture just that: research was conducted on an experimental platform that combines capabilities in mobility and manipulation. A series of mobile manipulation platforms is shown in Fig. 40.1.

With the introduction of such platforms it soon became apparent that combining mobility with manipulation was a game changer. Mobility enables manipulators to leave the lab. They now have to face the complexities of the real world. And in the real world, much of the research that had been successful in the carefully

controlled environments in research labs proved to be brittle or inadequate. This traditional research often relied on strong assumptions about the environment to be successful – an untenable assumption in unpredictable, nonstationary everyday environments.

In the context of mobile manipulation research, controlled environments, such as factory floors and specific experimental setups, are often referred to as *structured environments*. In contrast, *unstructured environments* are environments that have not been modified specifically to facilitate the execution of a task by a robot. These two types of environments are opposites on a continuous scale with many possible intermediates. And, of course, structured environments are still laden with uncertainty just the same as unstructured environments still contain a significant structure that can be exploited by the robot.

In unstructured environments, it becomes a necessity for the robot to perform a wide variety of tasks, instead of a single specific task. Any specific task, such as retrieving a book from the library, might require the robot to solve additional challenges, such as opening doors, operating elevators, moving a chair out of the way to approach the shelf, or asking for the location of the stacks. Today, no robotic system possesses such capabilities.

Researchers in mobile manipulation have learned that the progress achieved in traditional research areas of robotics cannot simply be combined to produce the level of competency they aim for. This insight motivated the founding of mobile manipulation as a research area. The research in this area seeks to develop robotic systems capable of autonomous task execution in unstructured or minimally structured environments. Clearly, the transition from single-purpose lab demonstration – a common practice in robotics – toward such mobile manipulation systems is a gradual one. Mobile manipulation research therefore gradually increases the autonomy, robustness, and task generality of robotic systems, while at the same time gradually reducing the dependence on prior information about the environment and the task.

The objective of mobile manipulation research is to:

1. Maximize task generality of autonomous robotic systems, while at the same time
2. Minimizing the dependence on task-specific, hard-coded, or narrowly-relevant information.

Some people argue that this is exactly what main stream robotics is doing and what most of this Handbook is about. And they might be right. But over the last 20 years, the mobile manipulation community, forced by failures of robotic systems in the real world, has begun to explore alternatives and extensions to the body of robotics knowledge collected in this tome. It is these alternatives or extensions that this chapter is about. In each of the subsequent sections, our goal is to showcase work in mobile manipulation that opens up possible novel directions for future advances toward robots operating in real-world environments. For the areas of grasping and manipulation, mobility, control, motion generation, learning, and perception, we will analyze how the main challenges of the field change when systems are operating in the real world. We will show how each of these fields has developed and advanced to increasingly address the challenges of unstructured environments.

Before we embark on this journey, it is worthwhile to agree on the challenges that are in fact encountered when robots are supposed to perform a variety of tasks in an unconstrained and uncontrolled envi-

Fig.40.1a–d Some exemplary mobile manipulation platforms developed throughout the decades **(a)** Herbert (MIT, 1985), **(b)** Stanford assistent mobile manipulator (1995), **(c)** PR2 (Willow Garage, 2009), **(d)** Herb (CMU, 2013)

ronment. These challenges are common to all of the subdisciplines we discuss in the following sections.

First, the necessity to perform a variety of tasks, instead of a single one, requires that the robot be equipped with general means of mobility, manipulation, and perception. Its capabilities must be the union of the minimum capabilities required for each task. This entails the need for versatile sensing and dexterous manipulation. This, in turn, implies that the sensor input of the robot is high dimensional and so is the configuration space of the robot. The first challenge therefore consists of dealing with the *high dimensionality* and the inherent complexity of the state spaces required for task generality. To make matters worse, task generality most often requires system dynamics to be hybrid, i. e., exhibit discrete as well as continuous behavior.

Second, by the definition of mobile manipulation, the robot cannot rely on a complete, continuously, and globally updated world model. While some part of the robotics community attempts to structure the world to become such a world model – by using RFID tags or by placing sensors everywhere – many in the mobile manipulation community believe it to be more effective and more insightful to make robots smarter. Not having a complete world model means that the robot has to use perception to acquire relevant information about the world; it means that the robot has to address the uncertainty in its sensing, but also the uncertainty in its world model which might change unbeknownst to the robot. The second challenge therefore consists of appropriately handling the *uncertainty* inherent to sensing and actuation and the uncertainty caused by a dynamic world.

Third, in the real world, objects exhibit large variability from the perspective of their appearance, even when they perform exactly the same function. This means that there is a layer of indirection through appearance when a robot wants to operate autonomously in the real world. There are so many different kinds of door handles, they differ significantly in appearance, but perform the same function. The challenge therefore is to cope with the *variability* in the world by revealing the functional principles underlying the world's appearance. A natural way of addressing variability are skills that generalize to novel situations.

As we will see in subsequent sections, attempts to address these challenges by the robotics community have produced innovative and successful concepts, algorithms, mechanisms, and integrated systems. Some of them were not covered in previous chapters, as they do not (yet?) belong to the canon of robotic wisdom. Maybe this chapter can contribute to changing this.

40.1 Grasping and Manipulation

Robots accomplish tasks by grasping, manipulating, and placing objects. Several chapters in this Handbook are concerned with associated mechanisms, methods, and concepts: Chapter 19 discusses robot hands, the most versatile type of end-effector for grasping and manipulation. Chapter 37 discusses manipulation in general and Chap. 38 presents the state of the art in grasping. Important for grasping and manipulation are also force and tactile sensors, which are discussed in Chap. 28. While these four chapters describe the well-established foundation of grasping and manipulation, in this chapter we want to take a higher level perspective, assess the suitability of these approaches in the context of mobile manipulation, and speculate about possible novel approaches.

Both grasping and manipulation exemplify the challenges for mobile manipulation we laid out above: uncertainty, high dimensionality, and task variability. Uncertainty is relevant as minor inaccuracies in the object model, minor errors in motion execution, and small differences in positioning relative to the object can cause grasp failure. The high dimensionality of grasping results from the many degrees of freedom of the hand as well as from those of the objects in the environment. Finally, grasping and manipulation occur in a large number of variations, each involving vastly different interactions with objects. It is therefore not surprising that decades of research on grasping and manipulation have made significant progress but so far were unable to create skills sufficiently robust for autonomous operation in the real world.

In this section, we describe a recent development in the field of grasping and manipulation away from the foundational techniques laid out elsewhere in this Handbook toward novel approaches that seem better suited for addressing the manipulation challenges in mobile manipulation.

40.1.1 Problem Description

First, a word on defining the problem. Traditionally, grasping and manipulation are viewed as two distinct problems. But there is signifiifant ambiguity with respect to these terms in the literature [40.3]. In this section, we limit ourselves to grasping and manipulation with robot hands. And in this context, we define the following terms:

Grasping refers to gaining reliable control over the extrinsic degrees of freedom of an object through the robot's degrees of freedom, i. e., the robot grasps an object so that it can change the object's extrinsic degrees of freedom by changing its own, excluding the degrees of freedom of the hand, which remain static once the grasp is achieved. To perform grasping, the overall degrees of freedom of the robot have traditionally been divided into two parts: those required to impart forces on an object (usually addressed using grasp planning approaches) and the remaining ones that can be used to reposition the object, once a grasp is obtained (general motion and manipulation planning). Note, however, that in contrast to this traditional decomposition, coordination of arm and hand degrees of freedom are usually required during grasping.

By *dexterous grasping* we refer to grasping while optimizing the grasp posture for (a) robustness and (b) suitability for the execution of a specific task. In the relevant literature, the overall degrees of freedom are still most commonly divided into those responsible for the grasp and those able to change the pose of the grasped object.

By *dexterous in-hand manipulation*, we refer to the ability to actuate the extrinsic degrees of freedom of the object using all degrees of freedom of the arm and the hand so that the object moves relative to a reference frame placed at the hand's wrist [40.4].

Classical approaches to all three of these problems rely on precise models of the environment and of the hand, carefully reason about contact points and states, and attempt to determine and precisely execute a detailed plan. We would like to point to the emergence of a novel approach to all three of these problems. Since this novel approach, however, has been mainly explored in the context of grasping at this early point in time, we will focus our discussion on grasping. But we will also make the point that in this new way of thinking about grasping and manipulation both of these problems become similar, if not identical.

40.1.2 Evaluation of the State of the Art

The earliest and still most fundamental considerations of grasping are based on the notions of force and form closure (Chap. 38).

Force and Form Closure

These notions express the effect of a set of disembodied contact points, i. e., ignoring the body on which that contact point lies, such as the finger, on the ability of an object to move. They reflect a static view of grasping, in which the physical interactions that will invariably occur during the grasp are not considered.

This line of research continues to be active and successful, as evidenced by a large number of sophisticated and capable grasp planners and simulators [40.5]. The resulting grasps, however, often do not result in successful real-world grasps. These failures are a consequence of model uncertainty and motion execution noise. The assumptions made during planning, i. e., that the disembodied contact points can be realized precisely by the hand, more often than not do not hold.

Interactions Between Hand and Object

There is a very simple strategy to overcome many of the theoretical failures of the approaches based on precise models and the notion of force closure – and this strategy is used tacitly by all grasping research: just keep closing the hand, even if you think you have attained the right configuration for your contact points. This strategy effectively leverages mechanical compliance of the hand to adapt the hand's configuration to the object's shape. This adaptation of shape has several positive effects for grasping:

1. Sensing and actuation uncertainty are compensated,
2. Large surface areas are established, and
3. Grasping forces are distributed and balanced.

These effects lead to an increase in grasp success and grasp quality. It is fair to say that this is a standard trick in the grasp planning community.

This insight is also strongly driving contemporary hand design. To maximize the effect of shape adaptation, *Rodriguez* and *Mason* [40.6] optimize finger shape to yield similar contact configuration irrespective of the size of the object. A different way of increasing shape adaptability is to include compliant components in the hand design [40.7]. The shape deposition manufacturing (SDM) hand [40.8], the Velo gripper [40.9], the i-HY hand [40.10], and the Pisa/IIT SoftHand [40.11] achieve shape adaptability through underactuation by coupling degrees of freedom of the hand, adapting the shape of the hand to the object while equalizing contact forces. An extreme case of this is the positive pressure gripper [40.12], which presses a bag filled with granular material onto the object to be grasped. The bag adapts to the shape of the object. When the air is evacuated from the bag, the granular material jams, forming a gripper perfectly adapted to the shape of the object.

From this brief analysis it is fair to say that the shape adaptability between hand and object is routinely leveraged in grasp planning, simply by *closing the hand* until a certain grasping force is reached. Also, shape adaptability has emerged as an important design criterion for hands, as it very obviously improves grasping performance.

Interactions Between Hand, Object, and Environment

Features in the environment, provided by other objects or support surfaces, for example, may constrain the motion of hand and object. This is most evident for support surfaces, such as tables and floors. These constraints, when used properly, can aid grasping. Furthermore, it might be the case that the necessary perceptual information for leveraging such constraints is often easier to obtain than the information required for the successful execution of an unconstrained grasp.

Recent research in robotic grasping leverages environmental constraints in the suggested manner, e.g., to position the hand relative to the object [40.13], to cage objects [40.13, 14], or to fixate an object during planar sliding [40.13, 15]. Some pregrasp manipulation relies on environmental constraints to improve grasp success. For example, *Chang* et al. [40.16] rotate pan handles into a specific orientation prior to grasping by exploiting the pan's friction and remote center of mass.

But, in addition to external contacts, environmental constraints can also be created by exploiting gravity, inertia, or dynamic motion of the arm and hand. In this way, in-hand manipulation is accomplished by using what has been called extrinsic dexterity [40.17].

All of the aforementioned grasp strategies rely on multiple interactions between hand, object, and environment prior to attaining the final grasp posture. These phases often are designed to reduce uncertainties in specific variables relevant to grasp success.

The idea of environmental constraints appears in early work by *Mason* et al. [40.18, 19]. Here, the intrinsic mechanics of the task environment are exploited to eliminate uncertainty and to achieve robustness.

The study of environmental constraint exploitation by humans so far has been limited to replicating instances of observed behavior on robots [40.16, 20]. For example, *Kaneko* et al. [40.20] extracted a set of grasping strategies from observations of a human subject. These strategies include interactions with environmental constraints.

We believe that this recent trend toward the exploitation of environmental constraints represents an important opportunity to improve robotic grasping capabilities. To take full advantage of this opportunity, we should understand the strategies humans employ, transfer them to robotic systems, and develop robotic hands tailored to this exploitation [40.21]. This will be discussed in the next section.

Insights from Human Grasping

Interestingly, the study of human grasping paralleled the development in the robotics community. Early studies of human grasping followed the static view captured by force closure concepts. This is reflected in grasp taxonomies, classifying grasp according to the final hand posture attained after the grasp process is completed [40.22, 23].

Even the early work on postural synergies, which has had a profound impact on robotics, initially only considered synergies of static grasp postures [40.24]. These studies do not capture the dynamic processes and the exploitation of environmental constraints we believe to be crucial for robust grasping. Nevertheless, these studies of human grasping have led to significant advances in robotic grasping [40.11].

The nature of hand–object interaction has also been studied in humans. The effect of shape adaptability is well known for human hands and studies have elucidated the degree to which humans vary their behavior to take advantage of it. *Christopoulos* and *Schrater* [40.25] showed that humans react to pose uncertainty of a cylinder and orient their hand to align it, presumably to be able to maximize the benefits of shape adaptability. However, other experiments may point at the fact that humans also rely on more complex interactions with the environment for grasping under difficult conditions. When the vision of humans is impaired, they fail more often at first grasp attempts of isolated (environmental-constraint free) objects [40.26]. The degree of the demonstrated effect seems surprising. We believe that in this specific experiment, it is due to the lack of environmental constraints exploitable for grasping.

More recently, studies of human grasping have shown the deliberate use of environmental constraints during grasping. For example, humans increase their use of environmental constraints when their vision is impaired [40.21]. This seems to indicate that the use of environmental constraints will play a crucial role in achieving robust and task-general grasping and manipulation in robotics.

Robot Hands

Many highly capable robotic hands exist. A historical overview, collecting robotic hands from over five decades, was compiled by *Controzzi* et al. [40.27]. An analysis of robot hand designs with respect to grasping capabilities was recently presented by *Grebenstein* [40.28]. As the notion of compliance is central to our hand design, we will limit our discussion to hands designs that deliberately include this concept.

We distinguish two main approaches for designing compliant hands. Compliance can be achieved using active control, and implemented on a fully actuated or even hyper-actuated systems, where every degree of freedom can be controlled. An impressive exam-

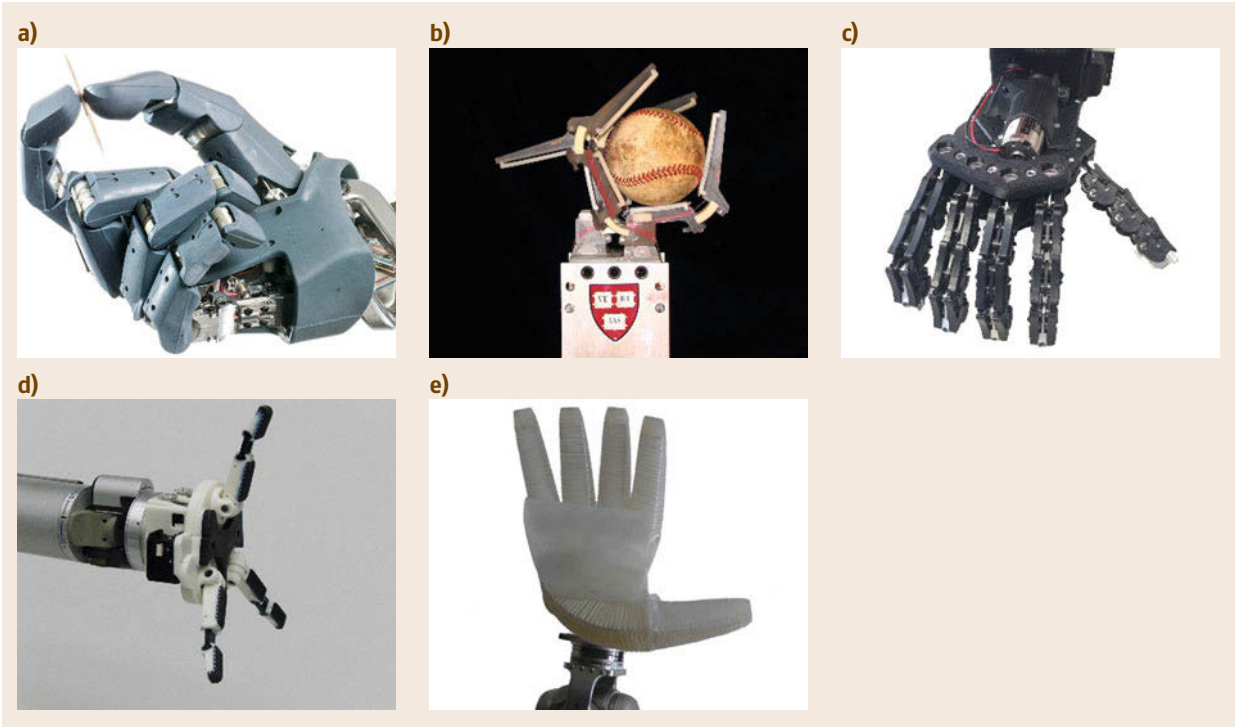


Fig.40.2a–e Various hand designs leveraging compliance to improve performance. **(a)** Awiwi hand (DLR), **(b)** SDM hand (Havard), **(c)** Pisa-IIT hand, **(d)** i-HY hand (iRobot, Havard, Yale), **(e)** RBO hand z (TU Berlin)

ple of this type of hand is the Awiwi hand [40.28], the ShadowRobot Shadow Dexterous Hand, and the SimLab Allegro Hand [40.29]. These hands achieve dexterity through accurate control, which comes at the price of mechanical complexity, making them difficult and costly to build and prone to failure.

An alternative approach is to make hands compliant by including elastic or flexible materials (passive compliance). Building a passively compliant joint is much cheaper than an actively controlled one, in terms of costs, volume and system complexity. Passive compliance can easily absorb impact forces – a desirable property for an end-effector designed to establish contact with the world. The cost of adding additional (passive) degrees of freedom is low, compared to actively compliant hands. The resulting ability to passively adapt to the shape of an object greatly enhances grasp success and grasp quality. At the same time, the hand can be underactuated, effectively offloading control to the physical material.

A pioneering work in grasping with passive compliance was the soft gripper by *Hirose* and *Umetani* [40.7]. Recently, a whole range of grippers and hands were built using passive compliance, such as the FRH-4 hand [40.30], the SDM hand and its successor [40.10, 31, 32], the starfish gripper [40.33], the THE Second Hand and the Pisa-IIT Soft Hand [40.34], the Positive Pressure Gripper [40.12], the RBO Hands [40.13, 35], and the Velo Gripper [40.9]. A different source of inspiration was taken by *Giannaccini* et al. [40.36], who built an octopus-inspired compliant gripper. Some of these hands are shown in Fig. 40.2.

The practical realization of underactuated hands is matched by theoretical approaches to analyze and evaluate their dexterity [40.37, 38]. However, these approaches require accurate knowledge of grasp posture, contact point locations, and contact forces. Given today's sensor technologies, this information is difficult to obtain in physical implementations.

The inclusion of compliance into the design of robotic hands has led to significant improvements in power-grasping of objects. Given the insights obtained from human grasping (see the previous section), it is plausible that these improvements are the result of these hands' capabilities to engage in beneficial contact interactions with the environment. While these benefits have been demonstrated extensively in the context of power grasping, little work has examined the beneficial effects of compliance and underactuation on the dexterity of robotic hands [40.35].

40.1.3 Toward Robust Grasping and Manipulation

The key to robust grasping and manipulation – an important prerequisite for mobile manipulation – appears to be the deliberate and purposeful use of interactions between hand, object, and environment. This is supported by evidence from human grasping as well as from the design features of successful robotic hands. If this claim is true, it fundamentally changes the game in grasping and manipulation. The notion of a specific contact point configuration, which has for the last 50 years been of pivotal importance, is superseded by deliberate constraint exploitations. The traditional, static view of grasping is replaced with a dynamic view that emphasizes motion of hand, object, and environment in contact with each other. Rather than placing fingers carefully at exactly the right place, this new way of thinking requires fingers to slide across surfaces, to comply to object shapes, and to balance and maintain contact forces through intelligent hardware design.

Especially in hardware design, there has been significant progress. The newest generation of hands are built for the exploitation of constraints: without sensing and explicit control, they support shape adaptation and compliant contact maintenance. One of the future challenges will be the development of grasp planning methods and perceptual capabilities to leverage the advantages afforded by these novel hands.

40.2 Control

Mobile manipulators are robotic systems with capabilities for both manipulation and mobility. A typical setup of mobile manipulator, therefore, consists of manipulator and mobile robot. Traditionally, mobile manipulators were implemented as wheeled mobile robots, providing mobility on two-dimensional flat surfaces. Mobility, however, can be provided by other types of robots depending on the operating environment. Legged robot systems operate well in uneven or rough terrain. Aerial or underwater robots allow operation in environments with totally different characteristics.

Controlling such mobile manipulators involves control for the subsystems. The control issues of manipulators are explained and discussed earlier in Chaps. 8 and 9. Modeling and control of various types of mobile platforms will be introduced later in detail in Chaps. from 48 to 52 in Part E. In these chapters, we would like to discuss unique control problems in mobile manipulation, which arise when the manipulator and mobile platform operate simultaneously.

Control of mobile manipulators also has to address the three challenges we stated initially: high dimensionality, uncertainty, and task variabilty. The mobile manipulator was first implemented to provide mobility to the manipulator. One of the main purposes is to increase the manipulation workspace of the robot (Fig. 40.3). This mobile platform naturally adds more degrees of freedom to the system by providing mobility. The additional degrees of freedom increase the dimensionality of the control problem. The robot be-

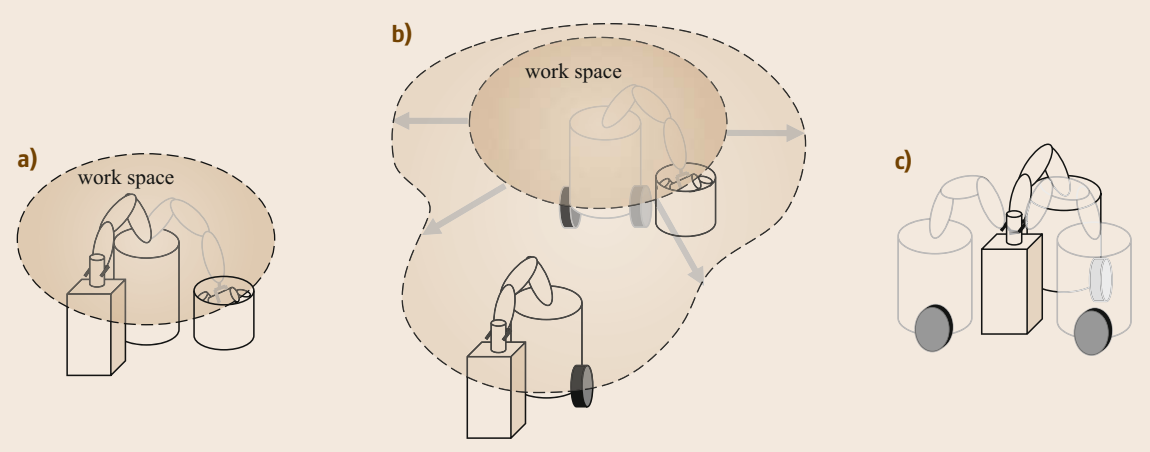


Fig.40.3a–c Increased workspace and redundancy of mobile manipulators. **(a)** A manipulator on a fixed base; **(b)** increased work space of a mobile manipulator; **(c)** redundancy due to the added mobility

comes a redundant system because the specified task to the robot is still the same, while the overall degrees of freedom have been increased. That is, there are infinite number of configurations that can execute the same task.

Second, uncertainty becomes severe in control of mobile manipulators due to the mobile platform. The uncertainty is larger when controlling a mobile platform compared to control of a manipulator. The reasons for the higher uncertainty are imperfect information or model about the operating environment, such as ground and underwater. Additionally mobile platforms with only internal sensors do not have access to perfect state information because drift raises uncertainty while moving. However, mobile manipulators need to solve the same manipulation tasks as stationary manipulators. They have to overcome this uncertainty to operate reliably.

Third, the variability of the mobile platform exists depending on the operating environment, although the high dimensionality and uncertainty are the common factors across the various mobile platforms. The different issues due to various mobile platforms are also to be discussed later in this chapter.

40.2.1 Problem Description

Control of mobile manipulators must provide capabilities for manipulation and mobility. Often times, these two are treated completely separate. The robot moves to a certain location using its mobile platform. Then, the manipulator performs specific tasks without moving the platform. In this approach, the mobile platform considers the manipulator as a static load. The mobile platform is then considered as a fixed base for the manipulator. There is no major difficulty implementing this approach, unless manipulator or mobile platform is unable to hold its position steadily.

The main problem to be discussed in this chapter arises when both manipulator and mobile base operate simultaneously. Although the previously mentioned approach of a separately controlling manipulator and a platform increases the workspace of the robot, it does not exploit the full capabilities of mobile platforms. Instead, the advantage of mobile manipulator can be maximized when the manipulator operates while the bases moves in coordination. This kind of operation will not only reduce the total operation time but will also increase task variability.

In this chapter, therefore, we will discuss how to control mobile manipulators by operating both subsystems at the same time. During this operation, the whole system becomes redundant due to the added degrees of freedom (DOF) from the mobile platform. How to exploit this increased number of DOF is one important problem in mobile manipulation. Secondly, the increased uncertainty due to the mobile platform requires more robust control strategies than that only for manipulator. Finally, various issues for different types of mobile platforms will be discussed.

40.2.2 Evaluation of the State of the Art

First mobile manipulators were implemented as wheeled robots and wheels are still the most popular tool to enable a manipulator to move. Most of the issues that have been dealt with in wheeled mobile manipulator can be shared with other types of mobile platforms. Therefore, we start discussing wheeled mobile platforms in this chapter and discuss issues specific to other types of mobile platforms afterward.

Redundancy Resolution and Control

One of the main issues in controlling mobile manipulators has been how to use the additional DOFs to execute manipulation tasks in a stable way. An early study [40.39] investigated how to obtain solutions to the inverse kinematics problem that prevent the robot from tipping over.

Other approaches exploited the increased number of DOFs to avoid obstacles, increase manipulability, handle singularities, or achieve lower priority tasks. In [40.40], a preferred region to maximize the manipulability of the robotic arm is defined and the nonholonomic mobile platform is controlled to be within the region. The same concept of a preferred region is also applied to the force control of a mobile manipulator in [40.41]. These approaches are typical examples for approaches that coordinate with the manipulator and base motion, but control the subsystems separately.

The redundancy of mobile manipulators allows dealing with multiple tasks simultaneously. In the configuration approach [40.42, 43], additional tasks or specifications are defined to be simultaneously controlled with the end-effector. To obtain solutions online, a weighted damped least-squares approach was presented in [40.44]. The configuration approach is further developed to be used both for holonomic and nonholonomic bases in [40.45].

In [40.46], the unified control of the arm and holonomic base was achieved by applying the operational space control framework to the mobile manipulator. The coordination of the arm and base can be implemented in the null-space of the task control, which does not affect the control of the end-effector. Later, this approach was extended to deal with nonholonomic bases in [40.47]. It uses a complete model of the system with nonholonomic constraints to devise one unified con-

troller for the whole system. The redundancy was used to deal with internal and external constraints. Another model-based control is derived in [40.48] for a car-like and differentially-driven system.

Similarly, an approach of exploiting redundancy for avoiding singularities and maximizing manipulability is proposed in [40.49] using an event-based planner and a nonlinear feedback controller with the dynamic model of the robot.

As can be seen from the above literature, the issue of redundancy resolution in mobile manipulator has been often dealt with as an extension from the control of redundant systems but with the special case of nonholonomic constraints.

Uncertainty Due to Mobile Platform

Another main problem arising when controlling mobile manipulators, is the additional uncertainty due to the mobile platform. This is because the mobile platform in real world moves on the imperfect ground, which creates a large uncertainty on the manipulation although it can be negligible for navigation. Early work in the nineties studied how to control the manipulator when the base is under disturbance due to the ground condition in [40.50]. The disturbance was assumed to be unknown and considered as rotational motion only.

Also, model-based approaches are not always suited for the control of mobile manipulator because accurate models of mobile platforms are harder to obtain than models of manipulators. The extended Jacobian transpose algorithm is proposed in [40.51] for controlling the manipulator to overcome the excessive end-effector error or instability due to unmodeled vehicle dynamics. Similarly, since precise modeling of the interaction is difficult, decentralized control is proposed in [40.52]. It regards the interaction force from each system as unknown disturbance.

In [40.53], the control system is divided into mobile platform and manipulator, and two low-level controllers are used to control each system. The dynamic interaction force from the base is modeled as an unknown disturbance in manipulator control, and an adaptive controller was designed to deal with this disturbance. A redundancy resolution scheme avoids singularities. On the other hand, the effects of dynamic interactions between manipulator and wheeled base on the performance of the end-effector task are studied in [40.54].

In another aspect, there have been approaches without modeling interaction or dynamics of each system. A robust damping controller is proposed in [40.55] for the motion control subject to kinematic constraints but without any knowledge of dynamic parameters.

Neural network-based control also has been proposed in this respect. In [40.56], a neural network-based joint space control is developed for a mobile manipulator with unknown dynamic model and disturbance. Two neural network controllers are applied to manipulator and vehicle, respectively. The radial basis function network with weight adaptation is used in [40.57] for online estimation of nonlinear dynamics. The algorithm is applied to a simulation of a two-link manipulator on a mobile platform. Adaptive robust motion/force control has been developed for both holonomic and nonholonomic constrained mobile manipulators in [40.58]. It ensures stability and the boundedness of tracking errors in the presence of uncertainties in dynamic parameters and disturbances.

Sliding mode control and adaptive neural network control are combined together in [40.59]. Multilayered perceptrons are applied to estimate the dynamic model as one system, and the adaptive control is designed in the task space control. A sliding mode approach with neural network-based control is proposed in [40.60] for omnidirectional wheeled mobile manipulators.

Control Issues on Various Types of Mobile Platform

Although wheeled vehicles are one of the most general types of platform to provide mobility, there is an increasing tendency of using other types (Fig. 40.4).

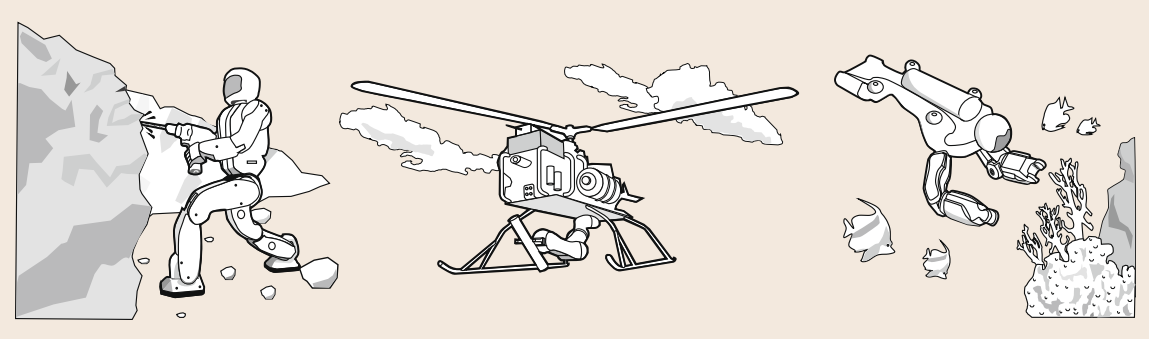


Fig. 40.4 Mobile manipulators in various environments

Legged locomotion systems like humanoids and quadrupeds are becoming popular. The interesting new aspect of these systems is that the platform for locomotion can actively participate in manipulation. That is, the legs of the humanoid system can provide not only locomotion but also assist in manipulation. This is especially enabled by using the whole-body control framework [40.61–64]. This control framework considers the robotic system as a whole and utilize all the joints for both manipulation and locomotion. This is especially justified in legged robots because the mobile platform and manipulator use the same type of actuator. Other platforms use different types of actuation for base and manipulator.

Also, quadruped robots such as Robosimian from NASA JPL uses the leg parts to increase the workspace and dexterity of the manipulation. The legged system can provide more capability of manipulation than wheeled mobile platform because it can provide the full three-dimensional (3-D) motion of the base part such that the manipulation part can have much larger manipulation capability. The whole-body control framework is also applied to a quadrupedal robot in [40.65]. The robot does not yet have manipulator in this system, but the control framework can support the manipulation control if needed.

Mobile manipulators operating underwater are becoming more prevalent. The main difference to wheeled mobile platform is the effect of hydrodynamic forces during motion. An efficient dynamic simulation algorithm is developed for underwater mobile manipulator in [40.66], where various hydrodynamic forces can be modeled and incorporated. In [40.67], the hydrodynamic effect between the manipulator and vehicle is modeled and compensated. The demonstrated results improved performance in controlling both vehicle and manipulator. The underwater mobile manipulator is modeled using Kane's method in [40.68], which also incorporates the major hydrodynamic forces such as added mass, profile drag, fluid acceleration, and buoyancy.

Underwater mobile manipulators also have redundant degrees of freedom. In [40.69], redundancy created by the underwater vehicle is utilized to achieve secondary tasks, such as reducing energy consumption and maximizing dexterity, while executing the main task at the end-effector. A task-priority redundancy resolution technique is applied for this purpose. The restoring moment is minimized in [40.70] in redundancy resolution so that the performance of the coordinated motion control is improved.

Similar to other mobile platforms, the uncertainty due to the underwater vehicle is one of the major difficulties. To deal with this issue, an adaptive control approach is proposed in [40.71] that is robust in the presence of uncertainties in the vehicle and environment. An iterative learning algorithm is developed for the hydrodynamic effects in [40.72]. An adaptive tracking control is proposed in [40.73] that keeps the advantage of model-based control and has a modular structure. An observer–controller strategy deals with the difficulty of obtaining precise velocity measurements in [40.74]. This significantly improves the chattering of the output actuator caused by noise and quantization.

Aerial mobile manipulation has been actively investigated in recent years. Aerial mobile manipulation has different characteristics in another sense: stability and load capability. The dynamics of the manipulator or interactions with the environment can create stability problem in aerial mobile manipulation. Also, interaction forces and load are limited due to the capability of lift provided by the aerial platform.

An aerial mobile manipulator system with three 2-DOF arms is developed in [40.75]. The interaction between the manipulators and quadrotors is modeled so that the reaction forces and torques during flight and manipulation are compensated for stable flight.

A Cartesian impedance control is implemented in [40.76] and [40.77], while the task redundancy due to the added DOF from the aerial vehicle is used for secondary tasks in [40.76]. Hybrid force and motion control of tool operation in quadrotors are proposed in [40.78]. The dynamics of quadrotors are transformed into the tool-tip position and decomposed into the tangential and normal directions with respect to the contact surface. Then, a stabilizing controller is designed for the hybrid motion and force control.

40.2.3 Toward Control in Mobile Manipulation

Control of mobile manipulators is one of the most investigated topics since the beginning of the concept of mobile manipulation. Although the issues discussed so far in this chapter are still very important and need to be further investigated, interaction with humans and the environment is another important aspect to handle to enable mobile manipulators to come into our daily life. In this respect, force control at the end-effector of the mobile manipulator has been studied in [40.79, 80]. The issue of contact transition has been studied for nonholonomic mobile manipulators in [40.81]. A control strategy is proposed to deal with impulsive contact force on the mobile manipulator in [40.82].

On the other hand, the use of compliant actuators such as series elastic actuators (SEA) and variable stiffness actuators (VSAs) is expected to become more

popular. The topic of compliant actuators is covered extensively Chap. 21. The SEA has been developed for the purpose of safety and energy efficiency in the compromise of precise control. The VSA can overcome the disadvantage of SEA with additional actuation for changing stiffness.

Despite the complexity of implementing these actuators in terms of both hardware and algorithms, they are expected to be used more because there is an increasing need for safe robot–human or robot–environment interaction. The application of compliant actuators to mobile manipulators would provide new challenges to the control issues of dealing with uncertainty and stability in addition to the precision of the manipulator itself. The compliance on the joints will provide larger uncertainty in the manipulator. This uncertainty affects not only the control of the manipulator but also the control of the mobile platform. The effect on the mobile base will be the stability problem in addition to the precision in control. Nevertheless, the introduction of soft/compliant actuators will be one of the important future directions to enable safe interaction in mobile manipulation.

40.3 Motion Generation

Methods to generate the motion of robots are fundamental to robotics. They are treated extensively in this Handbook (Chaps. 7, 8 as well as the entire Part E). As in the previous sections of this chapter, we will consider the state of the art in motion generation in the context of mobile manipulation. We will identify aspects of motion in mobile manipulation currently not fully addressed and will speculate about how these gaps might be closed.

In the context of mobile manipulation, the generation of robot motion poses a number of challenges, related to those laid out in the introduction to this chapter. Robotic systems able to perform a variety of tasks must possess versatile motion capabilities. This is usually realized through a large number of degrees of freedom, leading to high-dimensional motion generation problems. The humanoid robot Justin, for example, possesses 58 DOFs; the latest version of the Honda ASIMO has 57 DOFs. In addition to the high dimensionality of the associated configuration spaces, uncertainty represents a major challenges when generating robot motion. For applications in mobile manipulation, one simply cannot assume that a precise and complete world model is available at all times. The generation of motion therefore must consider the uncertainties present in the robot's world model. Furthermore, it is unrealistic to assume that the entire world will be perceivable all at once. This implies that the robot's world model will always be partial and potentially wrong in important ways. All this is compounded by the fact that the robots sensors and actuators themselves are prone to uncertainty. It is therefore apt to state that motion generation in mobile manipulation is subject to two of the problems we identified in the introduction: high dimensionality and uncertainty.

In this section, we will examine major branches of research in motion generation and analyze the degree to which they are poised to address the challenges of motion generation in manipulation. But before that, of course, we must discuss what exactly these challenges are.

40.3.1 Problem Description

What are the requirements for robot motion in the context of mobile manipulation? Just as in the classical motion planning problem, a robot must be able to move from one place to another. This requires information about the spatial extent of objects and the resulting global connectivity of world. Algorithms that address this most basic requirement of robot motion are discussed extensively in Chap. 7.

In addition to global, goal-oriented motion, a mobile manipulator must also be able to maintain task constraints during motion: holding a glass of water upright, pointing a camera in an inspection task, or coordinating motion with another robot during cooperative manipulation. At the same time, it must be able to react quickly to unforeseen changes in the environment; global motion planning might not be fast enough and therefore has to be complemented by reactive obstacle avoidance.

While performing this task-consistent, global, and reactive motion, the safe and efficient movements of a mobile manipulator may also depend on kinematic, dynamic, or postural constraints. Postural constraints may be used to perform additional tasks or simply to move in the most energy economical posture to extend battery life.

Finally, all of the motion requirements (global, task-consistent, reactive, joint-limit avoiding, posture-optimizing) must be achieved and maintained in the presence of various sources of uncertainty. In mobile manipulation, there will always be uncertainty about the world model, which might be incomplete or partially

wrong; changes to the world model may happen without knowing the robot. And, of course, there is uncertainty in regards to sensing, affecting the robot's knowledge of its own state or the ability to predict the outcome of its actions.

As we will see, most existing areas of research in motion generation today cannot address all the aforementioned requirements at the same time. Furthermore, these motion requirements also necessitate sensing capabilities. It is therefore difficult to treat motion generation for mobile manipulation independently from sensing and perception.

40.3.2 Evaluation of the State of the Art

Sampling-based motion planning is the de facto standard for motion planning in robotics; the state of the art in that field is described in detail in appendix of Chap. 7. Here, we want to show some work that extends standard methods, such as probabilistic roadmaps (PRMs) and rapidly-exploring random trees (RRTs), to address some of the motion requirements of mobile manipulation.

Motion Planning

Motion planners have been devised to account for the changes in the configuration space when robots manipulate objects by picking and placing them, possibly handing them off to another robot in order to fulfill a motion task [40.83]. This enables complex sequences of repeated reach and grasps in very tight environments to achieve a planning objective.

Other planners enable the combination of global motion with task constraints [40.84, 85]. The resulting planners generate globally goal-directed motion while maintaining task constraints at the robot's end-effector. This enables collision-free motion while maintaining a fixed orientation of the end-effector, for example. This is realized by identifying lower dimensional, task-consistent manifolds in configuration space and confining possible motions to those manifolds. This is usually achieved through iterative optimization of sampled configurations, adding the computational complexity of global motion planning.

Planners are also able to devise global motion that is consistent with kinodynamic constraints and with actuation limits of robotic platforms. Kinodynamic planning addresses, for example, constraints imposed by nonholonomicity or by the dynamics of the robot [40.86]. Actuation limits become important when a robot is unable to lift a heavy load with its arm fully extended. In such a situation, a planner can generate a motion to slide the object closer to a region of the workspace where the mechanical advantage of the robot kinematics enables the desired motion objective [40.87].

Global motion planning is computationally expensive. Therefore, it generally cannot operate at reactive rates. As a result, they are unable to satisfy the motion requirement of reactive obstacle avoidance. Of course, ideally one would determine a global motion plan at high frequencies; then planners would be able to satisfy global and reactive motion requirements. To overcome the provable computational complexity of planning, one has to resort to approximation algorithms. In other words, one has to trade completeness for computational efficiency. While there are probably many ways of accomplishing this, which may be the subject of future research, one possibility presented in the literature is the balancing of exploration and exploitation [40.88]. By replacing the exploration of motion planners with exploitation on those areas of configuration space where helpful information is available, it is possible to speed up planning by more than one order of magnitude.

Common to all of the motion planning approaches in this section is that they ignore uncertainty. They assume a perfect geometric world model and perfect execution capabilities. These assumptions generally do not hold in mobile manipulation. Hence, it will be difficult to extend traditional motion planning approaches so as to address the entire spectrum of motion requirements described above.

Further information regarding motion generation for mobile manipulation, in particular in the context of humanoid robots, has been collected in a special volume [40.89].

Feedback Motion Planning

Feedback motion planning [40.86, 90] addresses a particular type of uncertainty: actuation uncertainty. With this type of uncertainty, the robot might end up in a different state than it intended. An appropriate plan must therefore contain the correct action to take in every possible configuration the robot might end up in. Such a plan is called a navigation function. Given such a navigation function, the robot can perform Markovian gradient descent until the goal is reached.

Feedback motion planning also assumes a perfect knowledge of the world. The ability to address actuation uncertainty comes at a significant computational cost. Instead of having to compute a single path, an implicit representation of *all* paths has to be determined.

Trajectory Modification

There is an appealing complementarity between motion planning and control methods. While the former are good at addressing global motion constraints, the latter excel at reactive behavior, including obstacle avoid-

ance, force control, maintaining task constraints, etc. If the advantages of these two approaches could be combined, one would have made significant progress toward motion generation for mobile manipulation.

The simplest way of achieving this objective is through consecutive invocation: first determine a path consistent with global motion requirements, then apply control methods to the entire path to continuously optimize in response to changes in the environment or task constraints. This latter part, applying control to an entire path or trajectory as opposed to a single robot configuration, is called incremental trajectory modification.

All methods in this category perform optimization of a function, encoding various desirable properties of the trajectory, through some kind of gradient descent. The earliest method, called elastic bands [40.91], applies repulsive forces from obstacles and attractive internal forces to a discretization of the configuration space trajectory, leading to a behavior reminiscent of elastic material. This method was extended in the elastic strip framework [40.92] by applying these forces directly in the workspace to a discretization of the volume swept by the robot along its configuration space trajectory. Approaching obstacles deform the trajectory; when obstacles recede, the trajectory is shortened by the internal forces.

In contrast to the local reaction forces computed by the elastic band and elastic strip approaches, other approaches optimize trajectories globally in accordance with various criteria of optimality. This optimization problem can be cast as approximate probabilistic inference [40.93], then using message passing algorithms to find the maximum likelihood trajectory for a given objective. Alternatively, the optimization problem can be solved using the Hamiltonian Monte Carlo algorithm [40.94], where covariant functional gradient descent techniques perform local optimization. A stochastic variation of this method seems to be able to deal better with local minima [40.95].

Even though trajectory modification methods are often times labeled as motion planning methods, they do not perform global planning. They have been shown to work well in many realistic scenarios but can fail when gradient information is insufficient to find a solution. As a result, trajectory modification can serve as a powerful component for motion generation, but they cannot address all requirements of motion generation in mobile manipulation.

Integrating Planning and Control

Traditionally, global motion planners produce paths or trajectories that then get passed to the robot for execution. At execution time, these trajectories can be modified, for example, by the methods described in the previous section. It is striking, however, that a planner determines an entire trajectory from start to goal, knowing well that in dynamic environments most of the trajectory will be invalidated within seconds due to the motion of obstacles.

To fully leverage the complementarity of global planning and control mentioned in the previous section, it might be necessary to shift the boundary between these two fields. This would imply that, instead of computing a detailed trajectory, global motion planning would only provide as much information as is required to avoid local minima during control. This leads to the idea of sequencing local controllers, which can be visualized as funnels, in such a way that a global motion task can be achieved [40.96]. Referring to Fig. 40.5, each controller can be viewed as a local plan for region of the configuration space (in the image, the configuration space is the imagined plane above which the funnels are arranged). Each funnel has an exit (the converged state of the controller) that leads into the opening of the next funnel. Arrangements of these funnels represent robust motion plans as each funnel (controller) inherently rejects uncertainty.

Such a shift of the boundary between planning and control is realized in the Elastic Roadmap approach [40.97]. It combines global planning with trajectory modification to yield an approach to motion generation capable of generating global, task-consistent, reactive, and constraint-respecting motion. It does so, however, by deliberately trading completeness for computational efficiency. While this approach works well in practices, no performance guarantees can be made. Also, this approach is not able to address uncertainty; it still assumes that the world and its changes are perfectly known.

Reasoning About Uncertainty

Partially observable Markov decision processes (POMDP) are a general framework to formalize

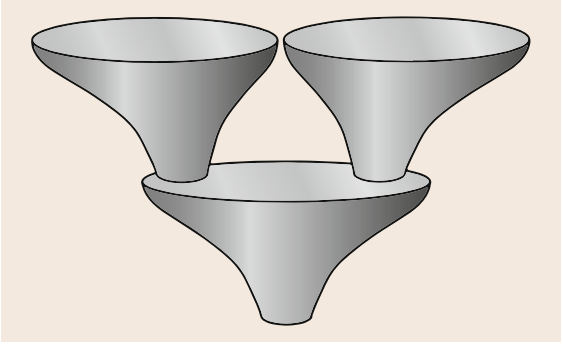


Fig. 40.5 Representing a motion plan as a sequence of funnels

problems involving actions that perform uncertain transition (uncertain outcome) between uncertain (not fully known) states (Chap. 15). POMDPs are so general that they can easily characterize the type of motion planning problem occurring in mobile manipulation. A downside of their generality and expressive power is the computational complexity of finding an exact solution, which is often intractable. To overcome this problem, one must resort to approximations.

Point-based POMDP solvers [40.98] provide such an effective approximation for the case that there is an uncertainty about the robot's state. The key idea of point-based solvers is to sample only parts of the solution space, avoiding regions unlikely to contain a solution [40.99]. Effectively, these methods exploit additional knowledge about the specific problem to guide the search for a solution.

In the context of motion generation, the roadmaps generated by traditional sampling-based motion planners can provide such knowledge for finding solutions to a specific motion planning problem [40.100, 101] or to any motion planning query in a given environment [40.4].

Considering uncertainty not only for the robot's state but also about the world model makes the problem even more difficult. Here also, it becomes necessary to approximate the state space of the world, leading to discrete approximations of the corresponding state space [40.102, 103].

While all the methods mentioned so far successfully address uncertainty, their computational complexity prevents them from being reactive. This means that all initial assumptions about the world, including assumptions about uncertainty distributions (encoded in the POMDP), must be correct and remain correct throughout the entire motion. In the context of mobile manipulation, it is difficult to think that even the initial assumptions could be correct.

If one does not want to rely on fixed assumptions about uncertainty, for example in the state transition function, one has to rely on sensing. For example, a robot might be able to reduce the uncertainty about its own state sufficiently well to be able to avoid reasoning about this type of uncertainty. But this in itself is a very strong assumption. By making such strong assumptions, it is possible to reduce the complexity of addressing uncertainty in a meaningful way, leading to a polynomial time to address uncertainty in the context of mobile manipulation [40.104]. While this approach achieves reactivity, task consistency, and is able to address other constraints, it is incomplete and no performance guarantees exist.

40.3.3 Toward Motion Generation for Mobile Manipulation

The literature contains a number of powerful technical tools and conceptual approaches that have proven effective in the context of motion generation: sampling-based motion planners, planning for feedback, optimization for incremental trajectory modification, sequencing of controllers, shifting the boundary between planning and control, and approximative solutions to POMDP-based formalization of motion planning problems. The use of assumptions to improve the computational efficiency is inherent to all of them – no surprise, given that even simple motion planning problems are already PSPACE-complete.

For each of the aforementioned motion requirements in mobile manipulation, when considered in isolation, an adequate solution exists. But no method or approach addresses all of them in coherent fashion with characterizable performance. Nevertheless, in practice, motion generation capabilities exist for many interesting applications in mobile manipulation.

With respect to the goals laid out in the introduction to this chapter, this discussion has exposed, however, that all approaches still depend on detailed prior knowledge about the environment. Most of this prior knowledge is difficult to obtain in the context of mobile manipulation. For example, accurate geometric models of the environment are unlikely to be obtainable at a global scale. And even if they were attainable, they will likely be outdated very quickly and would require constant updating. It might be even more difficult to obtain adequate characterizations of uncertainty for arbitrary motion problems.

In response to these difficulties, it is highly likely that the close integration of real-world sensing into motion generation will become a more active area of research. This sensing must be used to alleviate the needs for prior world models. This development might parallel the *shifting the boundary* between planning and control, only this time between planning and sensing. Instead of performing planning on very detailed and all encompassing world models, it might be more practical to plan in a coarse and approximate world world, leaving it to perception to fill in the details as they become relevant.

40.4 Learning

We mentioned high dimensionality, uncertainty, and variability as core challenges for robotic systems that leave prestructured lab environments. Learning is the core approach to cope with the variability of tasks a robot may need to perform in unstructured environments.

The topic of learning is orthogonal to all of the other topics discussed in this section: Learning methods are successfully applied to grasping and manipulation, mobility, control, motion generation, and perception. We will highlight representative approaches in these areas below, but first we want to discuss in more general terms the relation between the areas of machine learning and robotics.

A core question is whether existing machine learning methods can without need for modification be applied in robotic domains. Or whether genuinely novel issues are raised in the context of robot learning that necessarily go beyond core machine learning research. This is fundamentally related to the question *Is there a generic prior*?

In the Bayesian view, learning means computing the posterior

$$P(M|D) = \frac{P(D|M)P(M)}{P(D)} \,. \tag{40.1}$$

A similar discussion could refer to the choice of hypothesis space and regularization. But for simplicity we take the Bayesian view as a reference. The machine-learning community has made great progress in two respects: First, it developed a rapidly growing set of general and fundamental modeling tools to formalize learning problems, including standard parametric and nonparametric models, graphical models, infinite (hierarchical Dirichlet process) models, models combining probabilities with first-order logic, random set models, etc.

Second, the community has developed a large set of methods to solve learning problems, i. e., compute the posterior or optimize discriminative models, including a wide range of probabilistic inference methods and leveraging the tight relations to the field of (constrained) optimization.

With this range of modeling formalisms and computational methods available, what problems are left? Interestingly, a core issue remains the concrete specification of the prior $P(M)$ or hypothesis space itself. An example is the current research on deep representations: *Weston* et al. [40.105], for instance, argue that deep learning can concisely be understood as a form of regularization of intermediate representations, including auto-encoders as special case but generalizing it to arbitrary choices of embeddings. Such a regularization is one-to-one with the choice of model prior in the Bayesian formulation of learning methods. The large majority of (non-deep) regression and classification models are linear in some features (we include Hilbert space features here). The regularization is typically generic: L_2 or L_1 on the respective parameters. Therefore, the model prior is primarily determined by the choice of features.

Researchers generally agree that a core challenge of machine learning is in the specification of the prior $P(M)$ – respectively the choice of features [40.106, 107]. However, is the question about the *right* prior a genuine machine learning question? In other terms: Does there exist a *generic* prior that might perform well in any domain, or is a good choice of prior inherently domain specific, requiring expert knowledge?

Related to this question is the idea of *learning the prior* – or learning the right features. Of course, in the strict sense, a prior must not be data based. When discussing the learning of features, people usually presume a hyperprior that implies a class of potential features. However, it is questionable whether there exist generic hyperpriors over feature classes that would allow systems to learn arbitrary features suited in any domain. Some aspects of priors seem generic and are discussed extensively within machine learning: L_1 or L_2 regularization to enforce sparsity and penalize complexity; metric or topological embedding to enforce the preservation of information in internal (deep) representations. However, the structural complexity of suitable features in real-world domains seems – in the eyes of a roboticist – beyond such generic properties.

In this light, it is not surprising that the success of learning methods within robotics often relies on an insightful way to model the problem using machine learning formalisms combined with a good choice of features. Both depends on the roboticist's expert knowledge and his understanding of appropriate priors in real world domains, contrasting the idea that generic machine learning and priors could be applied for learning in robotic manipulation.

40.4.1 Problem Description

Perhaps a reason for the great success of machine learning is that its problems are – in constrast to robotics – rather straightforward to define rigorously. In most cases a learning problem is defined in terms of minimizing a loss or risk (often approximated using cross

validation), computing a posterior, or maximizing an expected return. Feature and hyperparameter selection are subsumed in these objectives.

To make this more concrete, we choose to detail one specific machine learning approach: conditional random fields (CRF) [40.108]. CRFs have become a default starting point for addressing many learning problems, especially in robotics. And CRFs are a very general way of modeling problems, as we discuss briefly in the following, and should therefore be a standard tool for roboticists.

A CRF could also be named a *conditional graphical model* or *conditional factor graph*. It describes a structured probability distribution over labels y where the factors that define this distribution additionally depend on some input x. The following summarizes the essentials of CRFs:

- We consider output variables $y = (y_1, \ldots, y_l)$ and input variables x. A *CRF* is defined by a joint function over x and y of the form

$$f(x, y) = \sum_{j=1}^{k} \phi_j(x, y_{\partial j})\beta_j = \phi(x, y)^{\mathrm{T}}\beta \,, \tag{40.2}$$

  which describes the predicted output as

$$y^*(x) = \mathrm{argmax}_y f(x, y) \,. \tag{40.3}$$

  Each feature $\phi_j(x, y_{\partial j})$ depends on a subset $y_{\partial j}$ of output variables, $\partial j \subseteq \{1, \ldots, l\}$. The function f is also called (neg-) *energy* or *discriminative function*. CRFs can be trained based on a probabilistic interpretation or with a hinge loss.
- In the *probabilistic interpretation* [40.108], CRFs define a conditional probability distribution over y,

$$p(y|x) = \frac{\mathrm{e}^{f(x,y)}}{\sum_{y'} \mathrm{e}^{f(x,y')}} = \mathrm{e}^{f(x,y)-Z(x,\beta)} \tag{40.4}$$

$$= \mathrm{e}^{-Z(x,\beta)} \prod_{j=1}^{k} \mathrm{e}^{\phi_j(x,y_{\partial j})\beta_j} \,, \tag{40.5}$$

  which is a Boltzmann distribution of f with log-partition function

$$Z(x, \beta) = \log \sum_{y'} \mathrm{e}^{f(x,y')}$$

  ensuring the conditional normalization. This is trained by minimizing

$$\beta^* = \mathrm{argmin}_\beta \, L(D, \beta) + \lambda \|\beta\| \,, \tag{40.6}$$

  where the second term is an L_2 (*Ridge*) or L_1 (*Lasso*) regularization and the first the neg-log-likelihood on data $D = \{(x_i, y_i)\}_{i=1}^N$. The optimum is found using a Newton method, using the following gradient and Hessian terms:

$$L(D, \beta) = -\sum_i \log p(y_i|x_i)$$

$$= -\sum_i [\phi(x, y)^{\mathrm{T}}\beta - Z(x_i, \beta)] \,, \tag{40.7}$$

$$\nabla_\beta Z(x, \beta) = \sum_y p(y|x) \, \phi(x, y) \,, \tag{40.8}$$

$$\nabla_\beta^2 Z(x, \beta) = \sum_y p(y|x) \, \phi(x, y) \, \phi(x, y)^{\mathrm{T}}$$

$$- \nabla_\beta Z \, [\nabla_\beta Z]^{\mathrm{T}} \,. \tag{40.9}$$

  Computing $p(y|x_i)$ (or the factor marginals $p(y_{\partial j}|x_i)$) for all x_i is analogous to the E-step of expectation maximization; the parameter update of β in each Newton iteration is analogous to the M-step. For arbitrary structures, the Hessian term $\sum_y p(y|x)\phi(x, y)\phi(x, y)^{\mathrm{T}}$ may be infeasible to compute exactly. Still, Newton methods with approximate Hessian may be significantly faster than gradient-based methods.
- In the *discriminative interpretation* [40.109], the CRF is trained by minimizing a *hinge loss*

$$\min_{\beta,\xi} \; \beta^2 + C \sum_{i=1}^{n} \xi_i$$

$$\text{s.t.} \;\; \forall_{y \neq y_i} : f(x_i, y_i) - f(x_i, y) \geq 1 - \xi_i, \;\; \xi_i \geq 1 \,, \tag{40.10}$$

  using either perceptron or linear programming methods similar to how SVMs use them [40.109]. This approach is also called *structured output SVM*.
- *Kernelization* works for both types of training. Further, in the probabilistic interpretation, the ridge regularization can be replaced by a Gaussian prior $P(\beta) = \mathcal{N}(\beta \mid 0, \sigma^2/\lambda)$ and a fully *Bayesian* predictive posterior $P(y|x, D)$ can be computed (just as for Bayesian ridge regression and Gaussian processes).
- CRFs are a very general class of learning tool. We list *special cases*, where the options concern the special case input and output spaces, *prob*abilistic or *hinge* loss training, *kernel*ization, *bayes*ianization, and of course the choice of features (explained below):
  SVM = CRF(kernel, hinge, $y \in \{0, 1\}, \phi_C$)
  logistic regression = CRF(prob, $y \in \{0, 1\}, \phi_C$)

Gaussian process (GP) = CRF(prob, bayes, kernel, $y \in \mathbb{R}$, ϕ_R).
GP classification = CRF(prob, bayes, kernel, $y \in \{0,1\}$, ϕ_C)
factor graph (MRF) = CRF(prob, $x = \emptyset$)
hidden Markov model (HMM) = CRF(prob, $y = (y_1, \ldots, y_T)$, $\{\phi_t\}$)
ridge regression = CRF(prob, $y \in \mathbb{R}$, ϕ_R)
The regression case is somewhat special, see below.

- We give three *feature examples* for the cases of multiclass classification, Markov chains (like HMMs), and the somewhat special case of regression.
For a three-class classification, one chooses

$$\phi_C(y,x) = \begin{pmatrix} [y=1]\phi(x) \\ [y=2]\phi(x) \\ [y=3]\phi(x) \end{pmatrix} \in \mathbb{R}^{3k}$$

where $y \in \{1,2,3\}$, $[expr.] \in \{0,1\}$ is the indicator function and $\phi(x) \in \mathbb{R}^k$ an arbitrary (e.g., polynomial) input feature vector. The effect of the indicators is that for each class, a different subset of β-elements define the function f.
For a Markov chain $y = (y_1, \ldots, y_T)$ with binary y_t, one chooses

$$\phi_t(y_{t+1}, y_t, x) = \begin{pmatrix} [y_t = 0 \wedge y_{t+1} = 0] \\ [y_t = 0 \wedge y_{t+1} = 1] \\ [y_t = 1 \wedge y_{t+1} = 0] \\ [y_t = 1 \wedge y_{t+1} = 1] \\ [y_t = 1]\phi_t(x) \end{pmatrix} \in \mathbb{R}^{4+k},$$

where $\beta_{1:4}$ determines the transition probabilities, and $\beta_{5:5+k-1}$ plays the same role as in the logistic regression of y_t on input features $\phi_t(x) \in \mathbb{R}^k$. If $x = (x_1, \ldots, x_T)$ is also a time series and if we choose input features $\phi_t(x) = \phi_t(x_t)$ to depend only on x_t, we are in the case of HMMs. The great strengths of Markov chain-CRFs over HMMs is that $\phi_t(x)$ may depend arbitrarily on the whole of x. In the (unusual) regression case, we may choose

$$\phi_R(y,x) = \begin{pmatrix} -\frac{1}{2}y^2/\sigma^2 \\ \phi(x)y/\sigma^2 \end{pmatrix} \in \mathbb{R}^{1+k},$$

which leads to $y^*(x) = \phi(x)^{\mathrm{T}}\beta$ and $p(y|x) = \mathcal{N}(y \mid y^*(x), \sigma^2)$ with a partition function independent of x and β. Therefore, a single Newton step leads to the well-known optimal parameters of ridge regression. GPs are the bayesian kernel variant of this.

An extreme special case is logistic regression, where the graphical model only concerns a single binary output variable y, the distribution over which depends on the input x. On the other side, also any kind of standard (non-conditional) graphical model over y is an extreme special case of a CRF that lacks the dependence on any external x. Conditional Markov chains (e.g., in linguistics), Markov random fields (in computer vision), and many other models can be viewed as instances of CRFs.

Given this concrete framework of CRFs, the problem – from the robotics perspective – essentially becomes to propose an appropriate graphical model structure over y, and to choose appropriate features $\phi(x,y)$. Both choices together correspond to the choice of hypothesis space or prior as discussed above.

40.4.2 Evaluation of the State of the Art

Perhaps the best developed area for the application of learning methods in robotics is perception.

Learning for Perception and Scene Understanding

Saxena et al. developed a series of methods that very successfully employ the general framework of CRFs in various perception tasks. In [40.110] they train CRFs to predict the depth in two-dimensional (2-D) views; [40.111] learns CRFs to classify a *grasp*-affordance in the 2-D view; [40.112] learns CRFs (or SVMs) to classify the *place*-affordance of locations; [40.113] learns CRFs to label and thereby anticipate the actions and movements of demonstrators. These works are exemplary for the power of current machine-learning methods for perception – but also in that their success relies on the creative choice of problem formalization and features. As an example, we would like to mention the unique features chosen to be indicative of concavity and place stability to predict cup-like place locations in [40.111]. Could such types of features ever be discovered autonomously by more advanced machine learning methods?

Also, for 3-D point cloud data the definition of clever features turned out to be essential for basic registration and classification problems [40.114].

With respect to our discussion of learning features, in standard image classification tasks there has recently been great success in using generic (sparse coding) priors to learn features that significantly outperform hand-coded features (like robust features (SURF) and histograms of oriented features (HOG) [40.115]. Such learned features correspond to local 2-D or 3-D appearance patterns (which define the codebook); the histogram of such patterns turns out to be an excellent basis for image classification. It seems promising that

typical 2-D CRF approaches to classify affordances – as those mentioned above – can be extended to also use learned local features. However, there may be limitations to such generically learned features. For instance, naively they do not respect any natural invariances (such as affine or illumination invariance; this can be fixed by artificially incorporating such invariances in the dataset). Using merely sparse coding priors, they are unlikely to reflect stronger priors (e.g., Gestalt laws, or kinematic understanding of rigid body scenes) of natural physical environments. For instance, the concavity feature for placement prediction seems in strong contrast to such learned features.

As a conclusion with respect to perception, the existing machine learning machinery, including novel feature learning methods have shown great success. For specific mobile manipulation tasks (like affordance prediction) this often relied on well chosen problem formalization. As is the case also in the following topics, a core question for future research are novel ideas for feature classes and priors, that are generic and support the learning of features for manipulation tasks, but incorporate stronger physical world priors than existing approaches.

Learning for Grasping

The ability to grasp is essential to manipulation and approaches to learning to grasp therefore deserves special attention. Learning to grasp has been addressed from at least two perspectives.

First, on the perceptual side, several authors have proposed methods to directly classify 2-D or 3-D appearance patterns that predict a grasp affordance [40.111]. On the positive side, this implies a very close perception–action cycle; the percept is directly translated to a potential pregrasp location without need to estimate a 3-D shape of the object or even a notion of objects and shapes. Such approaches provide a sort of heat map of potential affordances, e.g., for exploration or to reduce the number of options for manipulation. But they lack the generality of grasp approaches that take into account the goal of applying a certain force or motion to an object – but classically require a shape estimate.

Second, concerning the grasp motion itself, a basic approach is to employ policy search methods to improve a parametric dynamic movement primitive (DMP) model of the grasp motion [40.116].

Learning for Motion Generation

There exists extensive work on learning methods for control, e.g., in the form of system identification, (model-based and model-free) reinforcement learning and learning from demonstration. It is beyond the scope of this chapter to cover this. Instead, here we focus on learning in the context of path finding or trajectory planning for manipulation.

A core issue of motion generation for manipulation is the question of the right task space. For instance, the work by *Cakmak* and *Thomaz* on learning from human interaction [40.117] suggests that the question of what is the actual task space should be in the center of communication. In a certain formalization, the choice of the task space of a motion is perfectly analogous to the choice of features in machine learning: The mapping from a geometric problem setting to an appropriate motion can be formalized as a CRF [40.118], where the optimal motion maximizes a discriminative function (or neg-cost function). Typically such a cost function is a sum of squares of features, where each feature captures some error in some nonlinear task space. The problem of finding appropriate task spaces – the right features of motion – is therefore analogous to the problem of finding appropriate features in machine learning. And as for generic machine learning, the core questions are: what is a general class of potential features (of potential task spaces), and what is a possible prior over this class.

Jetchev and *Toussaint*'s task space retrieval (TRIC) approach [40.119] considers a combinatorial set of potential task dimensions that refer to various absolute and relative coordinates of objects and uses a Lasso prior to select those features that best explain a set of demonstrated trajectories. While this successfully extracts the ground-truth task spaces w.r.t. which the given trajectories have been optimized, the approach still seems limited by the crude set of allowed features. Relative coordinates (w.r.t. object centers) cannot express more meaningful and better generalizing concepts, e.g., fitting one piece into another. Further, features of manipulation motions might refer to the temporary interaction with contacts or other environmental constraints [40.120, 121]. It should be the subject of future research how such aspects could span a well-formalized set of features that leads to an appropriate prior over manipulation motions and respective optimization methods.

The above discussion views motion as being modeled by a cost (or discriminative) function. This is close to the view of motion as being modeled by an attractor (Lyapunov or potential) function; a cost function implies such a potential in the form of the cost-to-go function, and shifting the modeling directly to the cost-to-go function avoids the optimization step. The core issue of finding the right features for modeling the potential remains unchanged. TRIC [40.119] in fact models on the level of the cost-to-go instead of the cost function.

However, these views on modeling motion seem orthogonal to sample-based path finding approaches, as they traditionally operate only in configuration space and employ a discrete notion of feasible/infeasible and goal region.

Learning Sequential Manipulation from Demonstration

It is fairly recent that learning has also been applied on the level of higher level sequential manipulation. *Niekum* et al. [40.122] present a complete system to analyze a demonstrated sequential manipulation and *replay* it in a robust way on a robot, including its ability to search for alternative steps if some manipulation failed. Again, the choice of representation is the heart of this method: in [40.122] skill trees are used to represent the learned policy. *Alexandrova* et al. [40.123] extend such approaches by proposing an explicit user interface that allows the demonstrator to correct the robot's interpretation of the task and e.g., explicitly communicate the relevant task spaces.

Learning sequential manipulation is structurally one of the most interesting research challenges as it eventually requires choices of representations and priors on all levels, from perception and motion generation to the higher level representation of action and cooperation policies. The abovementioned work does first steps in the context of learning from demonstration. But much research needs to be done to enable robots to autonomously acquire skills that would allow them to control and manipulate the state of their environment as dexterously as we humans do.

40.4.3 Toward Learning in Mobile Manipulation

We above raised the question whether existing machine learning methods can without need for modification be applied in robotic domains, or whether genuinely novel issues are raised in the context of robot learning that necessarily go beyond core machine learning research. The above examples give two answers to this: First, standard machine learning tools like CRFs are widely and very successfully applied within the area mobile manipulation, employing off-the-shelf algorithms for training. Second, however, the above examples all show that the actual crux is in the modeling of the learning problem itself, in particular in the choice of features, the used regularization and the chosen structure of the output. These are issues that require substantial domain expertise. In some cases, modern approaches to feature learning or feature selection are indeed successful – particularly impressive are novel sparse coding principles to train perceptual features [40.115]. However, in many cases successful learning for mobile manipulation requires that a very specific feature space has been chosen based on the roboticist's insight (e.g., the potential operational spaces in TRIC, or the shape concavity feature to predict placement).

If we had a more fundamental theory of real-world mobile manipulation problems, we could hope that such a theory would also inform us on what are promising and general feature spaces and priors for learning in manipulation.

40.5 Perception

The need for perception is one of the main culprits in making problems in manipulation high dimensional. Technological as well as biological agents possess many sensors to perceive their environment. Humans, for example, possess on the order of 300 million nerve endings that produce a rich, continuous stream of information about the world, which is preprocessed and ultimately routed to the brain. The same is true in mobile manipulation: it is not uncommon for robots to be equipped with several cameras, each having millions of pixels, and each contributing to the high dimensionality of the input space.

To illustrate this point: A simple, black and white (not gray scale!) camera with only one thousand pixels can sense many, many more distinct images (namely, $2^{1000} \approx 10^{310}$) than there are atoms in the observable universe (about 10^{80}) or than nanoseconds have elapsed since the big bang (about 4.354×10^{26}). Even if every atom in the universe would generate one such image every nanosecond, all possible images would not have been generated by the time the universe dies its postulated heat death in 10^{100} years. Biological agents can cope with this high dimensionality, of course, because there is substantial structure in the world, i. e., many of the possible images simply do not occur in nature or some of the variability in the data is not relevant for the task at hand and can be ignored.

In this section, we will survey different successful approaches to perception for mobile manipulation that leverage this structure in perceptual data. We will mostly focus on visual perception, reflecting the emphasis of ongoing research activities. The foundations of perception in robotics are presented in great detail in Part C of this Handbook. Chapter 42 also contains relevant approaches to perception.

40.5.1 Problem Description

Perception is the process of interpreting sensory data to enable useful actions. In the context of robotics, this may be – given the state of the art – the most suitable, albeit very broad definition of the perception problem. It differs appropriately from the definition most commonly used in psychology in that the goal of perception is not solely to gain an understanding of the world, but also to use such an understanding to enable task-directed actions. This modification in fact can be seen as a simplification of the original perception problem, as only those aspects of the environment must be perceived that are relevant to a particular action. And, in fact, there is evidence from the human perceptual system that such a limitation does indeed take place, and sometimes leads to odd perceptual deficiencies [40.124].

In the context of robotics, the earliest perception work followed the psychologists' definition of perception. It focused on the use of visual perception to derive an exact 3-D model of the thing being perceived from its two-dimensional optical projection onto a camera sensor [40.125]. However, this problem is mathematically ill-posed (three-dimensional shape information cannot be reconstructed unambiguously from its two-dimensional projection) and, as we argued above, also high dimensional. To enable robust and competent perception in the context of mobile manipulation, it therefore seems sensible to attempt to reduce the complexity of perception, lending further support to our definition of perception above.

40.5.2 Evaluation of the State of the Art

Much of the work we will discuss in the remainder of this section follows the robotics-specific view of the perception problem given above. However, we will start out by surveying impressive advances made toward the objective of reconstructing three-dimensional, geometric models of the environment. Only the future can tell us if these methods are well suited to supporting the kind of complex and general manipulation tasks encountered in mobile manipulation. In the subsequent sections, we will then cover alternative approaches.

3-D Mapping and Localization

Recently, there has been remarkable progress in dense 3-D mapping and visual SLAM. What originated with DTAM [40.126] lead to current systems like LSD-SLAM [40.127] which efficiently and rather robustly estimates a 3-D environment model in real-time, without GPU, from monocular video.

Efficient Dense and Multiresolution Mapping

Recent multiscale surfel models [40.128, 129] using RGB-D sensors are an alternative to directly pixel-based reconstruction methods and provide a similarly dense 3-D model. In [40.130] and [40.131], a 3-D representation, Octree, is proposed having multiple resolutions based on a probabilistic approach. This 3-D multiresolution method can provide flexible occupancy mapping depending on the demand of the high or low resolutions. *Saarinen* et al. [40.132] proposed normal distributions transform occupancy map (NDT-OM), which has the compactness from NDT maps and the robustness from occupancy maps. This algorithm supports for multiresolution maps and it efficiently enables mapping in dynamic environments.

These developments imply that, leaving system integration problems aside, we may today assume to have access to precise 3-D mesh models of the environment. In the concrete mobile manipulation context 3-D information is needed with high resolution only when the robot is near the object or the environment. Therefore, practical techniques for multiresolution map are developed based on the distance from the robot. *Droeschel* et al. [40.133, 134] show 3-D multiresolution maps where sparse resolution is used for the environment far away and high resolution is used for the object to be interacted soon.

Mapping and Tracking with Moving Objects

Most methods for 3-D mesh estimation assume a static environment (but moving sensor). The segmentation into objects and dealing with moving objects – or rather exploiting the movement of objects – is left aside. However, fortunately recent approaches adopt a more integrated view on perception by jointly addressing the problems of dense 3-D reconstruction, moving object segmentation and tracking, and kinematic joint perception [40.135, 136], similarly as has previously been done on the basis of key points. We think that such integrated perception systems, fusing visual evidences and priors on multiple levels of representations (from low-level dense pixels to high-level rigid body hypotheses) should be the core effort of future research in this area.

Other approaches deal with general dynamic environments by efficiently updating the map continuously. Especially, in the real world problem, there are always moving obstacles and the environment itself can change over time. A incremental update algorithm is proposed in [40.137] for grid-based spatial representations for dynamic environment due to moving obstacles. Distance maps, Voronoi diagrams, and configuration-space collision maps are updated with the algorithm for the changes in the environment. In [40.138], velocity

occupancy space is proposed by adding the velocity obstacle concept into occupancy grid. This is to enable the robot to efficiently navigate through uncertain moving obstacles.

Indoor/Outdoor

Finally, the usage of a map can be different for indoor or outdoor applications. The outdoor map is mainly used for navigation whereas the indoor map is for both navigation and manipulation. The map for outdoor or field applications needs to cover wider range of area for navigation and the mobile platforms do not require high resolution maps compared to indoor applications. Indoor applications, however, typically require higher resolution even for navigation because there are more obstacles and less free space. Because of this space restriction, the navigation route needs to be planned carefully considering the obstacles. Manipulation tasks are more involved in the indoor applications such as cleaning, connecting hoses, and locking a valve, which tasks must require much higher resolution than typical navigation tasks.

For such different applications, the maps for indoor and outdoor can be quite different. Thus, in [40.139] pictures are used for the classification of the current environment. Specifically, Hue color component and color temperature values from each image are used with k-nearest neighbor (KNN) classifier to classify the environment into indoor or outdoor. *Payne* and *Singh* [40.140] proposed a simple system for the indoor–outdoor classification, which can be used as a real-time system. It is based on the assumption that indoor images contain more straight lines that outdoor environments.

Differently from indoors, the outdoor map often requires terrain information. In [40.141], the system classify if the current environment is indoor or outdoor, then terrain mapping is used for outdoor environment. It also provides seamless integration of mapping between indoor and outdoor environment.

Active and Interactive Perception

The success of the 3-D mapping approaches presented in the previous section is enabled by camera motion. This motion provides a continuous stream of incrementally changing two-dimensional projections of the three-dimensional scene. This enriched perceptual information helps resolve the ambiguities of the projection from 3-D to 2-D, thus overcoming limitations of *static* vision approaches.

Perceptual algorithms that deliberately change the sensor to optimize the perceptual processes are called *active*. Among active perception methods, active vision is probably the most prominent one [40.142–144]. But active haptic perception methods also hold promise as a perceptual approach for mobile manipulation. Active haptic perception has been used for object localization [40.145, 146], contact point localization [40.147], and for estimating the object's shape [40.148].

The concept of active perception can be extended to include interactions with the environment. Now the observer is able to manipulate the environment to reveal additional perceptual information, either by removing obstructions or by creating sensory signals that would otherwise not be observable. The class of perceptual algorithms relying on motion of the sensor as well as the deliberate application of forces to the environment is referred to as *interactive perception*.

Interactive perception eliminates the traditional boundary between perception and manipulation, combining both into a tightly coupled feedback loop. Manipulation now becomes an integral part of perception – in addition to being an objective in itself. At the same time, perception of task-relevant information is now critically enabled by manipulation. This combined perception-and-manipulation process links inseparably the goals of achieving a manipulation task and of acquiring perceptual information. These two goals must be balanced appropriately, as it is not desirable to perceive the world without progress toward the task, and it most often is not possible to achieve the task without perceiving something about the world first. Now, in addition to perception and manipulation, we must also consider aspects of machine learning: how do we balance progress toward the task (exploitation) with the goal of acquiring sufficient information to reach the goal robustly and reliably (exploration)? These and many related questions are current topics of research in this emerging field at the intersection of perception, manipulation, and machine learning.

Using interactive perception (or perceptual manipulation, if one prefers), it is possible to separate piles of objects so that they can be perceived, manipulated, and sorted [40.149–151]. It is also possible to interactively search for objects by removing obstructions [40.152]. Using interactive perception, it becomes possible to reveal sensory information about the articulations of objects in the environment, i. e., about their inherent degrees of freedom [40.153]. This ability is a perceptional prerequisite for mobile manipulation, as the degrees of freedom of an object are generally linked to its function. For example, by pushing on a door handle and observing its motion, it is possible to perceive the revolute joint connecting the handle to the door. Operating this revolute joint is necessary to open the door.

Interactive perception approaches based purely on vision data exist for planar objects [40.154] and three-

dimensional articulated objects [40.155, 156]. Their performance can be improved significantly by relying on RGB-D data [40.157, 158]. Interactive perception methods have been integrated with object databases so as to recognize previously seen objects and the associated kinematic model [40.159]. Online interactive perception enables the integration of interactive perception directly into the manipulation process [40.153]. This permits to detect failures, monitor progress, and identify the successful completion of a manipulation action. It is also possible to categorize novel types of articulation with interactive perception [40.160, 161]. Interactive perception can include additional modalities, such as proprioception and audition, for example, to identify objects [40.162].

One of the challenges remains the selection of appropriate actions for interactive perception. There are many possible ways to interact with the environment but only few of them will reveal relevant information. The selection of promising actions can be learned from experience [40.163] or by entropy-driven exploration [40.164]. The information obtained from such exploration can be represented in relational knowledge about the world [40.165].

The work discussed above has demonstrated that interactive perception is a promising approach to perception *and* manipulation in the context of mobile manipulation. It has resulted in robust perception that relies on few assumptions about the world, and in particular does not rely on the existence of prior geometric models. It also has resulted in robust manipulation, supported by the most task-relevant perceptual feedback. Interactive perception eliminates the boundaries between perception, action, and learning. While this seems like a complication, the close integration of these different aspects of robot behavior lead to novel solutions of long-standing challenges in perception and manipulation.

Affordance Detection

An affordance can be viewed as an opportunity of an agent to perform an action in its environment. The term dates back to 1977, when it was introduced by psychologist *Gibson* [40.166]. Since then the term has been used inconsistently with slight variations of meaning [40.167].

Here, we regard an affordance as an opportunity to instantiate a behavioral capability of the agent. Such an instantiation of a pushing action, for example, selects the objects involved (the robot's hand and a door handle within reach) and determines other free parameters of the behavior (pushing down on one side of the door handle).

It is appealing to focus perception of affordances as, by definition, this will focus perception only on those aspects of the environment that must be known to instantiate and successfully execute a certain behavior.

Research on affordance perception is intimately linked with research on affordance learning [40.168–170]. A modern view perspective is to consider affordance learning for manipulation as a statistical relational learning problem [40.171].

More low-level affordance perception research considers the problem of labeling scene segments by their affordance of certain actions, often using conditional random fields (CRFs) to train the labeled segmentation. This has extensively been addressed in the case of learning to detected grasp affordances [40.172–175].

40.5.3 Toward Robust Perception

Perception is one of the great challenges in mobile manipulation. We recall that the objective of mobile manipulation research is to a) maximize task generality of autonomous robotic systems, while at the same time b) minimizing the dependence on task-specific, hard-coded, or narrowly-relevant information. Given these objectives, it becomes clear that the complex and diverse information required for task generally must be acquired through perception. And perception must also compensate for the envisaged reduction of a priori knowledge available to the agent. Perception plays a central role in realizing mobile manipulation.

In this section, we surveyed some approaches designed to meet the challenges posed by the perception requirements of mobile manipulation. While a multitude of promising approaches exist, today there is no generally accepted framework or philosophy for machine perception. Evidence from human perception may suggest that such a common framework exists. Some results seem to imply that there are some shared working principles between vision and audition [40.3]. Irrespective of this, there is strong evidence that human perception is built to support action. This becomes apparent already in the retina, which is far from being a general sense organ; it contains highly specialized circuitry tailored for task-relevance [40.176]. Maybe this too can be a clue for how to address perception in robotics: Maybe we need much better low-level features?

40.6 Conclusions and Further Reading

While mobile manipulation requires the integration of methodologies from almost all aspects of robotics – we mentioned grasping, control, motion generation, learning and perception – it also suggests alternative approaches to the state of the art in these areas of research. Instead of considering each research area as independent, research in mobile manpulation exploits their interdependence when this simplifies the overall problem. Active and interactive perception, i. e., combining manipulation and 3-D perception, is one example that departs from the goals of the individual subareas: (1) general 3-D reconstruction from passive data and (2) manipulation planning given a 3-D environment model. Instead, these approaches view both as an integrated and potentially simpler problem.

In the present survey we only briefly covered the various aspects and their interdependencies. Which of the approaches will turn out to be the most successful for the integrated overall goal of mobile manipulation is an ongoing debate. An interesting starting point for further reading are the various workshops that focus on the integrated nature of mobile manipulation and robotics as a whole. A list of these workshops is maintained on the Website of the Technical Committee on Mobile Manipulation.

Video-References

VIDEO 650 Learning dexterous grasps that generalize to novel objects by combining hand and contact models
available from http://handbookofrobotics.org/view-chapter/40/videodetails/650

VIDEO 651 Atlas whole-body grasping
available from http://handbookofrobotics.org/view-chapter/40/videodetails/651

VIDEO 652 Handle localization and grasping
available from http://handbookofrobotics.org/view-chapter/40/videodetails/652

VIDEO 653 Catching objects in flight
available from http://handbookofrobotics.org/view-chapter/40/videodetails/653

VIDEO 654 Avian-inspired grasping for quadrotor micro UAVs
available from http://handbookofrobotics.org/view-chapter/40/videodetails/654

VIDEO 655 A compliant underactuated hand for robust manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/655

VIDEO 656 Yale Aerial Manipulator – Dollar Grasp Lab
available from http://handbookofrobotics.org/view-chapter/40/videodetails/656

VIDEO 657 Exploitation of environmental constraints in human and robotic grasping
available from http://handbookofrobotics.org/view-chapter/40/videodetails/657

VIDEO 658 Adaptive synergies for a humanoid robot hand
available from http://handbookofrobotics.org/view-chapter/40/videodetails/658

VIDEO 660 Universal gripper
available from http://handbookofrobotics.org/view-chapter/40/videodetails/660

VIDEO 661 DLR's Agile Justin plays catch with Rollin' Justin
available from http://handbookofrobotics.org/view-chapter/40/videodetails/661

VIDEO 662 Atlas walking and manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/662

VIDEO 664 Dynamic robot manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/664

VIDEO 665 CHOMP trajectory optimization
available from http://handbookofrobotics.org/view-chapter/40/videodetails/665

VIDEO 667 Motor skill learning for robotics
available from http://handbookofrobotics.org/view-chapter/40/videodetails/667

VIDEO 668 Policy learning
available from http://handbookofrobotics.org/view-chapter/40/videodetails/668

VIDEO 669 Autonomous robot skill acquisition
available from http://handbookofrobotics.org/view-chapter/40/videodetails/669

VIDEO 670 State representation learning for robotics
available from http://handbookofrobotics.org/view-chapter/40/videodetails/670

VIDEO 671 Extracting kinematic background knowledge from interactions using task-sensitive relational learning
available from http://handbookofrobotics.org/view-chapter/40/videodetails/671

VIDEO 673 DART: Dense articulated real-time tracking
available from http://handbookofrobotics.org/view-chapter/40/videodetails/673

VIDEO 674 Reaching in clutter with whole-arm tactile sensing
available from http://handbookofrobotics.org/view-chapter/40/videodetails/674
VIDEO 675 Adaptive force/velocity control for opening unknown doors
available from http://handbookofrobotics.org/view-chapter/40/videodetails/675
VIDEO 676 Interactive perception of articulated objects
available from http://handbookofrobotics.org/view-chapter/40/videodetails/676
VIDEO 776 A day in the life of Romeo and Juliet (Mobile Manipulators)
available from http://handbookofrobotics.org/view-chapter/40/videodetails/776
VIDEO 782 Flight stability in aerial redundant manipulator
available from http://handbookofrobotics.org/view-chapter/40/videodetails/782
VIDEO 783 HERMES, A humanoid experimental robot for mobile manipulation and exploration services
available from http://handbookofrobotics.org/view-chapter/40/videodetails/783
VIDEO 784 Task consistent Obstacle avoidance for mobile manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/784
VIDEO 785 Handling of a single object by multiple mobile robots based on caster-like dynamics
available from http://handbookofrobotics.org/view-chapter/40/videodetails/785
VIDEO 786 Rolling Justin – a platform for mobile manipulation
available from http://handbookofrobotics.org/view-chapter/40/videodetails/786
VIDEO 787 Combined mobility and manipulation – operational space control of free-flying space robots
available from http://handbookofrobotics.org/view-chapter/40/videodetails/787
VIDEO 788 Mobile robot helper
available from http://handbookofrobotics.org/view-chapter/40/videodetails/788
VIDEO 789 Free-floating autonomous underwater manipulation: Connector plug/unplug
available from http://handbookofrobotics.org/view-chapter/40/videodetails/789
VIDEO 790 Development of a versatile underwater robot – GTS ROV ALPHA
available from http://handbookofrobotics.org/view-chapter/40/videodetails/790

References

40.1 R.A. Brooks, J. Connell, P. Ning: *Herbert: A Second Generation Mobile Robot*, Technical Report, AI Memo 1016 (MIT, Cambridge 1988)
40.2 O. Khatib, K. Yokoi, O. Brock, K. Chang, A. Casal: Robots in human environments: Basic autonomous capabilities, Int. J. Robotics Res. **18**(7), 684–696 (1999)
40.3 R.J. Adams, P. Sheppard, A. Cheema, M.E. Mercer: Vision vs. hearing: Direct comparison of the human contrast sensitivity and audibility functions, J. Vis. **13**(9), 870 (2013)
40.4 A.-A. Agha-Mohammadi, S. Chakravorty, N.M. Amato: Firm: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements, Int. J. Robotics Res. **33**(2), 268–304 (2013)
40.5 A. Miller, P. Allen: Graspit! a versatile simulator for robotic grasping, IEEE Robotics Autom. Mag. **11**(4), 110–122 (2004)
40.6 A. Rodriguez, M.T. Mason: Grasp invariance, Int. J. Robotics Res. **31**(2), 236–248 (2012)
40.7 S. Hirose, Y. Umetani: The development of soft gripper for the versatile robot hand, Mech. Mach. Theory **13**(3), 351–359 (1978)
40.8 A.M. Dollar, R.D. Howe: The highly adaptive SDM hand: Design and performance evaluation, Int. J. Robotics Res. **29**(5), 585–597 (2010)
40.9 M. Ciocarlie, F.M. Hicks, S. Stanford: Kinetic and dimensional optimization for a tendon-driven gripper, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 217–224
40.10 L.U. Odhner, L.P. Jentoft, M.R. Claffee, N. Corson, Y. Tenzer, R.R. Ma, M. Buehler, R. Kohout, R.D. Howe, A.M. Dollar: A compliant, underactuated hand for robust manipulation, Int. J. Robotics Res. **33**(5), 736–752 (2014)
40.11 M.G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, A. Bicchi: Adaptive synergies for the design and control of the Pisa/IIT SoftHand, Int. J. Robotics Res. **33**(5), 768–782 (2014)
40.12 J. Amend, E. Brown, N. Rodenberg, H. Jaeger, H. Lipson: A positive pressure universal gripper based on the jamming of granular material, IEEE Trans. Robotics **28**(2), 341–350 (2012)
40.13 R. Deimel, O. Brock: A compliant hand based on a novel pneumatic actuator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) Karlsruhe (2013) pp. 472–480
40.14 M. Kazemi, J.-S. Valois, J.A.D. Bagnell, N. Pollard: *Robust object grasping using force compliant motion primitives*, Techn. Rep. CMU-RI-TR-12-04 (Carnegie Mellon University Robotics Institute, Pittsburgh 2012)
40.15 M. Dogar, S. Srinivasa: Push-grasping with dexterous hands: Mechanics and a method, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 2123–2130
40.16 L. Chang, G. Zeglin, N. Pollard: Preparatory object rotation as a human-inspired grasping strategy, IEEE-RAS Int. Conf. Humanoids (2008) pp. 527–534
40.17 N. Chavan-Dafle, A. Rodriguez, R. Paolini, B. Tang, S. Srinivasa, M. Erdmann, M.T. Mason, I. Lund-

berg, H. Staab, T. Fuhlbrigge: Extrinsic dexterity: In-hand manipulation with external forces, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)
40.18 T. Lozano-Pérez, M.T. Mason, R.H. Taylor: Automatic synthesis of fine-motion strategies for robots, Int. J. Robotics Res. **3**(1), 3–24 (1984)
40.19 M.T. Mason: The mechanics of manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1985) pp. 544–548
40.20 M. Kaneko, T. Shirai, T. Tsuji: Scale-dependent grasp, IEEE Trans. Syst. Man Cybern., A: Syst. Hum. **30**(6), 806–816 (2000)
40.21 R. Deimel, C. Eppner, J.L. Ruiz, M. Maertens, O. Brock: Exploitation of environmental constraints in human and robotic grasping, Proc. Int. Symp. Robotics Res. (2013)
40.22 M.R. Cutkosky: On grasp choice, grasp models, and the design of hands for manufacturing tasks, IEEE Trans. Robotics Autom. **5**(3), 269–279 (1989)
40.23 T. Feix, R. Pawlik, H. Schmiedmayer, J. Romero, D. Kragic: A comprehensive grasp taxonomy, Robotics Sci. Syst.: Workshop Underst. Hum. Hand Adv. Robotics Manip. (2009)
40.24 M. Santello, M. Flanders, J. Soechting: Patterns of hand motion during grasping and the influence of sensory guidance, J. Neurosci. **22**(4), 1426–1435 (2002)
40.25 V.N. Christopoulos, P.R. Schrater: Grasping objects with environmentally induced position uncertainty, PLoS Comput. Biol. **5**(10), 10 (2009)
40.26 D.R. Melmoth, A.L. Finlay, M.J. Morgan, S. Grant: Grasping deficits and adaptations in adults with stereo vision losses, Investig. Ophthalmol. Vis. Sci. **50**(8), 3711–3720 (2009)
40.27 M. Controzzi, C. Cipriani, M.C. Carozza: Design of artificial hands: A review. In: *The Human Hand as an Inspiration for Robot Hand Development*, Springer Tracts in Advanced Robotics, Vol. 95, ed. by R. Balasubramanian, V.J. Santos (Springer, Berlin, Heidlberg 2014) pp. 219–247
40.28 M. Grebenstein: Approaching Human Performance – The Functionality Driven Awiwi Robot Hand, Ph.D. Thesis (ETH, Zürich 2012)
40.29 J. Bae, S. Park, J. Park, M. Baeg, D. Kim, S. Oh: Development of a low cost anthropomorphic robot hand with high capability, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 4776–4782
40.30 I. Gaiser, S. Schulz, A. Kargov, H. Klosek, A. Bierbaum, C. Pylatiuk, R. Oberle, T. Werner, T. Asfour, G. Bretthauer, R. Dillmann: A new anthropomorphic robotic hand, 8th IEEE-RAS Int. Conf. Humanoid Robotics (Humanoids) (2008) pp. 418–422
40.31 A.M. Dollar, R.D. Howe: Simple, reliable robotic grasping for human environments, IEEE Int. Conf. Technol. Pract. Robot Appl. (TePRA) (2008) pp. 156–161
40.32 R. Ma, L. Odhner, A. Dollar: A modular, open-source 3-D printed underactuated hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)
40.33 F. Ilievski, A. Mazzeo, R.F. Shepherd, X. Chen, G.M. Whitesides: Soft robotics for chemists, Angew. Chem. Int. Ed. **50**(8), 1890–1895 (2011)
40.34 G. Grioli, M. Catalano, E. Silvestro, S. Tono, A. Bicchi: Adaptive synergies: An approach to the design of under-actuated robotic hands, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 1251–1256
40.35 R. Deimel, O. Brock: A novel type of compliant, underactuated robotic hand for dexterous grasping, Proc. Robotics Sci. Syst., Berkeley (2014)
40.36 M.E. Giannaccini, I. Georgilas, I. Horsfield, B.H.P.M. Peiris, A. Lenz, A.G. Pipe, S. Dogramadzi: A variable compliance, soft gripper, Auton. Robots **36**(1/2), 93–107 (2014)
40.37 D. Prattichizzo, M. Malvezzi, M. Gabiccini, A. Bicchi: On the manipulability ellipsoids of underactuated robotic hands with compliance, Robotics Auton. Syst. **60**(3), 337–346 (2012)
40.38 M. Gabiccini, E. Farnioli, A. Bicchi: Grasp analysis tools for synergistic underactuated robotic hands, Int. J. Robotics Res. **32**(13), 1553–1576 (2013)
40.39 Y. Li, A.A. Frank: A moving base robot, Am. Control Conf. 1986 (1986) pp. 1927–1932
40.40 Y. Yamamoto, X. Yun: Coordinating locomotion and manipulation of a mobile manipulator, Proc. 31st IEEE Conf. Decis. Control IEEE (1992) pp. 2643–2648
40.41 Y. Yamamoto, X. Yun: Control of mobile manipulators following a moving surface, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 1–6
40.42 H. Seraji: Configuration control of redundant manipulators: Theory and implementation, IEEE Trans. Robotics Autom. **5**(4), 472–490 (1989)
40.43 H. Seraji, R. Colbaugh: Improved configuration control for redundant robots, J. Robotics Syst. **7**(6), 897–928 (1990)
40.44 H. Seraji: An on-line approach to coordinated mobility and manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 28–35
40.45 H. Seraji: A unified approach to motion control of mobile manipulators, Int. J. Robotics Res. **17**(2), 107–118 (1998)
40.46 O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal: Coordination and decentralized cooperation of multiple mobile manipulators, J. Robotics Syst. **13**(11), 755–764 (1996)
40.47 V. Padois, J.-Y. Fourquet, P. Chiron: Kinematic and dynamic model-based control of wheeled mobile manipulators: A unified framework for reactive approaches, Robotica **25**(02), 157–173 (2007)
40.48 E. Papadopoulos, J. Poulakakis: Planning and model-based control for mobile manipulators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 1810–1815
40.49 J. Tan, N. Xi: Unified model approach for planning and control of mobile manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (2001) pp. 3145–3152
40.50 J. Joshi, A.A. Desrochers: Modeling and control of a mobile robot subject to disturbances, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1986) pp. 1508–1513
40.51 N. Hootsmans, S. Dubowsky: Large motion control of mobile manipulators including vehicle

suspension characteristics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1991) pp. 2336–2341
40.52 K. Liu, F.L. Lewis: Decentralized continuous robust controller for mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1990) pp. 1822–1827
40.53 J.H. Chung, S.A. Velinsky, R.A. Hess: Interaction control of a redundant mobile manipulator, Int. J. Robotics Res. **17**(12), 1302–1309 (1998)
40.54 Y. Yamamoto, X. Yun: Effect of the dynamic interaction on coordinated control of mobile manipulators, IEEE Trans. Robotics Autom. **12**(5), 816–824 (1996)
40.55 S. Lin, A.A. Goldenberg: Robust damping control of mobile manipulators, IEEE Trans. Syst. Man Cybern. B: Cybern **32**(1), 126–132 (2002)
40.56 S. Lin, A.A. Goldenberg: Neural-network control of mobile manipulators, IEEE Trans. Neural Netw. **12**(5), 1121–1133 (2001)
40.57 C.Y. Lee, I.K. Jeong, I.H. Lee, J.J. Lee: Motion control of mobile manipulator based on neural networks and error compensation, Proc. IEEE/RSJ Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4627–4632
40.58 Z. Li, S.S. Ge, A. Ming: Adaptive robust motion/force control of holonomic-constrained nonholonomic mobile manipulators, IEEE Trans. Syst. Man, Cybern. B: Cybern **37**(3), 607–616 (2007)
40.59 Y. Liu, Y. Li: Sliding mode adaptive neural-network control for nonholonomic mobile modular manipulators, J. Intell. Robotics Syst. **44**(3), 203–224 (2005)
40.60 D. Xu, D. Zhao, J. Yi, X. Tan: Trajectory tracking control of omnidirectional wheeled mobile manipulators: Robust neural network-based sliding mode approach, IEEE Trans. Syst. Man Cybern. **39**(3), 788–799 (2009)
40.61 L. Righetti, J. Buchli, M. Mistry, S. Schaal: Inverse dynamics control of floating-base robots with external constraints: A unified view, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 1085–1090
40.62 C. Ott, M.A. Roa, G. Hirzinger: Posture and balance control for biped robots based on contact force optimization, 11th IEEE-RAS Int. Conf. Humanoid Robotics (Humanoids) (2011) pp. 26–33
40.63 O. Khatib, L. Sentis, J. Park, J. Warren: Whole-body dynamic behavior and control of human-like robots, Int. J. Humanoid Robotics **1**(01), 29–43 (2004)
40.64 J. Park, O. Khatib: Contact consistent control framework for humanoid robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1963–1969
40.65 M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, R. Siegwart: Quadrupedal locomotion using hierarchical operational space control, Int. J. Robotics Res. **33**, 1062 (2014)
40.66 S. McMillan, D.E. Orin, R.B. McGhee: Efficient dynamic simulation of an underwater vehicle with a robotic manipulator, IEEE Trans. Syst. Man Cybern. **25**(8), 1194–1206 (1995)
40.67 T.W. McLain, S.M. Rock, M.J. Lee: Experiments in the coordinated control of an underwater arm/vehicle system. In: *Underwater Robots*, ed. by J. Yuh, T. Ura, G.A. Bekey (Kluwer, Boston 1996) pp. 139–158
40.68 T.J. Tarn, G.A. Shoults, S.P. Yang: A dynamic model of an underwater vehicle with a robotic manipulator using Kane's method. In: *Underwater Robots*, ed. by J. Yuh, T. Ura, G. Bekey (Kluwer, Boston 1996) pp. 195–209
40.69 G. Antonelli, S. Chiaverini: Task-priority redundancy resolution for underwater vehicle-manipulator systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (1998) pp. 768–773
40.70 J. Han, J. Park, W.K. Chung: Robust coordinated motion control of an underwater vehicle-manipulator system with minimizing restoring moments, Ocean Eng. **38**(10), 1197–1206 (2011)
40.71 H. Mahesh, J. Yuh, R. Lakshmi: A coordinated control of an underwater vehicle and robotic manipulator, J. Robotics Syst. **8**(3), 339–370 (1991)
40.72 S. Kawamura, N. Sakagami: Analysis on dynamics of underwater robot manipulators based on iterative learning control and time-scale transformation, IEEE Int. Conf. Robotics Autom (ICRA), Vol. 2 (2002) pp. 1088–1094
40.73 G. Antonelli, F. Caccavale, S. Chiaverini: Adaptive tracking control of underwater vehicle-manipulator systems based on the virtual decomposition approach, IEEE Trans. Robotics Autom. **20**(3), 594–602 (2004)
40.74 G. Antonelli, F. Caccavale, S. Chiaverini, L. Villani: Tracking control for underwater vehicle-manipulator systems with velocity estimation, IEEE J. Ocean. Eng. **25**(3), 399–413 (2000)
40.75 M. Orsag, C. Korpela, P. Oh: Modeling and control of mm-uav: Mobile manipulating unmanned aerial vehicle, J. Intell. Robotics Syst. **69**(1–4), 227–240 (2013)
40.76 V. Lippiello, F. Ruggiero: Exploiting redundancy in cartesian impedance control of uavs equipped with a robotic arm, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 3768–3773
40.77 F. Forte, R. Naldi, A. Macchelli, L. Marconi: Impedance control of an aerial manipulator, Am. Control Conf. (ACC) (2002) pp. 3839–3844
40.78 H.-N. Nguyen, D. Lee: Hybrid force/motion control and internal dynamics of quadrotors for tool operation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 3458–3464
40.79 F. Inoue, T. Murakami, K. Ohnishi: A motion control of mobile manipulator with external force, IEEE/ASME Trans. Mechatron. **6**(2), 137–142 (2001)
40.80 Z. Li, J. Gu, A. Ming, C. Xu, M. Shinojo: Intelligent compliant force/motion control of nonholonomic mobile manipulator working on the nonrigid surface, Neural Comput. Appl. **15**(3/4), 204–216 (2006)
40.81 V. Padois, J.-Y. Fourquet, P. Chiron, M. Renaud: On contact transition for nonholonomic mobile manipulators. In: *Experimental Robotics IX*, ed. by O. Khatib, V. Kumar, D. Rus (Springer, Berlin, Heidelberg 2006) pp. 207–216

40.82 S. Kang, K. Komoriya, K. Yokoi, T. Koutoku, B. Kim, S. Park: Control of impulsive contact force between mobile manipulator and environment using effective mass and damping controls, Int. J. Precis. Eng. Manuf. **11**(5), 697–704 (2010)

40.83 T. Simèon, J.-P. Laumond, J. Cortès, A. Sahbani: Manipulation planning with probabilistic roadmaps, Int. J. Robotics Res. **23**(7/8), 729–746 (2004)

40.84 M. Stilman: Global manipulation planning in robot joint space with task constraints, IEEE Trans. Robotics **26**(3), 576–584 (2010)

40.85 D. Berenson, S.S. Srinivasa, J. Kuffner: Task space regions: A framework for pose-constrained manipulation planning, Int. J. Robotics Res. **30**(12), 1435–1460 (2011)

40.86 S.M. LaValle: *Planning Algorithms* (Cambridge Univ. Press, Cambridge 2004)

40.87 D. Berenson, S. Srinivasa, D. Ferguson, J.J. Kuffner: Manipulation planning on constraint manifolds, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 625–632

40.88 M. Rickert, A. Sieverling, O. Brock: Balancing exploration and exploitation in sampling-based motion planning, IEEE Trans. Robotics **30**(6), 1305–1317 (2014)

40.89 K. Harada, E. Yoshida, K. Yokoi (Eds.): *Motion Planning for Humanoid Robots* (Springer, Berlin, Heidelberg 2010)

40.90 W. Choi, J.-C. Latombe: A reactive architecture for planning and executing robot motions with incomplete knowledge, Proc. RSJ/IEEE Int. Conf. Robotics Intell. Syst. (IROS) Osaka, Vol. 1 (1991) pp. 24–29

40.91 S. Quinlan, O. Khatib: Elastic Bands: Connecting path planning and control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (1993) pp. 802–807

40.92 O. Brock, O. Khatib: Elastic strips: A framework for motion generation in human environments, Int. J. Robotics Autm. **21**(10), 1031–1052 (2014)

40.93 M. Toussaint: Robot trajectory optimization using approximate inference, Proc. 26th Ann. Int. Conf. Mach. Learn. (2009) pp. 1049–1056

40.94 M. Zucker, N. Ratliff, A.D. Dragan, M. Pivtoraiko, M. Klingensmith, C.M. Dellin, J.A. Bagnell, S.S. Srinivasa: CHOMP: Covariant Hamiltonian optimization for motion planning, Int. J. Robotics Res. **32**(9/10), 1164–1193 (2013)

40.95 M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, S. Schaal: STOMP: Stochastic trajectory optimization for motion planning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 4569–4574

40.96 R.R. Burridge, A.A. Rizzi, D.E. Koditschek: Sequential composition of dynamically dexterous robot behaviors, Int. J. Robotics Res. **18**(6), 534–555 (1999)

40.97 Y. Yang, O. Brock: Elastic roadmaps—motion generation for autonomous mobile manipulation, Auton. Robots **28**(1), 113–130 (2010)

40.98 J. Pineau, G. Gordon, S. Thrun: Point-based value iteration: An anytime algorithm for POMDPs, Int. Jt. Conf. Artif. Intell. (IJCAI) (2003) pp. 1025–1032

40.99 H. Kurniawati, D. Hsu, W.S. Lee: SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces, Robotics Sci. Syst. (2008)

40.100 J. Van Den Berg, P. Abbeel, K. Goldberg: LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information, Int. J. Robotics Res. **30**(7), 895–913 (2011)

40.101 S. Prentice, N. Roy: The belief roadmap: Efficient planning in belief space by factoring the covariance, Int. J. Robotics Res. **28**(11), 1448–1465 (2009)

40.102 H. Kurniawati, T. Bandyopadhyay, N.M. Patrikalakis: Global motion planning under uncertain motion, sensing, and environment map, Auton. Robots **33**(3), 255–272 (2012)

40.103 L.P. Kaelbling, T. Lozano-Pérez: Integrated task and motion planning in belief space, Int. J. Robotics Res. **32**(9/10), 1194–1227 (2013)

40.104 A. Sieverling, N. Kuhnen, O. Brock: Sensor-based, task-constrained motion generation under uncertainty, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

40.105 J. Weston, F. Ratle, R. Collobert: Deep learning via semi-supervised embedding, Proc. 25th Int. Conf. Mach. Learn. (ICML) (2008)

40.106 T.G. Dietterich, P. Domingos, L. Getoor, S. Muggleton, P. Tadepalli: Structured machine learning: The next ten years, Mach. Learn. **73**, 3–23 (2008)

40.107 R. Douglas, T. Sejnowski: *Future Challenges for the Science and Engineering of Learning*, Final NSF Workshop Report (NSF, Arlington County 2008)

40.108 J. Lafferty, A. McCallum, F. Pereira: Conditional random fields: Probabilistic models for segmenting and labeling sequence data, Int. Conf. Mach. Learn. (ICML) (2001) pp. 282–289

40.109 Y. Altun, I. Tsochantaridis, T. Hofmann: Hidden Markov support vector machines, Proc. 20th Int. Conf. Mach. Learn. (2003) pp. 3–10

40.110 A. Saxena, M. Sun, A.Y. Ng: Make 3-D: Depth perception from a single still image, Proc. 23rd AAAI Conf. Artif. Intell. (2008) pp. 1571–1576

40.111 A. Saxena, J. Driemeyer, A.Y. Ng: Robotic grasping of novel objects using vision, Int. J. Robotics Res. **27**(2), 157–173 (2008)

40.112 Y. Jiang, M. Lim, C. Zheng, A. Saxena: Learning to place new objects in a scene, Int. J. Robotics Res. **31**(9), 1021–1043 (2012)

40.113 Y. Jiang, A. Saxena: Modeling and control of a mobile robot subject to disturbances, Proc. Robotics Sci. Syst., Berkeley (2014)

40.114 R.B. Rusu, N. Blodow, M. Beetz: Fast point feature histograms (fpfh) for 3-D registration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 3212–3217

40.115 K. Lai, L. Bo, D. Fox: Unsupervised feature learning for 3-D scene labeling, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

40.116 F. Stulp, E. Theodorou, J. Buchli, S. Schaal: Learning to grasp under uncertainty, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5703–5708

40.117 M. Cakmak, A.L. Thomaz: Designing robot learners that ask good questions, Proc. Int. Conf. Hum.-Robot Interact. (HRI) (2012)

40.118 N. Jetchev, M. Toussaint: Fast motion planning from experience: Trajectory prediction for speeding up movement generation, Auton. Robots **34**, 111–127 (2013)

40.119 N. Jetchev, M. Toussaint: Discovering relevant task spaces using inverse feedback control, Auton. Robots **37**, 169–189 (2014)

40.120 C. Eppner, O. Brock: Grasping unknown objects by exploiting shape adaptability and environmental constraints, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013)

40.121 M. Toussaint, N. Ratliff, J. Bohg, L. Righetti, P. Englert, S. Schaal: Dual execution of optimized contact interaction trajectories, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2014)

40.122 S. Niekum, S. Osentoski, G. Konidaris, A.G. Barto: Learning and generalization of complex tasks from unstructured demonstrations, IEEE/RSJ Int. Conf. Intell. Robotics Systems (IROS) (2012) pp. 5239–5246

40.123 S. Alexandrova, M. Cakmak, K. Hsiao, L. Takayama: Robot programming by demonstration with interactive action visualizations, Proc. Robotics Sci. Syst. (2014)

40.124 M.M. Chun, R. Marois: The dark side of visual attention, Curr. Opin. Neurobiol. **12**(2), 184–189 (2002)

40.125 D. Marr: *Vision: A Computational Investigation Into the Human Representation and Processing of Visual Information* (Henry Holt, New York 1982)

40.126 R.A. Newcombe, S.J. Lovegrove, A.J. Davison: Dtam: Dense tracking and mapping in real-time, IEEE Int. Conf. Comput. Vis. (ICCV) (2011) pp. 2320–2327

40.127 J. Engel, T. Schöps, D. Cremers: LSD-SLAM: Large-scale direct monocular SLAM, Eur. Conf. Comput. Vis. (ECCV) (2014)

40.128 P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox: Rgb-d mapping: Using kinect-style depth cameras for dense 3-D modeling of indoor environments, Int. J. Robotics Res. **31**(5), 647–663 (2012)

40.129 J. Stückler, S. Behnke: Multi-resolution surfel maps for efficient dense 3-D modeling and tracking, J. Vis. Commun. Image Represent. **25**(1), 137–147 (2014)

40.130 K.M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, W. Burgard: Octomap: A probabilistic, flexible, and compact 3-D map representation for robotic systems, Proc. ICRA 2010 Workshop Best Practice 3-D Percept. Model. Mob. Manip., Vol. 2 (2010)

40.131 A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard: Octomap: An efficient probabilistic 3-D mapping framework based on octrees, Auton. Robots **34**(3), 189–206 (2013)

40.132 J.P. Saarinen, H. Andreasson, T. Stoyanov, A.J. Lilienthal: 3-D normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments, Int. J. Robotics Res. **32**(14), 1627–1644 (2013)

40.133 D. Droeschel, J. Stückler, S. Behnke: Local multi-resolution representation for 6d motion estimation and mapping with a continuously rotating 3-D laser scanner, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

40.134 D. Droeschel, J. Stückler, S. Behnke: Local multiresolution surfel grids for MAV motion estimation and 3-D mapping, Proc. Int. Conf. Intell. Auton. Syst. (IAS) (2014)

40.135 M. McElhone, J. Stuckler, S. Behnke: Joint detection and pose tracking of multi-resolution surfel models in RGB-D, IEEE Eur. Conf. Mobile Robots (ECMR) (2013) pp. 131–137

40.136 J. Stückler, B. Waldvogel, H. Schulz, S. Behnke: Dense real-time mapping of object-class semantics from RGB-D video, J. Real-Time Image Process. **8**, 1–11 (2014)

40.137 B. Lau, C. Sprunk, W. Burgard: Efficient grid-based spatial representations for robot navigation in dynamic environments, Robotics Auton. Syst. **61**(10), 1116–1130 (2013)

40.138 R. Bis, H. Peng, G. Ulsoy: Velocity occupancy space: Robot navigation and moving obstacle avoidance with sensor uncertainty, ASME Dyn. Syst. Control Conf. (2009) pp. 363–370

40.139 A.N. Ghomsheh, A. Talebpour: A new method for indoor-outdoor image classification using color correlated temperature, Int. J. Image Process **6**(3), 167–181 (2012)

40.140 A. Payne, S. Singh: Indoor vs. outdoor scene classification in digital photographs, Pattern Recogn. **38**(10), 1533–1545 (2005)

40.141 J. Collier, A. Ramirez-Serrano: Environment classification for indoor/outdoor robotic mapping, IEEE Can. Conf. Comput. Robot Vis. (CRV) (2009) pp. 276–283

40.142 J. Aloimonos, I. Weiss, A. Bandyopadhyay: Active vision, Int. J. Comput. Vis. **1**(4), 333–356 (1988)

40.143 R. Bajcsy: Active perception, Proceedings IEEE **76**(8), 996–1006 (1988)

40.144 A. Blake, A. Yuille: *Active Vision* (MIT, Cambridge 1992)

40.145 A. Petrovskaya, O. Khatib: Global localization of objects via touch, IEEE Trans. Robotics **27**(3), 569–585 (2011)

40.146 P. Hebert, T. Howard, N. Hudson, J. Ma, J.W. Burdick: The next best touch for model-based localization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 99–106

40.147 H. Lee, J. Park: An active sensing strategy for contact location without tactile sensors using robot geometry and kinematics, Auton. Robots **36**(1/2), 109–121 (2014)

40.148 M.J. Kim, M. Choi, Y.B. Kim, F. Liu, H. Moon, J.C. Koo, H.R. Choi: Exploration of unknown object by active touch of robot hand, Int. J. Control Autom. Syst. **12**(2), 406–414 (2014)

40.149 L. Chang, J.R. Smith, D. Fox: Interactive singulation of objects from a pile, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

40.150 M. Gupta, G.S. Sukhatme: Using manipulation primitives for brick sorting in clutter, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

40.151 D. Katz, M. Kazemi, J.A. Bagnell, A. Stentz: Autonomous pile clearing using interactive perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

40.152 M.R. Dogar, M.C. Koval, A. Tallavajhula, S.S. Srinivasa: Object search by manipulation, Auton. Robots **36**(1/2), 153–167 (2014)

40.153 R. Martín Martín, O. Brock: Online interactive perception of articulated objects with multi-level recursive estimation based on task-specific priors, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2014)

40.154 D. Katz, O. Brock: Manipulating articulated objects with interactive perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 272–277

40.155 D. Katz, A. Orthey, O. Brock: Interactive perception of articulated objects, Int. Symp. Exp. Robotics (2010)

40.156 D. Katz, A. Orthey, O. Brock: Interactive perception of articulated objects, Springer Tract. Adv. Robotics **79**, 301–315 (2014)

40.157 D. Katz, M. Kazemi, J.A. Bagnell, A. Stentz: Interactive segmentation, tracking, and kinematic modeling of unknown articulated objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 5003–5010

40.158 X. Huang, I. Walker, S. Birchfield: Occlusion-aware multi-view reconstruction of articulated objects for manipulation, Robotics Auton. Syst. **63**(4), 497–505 (2014)

40.159 S. Pillai, M. Walter, S. Teller: Learning articulated motions from visual demonstration, Proc. Robotics Sci. Syst. (2014)

40.160 J. Sturm, K. Konolige, C. Stachniss, W. Burgard: Vision-based detection for learning articulation models of cabinet doors and drawers in household environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 362–368

40.161 J. Sturm, A. Jain, C. Stachniss, C. Kemp, W. Burgard: Operating articulated objects based on experience, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010)

40.162 J. Sinapov, T. Bergquist, C. Schenck, U. Ohiri, S. Griffith, A. Stoytchev: Interactive object recognition using proprioceptive and auditory feedback, Int. J. Robotics Res. **30**(10), 1250–1262 (2011)

40.163 D. Katz, Y. Pyuro, O. Brock: Learning to manipulate articulated objects in unstructured environments using a grounded relational representation, Proc. Robotics Sci. Syst., Zurich (2008) pp. 254–261

40.164 S. Otte, J. Kulick, M. Toussaint, O. Brock: Entropy-based strategies for physical exploration of the environment's degrees of freedom, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2014)

40.165 S. Höfer, T. Lang, O. Brock: Extracting kinematic background knowledge from interactions using task-sensitive relational learning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)

40.166 J.J. Gibson: The theory of affordances. In: *Perceiving, Acting, and Knowing*, ed. by R. Shaw, J. Bransford (Lawrence Erlbaum, Hilldale 1977)

40.167 E. Şahin, M. Çakmak, M.R. Doğar, E. Uğur, G. Üçoluk: To afford or not to afford: A new formalization of affordances toward affordance-based robot control, Adapt. Behav. **15**(4), 447–472 (2007)

40.168 L. Montesano, M. Lopes, A. Bernardino, J. Santos-Victor: Learning object affordances: From sensory–motor coordination to imitation, IEEE Trans. Robotics **24**(1), 15–26 (2008)

40.169 L. Paletta, G. Fritz, F. Kintzler, J. Irran, G. Dorffner: Learning to perceive affordances in a framework of developmental embodied cognition, IEEE 6th Int. Conf. IEEE Dev. Learn. (ICDL) (2007) pp. 110–115

40.170 P. Sequeira, M. Vala, A. Paiva: What can i do with this?: Finding possible interactions between characters and objects, Proc. 6th ACM Int. Jt. Conf. Auton. Agents Multiagent Syst. (2007) p. 5

40.171 B. Moldovan, M. van Otterlo, P. Moreno, J. Santos-Victor, L. De Raedt: Statistical relational learning of object affordances for robotic manipulation, Latest Adv. Inductive Logic Program. (2012) p. 6

40.172 C. de Granville, J. Southerland, A.H. Fagg: Learning grasp affordances through human demonstration, Proc. Int. Conf. Dev. Learn. (ICDL06) (2006)

40.173 L. Montesano, M. Lopes: Learning grasping affordances from local visual descriptors, IEEE 8th Int. Conf. Dev. Learn. (ICDL) (2009) pp. 1–6

40.174 D. Kraft, R. Detry, N. Pugeault, E. Baseski, J. Piater, N. Krüger: Learning objects and grasp affordances through autonomous exploration, Lect. Notes Comput. Sci. **5815**, 235–244 (2009)

40.175 R. Detry, D. Kraft, O. Kroemer, L. Bodenhagen, J. Peters, N. Krüger, J. Piater: Learning grasp affordance densities, Paladyn **2**(1), 1–17 (2011)

40.176 T. Gollisch, M. Meister: Eye smarter than scientists believed: Neural computations in circuits of the retina, Neuron **65**(2), 150–164 (2010)

Multimedia Contents

41. Active Manipulation for Perception

Anna Petrovskaya, Kaijen Hsiao

This chapter covers perceptual methods in which manipulation is an integral part of perception. These methods face special challenges due to data sparsity and high costs of sensing actions. However, they can also succeed where other perceptual methods fail, for example, in poor-visibility conditions or for learning the physical properties of a scene.

The chapter focuses on specialized methods that have been developed for object localization, inference, planning, recognition, and modeling in active manipulation approaches. We conclude with a discussion of real-life applications and directions for future research.

41.1 **Perception via Manipulation** 1037

41.2 **Object Localization** 1038
41.2.1 Problem Evolution 1038
41.2.2 Bayesian Framework 1039
41.2.3 Inference 1041
41.2.4 Advanced Inference Methods 1043
41.2.5 Planning 1045

41.3 **Learning About an Object** 1049
41.3.1 Shape of Rigid Objects 1049
41.3.2 Articulated Objects 1051
41.3.3 Deformable Objects 1052
41.3.4 Contact Estimation 1053
41.3.5 Physical Properties 1054

41.4 **Recognition** 1054
41.4.1 Object Shape Matching 1054
41.4.2 Statistical Pattern Recognition 1055
41.4.3 Recognition by Material Properties 1056
41.4.4 Combining Visual and Tactile Sensing 1056

41.5 **Conclusions** 1057
41.5.1 Real-Life Applications 1057
41.5.2 Future Directions of Research 1057
41.5.3 Further Reading 1057

Video-References 1058

References 1058

41.1 Perception via Manipulation

In this chapter, we focus on perceptual methods that rely on manipulation. As we will see, sensing-*via*-manipulation is a complex and highly intertwined process, in which the results of perception are used for manipulation and manipulation is used to gather additional data. Despite this complexity, there are three main reasons to use manipulation either together or instead of the more commonly used vision sensors.

First, manipulation can be used to sense in poor-visibility conditions: for example, sensing in muddy water or working with transparent objects. In fact, due to its high accuracy and ability to sense any hard surface regardless of its optical properties, contact sensing is widely used for accurate localization of parts in manufacturing. Second, manipulation is useful for determining properties that require physical interaction, such as stiffness, mass, or the physical relationship of parts. Third, if the actual goal is to manipulate the object, we might as well use data gathered from manipulation attempts to improve both perception and subsequent attempts at manipulation.

Sensing-*via*-manipulation also faces significant challenges. Unlike vision sensors, which provide a whole-scene view in a single snapshot, contact sensors are inherently local, providing information only about a very small area of the sensed surface at any given time. In order to gather additional data, the manipu-

lator has to be moved into a different position, which is a time-consuming task. Hence, unlike vision-based perception, contact-based methods have to cope with very sparse data and very low data acquisition rates. Moreover, contact sensing disturbs the scene. While in some situations this can be desirable, it also means that each sensing action can increase uncertainty. In fact, one can easily end up in a situation where the information gained via sensing is less than the information lost due to scene disturbance.

Due to these challenges, specialized perceptual methods had to be developed for perception-via-manipulation. Inference methods have been developed to extract the most information from the scarce data and to cope with cases in which the data are insufficient to fully solve the problem (i. e., under-constrained scenarios). Planning methods have been developed to make the most efficient sensing decisions based on the little data available and taking into account time, energy, and uncertainty costs of sensing actions. Modeling methods have been developed to build the most accurate models from small amounts of data.

In this chapter, the material is organized by perceptual goal: localizing an object (Sect. 41.2), learning about an object (Sect. 41.3), and recognizing an object (Sect. 41.4). Since the chapter topic is broad, to keep it concise, we provide a tutorial-level discussion of object localization methods, but give a birds-eye view of the methods for the other two perceptual goals. However, as we point out along the way, many object localization methods can be reused for the other two perceptual goals.

41.2 Object Localization

Tactile object localization is the problem of estimating the object's pose – including position and orientation – based on a set of data obtained by touching the object. A prior geometric model of the object is assumed to be known and the sensing is performed using some type of contact sensor (wrist force/torque sensor, tactile array, fingertip force sensor, etc.). The problem is typically restricted to rigid objects, which can be stationary or moving.

Tactile object localization involves several components, including manipulator control methods, modeling, inference, and planning. During data collection, some form of sensorimotor control is used for each sensing motion of the manipulator. For information on manipulator control, the reader is encouraged to reference Chaps. 8 and 9. With few exceptions, the sensing motions tend to be poking motions rather than following the surface. This is due to the fact that robots are usually unable to sense with all parts of the manipulator, and, hence, the poking motions are chosen to minimize the possibility of accidental non-sensed contact.

Due to the properties of sensing-via-contact, special methods for modeling, inference, and planning had to be developed for tactile localization. Modeling choices include not only how to model the object itself, but also models of the sensing process and models of possible object motion. We discuss models of sensing and motion in Sect. 41.2.2, but leave an in-depth discussion of object modeling techniques until Sect. 41.3. We also cover specialized inference and planning methods in Sect. 41.2.3 and 41.2.5, respectively.

41.2.1 Problem Evolution

Attempts to solve the tactile object localization date back to early 80s. Over the years, the scope of the problem as well as methods used to solve it have evolved as we detail below.

Early Methods

Early methods for tactile object localization generally ignore the sensing process uncertainties and focus on finding a single hypothesis that best fits the measurements. For example, *Gaston* and *Lozano-Perez* used interpretation trees to efficiently find the best match for 3 DOF (degrees of freedom) object localization [41.1]. *Grimson* and *Lozano-Perez* extended the approach to 6 DOF [41.2]. *Faugeras* and *Hebert* used least squares to perform geometrical matching between primitive surfaces [41.3]. *Shekhar* et al. solved systems of weighted linear equations to localize an object held in a robotic hand [41.4]. Several methods use geometric constraints together with kinematic and dynamic equations to estimate or constrain the pose of known planar objects in plane by pushing them with a finger [41.5] or parallel-jaw gripper [41.6], or by tilting them in a tray [41.7].

Workpiece Localization

Single hypothesis methods are also widely used to solve the workpiece localization problem in manufacturing applications for dimensional inspection [41.8], machining [41.9], and robotic assembly [41.10]. In these applications, the measurements are taken by a coordinate measurement machine (CMM) [41.11] or by on-machine sensors [41.12]. Workpiece localization makes

a number of restrictive assumptions that make it inapplicable to autonomous robot operation in unstructured environments. One important restriction is that there is a known correspondence between each measured data point and a point or patch on the object surface (called home point or home surface, respectively) [41.13]. In semiautomated settings, the correspondence assumption is satisfied by having a human direct the robot to specific locations on the object. In fully automated settings, the object is placed on the measurement table with low uncertainty to make sure each data point lands near the corresponding home point.

Further restrictions include assumptions that the data are sufficient to fully constrain the object, the object does not move, and there are no unmodeled effects (e.g., vibration, deformation, or temperature variation). All of these parameters are carefully controlled for in structured manufacturing environments.

The workpiece localization problem is usually solved in least squares form using iterative optimization methods, including the Hong-Tan method [41.14], the Variational method [41.15], and the *Menq* method [41.16]. Since these methods are prone to getting trapped in local minima, low initial uncertainty is usually assumed to make sure the optimization algorithm is initialized near the solution. Some attempts have been made to solve the global localization problem by rerunning the optimization algorithm multiple times from prespecified and random initial points [41.17]. Recent work has focused on careful selection of the home points to improve localization results [41.18–20] and on improving localization efficiency for complex home surfaces [41.21, 22].

Bayesian Methods

In the last decade, there has been increased interest in Bayesian state estimation for the tactile object localization problem [41.23–27]. These methods estimate the probability distribution over all possible states (the belief), which captures the uncertainty resulting from noisy sensors, inaccurate object models, and other effects present during the sensing process. Thus, estimation of the belief enables planning algorithms that are resilient to the uncertainties of the real world. Unlike workpiece localization, these methods do not assume known correspondences. In contrast to single hypothesis or set-based methods, belief estimation methods allow us to better describe and track the relative likelihoods of hypotheses in the under-constrained scenario, in which the data are insufficient to fully localize the object. These methods can also work with moving objects and answer important questions such as: *have we localized the object completely?* and *where is the best place to sense next?*.

41.2.2 Bayesian Framework

All Bayesian methods share a similar framework. We start with a general definition of a Bayesian problem (not necessarily tactile object localization), and then, explain how this formulation can be applied to tactile localization.

General Bayesian Problem

For a general Bayesian problem, the goal is to infer the state X of a system based on a set of sensor measurements $\mathcal{D} := \{D_k\}$. Due to uncertainty, this information is best captured as a probability distribution

$$bel(X) := p(X|\mathcal{D}) \,, \tag{41.1}$$

called the posterior distribution or the Bayesian belief. Figure 41.1a shows a Bayesian network representing all the random variables involved and all the relationships between them.

In a dynamic Bayesian system, the state changes over time, which is assumed to be discretized into small time intervals (Fig. 41.1b). The system is assumed to evolve as a Markov process with unobserved states. The goal is to estimate the belief at time t

$$bel_t(X_t) := p(X_t|\mathcal{D}_1, \ldots, \mathcal{D}_t) \,. \tag{41.2}$$

The behavior of the system is described via two probabilistic laws: (i) the measurement model $p(\mathcal{D}|X)$ captures how the sensor measurements are obtained and (ii) the dynamics model $p(X_t|X_{t-1})$ captures how the system evolves between time steps.

For brevity, it is convenient to drop the arguments in $bel(X)$ and $bel_t(X_t)$, and simply write bel and bel_t, but these two beliefs should always be understood as functions of X and X_t, respectively.

Tactile Localization in Bayesian Form

Tactile object localization can be formulated as an instance of the general Bayesian problem. Here, the

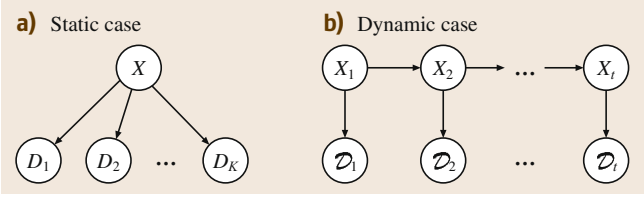


Fig.41.1a,b Bayesian network representation of the relationships between the random variables involved in a general Bayesian problem. The directional arrows are read as *causes*. **(a)** In the static case, a single unknown state X causes a collection of measurements $\{D_k\}$. **(b)** In the dynamic case, the state X_t changes over time and at each time step causes a set of measurements $\mathcal{D}_t$

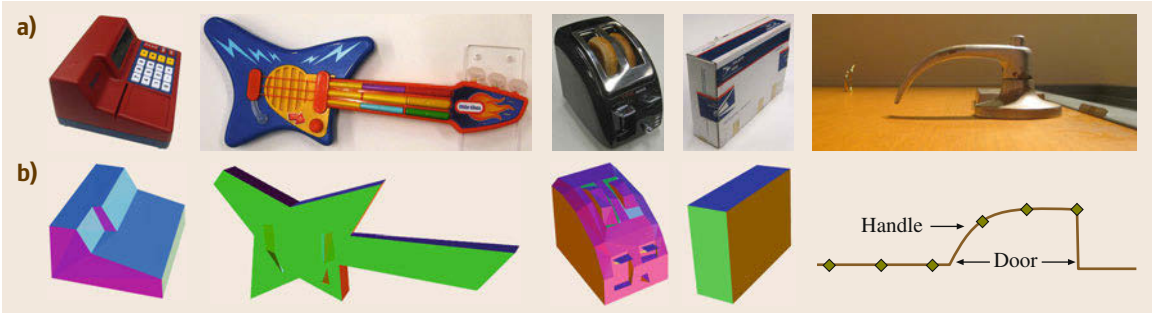


Fig.41.2a,b Examples of objects **(a)** and their polygonal mesh models **(b)**. Five objects are shown: cash register, toy guitar, toaster, box, and door handle. The first three models were constructed by collecting surface points with the robot's end effector. The last two were constructed from hand measurements made with a ruler. The door handle model is 2-D. Model complexity ranges from 6 faces (for the box) to over 100 faces (for the toaster) (after [41.28])

robot needs to determine the pose X of a known object $\mathcal{O}$ based on a set of tactile measurements $\mathcal{D}$. The state is the 6 DOF pose of the object – including position and orientation – in the manipulator coordinate frame. A number of state parameterizations are possible, including matrix representations, quaternions, Euler angles, and Rodrigues angles, to name a few. For our discussion here, assume that the state $X := (x, y, z, \alpha, \beta, \gamma)$, where (x, y, z) is the position and (α, β, γ) are orientation angles in Euler representation.

The measurements $\mathcal{D} := \{D_k\}$ are obtained by touching the object with the robot's end effector. The end effector can be a single probe, a gripper, or a hand. Thus, it may be able to sense multiple contacts D_k simultaneously, for example, in a single grasping attempt. We will assume that for each touch D_k, the robot is able to measure the contact point and also possibly sense the normal of the surface (perhaps as the direction of reaction force). Under these assumptions, each measurement $D_k := (D_k^{\text{pos}}, D_k^{\text{nor}})$ consists of the measured Cartesian position of the contact point D_k^{pos} and the measured surface normal D_k^{nor}. If measurements of surface normals are not available, then $D_k := D_k^{\text{pos}}$.

Measurement Model

In order to interpret tactile measurements, we need to define a model of the object and a model of the sensing process. For now, assume that the object is modeled as a polygonal mesh, which could be derived from a CAD model or a 3-D scan of the object, or built using contact sensors. Examples of objects and their polygonal mesh representations are shown in Fig. 41.2. Other types of object models can also be used. We provide an in-depth look at object modeling techniques in Sect. 41.3.

Typically, the individual measurements D_k in a data set $\mathcal{D}$ are considered independent of each other given the state X. Then, the measurement model factors over the measurements

$$p(\mathcal{D}|X) = \prod_k p(D_k|X) \,. \tag{41.3}$$

Sampled Models. Early Bayesian tactile localization work used sampled measurement models. For example, *Gadeyne* and *Bruyninckx* used numerical integration to compute the measurement model for a box and stored it in a look-up table for fast access [41.23]. *Chhatpar* and *Branicky* sampled the object surface by repeatedly touching it with the robot's end-effector to compute the measurement probabilities (Fig. 41.3) [41.24].

Proximity Model. One common model for the interpretation of tactile measurements is the proximity measurement model [41.29]. In this model, the measurements are considered to be independent of each other, with both position and normal components corrupted by Gaussian noise. For each measurement, the probability depends on the distance between the measurement and the object surface (hence the name *proximity*).

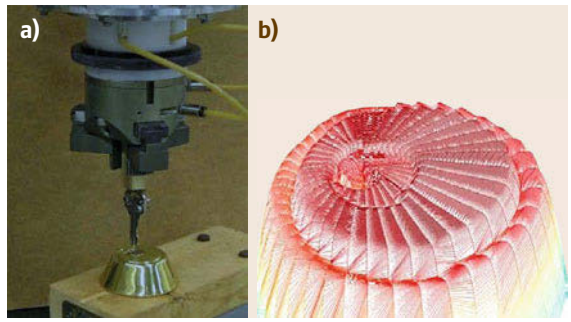


Fig.41.3a,b Sampled measurement model for a key lock. **(a)** The robot exploring the object with a key. **(b)** The resulting lock-key contact C-space (after [41.24])

Since the measurements contain both contact coordinates and surface normals, this distance is taken in the 6-D space of coordinates and normals (i. e., in the measurement space). Let $\hat{\mathcal{O}}$ be a representation of the object in this 6-D space. Let $\hat{o} := (\hat{o}^{\text{pos}}, \hat{o}^{\text{nor}})$ be a point on the object surface, and D be a measurement. Define $d_{\text{M}}(\hat{o}, D)$ to be the Mahalanobis distance between $\hat{o}$ and D

$$d_{\text{M}}(\hat{o}, D) := \sqrt{\frac{||\hat{o}^{\text{pos}} - D^{\text{pos}}||^2}{\sigma^2_{\text{pos}}} + \frac{||\hat{o}^{\text{nor}} - D^{\text{nor}}||^2}{\sigma^2_{\text{nor}}}} , \tag{41.4}$$

where σ^2_{pos} and σ^2_{nor} are Gaussian noise variances of position and normal measurement components, respectively. In the case of a sensor that only measures position (and not surface normal), only the first summand is used inside the square root. The distance between a measurement D and the entire object $\hat{\mathcal{O}}$ is obtained by minimizing the Mahalonobis distance over all object points $\hat{o}$

$$d_{\text{M}}(\hat{\mathcal{O}}, D) := \min_{\hat{o} \in \hat{\mathcal{O}}} d_{\text{M}}(\hat{o}, D) . \tag{41.5}$$

Let $\hat{\mathcal{O}}_X$ denote the object in state X. Then, the measurement model is computed as

$$p(\mathcal{D}|X) = \eta \exp\left[-\frac{1}{2}\sum_k d^2_{\text{M}}(\hat{\mathcal{O}}_X, D_k)\right] . \tag{41.6}$$

In the above equation and throughout the chapter, η denotes the normalization constant, whose value is such that the expression integrates to 1. VIDEO 723 shows the proximity model being used to localize a door handle.

Integrated Proximity Model. A variation of the proximity model is called the integrated proximity model. Instead of assuming that the closest point on the object caused the measurement, it considers the contribution from all surface points to the probability of the measurement [41.25]. This is a much more complex model that in general can not be computed efficiently. Moreover, for an unbiased application of this model, we need to compute a prior over all surface points, i. e., how likely each surface point is to cause a measurement. This prior is usually nonuniform and highly dependent on the object shape, the manipulator shape, and the probing motions. Nevertheless, in some cases this model can be beneficial as it is more expressive than the proximity model.

Negative Information. The models we have described so far do not take into account negative information, i. e., information about the absence rather than presence of measurements. This includes information that the robot was able to move through some parts of space without making contact with the object. Negative information is very useful for active exploration strategies and has been taken into account by [41.26] and [41.25]. Although adding negative information makes the belief more complex (e.g., discontinuous), in most cases, it can be superimposed on top of the belief computed using one of the above measurement models.

Dynamics Model

Free-standing objects can move during probing, and hence, a dynamics model is needed to describe this process. In most situations, little is known about possible object motions, and thus, a simple Gaussian model is assumed. Hence, $p(X_t|X_{t-1})$ is a Gaussian with mean at X_{t-1} and variances σ^2_{met} and σ^2_{ang} along metric and angular axes respectively. If additional properties of object motion are known, then a more informative dynamics model can be used. For example, if we know the robot shape and the magnitude and direction of the contact force, we can use Newtonian dynamics to describe object motion. This motion may be further restricted, if it is known that the object is sliding on a specific surface.

41.2.3 Inference

Once the models are defined and the sensor data are obtained, the next step is to estimate the resulting probabilistic belief. This process is called inference and can be viewed as numerical estimation of a real-valued function over a multidimensional space. We can distinguish two cases: a static case where the object is rigidly fixed and a dynamic case where the object can move. The dynamic case is solved recursively at each time step using a Bayesian filter. The static case can be solved either recursively or in a single step by combining all the data into a single batch. The other important differentiator is the amount of initial uncertainty. In low uncertainty problems, the pose of the object is approximately known and only needs to be updated based on the latest data. This type of problem usually arises during tracking of a moving object, in which case it is known as pose tracking. In global uncertainty problems, the object can be anywhere within a large volume of space and can have arbitrary orientation. This case is often referred to as global localization. Global localization problems arise for static objects, when the pose of the object is unknown, or as the initial step in pose tracking of a dynamic object. It is also the fallback

whenever the pose tracking method fails and uncertainty about the object pose becomes large.

The belief tends to be a highly complex function with many local extrema. For this reason, parametric methods (such as Kalman filters) do not do well for this problem. Instead, the problem is usually solved using nonparametric methods. The complexity of the belief is caused directly by properties of the world, the sensors, and models thereof. For more insight into causes of belief complexity and appropriate estimation methods, see the discussion of belief roughness in [41.28, Chap. 1].

We start by describing basic nonparametric methods, which are efficient enough to solve problems with up to three degrees of freedom (DOFs). Then, in Sect. 41.2.4, we describe several advanced methods that have been developed to solve the full 6 DOF problem in real-time.

Basic Nonparametric Methods

Nonparametric methods typically approximate the belief by points. There are two types of nonparametric methods: deterministic and Monte Carlo (i. e., nondeterministic). The most common deterministic methods are grids and histogram filters (HF). For these methods, the points are arranged in a grid pattern with each point representing a grid cell. The most common Monte Carlo methods are importance sampling (IS) and particle filters (PF). For Monte Carlo methods, the points are sampled randomly from the state space and called samples or particles.

Static Case

Via Bayes rule, the belief $bel(X)$ can be shown to be proportional to $p(\mathcal{D}|X)p(X)$. The first factor is the measurement model. The second factor, $\overline{bel}(X) := p(X)$, is called the Bayesian prior, which represents our belief about X before obtaining measurements $\mathcal{D}$. Hence, with this notation, we can write

$$bel = \eta\, p(\mathcal{D}|X)\, \overline{bel}\,. \tag{41.7}$$

In the most common case, where it is only known that the object is located within some bounded region of space, a uniform prior is most appropriate. In this case, the belief bel is proportional to the measurement model and the equation (41.7) simplifies to

$$bel = \eta\, p(\mathcal{D}|X)\,. \tag{41.8}$$

However, if an approximate object pose is known ahead of time (e.g., an estimate produced from a vision sensor), then it is common to use a Gaussian prior of some covariance around the approximate pose.

Grid methods compute the unnormalized belief $p(\mathcal{D}|X)\,\overline{bel}$ at the center of each grid cell and then normalize over all the grid cells to obtain an estimate of the belief bel.

Monte Carlo methods use importance sampling. For uniform priors, the particles are sampled uniformly from the state space. For more complex priors, the particles have to be sampled from the prior distribution $\overline{bel}$. The importance weight for each particle is set to the measurement model $p(\mathcal{D}|X)$ and the entire set of importance weights is normalized so that the weights add up to 1.

Dynamic Case

Given the measurement model $p(\mathcal{D}|X)$, the dynamics model $p(X_t|X_{t-1})$, and measurements $\{\mathcal{D}_1, \ldots, \mathcal{D}_t\}$ up to time t, the belief can be computed recursively using a Bayesian filter algorithm, which relies on the Bayesian recursion equation

$$bel_t = \eta\, p(\mathcal{D}_t|X_t)\, \overline{bel}_t\,. \tag{41.9}$$

To compute (41.9), the Bayesian filter alternates two steps: (a) the dynamics update, which computes the prior

$$\overline{bel}_t = \eta \int p\,(X_t|X_{t-1})\, bel_{t-1}\, \mathrm{d}X_{t-1}\,, \tag{41.10}$$

and (b) the measurement update, which computes the measurement model $p(\mathcal{D}_t|X_t)$.

A grid-based implementation of the Bayesian filter is called a histogram filter. Histogram filters compute (41.9) for each grid cell separately. At the start of time step t, the grid cell stores its estimate of the belief bel_{t-1} from the previous time step. During the dynamics update, the value of the prior $\overline{bel}_t$ at this grid cell is computed using (41.10), where the integral is replaced by a summation over all grid cells. During the measurement update, the measurement model is computed at the center of the grid cell and multiplied by the value of the grid cell (i. e., the prior). Then, values of all grid cells are normalized so they add up to 1. Since during dynamics update, a summation over all grid cells has to be carried out for each grid cell, the computational complexity of histogram filters is quadratic in the number of grid cells.

A Monte Carlo implementation of the Bayesian filter is called a particle filter. The particle set representing the prior $\overline{bel}_t$ is produced during the dynamics update as follows. The particles are resampled (see [41.30] for a tutorial on resampling) from the belief at the previous time step, bel_{t-1}, and then moved according to the dynamics model with the addition of some motion noise. This produces a set of particles representing the prior $\overline{bel}_t$. The measurement update is performed using importance sampling as in the static case above. Note that

unlike for the histogram filters, there is no per-particle summation during the dynamics update, and thus, the computational complexity of particle filters is linear in the number of particles.

41.2.4 Advanced Inference Methods

Although suitable for some problems, basic nonparametric methods have a number of shortcomings. There are two main issues.

1. While a rigid body positioned in space has 6 DOFs, the basic methods can only be used for localization with up to 3 DOFs. Beyond 3 DOFs, these methods become computationally prohibitive due to exponential growth in the number of particles (or grid cells) required. The reason for the exponential blow-up is as follows. In order to localize the object, we need to find its most likely poses, i. e., the peaks of bel_t. These peaks (also called modes) are very narrow, and, hence, the space needs to be densely populated with particles in order to locate them. The number of particles required to achieve the same density per unit volume goes up exponentially with the number of DOFs. Hence, the computational complexity of nonparametric methods also grows exponentially with the number of DOFs. This is known as the curse of dimensionality.
2. Another important issue is that the performance of basic nonparametric methods actually *degrades with increase in sensor accuracy*. This problem is especially pronounced in particle filters. Although it may seem very unintuitive at first, the reason is simple. The more accurate the sensor the narrower the peaks of the belief. Thus, more and more particles are required to find these peaks. We can call this problem curse of accurate sensor.

Several more advanced methods have been developed to combat the shortcomings of basic methods. Below, we detail one such method, called scaling series, and briefly discuss a few others.

Scaling Series

Scaling series is an inference algorithm designed to scale better to higher dimensional problems than the basic methods [41.28, Chap. 2]. In particular, it is capable of solving the full 6 DOF localization problem. Moreover, unlike for basic methods, the accuracy of scaling series *improves with increase in sensor accuracy*. For this reason, this algorithm may be preferable to basic methods even in low-dimensional problems.

Scaling series represents the belief by broad particles. Each particle is thought of as a representative for an entire δ-neighborhood, that is, the volume of space of radius δ surrounding the sample point. Instead of fixing the total number of particles (as in the basic methods), scaling series adjusts this number as needed to obtain good coverage of main peaks.

Peak width can be controlled using annealing, which means that for a given temperature τ, the measurement model is raised to the power of $1/\tau$. Hence, for $\tau = 1$, the original measurement model is obtained, whereas for $\tau > 1$ the measurement model is *heated-up*. Annealing broadens the peaks of bel_t, making them easier to find. However, it also increases ambiguity and decreases accuracy (Fig. 41.4). To make sure that accuracy is not compromised, scaling series combines annealing with iterative refinements as we detail below.

The Static Case Algorithm. Let us first consider the case, where the prior $\overline{bel}$ is uniform. Scaling series starts by uniformly populating the space with very broad particles (i. e., with large δ). It evaluates the annealed measurement model for each particle and prunes out low probability regions of space. Then, it refines the estimate with slightly narrower particles, and repeats again and again until a sufficiently accurate estimate is obtained. Peak width is controlled during iterations using annealing, to make sure that each particle is able to represent its δ-neighborhood well. As the value of δ decreases during iterations, the density of particles per volume of δ-neighborhood is kept constant to maintain good particle density near peaks. We can think of δ as peak width, which decreases (due to annealing) with iterations. At each iteration, the δ-neighborhood of each peak will have the same fixed number of particles.

The formal algorithm listing is given in Algorithm 41.1, where S_δ denotes a δ-neighborhood and $\dim X$ is the dimensionality of the state space. The algorithm relies on three subroutines. `Even_Density_Cover` samples a fixed number

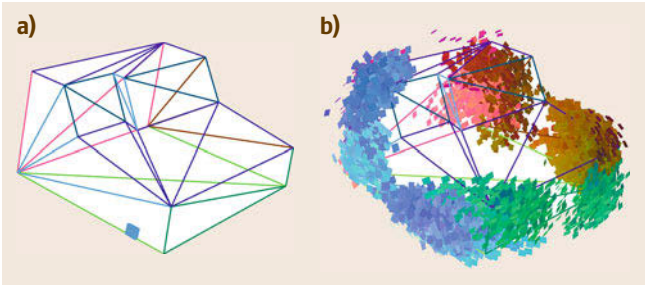


Fig.41.4a,b True (**a**) and annealed (**b**) belief for localization of the cash register [41.28]. The cash register model is shown as a wire frame. The small *colored squares* represent high likelihood particles. Note that annealing makes the problem much more ambiguous

of particles from each δ-neighborhood of a volume of space. It can be easily implemented using rejection sampling. `Importance_Weights` computes normalized importance weights using the annealed measurement model $p(\mathcal{D}|X)^{1/\tau}$. The `Prune` subroutine prunes out low probability regions. This step can be done either by weighted resampling or by thresholding on the weights.

The algorithm returns an approximation of the belief represented by a weighted particle set $\mathcal{X}$, where the weights $\mathcal{W}$ are set according to the measurement model, which in the case of uniform prior is proportional to the belief (41.8). Extension to the case of nonuniform prior can be done by simply multiplying the resulting weights by the prior. Sampling from the prior at the start of scaling series may also be useful. See VIDEO 721 for an example, where scaling series is successfully applied to 6-DOF localization of a cash register.

Algorithm 41.1 Scaling series algorithm for belief estimation

Input:
V_0 – initial uncertainty region,
$\mathcal{D}$ – data set,
M – number of particles per δ-neighborhood,
δ_* – terminal value of δ.

$\delta_0 \leftarrow Radius(V_0)$
$zoom \leftarrow 2^{-1/\dim X}$
$N \leftarrow \lfloor \log_2(Volume(S_{\delta_0})/Volume(S_{\delta_*})) \rfloor$
for $n = 1$ to N **do**
  $\delta_n \leftarrow zoom \cdot \delta_{n-1}$
  $\tau_n \leftarrow (\delta_n/\delta_*)^2$
  $\overline{\mathcal{X}}_n \leftarrow$ `Even_Density_Cover` (V_{n-1}, M)
  $\mathcal{W}_n \leftarrow$ `Importance_Weights` $(\overline{\mathcal{X}}_n, \tau_n, \mathcal{D})$
  $\mathcal{X}_n \leftarrow$ `Prune` $(\overline{\mathcal{X}}_n, \mathcal{W}_n)$
  $V_n \leftarrow$ `Union_Delta_Neighborhoods` $(\mathcal{X}_n, \delta_n)$
end for
$\mathcal{X} \leftarrow$ `Even_Density_Cover` (V_N, M)
$\mathcal{W} \leftarrow$ `Importance_Weights`$(\mathcal{X}, 1, \mathcal{D})$

Output:
$(\mathcal{X}, \mathcal{W})$ – a weighted particle set approximating the belief.

Dynamic Case. Scaling series can be extended to the dynamic case using the same technique as in the histogram filter. During the measurement update, the measurement model is estimated using scaling series with a uniform prior. This produces a set of weighted particles $\mathcal{X}_t$. During the dynamics update, the importance weights are adjusted to capture the motion model via the Bayesian recursion equation (41.9). To do this, for each particle X_t in $\mathcal{X}_t$, the importance weight is multiplied by the prior $\overline{bel}_t(X_t)$. Like in the histogram filter, the prior at a point X_t is computed by replacing the integral in (41.10) with a summation, except now the summation is done over all particles in $\mathcal{X}_{t-1}$. For other versions of dynamic scaling series, refer to [41.28, Chapter 2].

Other Advanced Inference Methods

Other advanced methods have been used to solve the tactile object localization problem, including the annealed particle filter (APF) [41.25], the GRAB algorithm [41.31], and the manifold particle filter (MPF) [41.32].

The APF algorithm is similar to scaling series as it also uses particles and iterative annealing. This algorithm was originally developed for articulated object tracking based on vision data [41.33, 34]. Unlike scaling series, APF keeps the number of particles constant at each iteration and the annealing schedule is derived from the particle set itself based on survival rate. Due to these properties, APF handles poorly in multimodal scenarios [41.35], which are prevalent in tactile object localization.

The GRAB algorithm is based on grids and measurement model bounds. It performs iterative grid refinements and prunes low-probability grid cells based on measurement model bounds. Unlike the majority of inference methods, GRAB is able to provide guaranteed results as long as the measurement model bounds are sound. It is well-suited for problems with many discontinuities, for example, whenever negative information is used in the measurement model. However, for smooth measurement models (such as the proximity model), GRAB has been shown to be slower than scaling series [41.28].

Similarly to the dynamic version of scaling series, the MPF algorithm samples particles from the measurement model and weighs them by the dynamics model. However, to sample from the measurement model, MPF draws a set of particles from the contact manifold, defined as the set of object states that contact a sensor without penetrating. Then, MPF weights these new particles by propagating prior particles forward using the dynamics model and applying kernel density estimation. Since the contact manifold is a lower dimensional manifold than the full state space of object poses, the MPF requires fewer particles than a traditional PF. It is not yet known if the improvement is sufficient to handle the full 6 DOF problem. However, it has been shown that MPF is capable of handling accurate sensors [41.32]. While MPF provides a method for sampling from the contact manifold for a single contact, it is unclear how to ex-

tend this approach to the case of multiple simultaneous contacts.

41.2.5 Planning

In order to localize an object, we need a strategy for gathering the data. Since each touch action can take significant time to execute, it is particularly important to accomplish the task with as few touches as possible. In this section, we will cover how to generate candidate actions and how to select which action to execute, with the goal of either localizing the object or grasping it with a high probability of task success.

Candidate Action Generation

The first step in selecting actions for tactile localization is to generate a set of candidate motions. While the object's pose is uncertain, we would like to avoid knocking over the object or damaging the robot. For this reason, actions are generally guarded motions, where the robot stops moving upon detecting contact. For multifingered robots, it may be advantageous to close the fingers upon detecting contact, in order to generate more than one contact point per action.

Candidate motions can be automatically generated in a similar manner to randomized or heuristic grasp planning methods (such as [41.36] or [41.37]). This approach is particularly useful if the goal is to generate multiple fingertip contacts, because any motion that is likely to grasp the object is also likely to result in informative contacts.

One option is to generate a small pool of good candidate motions based on the object shape, and to execute them relative to the current most-likely state of the object, as in [41.38]. Another option is to generate a larger pool of motions that are fixed with respect to the world, as in [41.39]. Finally, we can generate paths that specifically target disambiguating features. For example, *Schneiter* and *Sheridan* [41.40] show how to select paths that are guaranteed to disambiguate among remaining object shape and pose hypotheses, or if none such are possible, how to select paths that at least guarantee that something will be learned.

Bayesian Formulation of Planning

Once a pool of candidate actions has been created, the problem of selecting an optimal action to take next can be formulated as a partially observable Markov decision process (POMDP). Details on POMDPs can be found in Chap. 14. Briefly, a POMDP consists of a set of states $\mathcal{X} := \{X_i\}$, a set of actions $\mathcal{A} := \{A_j\}$, a set of observations $\mathcal{D} := \{D_k\}$. Both measurement model and dynamics model from Sect. 41.2.2 are now conditioned on the action A taken. This change is clearly visible

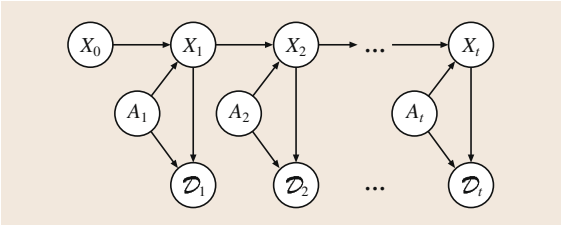


Fig. 41.5 Bayesian network representation of the planning problem. Note how it differs from Fig. 41.1b. Measurements $\mathcal{D}_t$ now depend on both the state X_t and the action chosen A_t. The state X_t evolves based on both the prior state X_{t-1} and the chosen action A_t

in the dynamic Bayesian network representation of the problem: compare Fig. 41.5 to Fig. 41.1b. Hence, the dynamics model is now $p(X_t|X_{t-1}, A_t)$, which is often called the transition model in the context of POMDPs. It represents the probability distribution of state X_t, given that we were in state X_{t-1} and executed action A_t. Similarly, the measurement model is now $p(D|X, A)$. For problems where the state X is continuous, the state space is typically discretized using either a grid or a set of particles.

For tactile object localization, the states are the actual object poses, the actions are our candidate robot motions (which may or may not touch the object), and the observations can include tactile and/or proprioceptive information gained while executing actions. We do not actually know the true underlying object pose, but at any given time t, we can estimate the uncertain belief bel_t about the object pose using methods described in Sect. 41.2.3. In the context of POMDPs, the belief is often referred to as the belief state.

A POMDP policy specifies an action A, for every possible belief. Intuitively, there is an optimal policy that would get the robot to the goal state most efficiently for any possible starting state. This optimal policy can be found using value or policy iteration algorithms. However, for the problem of tactile localization, these algorithms are typically computationally prohibitive for any practical discretization of the state space. Hence, this problem is usually solved using methods that approximate the optimal policy for a subset of states. For instance, online replanning using one-step or multistep lookahead is nonoptimal but usually provides good results.

For a subset of tactile localization problems, one-step lookahead has been shown to be nearly optimal [41.41]. Problems in this class satisfy two assumptions: (1) the object is stationary and does not move when touched (e.g., a heavy appliance), and (2) the actions being considered are a fixed set of motions regardless of the current belief. With these as-

sumptions, tactile localization can be formulated as an adaptive submodular information-gathering problem. Adaptive submodularity is a property of information-gathering problems where actions have diminishing returns, and taking additional actions never results in loss of information.

One-Step Lookahead

One way to select the next action is to consider the expected effect of each action on the belief about the object pose. Figure 41.6 shows examples of the effect of different actions on different beliefs.

Based on the possible object poses according to the current belief bel_t, we need to consider the possible outcomes of each action A. However, the same action can result in different outcomes depending on the actual object pose, which is not known to us exactly.

Simulation Experiments. Since we can not determine the effect of action A exactly, we approximate it by performing a series of simulation experiments. Let $\mathcal{X}_t$ be the current set of state hypotheses. For histogram filters, this can be the entire set $\mathcal{X}$. For particle filters, it is the current set of particles. For each particular object pose $X \in \mathcal{X}_t$, we can use a geometric simulation to determine where the hand will stop along a guarded motion A, and what observation D we are likely to see. Then, we can perform a Bayesian update of bel_t to obtain the simulated belief $\widetilde{bel}_{A,D}$ that would result if we chosen action A and observed D. Let $\mathcal{D}_A$ denote the set of all simulated observations we obtained for an action A using these experiments.

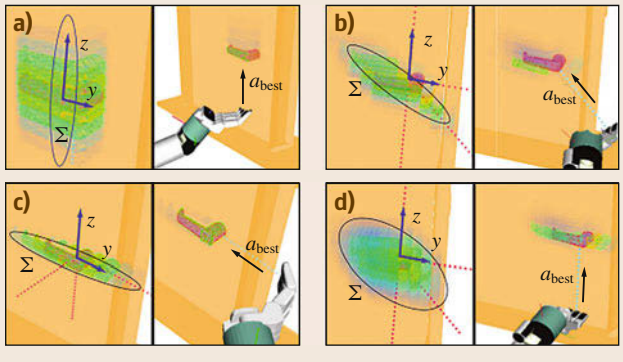


Fig.41.6a–d Selecting an action using a KLD-based utility function on examples of localizing a door handle. For each example, the starting belief bel_t is shown *on the left*. The action chosen $A := a_{\text{best}}$ and the resulting belief bel_{t+1} after executing the action are shown *on the right*. *Color* shows the probability of the state, ranging from high (*red*) to low (*blue*). **(a)** Sharp prior along the z-axis. **(b)** Sharp prior diagonal z–y-axis. **(c)** Sharp prior along the y-axis. **(d)** Wide prior along the diagonal z–y-axis (after [41.39])

To completely account for all possible outcomes, we need to simulate all possible noisy executions of action A and all possible resulting noisy observations D. However, restricting our consideration to the set of noise-free simulations works well in practice.

Each action A may also have an associated cost $C(A, D)$, which represents how expensive or risky it is to execute A. Note, that the action cost may depend on the observation D. For example, if the cost is the amount of time it takes to execute A, then depending on where the contact is sensed, the same guarded motion will stop at a different time along the trajectory.

Utility Function. In order to decide which action to take, we need a utility function $U(A)$ that allows us to compare different possible actions. The utility function should take into account both the usefulness of the action as well as its cost.

To localize the object, we want to reduce the uncertainty about its pose. Hence, we can measure action usefulness in terms of uncertainty of the resulting belief. Intuitively, the lower the uncertainty, the higher the usefulness of an action. The amount of uncertainty in a belief can be measured using entropy, which for a probability distribution $q(X)$ is defined as

$$H(q) := -\int q(X) \log q(X)\, \mathrm{d}X \,. \tag{41.11}$$

For each simulated belief $\widetilde{bel}_{A,D}$, we define its utility to be a linear combination of its usefulness (i. e., certainty) and cost

$$U(\widetilde{bel}_{A,D}) := -H(\widetilde{bel}_{A,D}) - \beta C(A, D) \,, \tag{41.12}$$

where β is a weight that trades off cost and usefulness [41.39]. Since we do not know what observation we will get after executing action A, we define its utility to be the expected utility of the resulting belief based on all possible observations

$$U(A) := \mathbb{E}_D\Big[U(\widetilde{bel}_{A,D})\Big] \,. \tag{41.13}$$

Based on the simulation experiments, this expectation can be estimated as a weighted sum over all the experiments

$$U(A) \approx \sum_{D \in \mathcal{D}_A} w_D U(\widetilde{bel}_{A,D}) \,, \tag{41.14}$$

where the weight w_D is the probability of the state X that was used to generate the observation D. More specifically, for a grid representation of state space, each $D \in \mathcal{D}_A$ was generated using some $X \in \mathcal{X}_t$, and thus, w_D

is the belief estimated for the grid point X. If $\mathcal{X}_t$ is represented by a set of weighted particles, then w_D is the weight of particle $X \in \mathcal{X}_t$ used to generate the observation D.

With this utility function in mind, the best action to execute is the one with the highest utility

$$A^{\text{best}} := \underset{A}{\text{argmax}}\Big[U(A)\Big] . \tag{41.15}$$

Since the utility function depends on negative entropy, methods that maximize this utility function are often called entropy minimization methods.

Alternative Utility Measures. Other measures of usefulness are also possible, such as reduction in average variance of the belief or Kullback–Leibler divergence (KLD) of the simulated belief from the current belief. However, KLD is only applicable for problems where the object is stationary as this metric does not support a changing state. An example of action selection using KLD metric is shown in Fig. 41.6.

If the goal is to successfully grasp the object, not just to localize it, then it makes more sense to select actions that maximize probability of grasp success rather than simply minimizing entropy. Success criteria for grasping can, for instance, be expressed geometrically as ranges in particular dimensions of object pose uncertainty that are required for a task to succeed. For a given belief, we can add up the probabilities corresponding to the object poses (states) for which executing a desired, task-directed grasp would succeed open-loop, and maximize that value rather than minimizing entropy. Action selection can still be done using one-step-lookahead. Using this metric instead of the one in (41.12) simply allows us to concentrate on reducing uncertainty along important dimensions, while paying less attention to unimportant dimensions [41.42]. For example, the axial rotation of a nearly cylindrical object is unimportant if it only needs to be grasped successfully.

Multi-step Planning in Belief Space

While in some situations one-step-lookahead methods give good results, in other situations it may be advantageous to look more than one step ahead. This is especially true if actions take a long time to execute, and/or if the plan can be stored and reused for multiple manipulation attempts.

Multi-step lookahead can be performed by constructing a finite-depth search tree (Fig. 41.7 for illustration). Starting with the current belief bel_t, we consider possible first-step actions $A_1, \ldots, A_3$ and simulate observations $D_{11}, \ldots, D_{32}$. For each simulated observation, we perform a belief update to obtain the simulated beliefs $B_{11}, \ldots, B_{32}$. For each of these simulated beliefs, we consider possible second-step actions (of which only $A_4, \ldots, A_6$ are shown). For each of the second-step actions, we again simulate observations and compute simulated beliefs. This process is repeated for third-step actions, fourth-step actions, and so on, until the desired number of steps is reached. The example in Fig. 41.7 shows the search tree for two-step lookahead.

After the tree is constructed from the top down to the desired depth, scoring is performed from the bottom up. Leaf beliefs are evaluated using the chosen utility metric. In the example in Fig. 41.7, the leafs shown are $B_{41}, \ldots, B_{62}$. For each leaf, we show the entropy H and cost βC. The utility of each corresponding action $A_4, \ldots, A_6$ is then computed by taking expectation over all observations that were simulated for that action. Note that we have to take expectations at this level because we have no control over which of the simulated observations we will actually observe. However, at the next level, we do have control over which action to execute, and we select the action with the maximum utility [41.42]. We compute the utility of beliefs $B_{11}, \ldots, B_{32}$ by subtracting their cost from the utility of the best action among their children. For example,

$$U(B_{22}) = U(A_6) - \beta C(A_2, D_{22}) . \tag{41.16}$$

In this manner, expectation and maximization can be carried out for multiple levels of the tree until the top is reached. Maximization at the very top level selects the most optimum action to execute next. Bold lines in Fig. 41.7 show the possible action sequences this algorithm would consider optimal depending on the possible observations made.

When searching to a depth greater than 1, branching on every possible observation is likely to be prohibitive. Instead, we can cluster observations based on how similar they are into a small number of canonical observations and only branch on these canonical observations.

Performing a multistep search allows us to include any desired final grasps as actions. These actions can result in early termination if the success criterion is met, but can also function as possible information-gathering actions if the criterion is not met. It also allows us to reason about actions that may be required to make desired grasps possible, such as reorienting the object to bring it within reach. As shown in [41.42], a two-step lookahead search is usually sufficient. Searching to a depth of 3 generally yields little additional benefit. One full sequence of information-gathering, grasping, and object reorientation using a multistep search with a depth of 2

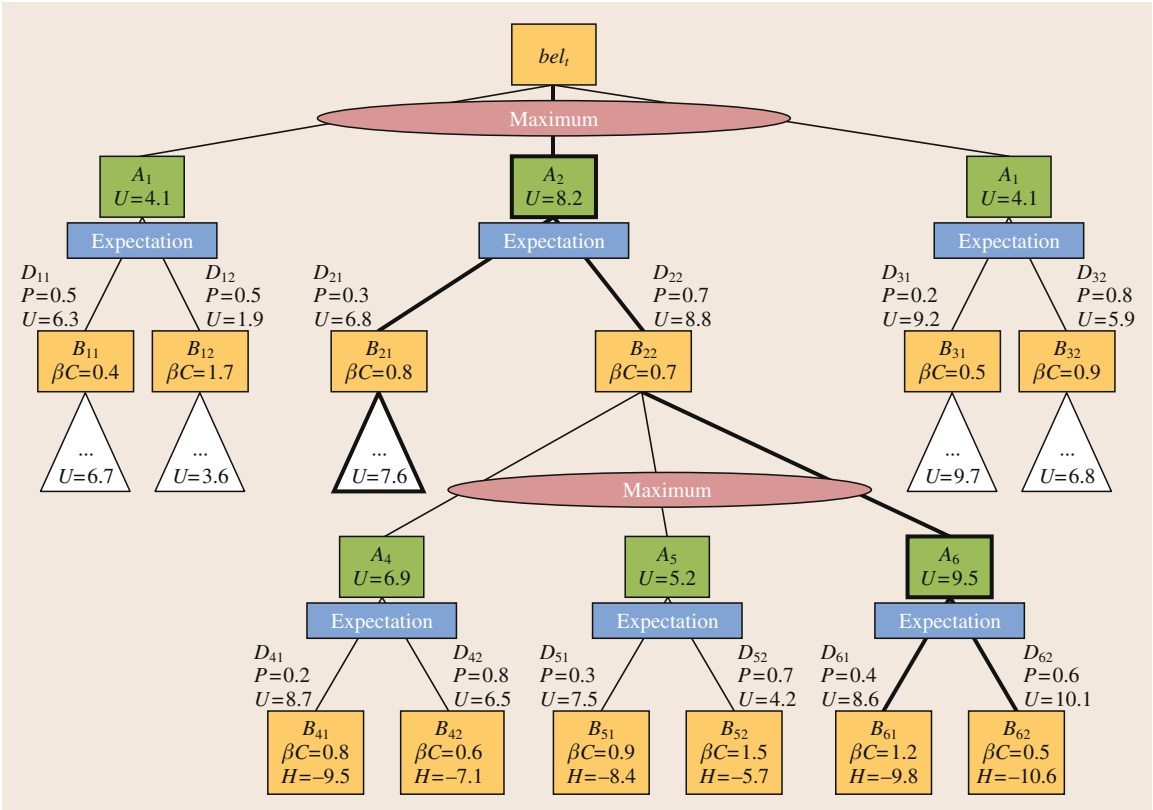


Fig. 41.7 Part of a depth-2 search tree through belief space, for three actions at each level and two canonical observations for each action; bel_t is the current belief. $A_1, \ldots, A_6$ are actions. $D_{11}, \ldots, D_{62}$ are simulated observations. And $B_{11}, \ldots, B_{62}$ are simulated beliefs. We can compute the entropy H and action cost $\beta\, C$ at the leafs, and thus compute the utility metric U for each leaf. Then, working upward from the bottom, we can compute the expected utility U of actions $A_4, \ldots, A_6$ by taking a weighted sum of their children. We then select the action with maximum utility (A_6 in this example) and use its utility $U(A_6)$ to compute the utility at node B_{22} using (41.16). The same operation is repeated for the upper levels, and at the top level we finally select the action with the highest utility as the best action to be performed (A_2 in this example) (after [41.42])

for grasping a power drill is shown in Fig. 41.8, and in VIDEO 77.

Other Planning Methods

There are many other possible methods for selecting actions while performing Bayesian state estimation for tactile object localization and grasping. As long as we continue to track the belief with one of the inference methods described above, even random actions can make progress toward successfully localizing the object. In this section, we will describe a few significantly different approaches to action selection for Bayesian object localization.

The planning methods we have described so far used either guarded motions or move-until-contact, to avoid disturbing or knocking over the object. However, one can instead use the dynamics of object pushing to both localize and constrain the object's pose in the process of push-grasping, as in *Dogar* and *Srinivasa* [41.43].

Also, instead of using entire trajectories as actions, we could consider smaller motions as actions out of which larger trajectories could be built. In this case, one- or two-step lookahead will be insufficient to get good results. Instead, we would need to construct plans consisting of many steps. In other words, we would need a much longer planning horizon. The challenge is that searching through belief space as described above is exponential in the planning horizon. In *Platt* et al. [41.44], the authors get around this problem by assuming the current most likely state is true and searching for longer horizon plans that will both reach the goal and also differentiate the current most likely hypothesis from other sampled, competing states. If we

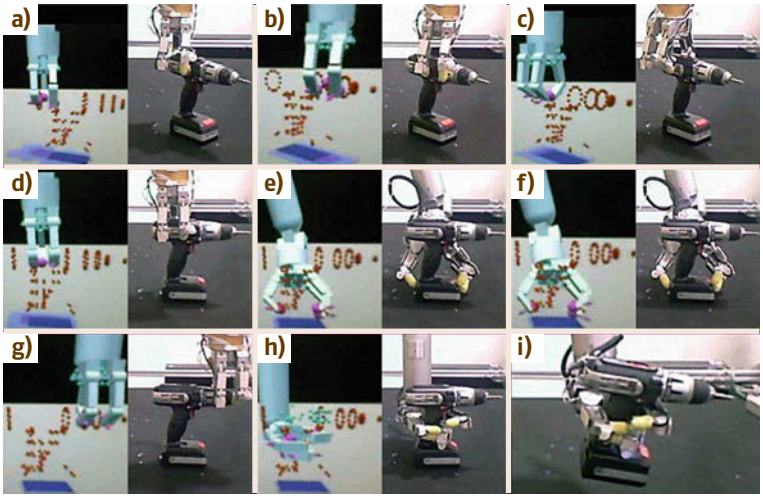


Fig.41.8a–i An example sequence of grasping a powerdrill based on a depth-2 search through belief space. The *right-hand side* of each pair of images shows the grasp just executed, while the *left-hand side* shows the resulting belief. The *dark blue flat rectangle* and the *red point-outline* of the powerdrill show the most likely state, while the *light blue rectangles* show states that are 1 standard deviation away from the mean in each dimension (x, y, θ). Panels **(a–c)** and **(e–g)** show information-gathering grasps being used to localize the drill, panel **(d)** shows the robot reorienting the drill to bring the desired grasp within reach, and panels **(h)** and **(i)** show the robot using the desired final grasp on the drill, lifting, and pulling the trigger successfully (after [41.42])

monitor the belief during execution and replan whenever it diverges by more than a set threshold, the goal will eventually be reached because each hypothesis will be either confirmed (and thus the goal reached) or disproved.

Finally, while the above methods can be used for planning for what we termed *global localization* (i. e., object localization under high uncertainty), if the object needs to be localized in an even broader context, as when trying to locate an object somewhere in a kitchen, then we may have to plan to gather information with higher level actions (e.g., opening cabinets). Such actions could be reasoned about using symbolic task planning, as in *Kaelbling* and *Lozano-Perez* [41.45].

41.3 Learning About an Object

While the previous section focused on localization of known objects, in this section, we discuss methods for learning about a previously unknown object or environment. The goal is to build a representation of the object that can later be useful for localization, grasp planning, or manipulation of the object. A number of properties may need to be estimated, including shape, inertia, friction, contact state, and others.

41.3.1 Shape of Rigid Objects

In this set of problems, the goal is to construct a 2-D or a 3-D model of the object's geometric shape by touching the object. Due to low data acquisition rate, touch-based shape reconstruction faces special challenges as compared to methods based on dense 3-D cameras or scanners. These challenges dictate the choice of object representation and exploration strategy.

While most of the work on shape reconstruction focuses on poking motions with the end-effector, some methods rely on techniques such as rolling the object in between planar palms equipped with tactile sensors. In these methods, shape of the object can be reconstructed from the contact curves traced out by the object on the palms. These methods were first developed for planar 2-D objects [41.46], and later extended to 3-D [41.47].

Representation

The shape reconstruction process is heavily dependent on the chosen representation. Since the data are sparse, a simpler shape representation can significantly speed up the shape acquisition process.

Shape Primitives. The simplest representation is a primitive shape: plane, sphere, cylinder, torus, tetrahe-

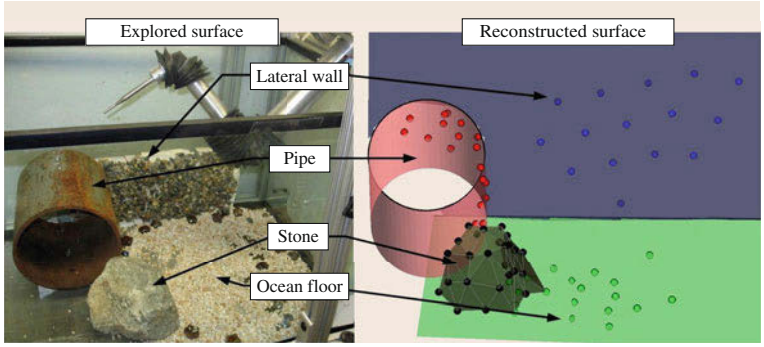


Fig. 41.9 A combination of primitive shapes and polygonal mesh is used for modeling objects during an underwater mapping experiment (after [41.48])

dra, or cube. In this case, only a few parameters need to be estimated based on the gathered data. For example, *Slaets* et al. estimate the size of a cube by performing compliant motions with the cube pressed against a table [41.49].

Patchwork of Primitives. If one primitive does not describe the whole object well, a *patchwork* of several primitives may be used. Parts of the object are represented with subsurfaces of different primitives, which are fused together [41.48] (Fig. 41.9), VIDEO 76. In this case, a strategy for refining the fusion boundaries between primitives is needed and additional data may be gathered for this specific purpose.

Super-quadrics. A slightly more complex primitive shape is a super-quadric surface, which gives more flexibility than a simple primitive. Super-quadric surfaces are surfaces described in spherical coordinates by

$$S(\omega_1,\omega_2) := \begin{pmatrix} c_x \cos^{\epsilon_1}(\omega_1) \cos^{\epsilon_2}(\omega_2) \\ c_y \cos^{\epsilon_1}(\omega_1) \sin^{\epsilon_2}(\omega_2) \\ c_z \sin^{\epsilon_1}(\omega_1) \end{pmatrix} . \quad (41.17)$$

The parameters c_x, c_y, and c_z describe the extent of the super-quadric along the axes x, y, and z, respectively. By adjusting the exponents ϵ_1 and ϵ_2, we can vary the resulting shape anywhere from an ellipse to a box. For example, setting $\epsilon_1, \epsilon_2 \approx 0$ results in box shapes; $\epsilon_1 = 1, \epsilon_2 \approx 0$ results in cylindrical shapes; $\epsilon_1, \epsilon_2 = 1$ results in ellipses. Hence, by estimating $c_x, c_y, c_z, \epsilon_1$, and ϵ_2, we can model a rich variety of shapes [41.50].

Polygonal Mesh. When none of the parametric representations capture the object's shape well enough, a polygonal mesh can be used. A polygonal mesh (or poly-mesh) can represent arbitrary shapes as it consists of polygons (typically, triangles) linked together to form a surface (Fig. 41.2). The simplest method is to link up collected data points to create faces (i. e., the polygons) of the mesh. However, this method tends to under-estimate the overall size of the object because data points at object corners are rarely gathered. This effect is present even for dense 3-D sensors, but especially noticeable here due to sparsity of the data. A more accurate representation can be obtained by collecting several data points for each polygonal face and, then, intersecting the polygonal faces to obtain the corners of the mesh. However, this tends to be a more involved, manual process [41.28].

Point Cloud. Objects can also be represented as clouds of gathered data points. In this case, the accuracy of the representation directly depends on the density of the gathered data and, hence, these methods typically spend a lot of time collecting data [41.51, 52].

Splines. Two-dimensional objects can be represented by splines. For example, *Walker* and *Salisbury* used proximity sensors and a planar robot to map smooth shapes placed on a flat surface [41.53] (Fig. 41.10).

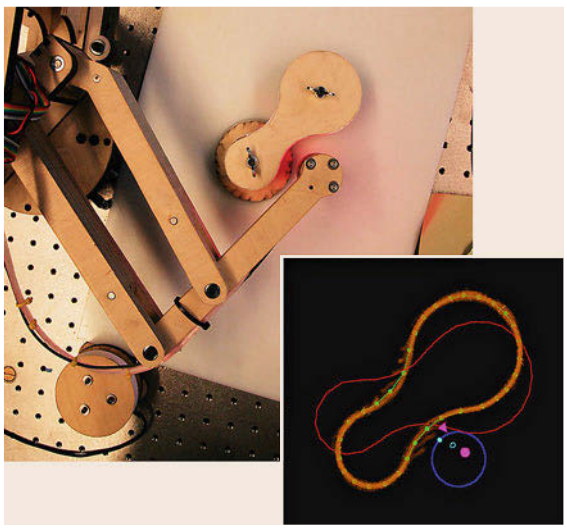


Fig. 41.10 A 2-D object is explored and represented by splines (after [41.53])

Volumetric Grid Maps. Objects can also be represented by volumetric grids. For 2-D objects, this representation is called an occupancy grid map. For 3-D objects, it is called a voxel grid map. Each grid cell represents a small part of the surface (or space) and records a binary value (e.g., whether the cell is occupied or not). This value is estimated based on gathered data using probabilistic methods similar to the methods used for mobile robot mapping with a 2-D laser [41.48, 54] (Fig. 41.11). For additional information on occupancy grid maps see Chap. 45.

Data Gathering

Different types of data can be collected during mapping. Tactile array images can be very informative for mapping planar objects. Sensed end-effector contact gives 3-D data points on the object surface. Link contacts can also be estimated and used for mapping even without sensory skin (Sect. 41.3.4 for more details). Moreover, the volume swept by the robot arm provides negative information, that is, the absence of obstacles in this volume. All of this information can be used both for mapping and for planning the next most informative move.

Guided Exploration. In many applications, it is acceptable to have a human operator guide the collection of data. This method allows for more accurate model building because humans can select the best points to sense on the object. Humans can also build variable accuracy models, where more data are collected near areas/features of interest and less data are collected elsewhere.

Autonomous Exploration. When autonomous model-building is required, a number of exploration strategies exist. The simplest approach is to select the sensing locations randomly. However, when using random sampling, the density of the gathered data can vary. An exhaustive strategy can ensure that points are gathered with a specified density. A triangular lattice arrangement has been shown to be the most optimal for such a strategy; however, this method takes considerable time. The most optimal decision about the next sensing location can be made by considering the amount of information that can be gained. These techniques are similar to the techniques described in Sect. 41.2.5. Several primitive and advanced exploration strategies for tactile mapping are compared in [41.48].

41.3.2 Articulated Objects

In all the prior sections, we were concerned with rigid objects, which tend to be simpler to model. The next step up in terms of complexity are objects comprised of several rigid parts, which can move with respect to each other. These objects are called articulated objects. For many objects, the motion of connected parts with respect to each other is restricted to only one or two DOFs by prismatic or revolute joints. Examples of articulated objects include simple objects such as doors, cabinet drawers, and scissors, as well as more complex objects such as robots and even humans (Fig. 41.12).

Manipulation can play an important role in modeling novel articulated objects. By applying a force to one part of an articulated object, we can observe how the other parts move and thus infer relationships between parts. Observation can be performed via the sense of touch or via some other sense, for example, vision. For example, if a robot arm pulls/pushes a door compliantly, its trajectory allows the robot to determine the width of the door [41.57]. Alternatively, if a robot arm pushes one handle of a pair of scissors, the location of the scissors joint can be inferred by tracking visual features with an overhead camera (Fig. 41.13).

A kinematic structure consisting of several revolute and prismatic joints can be represented as a relational model, which describes relationships between object parts. In order to build a relational model of a multijoint object, a robot needs an efficient exploration strategy for gathering the data. The exploration task can be described using a Markov decision process, similar to the ones discussed in Sect. 41.2.5. Since the goal is

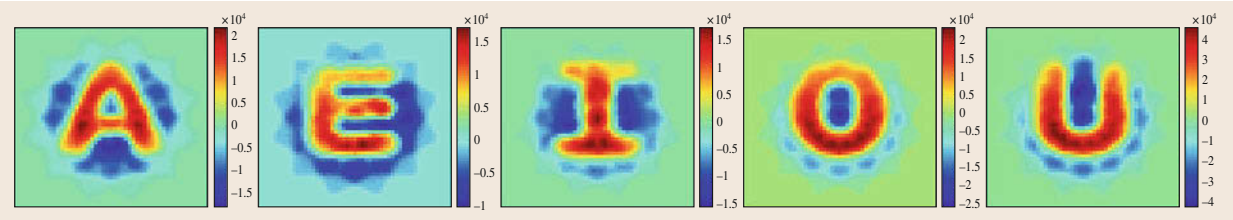


Fig. 41.11 Reconstruction of letters from tactile images. Pixels are colored by probability of occupancy in log-odds form. This probability can be seen to be high on the letters themselves, to be low in the surrounding area where measurements were taken, and to decay to the prior probability in the surrounding un-sensed area (after [41.54])

Fig. 41.12 Robot manipulating a variety of common articulated objects (*from left to right*): a cabinet door that opens to the right, a cabinet door that opens to the left, a dishwasher, a drawer, and a sliding cabinet door (after [41.55])

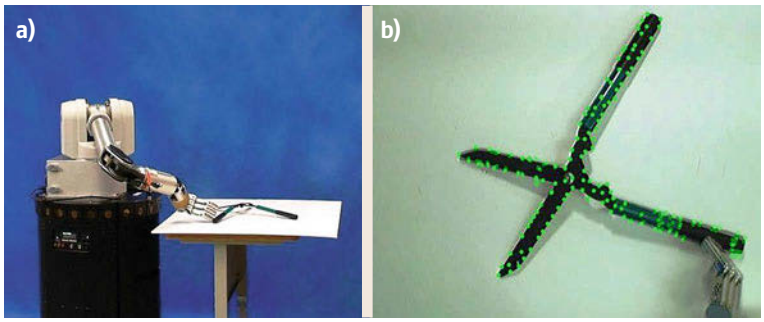


Fig.41.13a,b Robot exploring an articulated object (after [41.56]) **(a)** a robot arm pushes one handle of a pair of scissors to infer the location of its joint, **(b)** the view from the robot's camera with tracked visual features shown as green dots

to learn relational models, these Markov decision processes (MDPs) are called relational Markov decision processes (RMDPs). With the aid of RMDPs, robots can learn efficient strategies for exploring previously unknown articulated objects, learning their kinematic structure, and later manipulating them to achieve the goal state [41.56].

Some kinematic structures do not consist entirely of revolute and prismatic joints. A common example is garage doors, which slide in and out along a curved trajectory. General kinematic models can be represented by Gaussian processes, which can capture revolute, prismatic and other types of connections between parts [41.55], VIDEO 78.

41.3.3 Deformable Objects

Even more complex than articulated objects are deformable objects, which can be moved to form arbitrary shapes. The simplest of these are one-dimensional (1-D) deformable objects, such as ropes, strings, and cables. These objects are typically modeled as line objects with arbitrary deformation capability (except for stretching). A number of approaches can be found in [41.58].

A more complex type of deformable objects are planar deformable objects, such as fabric or deformable plastic sheets. While these objects can be represented as networks of nodes, modeling interaction between these nodes can get expensive and is not always necessary. For example, *Platt* et al. developed an approach to map planar deformable objects by swiping them between the robot's fingers [41.59] (Fig. 41.14). These maps can later be used for localization during subsequent swipes, in a manner similar to indoor robot localization on a map.

The most complex deformable objects are 3-D deformable objects. This category includes a great variety of objects in our everyday environments: sponges, sofas, and bread are just a few examples. Moreover, in medical robotics, tissues typically have to be modeled as 3-D deformable objects. Since the stiffness and composition of these objects can vary, they need to be modeled as networks of nodes, where different interaction can take place between distinct nodes. Such

Fig. 41.14 Robot exploring a 2-D deformable object (after [41.59])

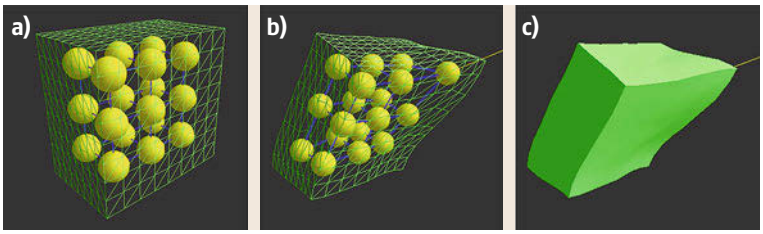


Fig.41.15a–c Model of a 3-D deformable object. The object at rest **(a)**. The object under influence of an external force **(b,c)** (after [41.60]

models typically require a lot of parameters to describe even reasonably small objects. Although, some efficiency improvements can be obtained by assuming that subregions of the object have similar properties. Despite the challenges, 3-D deformable object modeling has many useful applications, for example, a surgery simulator or 3-D graphics. *Burion* et al. modeled 3-D objects as a mass-spring system (Fig. 41.15), where mass nodes are interconnected by springs [41.60]. By applying forces to the object, we can observe the deformation and infer stiffness parameters of the springs (i. e., elongation, flexion, and torsion).

41.3.4 Contact Estimation

Up until now, we have assumed that the contact location during data gathering was somehow known. However, determining the contact point is not as straightforward for robots as it is for humans, because robots are not fully covered in sensory skin. Thus, during interaction between a robot and its environment, nonsensing parts of the robot can come in contact with the environment. If such contact goes undetected, the robot and/or the environment may be damaged. For this reason, touch sensing is typically performed by placing sensors at the end-effector and ensuring that the sensing trajectories never bring nonsensing parts of the robot in contact with the environment.

Clearly, end-effector-only sensing is a significant constraint, which is difficult to satisfy in real-world conditions. Work on whole-body sensory skin is ongoing [41.61], yet for now only small portions of the robot's surface tend to be covered with sensory skin.

In lieu of sensory skin, contact can be estimated from geometry of the robot and the environment via active sensing with compliant motions. Once a contact between the robot and the environment is detected (e.g., via deviation in joint torques or angles), the robot switches to the active sensing procedure. During this procedure, the robot performs compliant motions (either in hardware or software), by applying a small force toward the environment and moving back and forth. Intuitively, the robot's body carves out free space as it moves, and thus, via geometrical reasoning, the shape of the environment and contact points can be deduced. However, from a mathematical perspective, this can be done in several different ways.

One of the earliest approaches was developed by *Kaneko* and *Tanie*, who formulated the self-posture changability (SPC) method [41.63]. In SPC, the intersection of finger surfaces during the compliant motions is taken to be the point of contact. This method works well in areas of high curvature of the environment (e.g., corners), but can give noisy results in areas of shallow curvature. In contrast to SPC, the space sweeping algorithm by *Jentoft* and *Howe* marks all points on finger surfaces as possible contact, and then gradually rules out the possibilities [41.62] (Fig. 41.16). The space sweeping algorithm provides less noisy estimates in areas of shallow curvature, but can be more susceptible to even the smallest disturbances of the environment's surface.

Mathematically, contact estimation can also be formulated as a Bayesian estimation problem. If the robot moves using compliant motions as described above, the

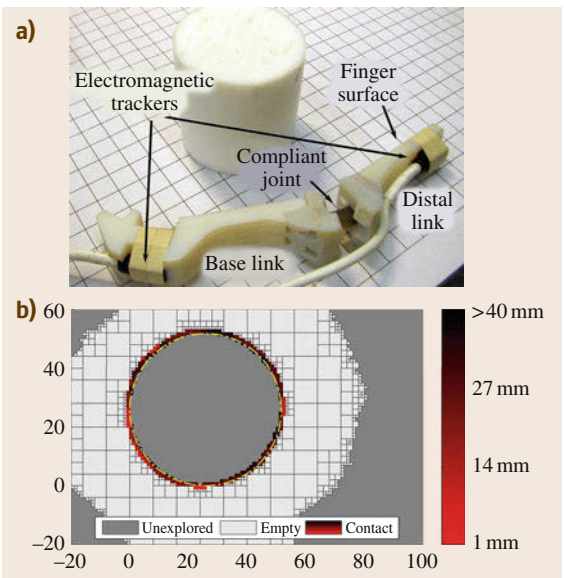


Fig. 41.16 **(a)** A two-link compliant finger used for contact sensing. **(b)** The resulting model built by the space sweeping algorithm (after [41.62])

sensor data are the joint angles of the robot. The state is the shape and position of the environment, represented using one of the rigid object models described in Sect. 41.3.1. Then, each possible state can be scored using a probabilistic measurement model, for example, the proximity model described in Sect. 41.2.2, and the problem can be solved using Bayesian inference methods [41.64].

41.3.5 Physical Properties

Up until now, we have focused on figuring out the shape and articulation of objects. While there are some advantages to doing so with tactile data as opposed to visual data, visual data can also be used to generate shape and articulation models. However, there are many properties of objects that are not easily detectable with vision alone, such as surface texture and friction, inertial properties, stiffness, or thermal conductivity.

Friction and Surface Texture

Surface friction, texture, and roughness can be estimated by dragging a fingertip over the object. The forces required to drag a fingertip over a surface at different velocities can be used to estimate surface friction parameters [41.65–67]. Accelerometer or pressure data is useful for estimating surface roughness (by the amplitude and frequency of vibration while sliding) [41.67, 68], or more directly identifying surface textures using classification techniques such as support vector machines (SVMs) or k-nearest neighbours (k-NNs) [41.69]. The position of the fingertip being dragged across the surface is also useful for estimating surface roughness and for mapping out small surface features [41.65]. Static friction is also often estimated as a by-product of trying to avoid object slip during a grasp, based on the normal force being applied at the moment when slip happens [41.70, 71]. Finally, the support friction distribution for an object sliding on a surface can be estimated by pushing the object at different locations and seeing how it moves [41.53, 72].

Inertia

Atkeson et al. [41.73] estimate the mass, center of mass, and moments of inertia of a grasped object, using measurements from a wrist force-torque sensor. Objects that are not grasped but that are sitting on a table can be pushed; if the pushing forces at the fingertips are known, the object's mass as well as the center of mass (COM) and inertial parameters in the plane can be estimated. *Yu* et al. [41.74] do so by pushing objects using two fingertips equipped with force/torque sensors and recording the fingertip forces, velocities, and accelerations. *Tanaka* and *Kushihama* [41.75] estimate the object's mass by pushing the object with known contact force and watching the resulting motion with a camera.

Stiffness

Identifying the stiffness (or the inverse, compliance) of an object is useful in terms of modeling the material properties of an object, but more directly, it is also useful for preventing the robot from crushing or damaging delicate objects. Gently squeezing an object and measuring the deflection seen for a given contact normal force is one method often used to estimate stiffness [41.68, 76, 77]. *Omata* et al. [41.78] use a piezoelectric transducer and a pressure sensor element in a feedback circuit that changes its resonant frequency upon touching an object, to estimate the object's acoustic impedance (which varies with both object stiffness and mass). *Burion* et al. [41.60] use particle filters to estimate the stiffness parameters of a mass-spring model for deformable objects, based on the displacements seen when applying forces to different locations on the object.

Thermal Conductivity

Thermal sensors are useful for identifying the material composition of objects based on their thermal conductivity. Fingertips with internal temperature sensors [41.68, 79] can estimate the thermal conductivity of the object by heating a robot fingertip to above room temperature, touching the fingertip to the object, and measuring the rate of heat loss.

41.4 Recognition

Another common goal of touch-based perception is to recognize objects. As with more typical, vision-based object recognition, the goal is to identify the object from a set of possible objects. Some methods assume known object shapes and try to figure out which object geometry matches the sensor data best. Other methods use feature-based pattern recognition to identify the object, often with a bag-of-features model that does not rely on geometric comparisons to overall object shape. Finally, there are methods that use object properties such as elasticity or texture to identify objects.

41.4.1 Object Shape Matching

Any of the methods for shape-based localization can also be used for recognition, by performing localiza-

tion on all potential objects. In terms of the equations used in Sect. 41.2, if we are considering a limited number of potential objects with known mesh models, we can simply expand our state space to include hypotheses over object identity and pose, instead of just pose. If we add an object identifier (ID) to each state in the state vector $\boldsymbol{X}$, and use the appropriate mesh model for that object ID for the measurement model for each state, all of the equations used in the previous sections still hold. The object ID for the combined object shape and pose that best matches the collected sensor data is then our top object hypothesis.

Nonprobabilistic methods for localization and object recognition often involve geometrically pruning objects from a set of possible object shapes/poses that do not match sensed data. For instance, *Gaston* and *Lozano-Perez* [41.1] and *Grimson* and *Lozano-Perez* [41.2] recognize and localize polyhedral objects by using the positions and normals of sensed contact points in a generate-and-test framework: generate feasible hypotheses about pairings between sensed contact points and object surfaces, and test them for consistency based on constraints over pairs of sensed points and their hypothesized face locations and normals. *Russell* [41.80] classify tactile array images as point, line, or area contacts, and use the time-sequence of contacts seen when rolling the object over the array to prune inconsistent object shapes.

Other methods recognize objects by forming a model of the object, then matching the new model with database objects. For instance, *Allen* and *Roberts* [41.81] build superquadric models of the object from contact data, then match the superquadric parameters against a database of object models. *Caselli* et al. [41.82] build polyhedral representations of objects (one based on contact points and normals that represents the space that the true object must fit inside, and one based on just contact points that the true object must envelop), and use the resulting representations to prune incompatible objects in a database of polyhedral objects.

41.4.2 Statistical Pattern Recognition

Many methods for tactile object recognition use techniques from statistical pattern recognition, summarizing sensed data with features and classifying the resulting feature vectors to determine the identity of the object. For instance, *Schöpfer* et al. [41.84] roll a tactile array over objects along a straight line, compute features such as image blob centroids and second or third moments over the resulting time series of tactile images, and use a decision tree to classify objects based on the resulting features. *Bhattacharjee* et al. [41.85] use k-nearest-neighbors on a set of features based on time series images from a forearm tactile array to identify whether an object is fixed or movable, and rigid or soft, in addition to recognizing the specific object.

Tactile array data is very similar to small image patches in visual data, and so techniques for object recognition using tactile arrays can look much like object recognition techniques that use visual data. For instance, *Pezzementi* et al. [41.86] and *Schneider* and *Sturm* [41.83] both use a Bag-of-Features model on fingertip tactile array data to recognize objects. The process for tactile data is almost the same as for visual images: for each training object (image), a set of tactile array images (2-D image patches) is obtained (Fig. 41.17), features (which are vectors of descriptive statistics) based on each tactile image are computed, and the features for all objects in the training set are clustered to form a vocabulary of canonical features (Fig. 41.18). Each object is represented as a histogram of how often each feature occurs. When looking at a new unknown object, the features computed from the tactile array images are similarly used to compute a his-

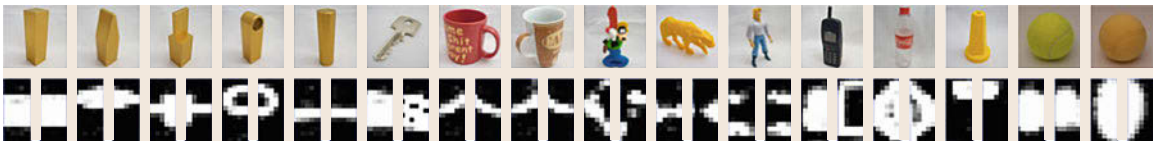


Fig. 41.17 Some objects and their associated tactile images for both left and right fingertips (after [41.83])

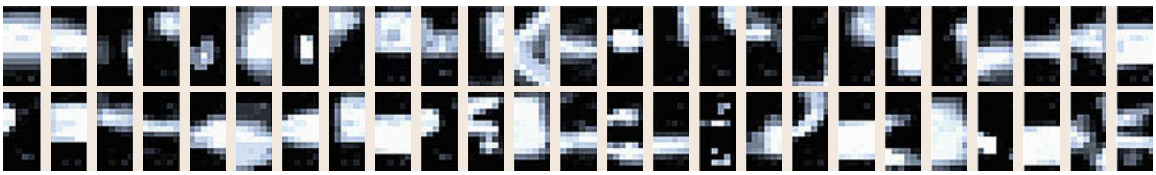


Fig. 41.18 Vocabulary of tactile images created using unsupervised clustering. Only the *left* finger's image is shown for each pair (after [41.83])

togram, and then the new histogram is compared against those of the objects in the database to find the best match. The one notable difference in the two domains is that tactile data is much more time-consuming and difficult to obtain than appropriate image patches: in 2-D visual object recognition, an interest point detection algorithm can quickly produce hundreds of salient patches from a given 2-D image, whereas for tactile object recognition, each tactile array requires touching the object, which in turn requires a planner (such as the ones described in Sect .41.2.5) to decide how to gather appropriate tactile data.

Tactile object exploration also allows one to gather and use information besides just tactile image patches: for instance, *Gorges* and *Navarro* [41.87] envelop objects in a multifingered hand, and use the resulting joint angles as features that directly relate to object dimensions, alongside additional features computed from tactile image patterns. In [41.88], *Pezzementi* and *Hager* combine techniques from vision-based object recognition with the type of Bayesian recognition-by-localization described in Sect. 41.4.1, using pairs of fingertip tactile image features along with their positions and surface normals to update a belief state over object poses and identities.

41.4.3 Recognition by Material Properties

Tactile object recognition sometimes has significant advantages over purely visual object recognition, particularly in cases where objects are visually similar but have different material properties detectable by tactile sensors. A number of recent methods classify objects using properties such as elasticity, surface texture, and thermal conductivity. Any of the methods listed in Sect. 41.3.5 can be used to identify objects, using many of the techniques described in Sect. 41.4.2. For instance, surface texture and roughness can be used to distinguish objects, as in [41.69] or [41.67], as can thermal conductivity, as in [41.79].

More specific properties can also be used to classify objects of particular types: for instance, *Chitta* et al. [41.89] identify bottles and also classify them as open or closed, full or empty by squeezing and rolling them from side to side and using decision tree classifiers on the resulting tactile measurements. *Drimus* et al. [41.90] also identify objects by squeezing them, but using k-nearest neighbor on the time series data of tactile measurements after performing dynamic time warping to align them.

41.4.4 Combining Visual and Tactile Sensing

Of course, visual object recognition also has advantages over purely tactile methods, and so combining data from both modalities can lead to even better object recognition and localization. For instance, *Allen* [41.91] uses sparse stereo data to provide regions of interest to explore with tactile sensors, then fuses the stereo data with the resulting tactile points to create surface and feature descriptions, which are then matched against objects in the database; object hypotheses are further verified with tactile exploration. *Boshra* and *Zhang* [41.92] recognize polyhedral objects by using tactile features and visual vertices and edges to formulate constraints in a constraint-satisfaction problem. *Nakamura* [41.93] grasps objects and learns object categories based on visual data (in the form of SIFT descriptors), audio information obtained while shaking (using a codebook of audio features based on mel-frequency cepstral coefficients), and haptic information obtained while squeezing (using hardness estimates based on finger pressure sensor readings); the learning and inference are done using probabilistic latent semantic analysis (PLSA).

Segmenting an object from a visual scene (which makes further efforts at recognition much easier) is also made easier by combining visual and tactile sensing; by pushing on an object and seeing which parts of the visual scene move together, objects can be segmented both from the background and from each other. For instance, *Gupta* and *Sukhatme* [41.94] use RGB-D data along with manipulation primitives such as spreading motions to segment and sort Duplo bricks, and *Hausman* et al. [41.95] segment textureless objects in RGB-D data by performing visual segmentation and then resolving ambiguities using pushing motions.

Work on categorizing objects by their affordances or functions often also uses both visual and tactile data. For instance, *Sutton* et al. [41.96] uses both shape-based reasoning (using shape derived from range images) as well as interaction-based reasoning (using data from the robot physically interacting with the object) to predict and confirm object affordances such as grasping, containing, and sitting (being suitable for humans to sit upon). *Bekiroglu* [41.97] performs inference on a Bayesian network using visual, proprioceptive, and tactile data to assess whether an object/grasp combination will be both stable and also good for a task such as pouring, hand-over, or dishwashing.

41.5 Conclusions

This chapter described perceptual methods that rely on manipulation. Although these methods are generally more complex than vision-based perception, they are useful in low-visibility conditions, in determining properties that require manipulation, and in situations where perception is needed for subsequent manipulation tasks. Perception-via-manipulation faces special challenges due to the low data acquisition rate and the fact that each sensing action disturbs the scene. To address these challenges, special inference methods have been developed to cope with situations where data are insufficient to fully constrain the problem (Sect. 41.2.3). Special planning methods had to be applied to minimize the number of sensing actions and reduce the possibility of significant scene disturbance or non-sensed contacts (Sect. 41.2.5). These challenges also dictate the choice of object models (Sect. 41.3), and play a role in object recognition methods (Sect. 41.4).

41.5.1 Real-Life Applications

Perception-*via*-manipulation is already used in a number of real-world applications and an even greater number of applications will likely benefit from it in the near future. In manufacturing, touch-based localization of parts for machining or assembly is often preferred to optical methods because it is not affected by optical properties of objects (e.g., transparency or reflectivity). At present, low uncertainty in part positioning is required prior to touch-based localization, which leads to high instrumentation and reconfiguration costs. However, the new localization methods presented in this chapter can cope with high uncertainty and thus enable flexible manufacturing lines capable of producing a variety of parts without reconfiguration. In the future, fully automated in-home manufacturing may even become possible.

Medical robotics is another mature real-world application, in which robotic contact with the environment (i. e., the patient) is not only unavoidable, it is in fact required for many of the procedures. Visibility is often poor, and sensing with devices that emit sound, light, or other waves can be harmful to the patient. Under these conditions, sensing-via-contact can sometimes be the most viable option. The environment is inherently deformable, and thus, medical robotics has been a strong driver for research on manipulation and perception of deformable objects.

Another area in which robots are indispensable is underwater robotics. It is not always safe or convenient for humans to be performing tasks deep underwater, and hence, remotely operated robots (ROVs) are a natural choice. ROVs even made international news during recovery operations from the BP oil spill disaster in the spring of 2010. Since visibility underwater is often poor, perception-via-contact plays an even more important role.

Beyond the more established robotic applications, as the fields of service robotics and disaster recovery develop, the necessity of sensing-via-contact becomes more and more obvious. In these applications, robots often have to work in clutter and in close proximity to humans. Under these conditions, visibility can be poor, yet at the same time, unsensed contact can be disastrous.

41.5.2 Future Directions of Research

Inspired by the many existing and potential applications, the field of perception-by-manipulation is evolving and receiving more attention. Despite the challenges, there are bountiful opportunities for further fruitful research:

- The vast majority of touch-based perception work deals with rigid objects, whereas many objects in our environments are articulated and/or deformable. These types of objects need to be studied in more detail.
- For rigid objects, touch-based perception of moving objects, object piles, and objects in clutter has been under-explored.
- Whole-body contact estimation needs further attention, with or without sensory skin. The challenges here are significant, but good solutions would increase the safety of robots in service applications and other unstructured environments.
- Last but not least, while Bayesian methods have found wide acceptance in other areas of robotics, touch-based perception has yet to fully benefit from these methods. Thus, application of Bayesian methods to any of the above listed areas is particularly promising.

41.5.3 Further Reading

For a broader view, a lot of related material can be found in other chapters throughout this handbook. Most sensing-via-manipulation is done using contact or near-contact (i. e., close-range) sensors. These sensors are described in Chap. 28. A few approaches use classical vision sensors in conjunction with manipulation (as in

Sect. 41.3.2, for example). Vision sensors and methods are covered in Chaps. 32 and 34. Manipulator design and control are covered in Chaps. 4, 8, and 9. Other relevant chapters include 7, 14 and 47 for planning methods, 37, 45, and 46 for modeling, 33 for recognition, 5 and 35 for inference.

Video-References

VIDEO 76 Tactile exploration and modeling using shape primitives
available from http://handbookofrobotics.org/view-chapter/41/videodetails/76

VIDEO 77 Tactile localization of a power drill
available from http://handbookofrobotics.org/view-chapter/41/videodetails/77

VIDEO 78 Modeling articulated objects using active manipulation
available from http://handbookofrobotics.org/view-chapter/41/videodetails/78

VIDEO 721 Touch-based door handle localization and manipulation
available from http://handbookofrobotics.org/view-chapter/41/videodetails/721

VIDEO 723 6-DOF object localization via touch
available from http://handbookofrobotics.org/view-chapter/41/videodetails/723

References

41.1 P.C. Gaston, T. Lozano-Perez: Tactile recognition and localization using object models: The case of polyhedra on a plane, IEEE Trans. Pattern Anal. Machine Intell. **6**(3), 257–266 (1984)

41.2 W.E.L. Grimson, T. Lozano-Perez: Model-based recognition and localization from sparse range or tactile data, Int. J. Robotics Res. **3**, 3–35 (1984)

41.3 O.D. Faugeras, M. Hebert: A 3-D recognition and positioning algorithm using geometrical matching between primitive surfaces, Proc. 8th Intl. Jt. Conf. Artif. Intell., Los Altos (1983) pp. 996–1002

41.4 S. Shekhar, O. Khatib, M. Shimojo: Sensor fusion and object localization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1986) pp. 1623–1628

41.5 Y.-B. Jia, M. Erdmann: Pose from pushing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 165–171

41.6 K.Y. Goldberg: Orienting polygonal parts without sensors, Algorithmica **10**(2-4), 201–225 (1993)

41.7 M.A. Erdmann, M.T. Mason: An exploration of sensorless manipulation, IEEE J. Robotics Autom. **4**(4), 369–379 (1988)

41.8 H.T. Yau, C.H. Menq: An automated dimensional inspection environment for manufactured parts using coordinate measuring machines, Int. J. Prod. Res. **30**(7), 1517–1536 (1992)

41.9 K.T. Gunnarsson, F.B. Prinz: CAD model-based localization of parts in manufacturing, Computer **20**(8), 66–74 (1987)

41.10 K.T. Gunnarsson: Optimal Part Localization by Data Base Matching with Sparse and Dense Data, Ph.D. Thesis (Dept. Mech. Eng., Carnegie Mellon Univ. Pittsburgh 1987)

41.11 H.J. Pahk, W.J. Ahn: Precision inspection system for aircraft parts having very thin features based on CAD/CAI integration, Int. J. Adv. Manuf. Technol. **12**(6), 442–449 (1996)

41.12 M.W. Cho, T.I. Seo: Inspection planning strategy for the on-machine measurement process based on CAD/CAM/CAI integration, Int. J. Adv. Manuf. Technol. **19**(8), 607–617 (2002)

41.13 Z. Xiong: Workpiece Localization and Computer Aided Setup System, Ph.D. Thesis (Hong Kong Univ. Sci. Technol., Hong Kong 2002)

41.14 J. Hong, X. Tan: Method and apparatus for determining position and orientation of mechanical objects, US Patent US520 8763 A (1993)

41.15 B.K.P. Horn: Closed-form solution of absolute orientation using unit quaternions, J. Opt. Soc. Am. A **4**(4), 629–642 (1987)

41.16 C.H. Menq, H.T. Yau, G.Y. Lai: Automated precision measurement of surface profile in CAD-directed inspection, IEEE Trans. Robotics Autom. **8**(2), 268–278 (1992)

41.17 Y. Chu: Workpiece Localization: Theory, Algorithms and Implementation, Ph.D. Thesis (Hong Kong Univ. Sci. Technol., Hong Kong 1999)

41.18 Z. Xiong, M.Y. Wang, Z. Li: A near-optimal probing strategy for workpiece localization, IEEE Trans. Robotics **20**(4), 668–676 (2004)

41.19 Y. Huang, X. Qian: An efficient sensing localization algorithm for free-form surface digitization, J. Comput. Inform. Sci. Eng. **8**, 021008 (2008)

41.20 L.M. Zhu, H.G. Luo, H. Ding: Optimal design of measurement point layout for workpiece localization, J. Manuf. Sci. Eng. **131**, 011006 (2009)

41.21 L.M. Zhu, Z.H. Xiong, H. Ding, Y.L. Xiong: A distance function based approach for localization and profile error evaluation of complex surface, J. Manuf. Sci. Eng. **126**, 542–554 (2004)

41.22 Y. Sun, J. Xu, D. Guo, Z. Jia: A unified localization approach for machining allowance optimization of complex curved surfaces, Precis. Eng. **33**(4), 516–523 (2009)

41.23 K. Gadeyne, H. Bruyninckx: Markov techniques for object localization with force-controlled robots, 10th Int. Conf. Adv. Robotics (ICAR) (2001) pp. 91–96

41.24 S.R. Chhatpar, M.S. Branicky: Particle filtering for localization in robotic assemblies with position uncertainty, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)

41.25 C. Corcoran, R. Platt: A measurement model for tracking hand-object state during dexterous manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

41.26 K. Hsiao, L. Kaelbling, T. Lozano-Perez: Task-driven tactile exploration, Robotics Sci. Syst. Conf. (2010)

41.27 A. Petrovskaya, O. Khatib: Global localization of objects via touch, IEEE Trans. Robotics **27**(3), 569–585 (2011)

41.28 A. Petrovskaya: Towards Dependable Robotic Perception, Ph.D. Thesis (Stanford Univ., Stanford 2011)

41.29 A. Petrovskaya, O. Khatib, S. Thrun, A.Y. Ng: Bayesian estimation for autonomous object manipulation based on tactile sensors, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 707–714

41.30 S. Arulampalam, S. Maskell, N. Gordon, T. Clapp: A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking, IEEE Trans. Signal Process. **50**(2), 174–188 (2002)

41.31 A. Petrovskaya, S. Thrun, D. Koller, O. Khatib: Guaranteed inference for global state estimation in human environments, Mob. Manip. Workshop Robotics Sci. Syst. (RSS) (2010)

41.32 M.C. Koval, M.R. Dogar, N.S. Pollard, S.S. Srinivasa: Pose estimation for contact manipulation with manifold particle filters, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 4541–4548

41.33 J. Deutscher, A. Blake, I. Reid: Articulated body motion capture by annealed particle filtering, IEEE Conf. Comput. Vis. Pattern Recog. (CVPR) (2000)

41.34 J. Deutscher, I. Reid: Articulated body motion capture by stochastic search, Int. J. Comput. Vis. **61**(2), 185–205 (2005)

41.35 A.O. Balan, L. Sigal, M.J. Black: A quantitative evaluation of video-based 3D person tracking, 2nd Jt. IEEE Int. Workshop Vis. Surveill. Perform. Eval. Track. Surveill. (2005) pp. 349–356

41.36 K. Huebner: BADGr – A toolbox for box-based approximation, decomposition and grasping, Robotics Auton. Syst. **60**(3), 367–376 (2012)

41.37 A.T. Miller, P.K. Allen: Graspit! a versatile simulator for robotic grasping, IEEE Robotics Autom, Magaz. **11**(4), 110–122 (2004)

41.38 K. Hsiao, T. Lozano-Pérez, L.P. Kaelbling: Robust belief-based execution of manipulation programs, 8 Int. Workshop Algorithm. Found. Robotics (WAFR) (2008)

41.39 P. Hebert, T. Howard, N. Hudson, J. Ma, J.W. Burdick: The next best touch for model-based localization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 99–106

41.40 J.L. Schneiter, T.B. Sheridan: An automated tactile sensing strategy for planar object recognition and localization, IEEE Trans. Pattern Anal. Mach. Intell. **12**(8), 775–786 (1990)

41.41 S. Javdani, M. Klingensmith, J.A. Bagnell, N.S. Pollard, S.S. Srinivasa: Efficient touch based localization through submodularity, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 1828–1835

41.42 K. Hsiao, L.P. Kaelbling, T. Lozano-Pérez: Robust grasping under object pose uncertainty, Auton. Robots **31**(2/3), 253–268 (2011)

41.43 M. Dogar, S. Srinivasa: A framework for push-grasping in clutter. In: *Robotics: Science and Systems VII*, ed. by H.F. Durrant-Whyte, N. Roy, P. Abbeel (MIT, Cambridge 2011)

41.44 R. Platt, L. Kaelbling, T. Lozano-Perez, R. Tedrake: Efficient planning in non-Gaussian belief spaces and its application to robot grasping, Int. Symp. Robotics Res. (2011)

41.45 L.P. Kaelbling, T. Lozano-Pérez: Integrated task and motion planning in belief space, Int. J. Robotics Res. **32**(9/10), 1194–1227 (2013)

41.46 M. Erdmann: Shape recovery from passive locally dense tactile data, Proc. Workshop Algorithm. Found. Robotics (1998) pp. 119–132

41.47 M. Moll, M.A. Erdmann: Reconstructing the shape and motion of unknown objects with active tactile sensors, Algorithm. Found. Robotics, Vol. V (2003) pp. 293–309

41.48 F. Mazzini: Tactile Mapping of Harsh, Constrained Environments, with an Application to Oil Wells, Ph.D. Thesis (MIT, Cambridge 2011)

41.49 P. Slaets, J. Rutgeerts, K. Gadeyne, T. Lefebvre, H. Bruyninckx, J. De Schutter: Construction of a geometric 3-D model from sensor measurements collected during compliant motion, 9th. Proc. Int. Symp. Exp. Robotics (2004)

41.50 A. Bierbaum, K. Welke, D. Burger, T. Asfour, R. Dillmann: Haptic exploration for 3D shape reconstruction using five-finger hands, 7th IEEE-RAS Int. Conf. Hum. Robots (2007) pp. 616–621

41.51 S.R. Chhatpar, M.S. Branicky: Localization for robotic assemblies using probing and particle filtering, Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (2005) pp. 1379–1384

41.52 M. Meier, M. Schopfer, R. Haschke, H. Ritter: A probabilistic approach to tactile shape reconstruction, IEEE Trans. Robotics **27**(3), 630–635 (2011)

41.53 S. Walker, J.K. Salisbury: Pushing using learned manipulation maps, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 3808–3813

41.54 Z. Pezzementi, C. Reyda, G.D. Hager: Object mapping, recognition, and localization from tactile geometry, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5942–5948

41.55 J. Sturm, A. Jain, C. Stachniss, C.C. Kemp, W. Burgard: Operating articulated objects based on experience, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 2739–2744

41.56 D. Katz, Y. Pyuro, O. Brock: Learning to manipulate articulated objects in unstructured environments using a grounded relational representation, Robotics Sci. Syst. (2008)

41.57 C. Rhee, W. Chung, M. Kim, Y. Shim, H. Lee: Door opening control using the multi-fingered robotic hand for the indoor service robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

41.58 D. Henrich, H. Wörn: *Robot Manipulation of Deformable Objects* (Springer, Berlin, Heidelberg 2000)

41.59 R. Platt, F. Permenter, J. Pfeiffer: Using bayesian filtering to localize flexible materials during manipulation, IEEE Trans. Robotics **27**(3), 586–598 (2011)

41.60 S. Burion, F. Conti, A. Petrovskaya, C. Baur, O. Khatib: Identifying physical properties of deformable objects by using particle filters, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 1112–1117

41.61 A. Jain, M.D. Killpack, A. Edsinger, C. Kemp: Reaching in clutter with whole-arm tactile sensing, Int. J. Robotics Res. **32**(4), 458–482 (2013)

41.62 L.P. Jentoft, R.D. Howe: Determining object geometry with compliance and simple sensors, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 3468–3473

41.63 M. Kaneko, K. Tanie: Contact point detection for grasping an unknown object using self-posture changeability, IEEE Trans. Robotics Autom. **10**(3), 355–367 (1994)

41.64 A. Petrovskaya, J. Park, O. Khatib: Probabilistic estimation of whole body contacts for multi-contact robot control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 568–573

41.65 A.M. Okamura, M.A. Costa, M.L. Turner, C. Richard, M.R. Cutkosky: Haptic surface exploration, Lec. Notes Control Inform. Sci. **250**, 423–432 (2000)

41.66 A. Bicchi, J.K. Salisbury, D.L. Brock: Experimental evaluation of friction characteristics with an articulated robotic hand, Lec. Notes Control Inform. Sci. **190**, 153–167 (1993)

41.67 J.A. Fishel, G.E. Loeb: Bayesian exploration for intelligent identification of textures, Frontiers Neurorobot. **6**, 4 (2012)

41.68 P. Dario, P. Ferrante, G. Giacalone, L. Livaldi, B. Allotta, G. Buttazzo, A.M. Sabatini: Planning and executing tactile exploratory procedures, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1992) pp. 1896–1903

41.69 J. Sinapov, V. Sukhoy: Vibrotactile recognition and categorization of surfaces by a humanoid robot, IEEE Trans. Robotics **27**(3), 488–497 (2011)

41.70 M.R. Tremblay, M.R. Cutkosky: Estimating friction using incipient slip sensing during a manipulation task, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 429–434

41.71 R. Bayrleithner, K. Komoriya: Static friction coefficient determination by force sensing and its application, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1639–1646

41.72 K.M. Lynch: Estimating the friction parameters of pushed objects, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1993) pp. 186–193

41.73 C.G. Atkeson, C.H. An, J.M. Hollerbach: Estimation of inertial parameters of manipulator loads and links, Int. J. Robotics Res. **5**(3), 101–119 (1986)

41.74 Y. Yu, T. Arima, S. Tsujio: Estimation of object inertia parameters on robot pushing operation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1657–1662

41.75 H.T. Tanaka, K. Kushihama: Haptic vision-vision-based haptic exploration, Proc. 16th Int. Conf. Pattern Recognit. (2002) pp. 852–855

41.76 H. Yussof, M. Ohka, J. Takata, Y. Nasu, M. Yamano: Low force control scheme for object hardness distinction in robot manipulation based on tactile sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 3443–3448

41.77 J. Romano, K. Hsiao, G. Niemeyer, S. Chitta, K.J. Kuchenbecker: Human-inspired robotic grasp control with tactile sensing, IEEE Trans. Robotics **27**, 1067–1079 (2011)

41.78 S. Omata, Y. Murayama, C.E. Constantinou: Real time robotic tactile sensor system for the determination of the physical properties of biomaterials, Sens. Actuators A **112**(2), 278–285 (2004)

41.79 C.H. Lin, T.W. Erickson, J.A. Fishel, N. Wettels, G.E. Loeb: Signal processing and fabrication of a biomimetic tactile sensor array with thermal, force and microvibration modalities, IEEE Int. Conf. Robotics Biomim. (ROBIO) (2009) pp. 129–134

41.80 R.A. Russell: Object recognition by a 'smart' tactile sensor, Proc. Aust. Conf. Robotics Autom. (2000) pp. 93–98

41.81 P.K. Allen, K.S. Roberts: Haptic object recognition using a multi-fingered dextrous hand, Proc. IEEE Conf. Robotics Autom. (ICRA) (1989) pp. 342–347

41.82 S. Caselli, C. Magnanini, F. Zanichelli, E. Caraffi: Efficient exploration and recognition of convex objects based on haptic perception, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (1996) pp. 3508–3513

41.83 A. Schneider, J. Sturm: Object identification with tactile sensors using bag-of-features, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 243–248

41.84 M. Schöpfer, M. Pardowitz, R. Haschke, H. Ritter: Identifying relevant tactile features for object identification, Springer Tracts Adv, Robotics **76**, 417–430 (2012)

41.85 T. Bhattacharjee, J.M. Rehg, C.C. Kemp: Haptic classification and recognition of objects using a tactile sensing forearm, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 4090–4097

41.86 Z. Pezzementi, E. Plaku, C. Reyda, G.D. Hager: Tactile-object recognition from appearance information, IEEE Trans. Robotics **27**(3), 473–487 (2011)

41.87 N. Gorges, S.E. Navarro: Haptic object recognition using passive joints and haptic key features, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 2349–2355

41.88 Z. Pezzementi, G.D. Hager: Tactile object recognition and localization using spatially-varying appearance, Int. Symp. Robotics Res. (ISRR) (2011)

41.89 S. Chitta, J. Sturm, M. Piccoli, W. Burgard: Tactile sensing for mobile manipulation, IEEE Trans. Robotics **27**(3), 558–568 (2011)

41.90 A. Drimus, G. Kootstra, A. Bilberg, D. Kragic: Design of a flexible tactile sensor for classification of rigid and deformable objects, Robotics Auton. Syst. **62**(1), 3–15 (2012)

41.91 P. Allen: Integrating vision and touch for object recognition tasks, Int. J. Robotics Res. **7**(6), 15–33 (1988)
41.92 M. Boshra, H. Zhang: A constraint-satisfaction approach for 3D vision/touch-based object recognition, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 2 (1995) pp. 368–373
41.93 T. Nakamura: Multimodal object categorization by a robot, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007) pp. 2415–2420
41.94 M. Gupta, G. Sukhatme: Using manipulation primitives for brick sorting in clutter, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 3883–3889
41.95 K. Hausman, F. Balint-Benczedi, D. Pangercic, Z.-C. Marton, R. Ueda, K. Okada, M. Beetz: Tracking-based interactive segmentation of textureless objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 1122–1129
41.96 M. Sutton, L. Stark, K. Bowyer: Function from visual analysis and physical interaction: A methodology for recognition of generic classes of objects, Image Vis. Comput. **16**(11), 745–763 (1998)
41.97 Y. Bekiroglu: Learning to Assess Grasp Stability from Vision, Touch and Proprioception Ph.D. Thesis (KTH Roy. Inst. Technol., Stockholm 2012)

Multimedia Contents

Haptics

42. Haptics

Blake Hannaford, Allison M. Okamura

The word *haptics*, believed to be derived from the Greek word *haptesthai*, means *related to the sense of touch*. In the psychology and neuroscience literature, haptics is the study of human touch sensing, specifically via *kinesthetic* (force/position) and *cutaneous* (tactile) receptors, associated with perception and manipulation. In the robotics and virtual reality literature, haptics is broadly defined as real and simulated touch interactions between robots, humans, and real, remote, or simulated environments, in various combinations. This chapter focuses on the use of specialized robotic devices and their corresponding control, known as *haptic interfaces*, that allow human operators to experience the sense of touch in remote (teleoperated) or simulated (virtual) environments.

42.1 **Overview** 1064
42.1.1 Human Haptics 1065
42.1.2 Application Examples 1066
42.2 **Haptic Device Design** 1068
42.2.1 Mechanisms 1068
42.2.2 Sensing 1069
42.2.3 Actuation and Transmission 1070
42.2.4 An Example Device 1070
42.3 **Haptic Rendering** 1071
42.3.1 Rendering Complex Environments 1072
42.3.2 Virtual Coupling 1073
42.4 **Control and Stability of Force Feedback Interfaces** 1073
42.5 **Other Types of Haptic Interfaces** 1075
42.5.1 Vibrotactile Feedback 1075
42.5.2 Contact Location, Slip, and Shear Display 1076
42.5.3 Slip and Shear 1076
42.5.4 Local Shape 1077
42.5.5 Surface Displays 1078
42.5.6 Temperature 1078
42.6 **Conclusions and Further Reading** 1079
References 1079

Haptic technology is intimately connected with robotics through its reliance on dexterous mechatronic devices and draws heavily on the theoretical foundations of manipulator design, actuation, sensing, and control. In this chapter, we begin with motivation for haptic interface design and use, including the basic design of a haptic interface, information about human haptics, and examples of haptic interface applications. Next, we review concepts in the mechatronic design of kinesthetic haptic interfaces, including sensors, actuators, and mechanisms. We then examine the control aspect of kinesthetic haptic interfaces, particularly the rendering of virtual environments and stable and accurate display of forces. We next review tactile displays, which vary widely in their design due to the many types of tactile information that can be presented to a human operator. Finally, we provide resources for further study of haptics.

42.1 Overview

Haptics is the science and technology of experiencing and creating touch sensations in human operators. Imagine trying to button a coat, shake someone's hand, or write a note without the sense of touch. These simple tasks become extremely difficult to perform without adequate haptic feedback. To improve human operator performance in simulated and teleoperated environments, haptic interfaces seek to generate a compelling sensation that the operator is directly touching a real environment.

Haptic interfaces attempt to replicate or enhance the touch experience of manipulating or perceiving a real environment through mechatronic devices and computer control. They consist of a haptic device (HD, a manipulandum with sensors and actuators) and a control computer with software relating human operator inputs to haptic information display. While the low-level design of haptic interfaces varies widely depending on the application, their operation generally follows the *haptic loop* shown in Fig. 42.1. First, the haptic device *senses* an operator input, which may be position (and its derivatives), force, muscle activity, etc. Second, the sensed input is applied to a virtual or teleoperated environment. For a virtual environment, the effect of the operator's input on virtual objects and the subsequent response to be displayed to the operator are computed based on models and a *haptic rendering* algorithm. In teleoperation, a manipulator that is remote in space, scale, or power attempts to track the operator's input. When the manipulator interacts with its real environment, haptic information to be relayed to the operators is recorded or estimated. Finally, *actuators* on the haptic device are used to physically convey touch sensations to the human operator. Based on the haptic feedback, whether through unconscious or conscious human control, or simply system dynamics, the operator input is modified. This begins another cycle of the haptic loop.

Despite the simplicity of the concept of haptic display, there exist many challenges to developing compelling haptic interfaces. Many of these are addressed through fundamental robotics theory and an understanding of human haptic capabilities. In general, haptic interface performance specifications are based on human sensing and motor control characteristics. One major challenge in artificially generating haptic sensations is that the human operator's motion should be unrestricted when there is no contact with a virtual or remote object. Haptic devices must allow the human operator to make desired motions, thus requiring back-drivability and sufficient degrees of freedom of motion. A variety of robotic designs are used in haptic devices, including exoskeletons, actuated grippers, parallel and serial manipulators, small-workspace *mouse*-like devices, and large-workspace devices that capture whole arm, and even whole body, movement. Another challenge is that humans integrate kinesthetic (force/position) and cutaneous (tactile) information with motion and control cues to form haptic perceptions. Haptic devices would ideally include both force and tactile displays, although this has been rarely done due to size and weight limitations of actuators. Because of the human's sensitivity to high-frequency information, for many haptic interfaces and applications, this loop must repeat at a high

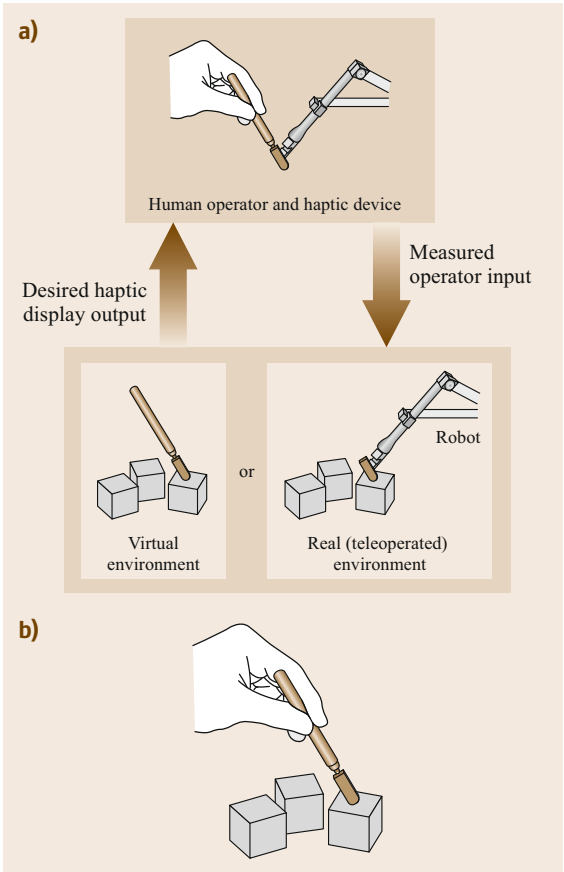


Fig. 42.1 (a) The *haptic loop* of a generic haptic interface. A haptic device senses human operator input, such as position or force, and the system applies this input to a virtual or teleoperated environment. The response of the environment to be relayed to the human operator is computed through models, haptic rendering, sensing, and/or estimation. Finally, actuators on the haptic device display corresponding touch sensations to the human operator. (b) The ideal result is that the human operator feels that he or she is interacting directly with a real environment

frequency, typically 1 kHz. Not only does a high update rate provide realistic (nondiscretized) touch sensations to the human operator, it also typically helps to maintain system stability. Controls analysis for haptic devices must consider both the continuous nature of the physical dynamics and the discrete nature of the computer control.

Before we examine the various components of haptic interfaces in detail, it is useful to motivate their design through a review of human haptics and applications of haptics. The remainder of this section is devoted to those topics.

42.1.1 Human Haptics

Two functions of the human nervous system play a primary role in haptics: kinesthesia (the internal sensing of forces and displacements inside muscles, tendons, and joints), and tactile sensing (the sensation of deformations of the skin).

Anatomy and Physiology

Haptics incorporates both, and is associated with an activity such as manipulation or exploration. Most of this chapter will address systems meant to interact primarily with the kinesthetic modality. Devices specifically aimed at tactile perception are described in Sect. 42.5. Even if tactile stimuli are not explicitly generated by a haptic device, tactile receptors are still stimulated, and are known to respond to frequencies as high as 10 000 Hz [42.1] and displacements as small as 2–4 μm [42.2–4].

Kinesthesia is mediated by muscle spindles, which transduce stretch of muscles, and Golgi tendon organs, which transduce joint rotation, especially at the extremes of motion. In principle, these and similar receptors could be stimulated directly to produce haptic sensations. For example, a vibration applied to a muscle tendon creates a strong sensation of muscle lengthening and corresponding joint motion in humans [42.5, 6]. Research in peripheral nerve stimulation for prosthesis control has demonstrated that electrodes implanted within individual fascicles of peripheral nerve stumps in amputees can be stimulated to produce sensations of touch or movement referred to the amputee's phantom hand [42.7].

Psychophysics

At the next level up from physiology and anatomy, psychophysics [42.8], the science of the physical capabilities of the senses, has been a rich source of design data for haptic device development. Its chief contribution has been methodologies that haptics researchers have applied to answer questions about what capabilities are needed in haptic devices. These sensory capabilities could then be translated into design requirements. Some of the chief psychophysical methods that have been fruitfully applied to haptics include threshold measurement by the method of limits and adaptive up-down methods. However, perception at threshold is not 100% reliable. Perception accuracy tends to depend on the strength of the stimulus, and the tradeoff between hit rate and false alarms depends strongly on the probability that a stimulus will be present in a given time interval (P_{stim}).

A more general notion is the receiver operating curve (Fig. 42.2, borrowed by psychophysics from radar theory), which plots the probability of a subject response given the existence of a stimulus, versus the probability of response given no stimulus. The curve is generated by measuring both probabilities at several different values of P_{stim}. The ideal response is the point (0,1): 100% response for stimuli and 0% response for nonstimuli. Human response is near this point for stimuli much above threshold, but declines to a rounded curve and eventually the 45° line as response below threshold becomes equal to chance.

Another relevant concept from psychophysics is the just noticeable difference (JND), commonly expressed as a percentage. This is the magnitude of a relative change in a stimulus, such as a force or displacement applied to the finger that is just perceivable by subjects.

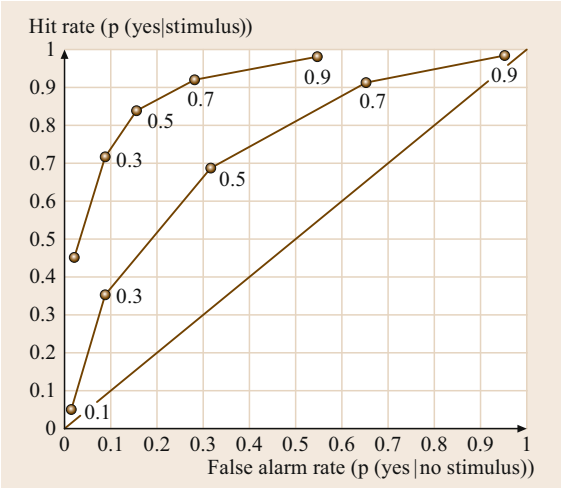


Fig. 42.2 Receiver operating curve (ROC). The ROC encodes the tradeoffs made by a subject between risk of false alarm and risk of missing a valid stimulus. Each point encodes a specific tradeoff between these two risks observed when stimuli are presented with a specified probability. ROCs for stronger signals tend towards the upper left hand corner (after [42.8] with permission from Lawrence Erlbaum and Associates, Mahwah)

For example, *Jones* [42.9] has measured JNDs of 6% for force applied to the human finger over a range of 0.5–200 N.

Psychology: Exploratory Procedures

In influential research starting in the 1980s, *Lederman* and *Klatzky* defined stereotyped hand motions called *exploratory procedures* (EPs), which are characteristic of human haptic exploration [42.10–12]. They placed objects into the hands of blindfolded subjects, and videotaped their hand motions. Their initial experiments [42.11] showed that the EPs used by subjects could be predicted based on the object property (texture, mass, temperature, etc.) that the subjects needed to discriminate. They also showed that the EPs chosen by subjects were the ones best able to discriminate that property. Furthermore, when asked to answer specific questions about objects (*Is this an eating utensil, further a fork?*), subjects used a two-stage sequence in which a more general lifting EP preceded more specific EPs [42.12].

Lederman and Klatzky's eight EPs (Fig. 42.3) and the property for which they are optimal are:

1. Lateral motion (texture)
2. Pressure (hardness)
3. Static contact (temperature)
4. Unsupported holding (weight)
5. Enclosure (global shape, volume)
6. Contour following (exact shape, volume)
7. Part motion test (part motion)
8. Function testing (specific function).

Each of these EPs is a bimanual task involving contact with all interior surfaces of the hand, motion of the wrist and various degrees of freedom of the hand, and tactile and temperature sensors in the skin (e.g., EPs 1 and 3), and kinesthetic sensors in the arm (EP 4). A haptic device capable of supporting all of these EPs is far beyond today's state of the art. However, the significance of these results for the design of haptic interface is great, since they allow us to derive device requirements from EPs.

42.1.2 Application Examples

The most common haptic device encountered by the general population is a vibration display device that provides haptic feedback while an operator plays a video game. For example, when the operator drives off the virtual road or bumps into a virtual wall, the hand controller shakes to imply driving over a rough surface or displays an impulse to represent the shock of hitting a hard surface. We examine two more pragmatic examples, medical simulators and computer-aided design (CAD) systems, in detail below. In addition, we review several commercially available haptic devices. Although haptic interfaces are not yet in widespread commercial use outside of entertainment, they are being integrated into numerous applications where the potential benefits are clear enough to justify the adoption of new technologies. A variety of novel and creative applications are being developed regularly in numerous fields, including:

- Assistive technology
- Automotive
- Design
- Education
- Entertainment
- Human–computer interaction
- Manufacturing/assembly
- Medical simulation
- Micro/nanotechnology
- Molecular biology
- Prosthetics
- Rehabilitation
- Scientific visualization
- Space
- Surgical robotics.

Medical Simulations

A major example driving much of today's haptic virtual environment research is simulation for training of hands-on medical procedures. Medical invasive thera-

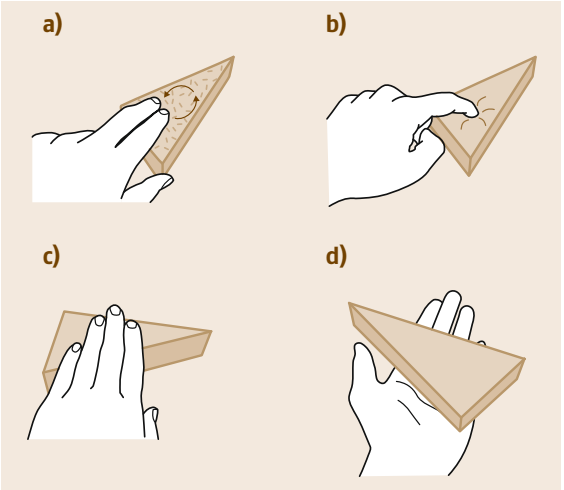


Fig.42.3a–d Four of the eight human exploratory procedures (EPs). **(a)** Lateral motion (texture); **(b)** pressure (hardness); **(c)** static contact (temperature); **(d)** unsupported holding (weight) (after *Lederman* and *Klatzky* [42.11])

peutic and diagnostic procedures, ranging from drawing blood samples to surgery, are potentially dangerous and painful for the patient and require the student to learn hands-on skills mediated by haptic information pathways [42.13]. Simulators both with and without haptic feedback aim to replace supervised learning directly on human patients or on animals. Simulators have proven highly effective in developing minimally invasive surgery skills [42.14], especially when haptic feedback is provided in early training [42.15]. Expected benefits of training with haptic simulators include:

- Reduced risk to patients both during and immediately after training
- Increased ability to simulate unusual conditions or medical emergencies.
- Ability to collect physical data during the training process and provide specific and directed feedback to the student.
- Increased training time per unit of instructor effort.

Approaches for simulator designs, specific medical applications, and training evaluation methods have also been widely studied in the last two decades, e.g., [42.16, 17]. However, the costs of this technology are still high. In addition, it is not always clear which improvements in simulator technology, such as haptic device performance or accuracy of soft-tissue modeling, lead to improved clinical performance and, ultimately, patient outcomes.

Computer-Aided Design

The Boeing Company [42.18] has studied the use of haptic interfaces for solving advanced problems in CAD. One such problem is verification of the ability to efficiently maintain a complex system such as an aircraft. In the past, mechanics could verify the procedures (such as change-out of parts) on a physical prototype. However, this analysis is difficult or impossible to perform visually on an advanced CAD system. The VoxMap Pointshell system (Fig. 42.4) was developed to allow test extraction of parts with a haptic interface. Force sensations from the haptic interface reproduce for the operator the physical constraints of the part bumping into elements of the complex workplace. If the operator can remove the part in the haptic interface, it is verified that this part can be maintained without undue disassembly of the aircraft. This capability has been proved useful in actual design activities.

Commercially Available Haptic Devices and Systems

There are a wide variety of haptic devices available from companies, although many researchers build their own haptic devices for special purposes. At the time of this writing, one of the most popular commercially available haptic devices is the Geomagic Touch [42.19], formerly known as the Phantom Omni from SensAble Technologies (Fig. 42.5). In the 1990s and 2000s, SensAble developed the Phantom line of stylus-type haptic devices. The Phantom Premium [42.20], a higher fidelity, larger workspace device, has also been used widely in haptics research. The high price of haptic devices (compared to visual displays) restricts the development of some commercial applications. The Geomagic Touch (Phantom Omni), which is an order of magnitude less expensive than the Phantom Premium, has gained popularity among haptics and robotics re-

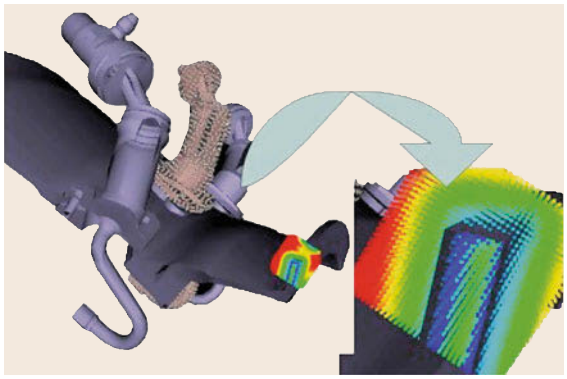


Fig. 42.4 Boeing computer-aided design (CAD) application for assembly and maintenance verification of complex aircraft systems. Boeing researchers developed the Voxmap/Pointshell software for haptically rendering very complex models in six degrees of freedom at high rates (courtesy of Bill McNeely, Boeing Phantom Works)

Fig. 42.5 The Phantom Omni device from SensAble Technologies, now marketed as the Geomagic Touch. This relatively low-cost device senses motion in six degrees of freedom from the stylus and can apply forces in the x, y, and z directions to the stylus tip (courtesy SensAble Technologies, Inc., Woburn)

searchers. In 2007, Novint Technologies [42.21] released the Novint Falcon, an inexpensive 3-DOF (three-degree-of-freedom) haptic device that is in turn an order of magnitude less expensive than the Phantom Omni.

Immersion has aimed at the mass market and consumer segments with a wide variety of haptics-based products, many of them involving a single degree of freedom. For example, they have licensed technology to makers of various video games, as well as mobile phone manufacturers, in the form of vibratory feedback in handheld devices and haptic-enabled steering wheels for driving games. Immersion also has a medical division selling medical simulators with haptic feedback.

Software for haptic rendering has also become widely available, through both commercial sources and research groups. Most companies that sell haptic devices also provide a standard development kit (SDK) with haptic rendering capability. In addition, not-for-profit open-source projects such as Chai3D [42.22] aim to make rendering algorithms from different groups publicly available, shortening application development time and allowing direct comparison of algorithms for benchmarking purposes.

42.2 Haptic Device Design

There are two broad classes of haptic devices: *admittance* and *impedance* devices. Admittance devices sense the force applied by the operator and constrain the operator's position to match the appropriate deflection of a simulated object or surface in a virtual world. In contrast, an impedance haptic device senses the position of the operator, and then applies a force vector to the operator according to computed behavior of the simulated object or surface.

Robots of the impedance type are back-drivable, have low friction and inertia, and have force-source actuators. A commonly used impedance haptic device in robotics-related research is the Phantom Premium [42.20, 23]. Robots of the admittance type, such as typical industrial robots, are non-back-drivable and have velocity-source actuators. The velocity is controlled with a high-bandwidth low-level controller, and is assumed to be independent of applied external forces. Some commercially available haptic devices, such as the HapticMaster [42.24], do operate under admittance control. While such closed-loop force control has been used for haptic display, more commonly designers have opted for mechanisms specially designed for open-loop force control to achieve simultaneously low cost and high bandwidth.

The choice of admittance or impedance architecture has many profound implications in the design of the software and hardware system. For a variety of reasons, including cost, the majority of haptic devices implemented today are of the impedance type. Because the preponderance of systems today are impedance devices and limitations on space, we limit our subsequent discussion to that class.

42.2.1 Mechanisms

Creating high-fidelity haptic sensations in the operator requires attention to mechanism design (Chap. 5). The requirements for impedance haptic devices are similar to those for designing manipulators suitable for force control. Desirable mechanism attributes for open-loop force control include low inertia, high stiffness, and good kinematic conditioning throughout a workspace designed to effectively match the appropriate human limb, primarily the finger or arm. The weight of the mechanism should be minimized, as it is perceived by the operator as weight and inertia of the virtual or teleoperated environment. Kinematic singularities (Chaps. 2, 5, and 18) are detrimental to haptic interfaces because they create directions in space in which the end-point cannot be moved by the human operator and thus impose disturbances on the illusion of haptic contact with virtual objects. High transmission ratios must be avoided as they introduce significant amounts of friction. This constraint requires haptic interfaces to make high demands on actuator performance.

Measures of Mechanism Performance

The ideal haptic device can move freely in any direction and is free of singular configurations as well as the *bad* effects of operating in their neighborhood. Traditionally, kinematic performance has been derived from the mechanism's Jacobian matrix, $\mathbf{J}(p, q)$, using some of the following well-known measures:

- Manipulability [42.25]: the product of the singular values of $\mathbf{J}(p, q)$
- Mechanism isotropy [42.26]: the ratio of smallest to the largest singular value of $\mathbf{J}(p, q)$
- Minimum force output [42.25, 27, 28]: maximizing the force output in the *worst* direction.

Dynamics can also be introduced into the cost function using measures such as dynamic manipulability [42.29]. This is still an active area of research and

there is no consensus yet on which dexterity measure is most appropriate for haptic devices.

Kinematic and Dynamic Optimization

This aspect of design requires synthesis of mechanisms that match the workspace of the (most often) human finger or arm while simultaneously avoiding kinematic singularities.

A haptic device workspace is defined to match that of the targeted human limb. This can be assisted by the use of anthropometric data [42.30]. The performance goals, such as low inertia and avoidance of kinematic singularities, must be formalized into a quantitative performance measure which can be computed for any candidate design. Such a measure must account for:

- Uniformity of kinematic conditioning throughout the target workspace
- Favoring designs with lower inertia
- Guaranteeing that the target workspace is reachable.

The measures of mechanism performance defined above operate at a single point in space and thus must be integrated over the entire workspace to derive a figure of merit for a proposed haptic device design. For example, if S is the set of all joint angles Θ, such that the end effector is inside the target workspace, one such measure is

$$M = \min_S W(\Theta) \,, \tag{42.1}$$

where $W(\Theta)$ is a measure of design performance. The performance measure should include a link length penalty such as

$$M = \min_S \frac{W(\Theta)}{l^3} \,, \tag{42.2}$$

in order to avoid solutions which have excessive size, compliance, and mass of long links. We could search a large family of mechanism designs to maximize M. For example, if a design has five free parameters (typically link lengths and offsets), and we study ten possible values for each parameter, 10^5 designs must be evaluated.

Available computing power has grown much faster than the complexity of realizable mechanisms on the human scale (as measured by their DOF). Thus, brute-force search of design spaces is often sufficient, and sophisticated optimization techniques are not necessary.

Grounded Versus Ungrounded Devices

Most current devices that provide kinesthetic feedback are physically grounded, that is, forces felt by the operator are with respect to the operator's *ground*, such as the floor or desktop. Ungrounded haptic feedback devices are more mobile and can operate over larger workspaces compared to grounded devices, which enables them to be used in large-scale virtual environments. A number of ungrounded kinesthetic feedback devices have been developed, for example [42.31–33]. Comparisons have been made between the performance of ungrounded and grounded haptic displays [42.34]. Some ungrounded devices provide tactile rather than kinesthetic sensations, and these are described in Sect. 42.5.

42.2.2 Sensing

Haptic devices require sensors to measure the state of the device. This state may be modified by the operator's applied position/force, the haptic control law, and/or device and environment dynamics. The operator's input is sensed in the form of an applied position or an applied force. Sensing requirements for haptics are similar to those of other robotic devices (Chap. 29) so only haptics-specific sensing issues are discussed here.

Encoders

Rotary optical quadrature encoders are typically used as position sensors on the joints of haptic devices. They are often integrated with rotary motors, which serve as actuators. The underlying sensing mechanism for encoders is described in Sect. 29.1. The required resolution of an encoder for a haptic device depends on the ratio between the angular distance of a single encoder tick to the end-point motion in Cartesian space. The resolution of the selected position encoder has effects beyond simple spatial resolution of the end-point, including the maximum stiffness that can be rendered (Sect. 42.4) without unstable or nonpassive behavior [42.35].

Many haptic applications, such as the rendering of virtual environments with damping (in which force is proportional to velocity), require velocity measurement. Velocity is typically obtained by numerical differentiation of the position signal obtained by an encoder. An algorithm for velocity estimation must be selected which is free of noise but minimizes phase lag at the frequencies of interest [42.36]. Thus, an alternative method is to use specialized hardware that measures the time between encoder ticks in order to compute the velocity [42.37].

Force Sensors

Force sensors are used in haptic devices as the operator input to an admittance-controlled device, or as a mechanism for canceling device friction and other undesirable dynamic properties in an impedance-controlled device.

When a force sensor such as a strain gauge or load cell measures the operator's applied force, care must be taken to thermally isolate the sensor, since thermal gradients in the sensor caused by body heat can affect force readings.

42.2.3 Actuation and Transmission

Haptic devices are differentiated from traditional computer input devices by actuators that are controlled to provide appropriate haptic sensations to the human operator. The performance of the haptic device depends heavily on the actuator properties and the mechanical transmission between the actuator and the haptic interaction point (HIP).

Requirements for Haptics

The primary requirements for actuators and mechanical transmission in impedance-type haptic devices are: low inertia, low friction, low torque ripple, backdriveability, and low backlash. In addition, if the design is such that the actuator itself moves as the user's position changes, a higher power-to-weight ratio is desired. Although closed-loop force control has been used for haptic display in impedance devices, most often the mechanism is designed to have sufficiently low friction and inertia so that open-loop force control is accurate enough.

One common mechanical transmission for haptic devices is the capstan drive (Fig. 42.6), which consists of smooth cables wrapped around pulleys of differing diameter to provide a gear ratio. A no-slip, high-friction contact between the cable and the pulleys is maintained through several wraps of the cable. The capstan drive minimizes friction forces felt by the human operator because it prevents translational forces on motor and joint axes.

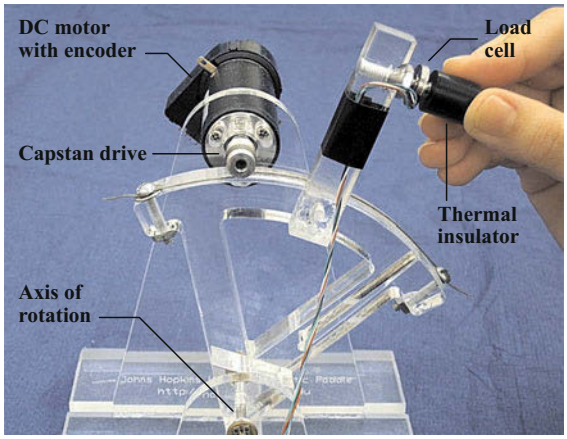


Fig. 42.6 This version of the Haptic Paddle [42.38] includes an encoder for position sensing, a single-axis load cell for force sensing, and a brushed motor with capstan transmission for actuation

Current amplifiers are typically used to create a direct relationship between the voltage output by the computer via a digital-to-analog (D/A) converter and the torque output by the motor. The effect of actuator and amplifier dynamics and D/A resolution on system stability is typically negligible in comparison to position sensor resolution and sampling rate for most haptic devices. Actuator or amplifier saturation can produce undesirable behavior, particularly in multi-DOF haptic devices where a single saturated motor torque may change the apparent geometry of virtual objects. The force vector, and thus the corresponding actuator torques, must be scaled appropriately if any actuator is saturated.

42.2.4 An Example Device

As an illustrative example, we will provide detailed design information for a simple 1-DOF haptic device known as the Haptic Paddle [42.38]. This section is meant to provide a concrete description of the types of components that are used in kinesthetic haptic devices, and the device can also be constructed following the instructions provided by Johns Hopkins University. Many widely available haptic devices share the common working principles of this device and differ chiefly in kinematic details arising from a greater number of DOF.

The haptic paddle shown in Fig. 42.6 is equipped with two sensors: a position a position encoder and a force sensor. A 500-counts-per-turn Hewlett-Packard HEDS 5540 encoder that is mounted directly on the motor. The quadrature process yields 2000 counts per revolution; the capstan transmission gear ratio and lever arm result in a position resolution of 2.24×10^{-5} m at the haptic interaction point (HIP). An optional load cell is used to measure the applied operator force. A plastic cap thermally insulates the load cell. In this device, the load cell can be used to minimize the effect of friction through a control law that attempts to zero the applied operator force when the HIP is not in contact with a virtual object.

The Haptic Paddle shown uses a brushed DC (direct current) motor with an aluminum pulley attached to the shaft. Like many commercial haptic devices, it uses a capstan drive: a cable is wrapped several times around the motor pulley and attached at each end of the large partial pulley. In one instantiation, the output of the digital-to-analog (D/A) converter from the microprocessor is passed through a current amplifier that gives a current through the motor that is proportional

to the D/A voltage. This gives direct control of applied torque on the motor. The resulting system has a force felt at the driving point that is proportional to the output voltage during static operation. When the system is moving, the force applied to the operator may differ due to human and device dynamics.

42.3 Haptic Rendering

Haptic rendering (in impedance systems) is the process of computing the force required by contacts with virtual objects based on measurements of the operator's motion. This section describes haptic rendering for virtual environments. Haptic feedback for teleoperators is described in Chap. 43.

An important property of haptic systems is that their timing constraints are quite severe. To illustrate this point, tap a pencil on the table top. You hear a sound which is an audio representation of the contact dynamics between pencil tip and table top. Tactile receptors in the human fingers are known to respond up to 10 kHz [42.1]. To realistically render this type of contact between hard surfaces would require response well into the audio range of frequencies (up to 20 kHz) and thus sampling times of around 25 μs. Even with specialized designs, haptic devices do not have this bandwidth, and such high fidelity is not usually a goal of haptic rendering. Achieving stability in any type of hard contact requires very high sampling rates. In practice, most haptic simulation systems are implemented with at least 1000 Hz sampling rate. This can be reduced to the low hundreds of Hertz if the virtual environment is limited to soft materials.

Basic Haptic Rendering

The computational process of haptic rendering can be formulated into the following seven sequential steps for each cycle (Fig. 42.7). The rendering cycle must typically be completed in under 1 ms for stability and realism:

1. Sensing (Sect. 42.2.2)
2. Kinematics
3. Collision detection
4. Determining surface point
5. Force calculation
6. Kinematics
7. Actuation (Sect. 42.2.3).

Kinematics

The position and velocity measurements acquired by sensors are typically in joint space. These must be converted through the forward kinematics model and the Jacobian matrix (Chap. 2) to the Cartesian position and velocity of the operator's hand or fingertip. In some applications the operator is virtually holding a tool or object whose shape is represented in the virtual environment but whose position and orientation are determined by the operator. In whose others, the operator's fingertip or hand is represented by a point which makes only point contact with objects in the virtual environment (VE). We refer to a virtual handheld object as a virtual tool and, following [42.39], we refer to the single end-point as the haptic interaction point (HIP).

Collision Detection

For the point contact case, the collision detection software must determine if the position of the HIP at the current instant of time represents contact with a virtual object. In practice this usually means to determine if the HIP is penetrating or inside the object surface. The object surface is represented by a geometric model such as polygons or splines.

Although there is an extensive literature on collision detection in computer graphics, there are unique aspects of the collision detection problem for haptics. In particular, speed of computation is paramount and worst-case speed, as opposed to average speed, is what counts. Solutions that evaluate in constant time are pre-

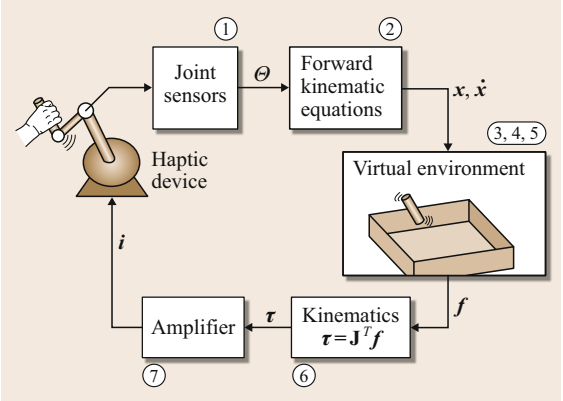


Fig. 42.7 Schematic diagram of the haptic rendering cycle for an impedance haptic display system. Virtual object moves in the virtual environment according to operator's displacement of the haptic device. Joint displacements (Θ) sensed in the device (1) are processed through kinematics (2), collision detection (3), surface point determination (4), force calculation (5), kinematics (6), and actuation (7)

ferred. Section 42.3.1 addresses collision detection and haptic rendering for complex environments.

If the HIP is found to be outside all objects, then a force of zero is returned.

Determining Surface Point

Once it is determined that the HIP is inside a surface, the force to be displayed to the operator must be computed. Many researchers have used the idea of a virtual spring connecting the HIP to the nearest point on the surface as a model of interpenetration and force generation [42.39–41]. Basdogan and Srinivasan named this point the intermediate haptic interaction point (IHIP). However all of these authors realized that the closest surface point is not always the most faithful model of contact. For example, as the HIP moves laterally below the top surface of a cube (Fig. 42.8), eventually it becomes close enough to the edge that the closest surface point becomes the side of the cube. In this situation, the algorithm needs *memory* to keep the IHIP on the top surface and generate an upward force at all times or the operator is suddenly ejected out the side of the cube.

Force Calculation

Force is commonly computed by using the spring model (Hooke's Law)

$$\boldsymbol{f} = k\boldsymbol{x}\,, \tag{42.3}$$

where $\boldsymbol{x}$ is the vector from the HIP to the IHIP, and $k > 0$. When k is sufficiently large, the object surface will feel like a wall perpendicular to $\boldsymbol{x}$. This *virtual wall*, or impedance surface, is a fundamental building block of most haptic virtual environments. Because the virtual wall is only displayed if a collision between the HIP and virtual object is detected, the wall is a unilateral constraint, governed by a nonlinear switching condition. As will be described in the following section, haptic virtual environments with complex geometries are often formed using a polygonal mesh in which each polygon is essentially a virtual wall. A virtual surface may also be allowed to deform globally, while the local interaction with the operator is governed by a virtual wall. *Virtual fixtures*, which are often constructed from virtual walls, can be overlaid on haptic feedback teleoperators to assist the operator during a teleoperated task (Chap. 43).

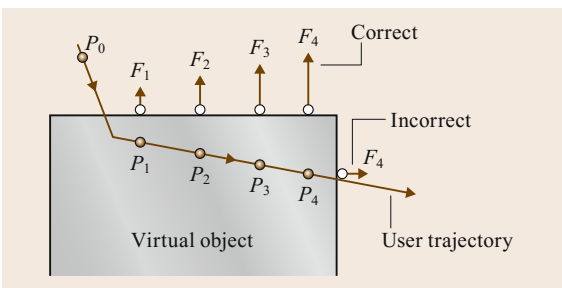


Fig. 42.8 Illustration of subtle aspects of haptic rendering of contact force. The operator fingertip trajectory enters object surface moving down and to right. Haptic interaction points (HIP) are shown at times 1–4 (*solid circles*, P_0–P_4). Intermediate haptic interaction points (IHIP) are shown when the HIP is inside the object (*open circles*). At position P_4, the algorithm must not render force based on the closest surface point or the operator will be ejected from the side of the object (feeling unnatural force tangential to the top surface)

The pure stiffness model described above can be augmented to provide other effects, particularly through the use of the *virtual coupling* described in Sect. 42.3.2. Damping can be added perpendicular or parallel to the surface. In addition, Coulomb or other nonlinear friction may be displayed parallel to the surface. To provide a more realistic display of hard surfaces, vibrations can also be displayed open loop at the moment of collision between the HIP and the surface, as is described in Sect. 42.5.

Kinematics

The computed force in Cartesian space must then be transformed into torques in the actuator space. Typically the calculation is

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\boldsymbol{f}\,, \tag{42.4}$$

where $\boldsymbol{\tau}$ is the torque command to the actuators, $\boldsymbol{f}$ is the desired force vector, and $\mathbf{J}^{\mathrm{T}}$ is the transpose of the haptic device Jacobian matrix (Chap. 2). If the haptic device has no dynamics and the actuators are perfect, the exact desired force is displayed to the operator. However, real device dynamics, time delays, and other nonidealities result in applied operator forces that differ from the desired forces.

42.3.1 Rendering Complex Environments

Today's computer power is sufficient that a variety of relatively simple algorithms can effectively render haptics for simple virtual environments, say consisting of a handful of simple geometric primitives such as spheres, cubes, and planes. However, the challenge is to scale these algorithms to complex environments such as those we are used to seeing in computer graphic renderings consisting of 10^5–10^7 polygons. A variety of approaches have been tried in the literature for efficient rendering of complex scenes. *Zilles* and *Salisbury* [42.40] found planar constraints due to

the nearby surface polygons and solved for the closest IHIP point using Lagrange multipliers. *Ruspini* and *Khatib* [42.42] added force shading and friction models. *Ho* et al. [42.39] used a hierarchy of bounding spheres to determine the initial contact (collision) point, but thereafter searched neighboring surfaces, edges, and vertices (referred to as geometric primitives) of the current contacted triangle to find the closest point for the IHIP. *Gregory* et al. [42.43] imposed a discretization of three-dimensional (3-D) space onto the hierarchy to speed up the detection of initial contact with the surface. *Johnson* et al. [42.44] performed haptic rendering on moving models based on local extrema in distance between the model controlled by the haptic device and the rest of the scene. *Lin* and *Otaduy* [42.45, 46] used level-of-detail representations of the objects for performing multiresolution collision detection, with the goal of satisfying real-time constraints while maximizing the accuracy of the computed proximity information.

Alternative algorithms and efficiencies can be derived by not representing the object surface as polygons. *Thompson* and *Cohen* [42.47] derived the mathematics for computing the interpenetration depth for surfaces directly from non-uniform rational B-spline (NURBS) models. *McNeely* et al. [42.18] took the extreme approach of voxelizing space at the millimeter scale. Each voxel contained a precomputed normal vector, stiffness properties, etc. 1000 Hz rendering was achieved with very complex CAD models containing millions of polygons (in the graphical equivalent representation), but large amounts of memory and precomputation are required. The performance of this algorithm is sufficiently high that it can be used to render hundreds of contact points simultaneously. This allows the operator to hold an arbitrarily shaped tool or object. The tool/object is populated by points surrounding its surface and the resultant force and moment on the tool is the sum of interaction forces computed on all of the surface points.

For surgical simulation, researchers have focused on the modeling and haptic rendering of the interaction between surgical instruments and organs. Researchers have attempted to model virtual tissue behavior in a wide variety of ways, which can be broadly classified as:

1. Linear elasticity based
2. Nonlinear (hyperelastic) elasticity-based finite-element (FE) methods
3. Other techniques that are not based on FE methods or continuum mechanics.

While most conventional linear and nonlinear FE algorithms cannot be run in real time, methods such as preprocessing can allow them to run at haptic rates [42.48]. Many researchers rely on data acquired from real tissues to model organ deformation and fracture accurately. Major challenges in this field include the modeling of connective tissue supporting the organ, friction between instruments and tissues, and topological changes occurring during invasive surgical procedures.

42.3.2 Virtual Coupling

So far we have rendered forces by computing the length and direction of a virtual spring and applying Hooke's Law (42.3). This spring is a special case of a *virtual coupling* [42.41] between the HIP and the IHIP. The virtual coupling is an abstraction of the interpenetration model of force rendering. Instead of viewing the objects as compliant, we assume them to be rigid but connect them to the operator through a virtual spring. This imposes an effective maximum stiffness (that of the virtual coupling).

Problems with stable contact rendering (Sect. 42.4) often require more sophisticated virtual couplings than a simple spring. For example, damping can be added, generalizing the force rendering model of (42.3) to

$$\boldsymbol{f} = k\boldsymbol{x} + b\dot{\boldsymbol{x}} \,. \tag{42.5}$$

The parameters k and b can be empirically tuned for stable and high-performance operation. More-formal design methods for the virtual coupling are covered in Sect. 42.4.

42.4 Control and Stability of Force Feedback Interfaces

Introduction to the Problem

The haptic rendering system depicted in Fig. 42.7 is a closed-loop dynamical system. It is a challenge to render realistic contact forces yet retain the stable behavior of human-environment contact in the natural world. Instability in haptic interfaces manifests itself as buzzing, bouncing, or even wildly divergent behavior. The worst case for impedance devices is during attempted contact with stiff objects. Empirically, instability is frequently encountered when developing haptic interfaces with stiff virtual objects, but this instability can be eliminated by reducing the stiffness of the virtual object or by the operator making a firmer grasp on the haptic device.

Problem Description in Classical Control Terms

Although linear theory is of very limited use in haptic control applications, it can be applied to a basic analysis of the factors affecting instability [42.49]. Such a highly simplified model of an impedance device is shown in Fig. 42.9. $G_1(s)$ and $G_2(s)$ represent dynamics of the haptic device for both operator position sensing and force display respectively. Assume that the virtual environment and human operator/user (HO) can each be represented by a linear impedance such as

$$Z_{\mathrm{VE}} = \frac{F_{\mathrm{VE}}(s)}{X_{\mathrm{VE}}(s)} , \tag{42.6}$$

$$Z_{\mathrm{HO}} = \frac{F_{\mathrm{HO}}(s)}{X_{\mathrm{HO}}(s)} . \tag{42.7}$$

Then the loop gain of the closed-loop system from the human operator and back again is

$$G_l(s) = G_1(s)G_2(s)\frac{Z_{\mathrm{VE}}(s)}{Z_{\mathrm{HO}}(s)} . \tag{42.8}$$

Stability in the classical sense is assessed by applying the magnitude and phase criteria of Nyquist to $G_l(s)$. Increasing Z_{VE} (corresponding to stiffer or heavier virtual objects) increases the magnitude of $G_l(s)$ and thus destabilizes the system while a firmer grasp by the human operator, which increases the magnitude of Z_{HO}, has a stabilizing effect. Similar arguments apply to phase shifts that might be present in any part of the system.

Limitations of Linear Theory

Although the model of Fig. 42.9 illustrates some qualitative features of haptic interface stability, linear continuous time theory is of little use in designing methods to stabilize the loop. Interesting virtual environments are nonlinear. Furthermore, they can rarely be linearized because applications often simulate discontinuous contact, for example, between a stylus in free space and a hard surface. A second feature is digital implementation, which introduces sampling and quantization – both of which have significant effects.

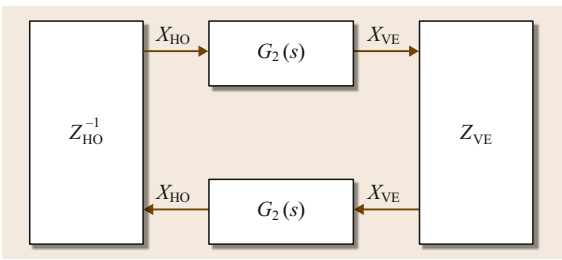


Fig. 42.9 Highly simplified linear model of haptic rendering to highlight some stability issues

Sampling

Colgate et al. [42.41] incorporated consideration of discrete time sampling behavior in the stability analysis. They considered the problem of implementing a virtual wall of stiffness

$$H(z) = K + B\frac{z-1}{Tz} , \tag{42.9}$$

where K is the virtual wall stiffness, B is the virtual wall damping coefficient, z is the z-transform variable, and T is the sampling time. They further modeled the haptic device (HD) in continuous time as

$$Z_{\mathrm{HD}}(s) = \frac{1}{ms+b} , \tag{42.10}$$

where m and b are the mass and damping of the haptic device, respectively. They derived the following condition for passivity of the device

$$b > \frac{KT}{2} + |B| , \tag{42.11}$$

showing a significant stabilizing effect of high sampling rates and also of high mechanical damping in the haptic device.

Quantization

Additional factors include delays due to numerical integration schemes and quantization. These contributing factors to instability have been termed *energy leaks* by *Gillespie* and *Cutkosky* [42.50].

Passivity

Interesting virtual environments are always nonlinear and the dynamic properties of a human operator are important. These factors make it difficult to analyze haptic systems in terms of known parameters and linear control theory. One fruitful approach is to use the idea of passivity to guarantee stable operation. Passivity is a sufficient condition for stability, and is reviewed more completely in Chap. 43 on telerobotics. There are many similarities between the control of haptic interfaces and bilateral teleoperation.

The major problem with using passivity for design of haptic interaction systems is that it is overly conservative, as shown in [42.35]. In many cases, performance can be poor if a fixed damping value is used to guarantee passivity under all operating conditions. *Adams* and *Hannaford* derived a method of virtual coupling design from two-port network theory which applied to all causality combinations and was less conservative than passivity based design [42.51]. They were able to derive optimal virtual coupling parameters using a dynamic model of the haptic device and by satisfying Lewellyn's *absolute stability criterion*, an inequality

composed of terms in the two-port description of the combined haptic interface and virtual coupling system. *Miller* et al. derived another design procedure which extended the analysis to nonlinear environments and extracted a damping parameter to guarantee stable operation [42.52–54].

42.5 Other Types of Haptic Interfaces

While kinesthetic (force feedback) haptic interfaces have the closest relationship to robotics, there are a variety of other types of haptic interfaces which are usually classified as tactile displays. Tactile displays are used to convey force, contact and shape information to the skin. They purposely stimulate cutaneous receptors, with little effect on kinesthetic sensation. This is in contrast to kinesthetic displays, which must inherently provide some cutaneous sensations through physical contact with a tool or thimble, but whose primary output is force or displacement to the limbs and joints. Tactile displays are usually developed for a specific purpose, such as display of contact events, contact location, slip/shear, texture, and local shape. Special purpose tactile displays can be targeted at different types of cutaneous receptors, each with its own frequency response, receptive field, spatial distribution, and sensed parameter (e.g., local skin curvature, skin stretch, and vibration) and each receptor type is associated with different exploratory procedures described in Sect. 42.1.1.

In contrast, tactile displays that render contact information for virtual reality or teleoperation have proven far more challenging. Accurate recreation of the local shape and pressure distribution at each fingertip requires a dense array of actuators. Devices specifically aimed at tactile perception are an active area of research but most have not reached the stage of applications or commercial distribution, with the notable exception of Braille displays for the blind. In this section, we describe the various types of tactile displays, their design considerations and specialized rendering algorithms, and applications.

42.5.1 Vibrotactile Feedback

Vibrotactile feedback is a popular method of providing tactile feedback. It can be used as a stand-alone method for haptic feedback or as an addition to a kinesthetic display. Vibrating elements, such as piezoelectric materials and small voice-coil motors, are lighter than the actuators used in kinesthetic devices, and can often be added to kinesthetic devices with little impact on existing mechanisms. In addition, high-bandwidth kinesthetic displays can be programmed to display open-loop vibrations through their normal actuators. The sensitivity for human vibration sensing ranges from DC to over 1 kHz, with peak sensitivity around 250 Hz.

We first consider the use of vibrations to convey impact or contact events – a technique that straddles kinesthetic and tactile feedback. When humans touch an environment, fast-acting sensors embedded in the skin record the minute vibrations occurring from this interaction. As described in Sect. 42.3, conventional approaches to haptic display usually consist of designing a virtual model with simple geometry, then using a first-order stiffness control law to emulate a surface. However, such first-order models often lack the realism of higher-order effects such as impact. With common haptic rendering algorithms, surfaces feel squishy or unrealistically smooth. One solution to improving the realism of such environments is to add higher-order effects such as textures and contact vibrations. These effects can use a library of surface models based on ad hoc analytical descriptions [42.55], which are sometimes tuned using qualitative operator feedback, physical measurements (reality-based models created from empirical data) [42.56, 57], or a combination of the two [42.58]. At the instant collision is detected between the HIP and a surface of the virtual object, the appropriate waveform is called out of the library, scaled according to the context of the motion (such as velocity or acceleration), and played open loop through an actuator. That actuator may be the same one simultaneously displaying lower-frequency force information, as shown in Fig. 42.10, or it might be a separate transducer. *Kuchenbecker* et al. [42.59] considered the dynamics of the haptic device to display the most accurate vibration waveforms possible, and compared a number of different vibration waveform generation techniques meant to convey impact superimposed on force feedback in virtual environments. Most of the vibration feedback methods performed similarly in terms of realism, and they were also significantly more realistic than conventional force feedback alone.

Vibration feedback can also be used to provide information about patterned textures, roughness, and other phenomena that have clear vibratory signals. This type of vibration feedback is often termed *vibrotactile* feedback. In teleoperated environments, *Kontarinis* and *Howe* [42.60] showed that damaged ball bear-

ings could be identified through vibration feedback, and *Dennerlein* et al. [42.61] demonstrated that vibration feedback improved performance over no haptic feedback, for a telemanipulation task inspired by undersea field robotics. In these teleoperator systems, vibration-sensitive sensors such as accelerometers and piezoelectric sensors are used to pick up and, in most cases, directly provide the vibration signals as inputs to a vibrotactile actuator. In virtual environments, *Okamura* et al. [42.62] displayed vibrations modeled based on textures and puncture of a membrane. Similar to the event-based haptics above, the vibration waveforms were modeled based on earlier experiments and played open loop during interaction with the virtual environment.

Finally, vibration feedback has been used as a method of sensory substitution, to convey direction, attention, or other information, e.g., [42.63–65]. In this case, the strength and clarity of the signal, not realism, is the goal. Vibration frequencies near the peak human sensitivity are most effective, and there exist commercially available vibrotactile actuators (or *tactors*) suitable for such applications, e.g., Engineering Acoustic, Inc.'s C2 Tactor [42.66]. Elements within an array of tactors can be selectively turned on and off to evoke the sensory saltation phenomenon, in which a pattern of brief pulses on a series of tactors is perceived not as successive taps at different locations, but as a single tap that is traveling or hopping over the skin.

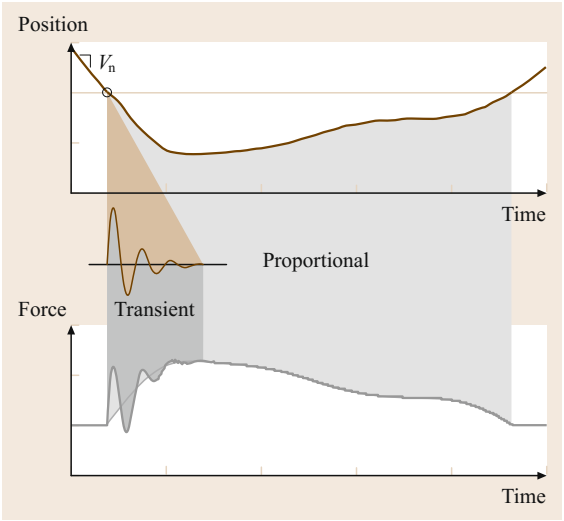


Fig. 42.10 Event-triggered open-loop force signals superimpose on traditional penetration-based feedback forces, providing vibration feedback that improves the realism of hard surfaces in virtual environments (after [42.59] with permission)

42.5.2 Contact Location, Slip, and Shear Display

In early work related to robotic dexterous manipulation, it was found that knowledge of the contact point between a robot hand and a grasped object is essential for manipulation. Without this knowledge, the robot will easily drop the object due to rapid accumulation of grasp errors. While many robotics researchers and some companies have developed tactile array sensors capable of measuring contact location, pressure distribution, and local object geometry (Chap. 28), practical methods for display of this information to the human operator of a virtual or teleoperated environment have proven much more difficult. We begin our discussion of contact display by considering contact location, slip, and shear display, which have the common goal of displaying the motion of a single area of contact relative to the skin (almost invariably on a finger). Arrays of pins that rise and fall to create a pressure distribution on the skin have been the most popular method to date for displaying contact information, but we will address those designs in the following section on local shape, since their primary advantage is the display of spatially distributed information. Instead, we will focus here on tactile devices that are designed to specifically address the problem of contact location and motion.

As an example of contact location display, *Provancher* et al. [42.67] developed a system that renders the location of the contact centroid moving on the user's fingertip. The tactile element is a free-rolling cylinder that is normally suspended away from the fingertip, but comes into contact with the skin when the operator pushes on a virtual object. The motion of the cylinder over the skin is controlled by sheathed push-pull wires. This allows the actuators to be placed remotely, creating a lightweight, thimble-sized package that can be unobtrusively mounted on a kinesthetic haptic device. An experiment demonstrated that human operators performed similarly during real manipulation and virtual manipulation (using the tactile display) in an object curvature discrimination task. In addition, operators were able to use the device to distinguish between manipulations of rolling and anchored but rotating virtual objects.

42.5.3 Slip and Shear

Humans use slip and incipient slip widely during manipulation tasks [42.68]. To reproduce these sensations for experiments to characterize human slip sensation, researchers have created stand-alone 1-DOF slip displays [42.69–71]. *Webster* et al. [42.71] created a 2-DOF tactile slip display, which uses an actuated ro-

tating ball positioned under the user's fingertip. The lightweight, modular tactile display can be attached to a multi-DOF kinesthetic interface and used to display virtual environments with slip. Experimental results demonstrate that operators complete a virtual manipulation task with lower applied forces using combined slip and force feedback in comparison with conventional force feedback alone. Skin stretch can also be integrated with a slip display to provide information about pre-slip conditions. For example, *Tsagarakis* et al. [42.72] developed a lightweight device that uses a V configuration of miniature motors to provide sensations of relative lateral motion (direction and velocity) onto the operator's fingertips. Generation of two-dimensional (2-D) slip/stretch is achieved by coordinating the rotational speed and direction of the two motors.

In terms of tactile device kinematics, slip and shear displays can be quite similar. However, the goal of shear, or skin stretch, displays is to maintain a no-slip condition such that shear forces/motions can be accurately controlled with respect to the human operator, typically on the finger. The use of shear (tangential skin stretch) is motivated by perceptual experiments demonstrating that the human fingerpad is more sensitive to tangential displacement compared to normal displacement [42.73]. A variety of tangential skin stretch devices have been designed, with applications in wearable haptics, high-fidelity tactile rendering, and teleoperation. *Hayward* and *Cruz-Hernandez* [42.74] developed a tactile device consisting of closely packed piezoelectric actuators, which generates a programmable stress field within the fingerpad. Single skin-stretch *tactors* have also been used to convey two-dimensional directional information (for navigation) [42.75] and to enhance perception of virtual environments (e.g., friction) [42.76] alone or in combination with kinesthetic haptic interfaces. A similar approach has been used for 3-DOF skin stretch, in combination with cutaneous normal force [42.77, 78]. In a different form factor, researchers have developed rotational skin stretch devices that can act as a substitute for natural proprioceptive feedback [42.79, 80].

42.5.4 Local Shape

Most tactile devices for local shape display consist of an array of individual pin elements that move normal to the surface. Often, a layer of elastic material is used to cover the pins so that the operator contacts a smooth surface rather than the pins directly. Other systems use individual elements that move laterally, and some substitute electrodes for moving parts to form an array of electrocutaneous elements. A number of researchers have used psychophysical and perceptual experimental results to define design parameters such as number of pins, spacing, and amplitude of pin-based tactile displays. A commonly used metric is the two-point discrimination test, which defines the minimum distance between two contact points on the skin at which they are perceived as two, rather than one point. This discrimination limit varies widely for skin on different parts of the body, with the fingertips having one of the smallest (usually cited as less than 1 mm, although this depends on the shape and size of the contacts) [42.81, 82]. *Moy* et al. [42.83] quantified several perceptual capabilities of the human tactile system based on predicted subsurface strain and psychophysical experiments that measured amplitude resolution, the effects of shear stress, and the effects of viscoelasticity (creep and relaxation) on tactile perception for static touch. They found find that 10% amplitude resolution is sufficient for a teletaction system with a 2 mm elastic layer and 2 mm tactor spacing. A different type of experiment examines the kind of tactile information relevant to a particular application. For example, *Peine* and *Howe* [42.84] found that sensed deformation of the fingerpad, and not changes in pressure distribution, were responsible for localizing lumps in a soft material, such as a tumor in tissue.

We will now highlight a few distinctive designs of array-type tactile displays. A number of actuator technologies have been applied to create tactile arrays, including piezoelectric, shape-memory alloy (SMA), electromagnetic, pneumatic, electrorheological, microelectromechanical system (MEMS), and electrotactile. Further reading on tactile display design and actuation is available in review papers, including [42.85–90].

We will first consider two ends of the spectrum in complexity/cost of the pin-based approach. *Killebrew* et al. [42.91] developed a 400-pin, 1 cm^2 tactile stimulator to present arbitrary spatiotemporal stimuli to the skin for neuroscience experiments. Each pin is under independent computer control and can present over 1200 stimuli per minute. While not practical for most haptic applications due to the size and weight of the actuation unit, it is the highest-resolution tactile display built to date and can be used to evaluate potential designs for lower-resolution displays. *Wagner* et al. [42.92] created a 36-pin, 1 cm^2 tactile shape display that uses commercially available radio-controlled (RC) servomotors. The display can represent maximum frequencies of 7.5−25 Hz, pending on the amount of pin deflection, and is shown in Fig. 42.11. *Howe* et al. [42.93, 94] have also explored the use of shape-memory alloys for pin actuation, for the application of remote palpation.

In contrast to pins that move normal to the surface, recent tactile array designs have incorporated

pins that move laterally. First introduced by *Hayward* and *Cruz-Hernandez* [42.95], the most recent compact, lightweight, modular design [42.96] uses a 6×10 piezo bimorph actuator array with a spatial resolution of $1.8\,\text{mm} \times 1.2\,\text{mm}$. The force of the individual actuators provides sufficient skin pad motion/stretch to excite mechanoreceptors [42.97]. A pilot test demonstrated that subjects could detect a virtual line randomly located on an otherwise smooth virtual surface, and the device has also been tested as a Braille display [42.98]. Another lateral stretch display and its evaluation is described in [42.99]. Other novel approaches to tactile display include sending small currents through the skin or tongue using an electrocutaneous array [42.100] and the application of air pressure to stimulate only superficial mechanoreceptors [42.101].

42.5.5 Surface Displays

In recent years, the concept of surface displays, which modulate surface friction in order to display changing shear forces as a user moves the finger over the surface. One example device that uses slip and friction to display compelling tactile sensations is the TPaD (tactile pattern display) [42.102]. Ultrasonic frequency, low amplitude vibrations of a flat plate create a film of air between the plate and a human finger touching the plate, thereby reducing friction. The 33 kHz vibration of the plate cannot be perceived by the human. The amount of friction reduction varies with vibration amplitude, allowing indirect control of shear forces on the finger during active exploration. Finger position and velocity feedback enables haptic rendering of spatial texture sensations. This work has been expanded into a variety of different surface displays, including devices that generate active forces [42.103] and those that use electrostatic forces to modulate friction [42.104, 105].

Fig. 42.11 **(a)** A low-cost 36-pin tactile display using RC servomotor actuation. **(b)** A closeup of the 6×6 display, showing a sine wave grating (after [42.92] with permission)

42.5.6 Temperature

Because the human body is typically warmer than objects in the environment, thermal perceptions are based on a combination of thermal conductivity, thermal capacity, and temperature. This allows us to infer not only temperature difference, but also material composition [42.106]. Most thermal display devices are based on thermoelectric coolers, also known as Peltier heat pumps. Thermoelectric coolers consist of a series of semiconductor junctions connected electrically in series and thermally in parallel. The thermoelectric cooler is designed to pump heat from one ceramic faceplate to the other, but if used in reverse, a temperature gradient across the device produces a proportional potential; as a measure of relative temperature change. The designs of haptic thermal displays mostly use off-the-shelf components, and their applications are typically straightforward, enabling identification of objects in a virtual or teleoperated environment by their temperature and thermal conductivity.

Ho and *Jones* [42.107] provide a review of haptic temperature display, as well as promising results suggesting that a thermal display is capable of facilitating object recognition when visual cues are limited. Although numerous systems have integrated thermal display with other types of haptic display, the Data Glove Input System designed by *Caldwell* et al. [42.108, 109] was one of the first to do so. Their haptic interface provides simultaneous force, tactile, and thermal feedback. The Peltier device used for thermal display contacts the dorsal surface of the index finger. Subjects achieved a 90% success rate in identifying materials such as a cube of ice, a soldering iron, insulating foam, and a block of aluminum, based only on thermal cues. The study of human temperature perception is particularly interesting, including issues such as spatial summation and the psychological relevance of temperature display. For example, in prosthetic limbs, temperature display

may be useful not only for practical reasons such as safety and material identification, but also for reasons of personal comfort, such as feeling the warmth of a loved one's hand.

42.6 Conclusions and Further Reading

Haptic technology, which attempts to provide compelling sensations to human operators in virtual and teleoperated environments, is a relatively new, but fast-growing and dynamic area of research. The field relies not only on fundamental foundations from robotics and control theory, but also on fields in the human sciences, particularly neuroscience and psychology. To date, commercial success of haptics has been in the areas of entertainment, medical simulation, and design, although novel devices and applications are regularly appearing.

There exist many books on the topic of haptic technology, most of them compendiums from workshops or conferences on the subject. One of the earliest books on haptics, by *Burdea* [42.110], provides a thorough review of applications and haptic devices up to 1996. A book specifically focused on haptic rendering, designed for potential use as a textbook, has been edited by *Lin* and *Otaduy* [42.111]. In addition, we recommend the following useful articles: *Hayward* and *MacLean* [42.112, 113] describe the fundamentals of constructing experimental haptic devices of modest complexity, the software components needed to drive them, and the interaction design concepts important to creating usable systems. *Hayward* et al. [42.114] also provide a tutorial on haptic devices and interfaces. *Salisbury* et al. [42.115] describe the basic principles of haptic rendering. *Hayward* and *MacLean* [42.116] describes a number of tactile illusions that can inspire creative solutions for haptic interface design. *Robles-De-La-Torre* [42.117] underscores the importance of haptics with compelling examples of humans who have lost the sense of touch.

Finally, there exist two journals that are specific to the field of haptics: *Haptics-e* [42.118] and *IEEE Transactions on Haptics* (first issue expected 2008). Several conferences are specifically devoted to haptics: Eurohaptics and the Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems are held separately in even years, and on the odd years become a single conference, World Haptics. The IEEE (Institute of Electrical and Electronics Engineers) technical committee on haptics [42.119] provides information about relevant publication forums.

References

42.1 K.B. Shimoga: A survey of perceptual feedback issues in dexterous telemanipulation. I. Finger force feedback, Proc. Virtual Real. Annu. Int. Symp. (1993) pp. 263–270

42.2 M.A. Srinivasan, R.H. LaMotte: Tactile discrimination of shape: responses of slowly and rapidly adapting mechanoreceptive afferents to a step indented into the monkey fingerpad, J. Neurosci. **7**(6), 1682–1697 (1987)

42.3 R.H. LaMotte, R.F. Friedman, C. Lu, P.S. Khalsa, M.A. Srinivasan: Raised object on a planar surface stroked across the fingerpad: Responses of cutaneous mechanoreceptors to shape and orientation, J. Neurophysiol. **80**, 2446–2466 (1998)

42.4 R.H. LaMotte, J. Whitehouse: Tactile detection of a dot on a smooth surface: Peripheral neural events, J. Neurophysiol. **56**, 1109–1128 (1986)

42.5 R. Hayashi, A. Miyake, H. Jijiwa, S. Watanabe: Postureal readjustment to body sway induced by vibration in man, Exp. Brain Res. **43**, 217–225 (1981)

42.6 G.M. Goodwin, D.I. McCloskey, P.B.C. Matthews: The contribution of muscle afferents to kinesthesia shown by vibration induced illusions of movement an the effects of paralysing joint afferents, Brain **95**, 705–748 (1972)

42.7 G.S. Dhillon, K.W. Horch: Direct neural sensory feedback and control of a prosthetic arm, IEEE Trans. Neural Syst. Rehabil. Eng. **13**(4), 468–472 (2005)

42.8 G.A. Gescheider: *Psychophysics: The Fundamentals* (Lawrence Erlbaum, Hillsdale 1985)

42.9 L.A. Jones: Perception and control of finger forces, Proc. ASME Dyn. Syst. Control Div. (1998) pp. 133–137

42.10 R. Klatzky, S. Lederman, V. Metzger: Identifying objects by touch, An 'expertt system', Percept. Psychophys. **37**(4), 299–302 (1985)

42.11 S. Lederman, R. Klatzky: Hand movements: A window into haptic object recognition, Cogn. Psychol. **19**(3), 342–368 (1987)

42.12 S. Lederman, R. Klatzky: Haptic classification of common objects: Knowledge-driven exploration, Cogn. Psychol. **22**, 421–459 (1990)

42.13 O.S. Bholat, R.S. Haluck, W.B. Murray, P.G. Gorman, T.M. Krummel: Tactile feedback is present

during minimally invasive surgery, J. Am. Coll. Surg. **189**(4), 349–355 (1999)
42.14 C. Basdogan, S. De, J. Kim, M. Muniyandi, M.A. Srinivasan: Haptics in minimally invasive surgical simulation and training, IEEE Comput. Graph. Appl. **24**(2), 56–64 (2004)
42.15 P. Strom, L. Hedman, L. Sarna, A. Kjellin, T. Wredmark, L. Fellander-Tsai: Early exposure to haptic feedback enhances performance in surgical simulator training: A prospective randomized crossover study in surgical residents, Surg. Endosc. **20**(9), 1383–1388 (2006)
42.16 A. Liu, F. Tendick, K. Cleary, C. Kaufmann: A survey of surgical simulation: Applications, technology, and education, Presence Teleop. Virtual Environ. **12**(6), 599–614 (2003)
42.17 R.M. Satava: Accomplishments and challenges of surgical simulation, Surg. Endosc. **15**(3), 232–241 (2001)
42.18 W.A. McNeely, K.D. Puterbaugh, J.J. Troy: Six degree-of-freedom haptic rendering using voxel sampling, Proc. SIGGRAPH 99 (1999) pp. 401–408
42.19 Geomagic Touch: http://www.geomagic.com
42.20 T.H. Massie, J.K. Salisbury: The phantom haptic interface: A device for probing virtual objects, Proc. ASME Dyn. Syst. Contr. Div., Vol. 55 (1994) pp. 295–299
42.21 Novint Technologies: http://www.novint.com
42.22 Chai3D: http://www.chai3d.org
42.23 M.C. Cavusoglu, D. Feygin, F. Tendick: A critical study of the mechanical and electrical properties of the PHANToM haptic interface and improvements for high-performance control, Presence **11**(6), 555–568 (2002)
42.24 R.Q. van der Linde, P. Lammerste, E. Frederiksen, B. Ruiter: The HapticMaster, a new high-performance haptic interface, Proc. Eurohaptics Conf. (2002) pp. 1–5
42.25 T. Yoshikawa: Manipulability of robotic mechanisms, Int. J. Robotics Res. **4**(2), 3–9 (1985)
42.26 J.K. Salibury, J.T. Craig: Articulated hands: Force control and kinematics issues, Int. J. Robotics Res. **1**(1), 4–17 (1982)
42.27 P. Buttolo, B. Hannaford: Pen based force display for precision manipulation of virtual environments, Proc. Virtual Reality Annu. International Symposium (VRAIS) (1995) pp. 217–225
42.28 P. Buttolo, B. Hannaford: Advantages of actuation redundancy for the design of haptic displays, Proc. ASME 4th Annu. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst., Vol. 57-2 (1995) pp. 623–630
42.29 T. Yoshikawa: *Foundations of Robotics* (MIT Press, Cambridge 1990)
42.30 S. Venema, B. Hannaford: A probabilistic representation of human workspace for use in the design of human interface mechanisms, IEEE Trans. Mechatron. **6**(3), 286–294 (2001)
42.31 H. Yano, M. Yoshie, H. Iwata: development of a non-grounded haptic interface using the gyro effect, Proc. 11th Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (2003) pp. 32–39
42.32 C. Swindells, A. Unden, T. Sang: TorqueBAR: an ungrounded haptic feedback device, Proc. 5th Int. Conf. Multimodal Interface (2003) pp. 52–59
42.33 Immersion Corporation: CyberGrasp – Groundbreaking haptic interface for the entire hand, http://www.immersion.com/3d/products/cyber_grasp.php (2006)
42.34 C. Richard, M.R. Cutkosky: Contact force perception with an ungrounded haptic interface, Proc. ASME Dyn. Syst. Control Div. (1997) pp. 181–187
42.35 J.J. Abbott, A.M. Okamura: Effects of position quantization and sampling rate on virtual-wall passivity, IEEE Trans. Robotics **21**(5), 952–964 (2005)
42.36 S. Usui, I. Amidror: Digital low-pass differentiation for biological signal processing, IEEE Trans. Biomed. Eng. **29**(10), 686–693 (1982)
42.37 P. Bhatti, B. Hannaford: Single chip optical encoder based velocity measurement system, IEEE Trans. Contr. Syst. Technol. **5**(6), 654–661 (1997)
42.38 A.M. Okamura, C. Richard, M.R. Cutkosky: Feeling is believing: Using a force-feedback joystick to teach dynamic systems, ASEE J. Eng. Educ. **92**(3), 345–349 (2002)
42.39 C.H. Ho, C. Basdogan, M.A. Srinivasan: Efficient point-based rendering techniques for haptic display of virtual objects, Presence **8**, 477–491 (1999)
42.40 C.B. Zilles, J.K. Salisbury: A constraint-based god-object method for haptic display, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1995) pp. 146–151
42.41 J.E. Colgate, M.C. Stanley, J.M. Brown: Issues in the haptic display of tool use, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1995) pp. 140–145
42.42 D. Ruspini, O. Khatib: Haptic display for human interaction with virtual dynamic environments, J. Robot. Syst. **18**(12), 769–783 (2001)
42.43 A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, D. Manocha: Six degree-of-freedom haptic display of polygonal models, Proc. Conf. Vis. 2000 (2000) pp. 139–146
42.44 D.E. Johnson, P. Willemsen, E. Cohen: 6-DOF haptic rendering using spatialized normal cone search, Trans. Vis. Comput. Graph. **11**(6), 661–670 (2005)
42.45 M.A. Otaduy, M.C. Lin: A modular haptic rendering algorithm for stable and transparent 6-DOF manipulation, IEEE Trans. Vis. Comput. Graph. **22**(4), 751–762 (2006)
42.46 M.C. Lin, M.A. Otaduy: Sensation-preserving haptic rendering, IEEE Comput. Graph. Appl. **25**(4), 8–11 (2005)
42.47 T. Thompson, E. Cohen: Direct haptic rendering of complex trimmed NURBS models, Proc. ASME Dyn. Syst. Control Div. (1999)
42.48 S.P. DiMaio, S.E. Salcudean: Needle insertion modeling and simulation, IEEE Trans. Robotics Autom. **19**(5), 864–875 (2003)
42.49 B. Hannaford: Stability and performance tradeoffs in bi-lateral telemanipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1989) pp. 1764–1767

42.50 B. Gillespie, M. Cutkosky: Stable user-specific rendering of the virtual wall, Proc. ASME Int. Mech. Eng. Cong. Exhib., Vol. 58 (1996) pp. 397–406

42.51 R.J. Adams, B. Hannaford: Stable haptic interaction with virtual environments, IEEE Trans. Robotics Autom. **15**(3), 465–474 (1999)

42.52 B.E. Miller, J.E. Colgate, R.A. Freeman: Passive implementation for a class of static nonlinear environments in haptic display, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 2937–2942

42.53 B.E. Miller, J.E. Colgate, R.A. Freeman: Computational delay and free mode environment design for haptic display, Proc. ASME Dyn. Syst. Cont. Div. (1999)

42.54 B.E. Miller, J.E. Colgate, R.A. Freeman: Environment delay in haptic systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 2434–2439

42.55 S.E. Salcudean, T.D. Vlaar: On the emulation of stiff walls and static friction with a magnetically levitated input/output device, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 119 (1997) pp. 127–132

42.56 P. Wellman, R.D. Howe: Towards realistic vibrotactile display in virtual environments, Proc. ASME Dyn. Syst. Control Div. (1995) pp. 713–718

42.57 K. MacLean: The haptic camera: A technique for characterizing and playing back haptic properties of real environments, Proc. 5th Annu. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (1996)

42.58 A.M. Okamura, J.T. Dennerlein, M.R. Cutkosky: Reality-based models for vibration feedback in virtual environments, ASME/IEEE Trans. Mechatron. **6**(3), 245–252 (2001)

42.59 K.J. Kuchenbecker, J. Fiene, G. Niemeyer: Improving contact realism through event-based haptic feedback, IEEE Trans. Vis. Comput. Graph. **12**(2), 219–230 (2006)

42.60 D.A. Kontarinis, R.D. Howe: Tactile display of vibratory information in teleoperation and virtual environments, Presence **4**(4), 387–402 (1995)

42.61 J.T. Dennerlein, P.A. Millman, R.D. Howe: Vibrotactile feedback for industrial telemanipulators, Proc. ASME Dyn. Syst. Contr. Div., Vol. 61 (1997) pp. 189–195

42.62 A.M. Okamura, J.T. Dennerlein, R.D. Howe: Vibration feedback models for virtual environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 674–679

42.63 R.W. Lindeman, Y. Yanagida, H. Noma, K. Hosaka: Wearable vibrotactile systems for virtual contact and information display, Virtual Real. **9**(2-3), 203–213 (2006)

42.64 C. Ho, H.Z. Tan, C. Spence: Using spatial vibrotactile cues to direct visual attention in driving scenes, Transp. Res. F Traffic Psychol. Behav. **8**, 397–412 (2005)

42.65 H.Z. Tan, R. Gray, J.J. Young, R. Traylor: A haptic back display for attentional and directional cueing, Haptics-e Electron. J. Haptics Res. **3**(1), 20 (2003)

42.66 C2 Tactor: Engineering Acoustic Inc.: http://www.eaiinfo.com

42.67 W.R. Provancher, M.R. Cutkosky, K.J. Kuchenbecker, G. Niemeyer: Contact location display for haptic perception of curvature and object motion, Int. J. Robotics Res. **24**(9), 691–702 (2005)

42.68 R.S. Johansson: Sensory input and control of grip, Novartis Foundat. Symp., Vol. 218 (1998) pp. 45–59

42.69 K.O. Johnson, J.R. Phillips: A rotating drum stimulator for scanned embossed patterns and textures across the skin, J. Neurosci. Methods **22**, 221–231 (1998)

42.70 M.A. Salada, J.E. Colgate, P.M. Vishton, E. Frankel: Two experiments on the perception of slip at the fingertip, Proc. 12th Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (2004) pp. 472–476

42.71 R.J. Webster III, T.E. Murphy, L.N. Verner, A.M. Okamura: A novel two-dimensional tactile slip display: Design, kinematics and perceptual experiment, ACM Trans. Appl. Percept. **2**(2), 150–165 (2005)

42.72 N.G. Tsagarakis, T. Horne, D.G. Caldwell: SLIP AESTHEASIS: A portable 2D slip/skin stretch display for the fingertip, 1st Jt. Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (World Haptics) (2005) pp. 214–219

42.73 J. Biggs, M. Srinivasan: Tangential versus normal displacements of skin: Relative effectiveness for producing tactile sensations, Proc. 10th Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (2002) pp. 121–128

42.74 V. Hayward, J.M. Cruz-Hernandez: Tactile display device using distributed lateral skin stretch, Proc. 8th Symp. Haptic Interfaces Virtual Environ. Teleoperator Syst. (2000) pp. 1309–1314

42.75 B. Gleeson, S. Horschel, W. Provancher: Perception of direction for applied tangential skin displacement: Effects of speed, displacement, and repetition, IEEE Trans. Haptics **3**(3), 177–188 (2010)

42.76 W.R. Provancher, N.D. Sylvester: Fingerpad skin stretch increases the perception of virtual friction, IEEE Trans. Haptics **2**(4), 212–223 (2009)

42.77 Z.F. Quek, S.B. Schorr, I. Nisky, W.R. Provancher, A.M. Okamura: Sensory substitution using 3-degree-of-freedom tangential and normal skin deformation feedback, IEEE Haptics Symp. (2014) pp. 27–33

42.78 A. Tirmizi, C. Pacchierottie, D. Prattichizzo: On the role of cutaneous force in teleoperation: Subtracting kinesthesia from complete haptic feedback, IEEE World Haptics Conf. (2013) pp. 371–376

42.79 K. Bark, J. Wheeler, P. Shull, J. Savall, M. Cutkosky: Rotational skin stretch feedback: A wearable haptic display for motion, IEEE Trans. Haptics **3**(3), 166–176 (2010)

42.80 P.B. Shull, K.L. Lurie, M.R. Cutkosky, T.F. Besier: Training multi-parameter gaits to reduce the knee adduction moment with data-driven models and haptic feedback, J. Biomech. **44**(8), 1605–1609 (2011)

42.81 K.O. Johnson, J.R. Phillips: Tactile spatial resolution. I. Two-point discrimination, gap detection,

grating resolution, and letter recognition, J. Neurophysiol. **46**(6), 1177–1192 (1981)
42.82 N. Asamura, T. Shinohara, Y. Tojo, N. Koshida, H. Shinoda: Necessary spatial resolution for realistic tactile feeling display, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 1851–1856
42.83 G. Moy, U. Singh, E. Tan, R.S. Fearing: Human psychophysics for teletaction system design, Haptics-e Electron. J. Haptics Res. **1**, 3 (2000)
42.84 W.J. Peine, R.D. Howe: Do humans sense finger deformation or distributed pressure to detect lumps in soft tissue, Proc. ASME Dyn. Syst. Contr. Div., Vol. 64 (1998) pp. 273–278
42.85 K.B. Shimoga: A survey of perceptual feedback issues in dexterous telemanipulation: Part II, Finger touch feedback, Proc. IEEE Virtual Real. Annu. Int. Symp. (1993) pp. 271–279
42.86 K.A. Kaczmarek, P. Bach-Y-Rita: Tactile displays. In: *Virtual Environments and Advanced Interface Design*, ed. by W. Barfield, T.A. Furness (Oxford Univ. Press, Oxford 1995) pp. 349–414
42.87 M. Shimojo: Tactile sensing and display, Trans. Inst. Electr. Eng. Jpn. E **122**, 465–468 (2002)
42.88 S. Tachi: Roles of tactile display in virtual reality, Trans. Inst. Electr. Eng. Jpn. E **122**, 461–464 (2002)
42.89 P. Kammermeier, G. Schmidt: Application-specific evaluation of tactile array displays for the human fingertip. In: *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)* 2002)
42.90 S.A. Wall, S. Brewster: Sensory substitution using tactile pin arrays: Human factors, technology and applications, Signal Process. **86**(12), 3674–3695 (2006)
42.91 J.H. Killebrew, S.J. Bensmaia, J.F. Dammann, P. Denchev, S.S. Hsiao, J.C. Craig, K.O. Johnson: A dense array stimulator to generate arbitrary spatio-temporal tactile stimuli, J. Neurosci. Methods **161**(1), 62–74 (2007)
42.92 C.R. Wagner, S.J. Lederman, R.D. Howe: Design and performance of a tactile shape display using RC servomotors, Haptics-e Electron. J. Haptics Res. **3**, 4 (2004)
42.93 R.D. Howe, W.J. Peine, D.A. Kontarinis, J.S. Son: Remote palpation technology, IEEE Eng. Med. Biol. **14**(3), 318–323 (1995)
42.94 P.S. Wellman, W.J. Peine, G. Favalora, R.D. Howe: Mechanical design and control of a high-bandwidth shape memory alloy tactile display, Lect. Notes Comput. Sci. **232**, 56–66 (1998)
42.95 V. Hayward, M. Cruz-Hernandez: Tactile display device using distributed lateral skin stretch, Haptic Interfaces Virtual Environ. Teleop. Syst. Symp., Vol. 69-2 (2000) pp. 1309–1314
42.96 Q. Wang, V. Hayward: Compact, portable, modular, high-performance, distributed tactile transducer device based on lateral skin deformation, Haptic Interfaces Virtual Environ. Teleop. Syst. Symp. (2006) pp. 67–72
42.97 Q. Wang, V. Hayward: In vivo biomechanics of the fingerpad skin under local tangential traction, J. Biomech. **40**(4), 851–860 (2007)
42.98 V. Levesque, J. Pasquero, V. Hayward: Braille display by lateral skin deformation with the STReSS2 tactile transducer, 2nd Jt. Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (World Haptics) (2007) pp. 115–120
42.99 K. Drewing, M. Fritschi, R. Zopf, M.O. Ernst, M. Buss: First evaluation of a novel tactile display exerting shear force via lateral displacement, ACM Trans. Appl. Percept. **2**(2), 118–131 (2005)
42.100 K.A. Kaczmarek, J.G. Webster, P. Bach-Y-Rita, W.J. Tompkins: Electrotactile and vibrotactile displays for sensory substitution systems, IEEE Trans. Biomed. Eng. **38**, 1–16 (1991)
42.101 N. Asamura, N. Yokoyama, H. Shinoda: Selectively stimulating skin receptors for tactile display, IEEE Comput. Graph. Appl. **18**, 32–37 (1998)
42.102 L. Winfield, J. Glassmire, J.E. Colgate, M. Peshkin: T-PaD: Tactile pattern display through variable friction reduction, 2nd Jt. Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleop. Syst. (World Haptics) (2007) pp. 421–426
42.103 J. Mullenbach, D. Johnson, J.E. Colgate, M.A. Peshkin: ActivePaD surface haptic device, IEEE Haptics Symp. (2012) pp. 407–414
42.104 O. Bau, I. Poupyrev, A. Israr, C. Harrison: TeslaTouch: Electrovibration for touch surfaces, Proc. 23nd Annu. ACM Symp. User Interface Softw. Technol. (2012) pp. 283–292
42.105 D.J. Meyer, M.A. Peshkin, J.E. Colgate: Fingertip friction modulation due to electrostatic attraction, IEEE World Haptics Conf. (2013)
42.106 H.-N. Ho, L.A. Jones: Contribution of thermal cues to material discrimination and localization, Percept. Psychophys. **68**, 118–128 (2006)
42.107 H.-N. Ho, L.A. Jones: Development and evaluation of a thermal display for material identification and discrimination, ACM Trans. Appl. Percept. **4**(2), 118–128 (2007)
42.108 D.G. Caldwell, C. Gosney: Enhanced tactile feedback (tele-taction) using a multi-functional sensory system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 955–960
42.109 D.G. Caldwell, S. Lawther, A. Wardle: Tactile perception and its application to the design of multimodal cutaneous feedback systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 3215–3221
42.110 C.G. Burdea: *Force and Touch Feedback for Virtual Reality* (Wiley, New York 1996)
42.111 M.C. Lin, M.A. Otaduy (Eds.): *Haptic Rendering: Foundations, Algorithms, and Applications* (AK Peters, Wellesley 2008)
42.112 V. Hayward, K.E. MacLean: Do it yourself haptics, Part I, IEEE Robotics Autom. Mag. **14**(4), 88–104 (2007)
42.113 K.E. MacLean, V. Hayward: Do It Yourself Haptics, Part II, IEEE Robotics Autom. Mag. **15**(1), 104–119 (2008)
42.114 V. Hayward, O.R. Astley, M. Cruz-Hernandez, D. Grant, G. Robles-De-La-Torre: Haptic interfaces and devices, Sensor Rev. **24**(1), 16–29 (2004)

42.115 K. Salisbury, F. Conti, F. Barbagli: Haptic rendering: Introductory concepts, IEEE Comput. Graph. Appl. **24**(2), 24–32 (2004)
42.116 V. Hayward, K.E. MacLean: A brief taxonomy of tactile illusions and demonstrations that can be done in a hardware store, Brain Res. Bull. **75**(6), 742–752 (2007)
42.117 G. Robles-De-La-Torre: The importance of the sense of touch in virtual and real environments, IEEE Multimedia **13**(3), 24–30 (2006)
42.118 Haptics-e: The Electronic Journal of Haptics Research, http://www.haptics-e.org
42.119 Haptics Technical Committee: http://www.worldhaptics.org

Multimedia Contents

43. Telerobotics

Günter Niemeyer, Carsten Preusche, Stefano Stramigioli, Dongjun Lee

In this chapter we present an overview of the field of telerobotics with a focus on control aspects. To acknowledge some of the earliest contributions and motivations the field has provided to robotics in general, we begin with a brief historical perspective and discuss some of the challenging applications. Then, after introducing and classifying the various system architectures and control strategies, we emphasize bilateral control and force feedback. This particular area has seen intense research work in the pursuit of telepresence. We also examine some of the emerging efforts, extending telerobotic concepts to unconventional systems and applications. Finally, we suggest some further reading for a closer engagement with the field.

43.1 **Overview and Terminology** 1085
43.2 **Telerobotic Systems and Applications** 1087
43.2.1 Historical Perspective 1087
43.2.2 Applications 1090
43.3 **Control Architectures** 1090
43.3.1 Supervisory Control 1091
43.3.2 Shared Control 1092
43.3.3 Direct and Bilateral Teleoperation 1093
43.4 **Bilateral Control and Force Feedback** 1095
43.4.1 Position/Force Control 1095
43.4.2 Passivity and Stability 1096
43.4.3 Transparency and Multichannel Feedback 1097
43.4.4 Time Delay and Scattering Theory 1098
43.4.5 Wave Variables 1098
43.4.6 Teleoperation with Lossy Communication 1099
43.4.7 Port-Based Approaches 1099
43.5 **Emerging Applications of Telerobotics** ... 1101
43.5.1 Telerobotics for Mobile Robots 1101
43.5.2 Multilateral Telerobotics 1102
43.6 **Conclusions and Further Reading** 1104
Video-References 1104
References 1105

43.1 Overview and Terminology

Telerobotics is perhaps one of the earliest aspects and manifestations of robotics. Literally meaning robotics at a distance, it is generally understood to refer to robotics with a human operator in control or *human-in-the-loop*. Any high-level, planning, or cognitive decisions are made by the human user, while the robot is responsible for their mechanical implementation. In essence, the *brain* is removed or distant from the *body*.

Herein the term *tele*, which is derived from the Greek and means distant, is generalized to imply a barrier between the user and the environment. This barrier is overcome by remote-controlling a robot at the environment, as indicated in Fig. 43.1. Besides distance, barriers may be imposed by hazardous environments or scaling to very large or small environments. All barriers prevent the user from physically reaching or directly interacting with the environment.

While the physical separation may be very small, with the human operator and the robot sometimes occupying the same room, telerobotic systems are often at least conceptually split into two sites. The *local site* encompasses the human operator and all elements necessary to support the system's connection with the user, which could be joysticks, monitors, keyboards, or other

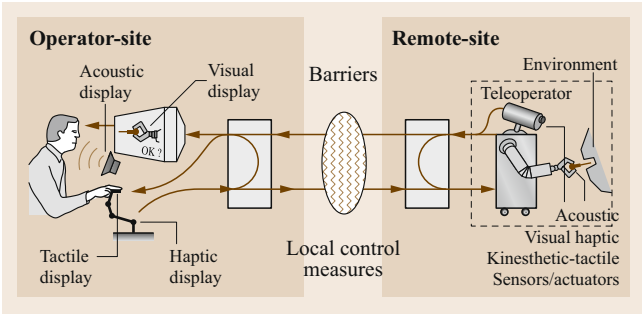


Fig. 43.1 Overview of a telerobotic system (after [43.1], adapted from [43.2])

input/output devices. The *remote site* contains the robot, supporting sensors and control elements, and the environment to be manipulated.

To support its operation, telerobotics integrates many areas of robotics. At the remote site, to operate the robot and execute the human's commands, the system may control the motion and/or forces of the robot. We refer to Chaps. 7 and 8 for detailed descriptions of these areas. Also, sensors are invaluable (Chap. 5), including force sensors (Chap. 28) and others (Part C). Meanwhile, at the local site information is often displayed haptically (Chap. 41).

A recent addition to telerobotics is the use of computer networks to transmit information between the sites. The ubiquity of network access is allowing remote control from anywhere on demand. Chapter 44 discusses some of these developments, detailing network infrastructure and focusing on visual, often web-based, user interfaces, avoiding the need for specialized mechanical I/O (input/output) devices. Computer networks also allow new multilateral telerobotic architectures. For example, multiple users may share a single robot or a single user may control multiple robots (Sect. 43.5.2). Unfortunately, computer networks often see transmission delays and can introduce nondeterministic effects such as variable delay times and data losses. These effects can easily destabilize force feedback loops and require particular countermeasures (Sects. 43.4.4–43.4.6).

We should also point out the relation between telerobotics and human exoskeletons, as described in Chap. 69. Exoskeletons are also controlled by a human operator, leaving all planning and high-level challenges to the user, and their control systems share many aspects with telerobotics. However, the two sites are physically combined in an exoskeleton as the user directly touches and interacts with the robot. In this chapter, we will disallow any such direct mechanical connection.

The inclusion of the human operator makes telerobotics very attractive to handle unknown and unstructured environments. Applications are plentiful (Part F) and range from space robotics (Chap. 55) to dealing with hazardous environments (Chap. 58), from search and rescue situations (Chap. 60), to medical systems (Chap. 63) and rehabilitation (Chap. 64).

Before proceeding, we define some basic terminology. Indeed many other terms are used nearly synonymously with telerobotics, in particular *teleoperation* and *telemanipulation*. Telerobotics is the most common, emphasizing a human's (remote) control of a robot. Teleoperation stresses the task-level operations, while telemanipulation highlights object-level manipulation.

Within telerobotics, a spectrum of control architectures is used. *Direct control* or *manual control* falls at one extreme, indicating that the user is controlling the motion of the robot directly and without any automated help. At the other extreme, *supervisory control* implies that user's commands and feedback occur at a very high level and the robot requires substantial intelligence or autonomy to fulfill its function. Between the two extrema lie a variety of *shared control* architectures, where some degree of autonomy or automated help is available to assist the user.

In practice, many systems involve at least some level of direct control and accept the user's motion commands via a joystick or similar device in the user interface. The joystick is an instrumented mechanism and can itself be viewed as a robot. The local and remote robots are called *master* and *slave* respectively, while the system is referred to as a *master–slave system*. To provide direct control, the slave robot is programmed to follow the motions of the master robot, which is positioned by the user. It is not uncommon for the master robot (joystick) to be a kinematic replica of the slave, providing an intuitive interface.

Some master–slave systems provide force feedback, such that the master robot not only measures motions but also displays forces to the user. The user interface becomes fully bidirectional and such telerobotic systems are often called *bilateral*. The human–master interactions are a form of human–robot interaction (Chap. 69). The field of haptics (Chap. 41) also discusses bidirectional user interfaces, involving both motion and force, though more commonly to interface the user with virtual instead of remote environments. We should note that both motion and force may become the input or output to/from the user, depending on the system architecture.

Finally, *telepresence* is often discussed as an ultimate goal of master–slave systems and telerobotics in general. It promises to the user not only the abil-

ity to manipulate the remote environment, but also to perceive the environment as if encountered directly. The human operator is provided with enough feedback and sensations to feel present in the remote site. This combines the haptic modality with other modalities serving the human senses of vision, hearing or even smell and taste. See videos VIDEO 297, VIDEO 318 and VIDEO 319, and VIDEO 321 for some early and recent results aiming for this telepresence. We focus our descriptions on the haptic channel, which is created by the robotic hardware and its control systems. The master–slave system becomes the medium through which the user interacts with the remote environment and ideally they are fooled into forgetting about the medium itself. If this is achieved, we say that the master–slave system is transparent.

While bilateral master–slave systems have held the biggest promise for telepresence and intuitive operations, they have also posed some of the largest stability and control problems. Especially considering force feedback from sensors at the remote site, these systems close multiple interwoven feedback loops and have to deal with large uncertainties in the environment. They have received heavy research attention and will therefore be a repeated focus in some of our following discussions.

43.2 Telerobotic Systems and Applications

Telerobotic systems, like most robotic devices, are typically designed for specific tasks and according to explicit requirements. As such, many unique systems have evolved, of which we present an overview for different applications. We begin with a short historical perspective, then describe different applications with various robot designs and user interfaces.

43.2.1 Historical Perspective

Teleoperation enjoys a rich history and dates back to nuclear research by Raymond C. Goertz in the 1940s and 1950s. In particular, he created systems for humans to handle radioactive material from behind shielded walls. The first systems were electrical, controlled by an array of on–off switches to activate various motors and move various axes [43.3]. Without any feel, these manipulators were *slow and somewhat awkward to operate*, leading *Goertz* to build pairs of mechanically linked master–slave robots [43.3, 4]. Connected by gears, linkages, and cables, these systems allowed the operator to use natural hand motions and transmitted forces and vibrations through the connecting structure (Fig. 43.2). Unfortunately they limited the distance between the operator and environment and required the use of kinematically identical devices. *Goertz* quickly recognized the value of electrically coupled manipulators and laid the foundations of modern telerobotics and bilateral force-reflecting positional servos [43.5].

At the beginning of the 1960s the effects of time delay on teleoperation started to become a topic of research [43.6, 7]. To cope with this problem the concept of supervisory control was introduced [43.2] and inspired the next years of development. In the late 1980s and early 1990s theoretical control came into play with Lyapunov-based analysis and network theory [43.8–13]. Using these new methods, bilateral control of telerobotic systems became the vital research area it is today (Sect. 43.4). The growth of the Internet and its use as a communication medium has further fueled this trend, adding the challenges of nondeterministic time delay.

On the hardware side, the Central Research Laboratory model M2 of 1982 was the first telerobotic system which realized force feedback while separating master and slave electronics. It was developed together with the Oak Ridge National Laboratory and was used for some time for a wide range of demonstration tasks including military, space or nuclear applications. The National Aeronautics and Space Administration (NASA) tested the M2 system to simulate the ACCESS space truss assembly with excellent results (Fig. 43.3). The ad-

Fig. 43.2 Raymond C. Goertz used electrical and mechanical teleoperators in the early 1950s to handle radioactive material (courtesy Argonne National Labs)

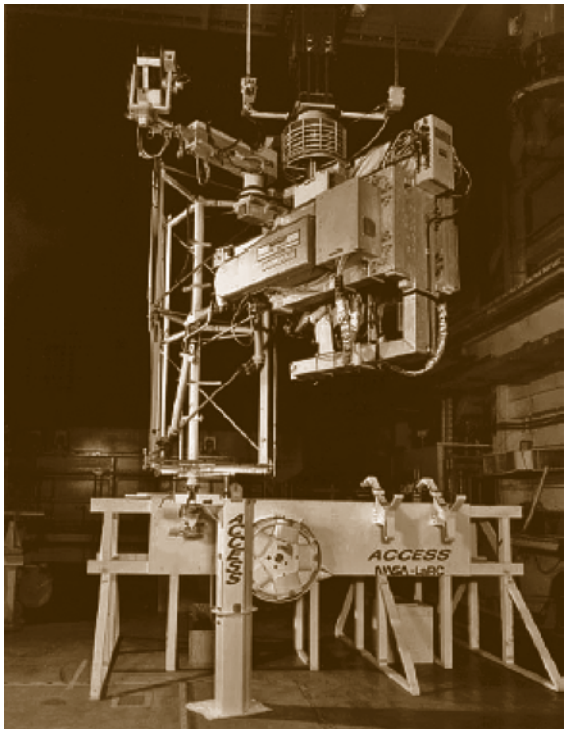


Fig. 43.3 The telerobotic system CRL Model M2 is used to verify the assembly of space truss structures (1982) (courtesy Oak Ridge National Laboratory)

vanced servomanipulator (ASM) was developed from the M2 to improve the remote maintainability of manipulators and intended as a foundation for telerobotic systems [43.14].

Also driven by the nuclear application, bilateral servomanipulators for teleoperation were developed in France at the Commission de Energie Atomique (CEA) by *Vertut* and *Coiffet* [43.15]. With the MA 23 they demonstrated telerobotic operation including computer-assisted functionalities to improve the operator's performance [43.16]. The assistance included software jigs and fixtures or virtual walls and restrictions [43.17] (Sect. 43.3.2).

For space applications a dual-arm force reflecting telerobotic system was developed by *Bejczy* at the Jet Propulsion Laboratory (JPL) [43.18] and VIDEO 298. This system was the first use of kinematically and dynamically different master and slave robots. It required control in the Cartesian space coordinates of the operator's hand and slave robot's tool. Figure 43.4 shows the master control station with its two back-drivable hand controllers. The system was used to simulate teleoperation in space.

In 1993 the first telerobotic system was flown in space with the German Spacelab Mission D2 on board

Fig. 43.4 JPL ATOP control station (early 1980s) (JPL No. 19902Ac, courtesy NASA/JPL-CALTECH)

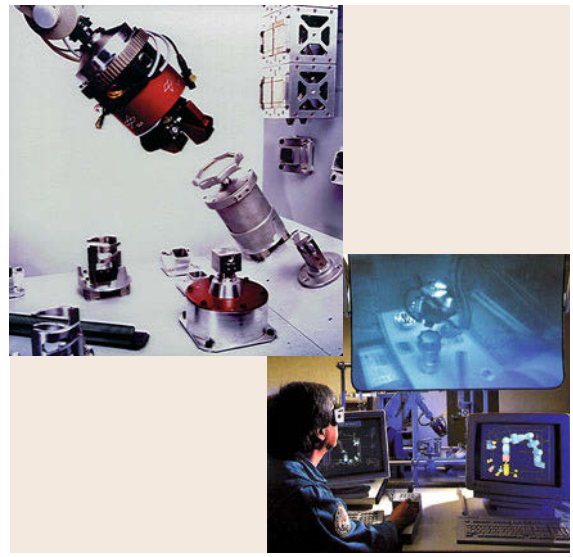


Fig. 43.5 ROTEX, the first remotely controlled robot in space (1993). Telerobot in space and ground operator station (courtesy German Aerospace Center, DLR)

the Space Shuttle Columbia. The robot technology experiment (ROTEX), shown in Fig. 43.5, demonstrated remote control of a space robot by means of local sensory feedback, predictive displays, and teleoperation [43.19]. In this experiment the round trip delay was 6–7 s, such that it was not feasible to include force feedback into the control loop.

Throughout the 1980s and 1990s, as nuclear power activities began to decline, interests expanded into new areas including medicine and undersea operations. Ef-

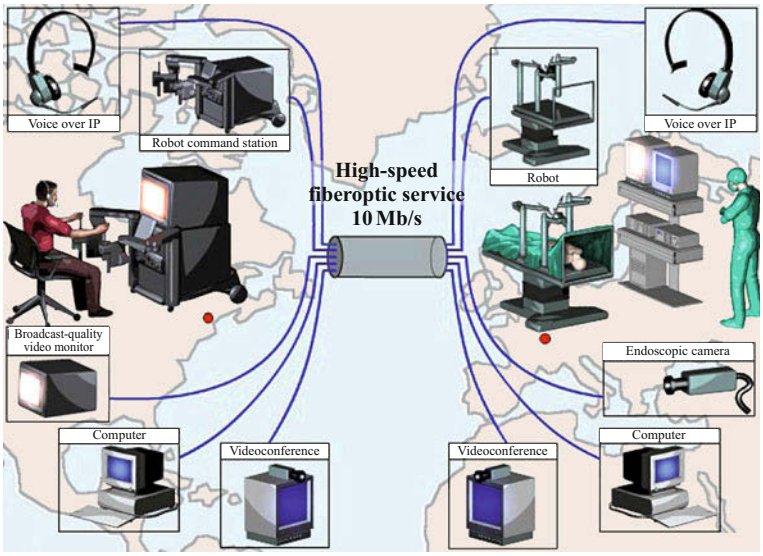


Fig. 43.6 Operation Lindberg. The first transcontinental telerobotic surgery (2001) (courtesy M. Ghodoussi)

forts were accelerated by the availability of increasing computer power as well as the introduction of novel hand controllers, e.g., the PHANToM device [43.20], popularized by haptic applications in virtual reality (Chap. 41).

Simultaneously, surgery was seeing the trend toward minimally invasive techniques, highlighted by the first laparoscopic cholecystectomy (removal of the gallbladder) in 1987. Several groups saw the potential for telerobotics and pursued telesurgical systems. Most noteworthy are the Telepresence Surgery System developed at the Stanford Research Institute (now SRI International) in 1987 [43.21], the Laparoscopic Assistant Robotic System (LARS) created at the IBM Watson Research Center [43.22], the teleoperated surgical instrument Falcon designed at MIT (Massachusetts Institute of Technology) [43.23], and the Robot Assisted Microsurgery (RAMS) workstation developed at JPL [43.24].

In 1995 Intuitive Surgical Inc. was founded to leverage several of these concepts, leading to the da Vinci telesurgical system [43.25] and its introduction to market in 1999. Meanwhile Computer Motion started with a voice-controlled robot moving an endoscopic camera [43.26] and extended those capabilities into the

Fig. 43.7 Intuitive Surgical Inc. makes the da Vinci telerobotic system, which is used in minimally invasive surgery (courtesy 2008 Intuitive Surgical, Inc.)

Fig. 43.8 tEODor, a telerobotic system for disarming of explosives (courtesy telerob Gesellschaft für Fernhantierungstechnik mbH, Ostfildern, Germany)

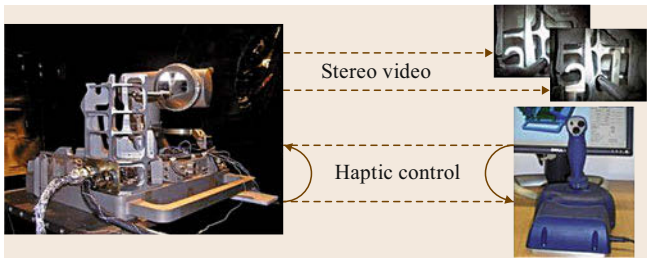


Fig. 43.9 ROKVISS, a telerobotic system providing stereo vision and haptic feedback to the ground operator (courtesy German Aerospace Center, DLR)

ZEUS system. In 2001 a surgeon in New York (USA) used a ZEUS system to perform the first transatlantic telesurgical laparoscopic cholecystectomy on a patient located in Strasbourg (France) [43.27], as depicted in Fig. 43.6. The system did not include force feedback, so the surgeon had to rely on visual feedback only.

In this perspective we have given reference only to the systems that may be seen as milestones within the history of telerobotics. Other systems, which have been developed and added value to the research field, unfortunately could not be mentioned here.

43.2.2 Applications

Telerobotic systems have been motivated by issues of human safety in hazardous environments (e.g., nuclear or chemical plants), the high cost of reaching remote environments (e.g., space), scale (e.g., power amplification or position scaling in micromanipulation or minimally invasive surgery), and many others. Not surprisingly, after their beginning in nuclear research, telerobotic systems have evolved to many fields of application. Nearly everywhere a robot is used, telerobotic systems can be found. The following are some of the more exciting uses.

In minimally invasive surgery telerobots allow procedures to be performed through small incisions, reducing the trauma to the patient compared to traditional surgery [43.28]. The da Vinci system, made by Intuitive Surgical Inc. [43.25] and shown in Fig. 43.7, is the only commercially available device at present. Other efforts, however, have included computer motion [43.26] and endoVia Medical [43.29] on the commercial side, as well as the University of Washington [43.30], Johns Hopkins University [43.31], the German Aerospace Center [43.32], and many others (VIDEO 322).

Protecting the operator from having to reach into a hazardous environment, telerobotic systems are widely used in nuclear or chemical industry. Some systems have been developed for the maintenance of high-voltage electrical power lines, which can be safely repaired without service interruption by a human operator using a telerobotic system. Disarming of explosives is another important task. Many systems like the telerob explosive ordnance disposal and observation robot (tEODor) shown in Fig. 43.8 or PackBot, made by iRobot [43.33], are used by police and military to disarm mines and other explosives. Similar vehicles are remote controlled for search and rescue in disaster zones [43.34].

Space robotics is a classic application, in which distance is the dominating barrier, as discussed in Chap. 55. The NASA rovers on Mars are a famous example. Due to the time delay of several minutes, the rovers are commanded using supervisory control, in which the human operator is defining the goal of a movement and the rover achieves the goal by local autonomy using sensory feedback directly [43.35].

In orbital robotics the German technology experiment ROKVISS (robot component verification on the international space station (ISS)) was the most advanced telerobotic system [43.36]. Launched in 2004 and operational through 2010, it was installed outside the Russian module of the international space station. It validated advanced robot components in the slave system, including torque sensors and stereo video cameras, in real space conditions. Using a direct communication link between the space station and the operator station at DLR (German Aerospace Center), the time delay was kept at about 20 ms allowing a bilateral control architecture with high-fidelity force feedback to the operator [43.37] (Fig. 43.9). This technology is leading toward robotic service satellites, called Robonauts, which can be controlled remotely from the ground to help real astronauts during extravehicular activities (EVA) or to perform repair and maintenance tasks [43.38].

43.3 Control Architectures

Compared to plain robotic systems, in which a robot executes a motion or other program without further consultation of a user or operator, telerobotic systems provide information to and require commands from the user. Their control architectures can be described by the style and level of this connection, as shown in Fig. 43.10. Organized in a spectrum, the three main categories are:

- Direct control
- Shared control
- Supervisory control.

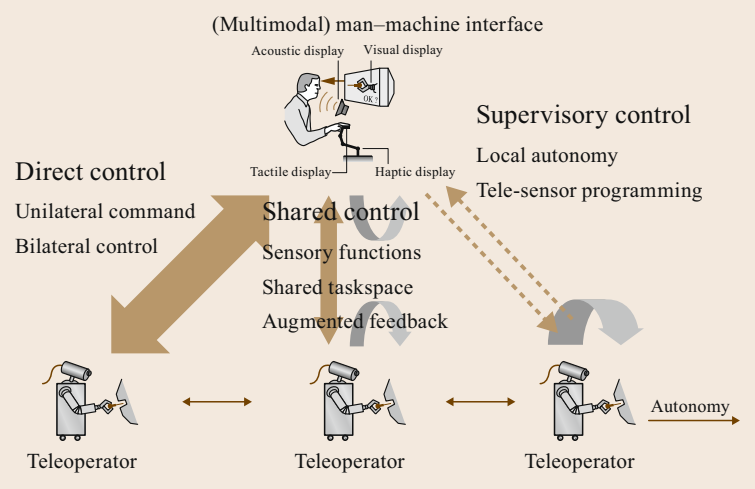


Fig. 43.10 Different concepts for telerobotic control architectures

In practice, however, control architectures often include parts of all strategies.

Direct control implies no intelligence or autonomy in the system, so that all slave motion is directly controlled by the user via the master interface. This may incorporate sensory feedback to the user in a bilateral configuration. If the slave motion is controlled by a combination of direct user commands and local sensory feedback or autonomy, the architecture is denoted as shared control. It is similarly shared if user feedback is augmented from virtual reality or by other automatic aids. In supervisory control user commands and feedback occur at a higher level. The connection is more loose and the slave has to rely on stronger local autonomy to refine and execute tasks. The following explains the architectures in reverse order, leading to a detailed treatment of direct and bilateral control in Sect. 43.3.3, which introduces the basic ideas for Sect. 43.4.

43.3.1 Supervisory Control

Supervisory control, introduced by *Ferell* and *Sheridan* in 1967 [43.2], is derived from the analog of supervising a human subordinate staff member. The supervisor gives high-level directives to and receives summary information from, in this case, the robot. *Sheridan* describes this approach in comparison with manual and automatic robot control [43.39]:

> *Human operators are intermittently programming and continually receiving information from a computer that itself closes an autonomous control loop through artificial effectors and sensors.*

In general, supervisory control techniques will allow more and more autonomy and intelligence to shift to the robot system. Today simple autonomous control loops may be closed at the remote site, with only state and model information being transmitted to the operator site. The operator supervises the telerobotic system closely and decides exactly how to act and what to do. A specific implementation of supervisory control is the telesensor programming approach, which is presented hereafter. See also VIDEO 299 for another implementation of supervisory control for space operation.

Telesensor Programming

Developed for space applications with large communication delays, the telesensor programming (TSP) approach has been characterized as a *task-level-oriented programming technique* and *sensor-based teaching by showing* [43.40, 41]. In essence, operators interact with a complex simulation of the robot and remote environment, in which they can test and adjust tasks. The tasks, consisting of robot and environment signals and configuration parameters, are then uploaded to the remote site. The approach presumes that the sensor systems provide sufficient information about the actual environment so that the tasks can be executed autonomously. Specifications and high-level planning remain the responsibility of the human operator.

Figure 43.11 shows the structure of a TSP implementation, consisting of two control loops working in parallel. One loop controls the real (remote) system, which contains internal feedback for local autonomy. The other loop establishes a simulation environment which is structurally equivalent to the real system, with a few exceptions. Most importantly, any signal delay which may result from communication to the remote system, in particular in space applications, is not duplicated in the simulation. This makes the simulation predictive with respect to the real system. A second exception is the display of internal variables in the sim-

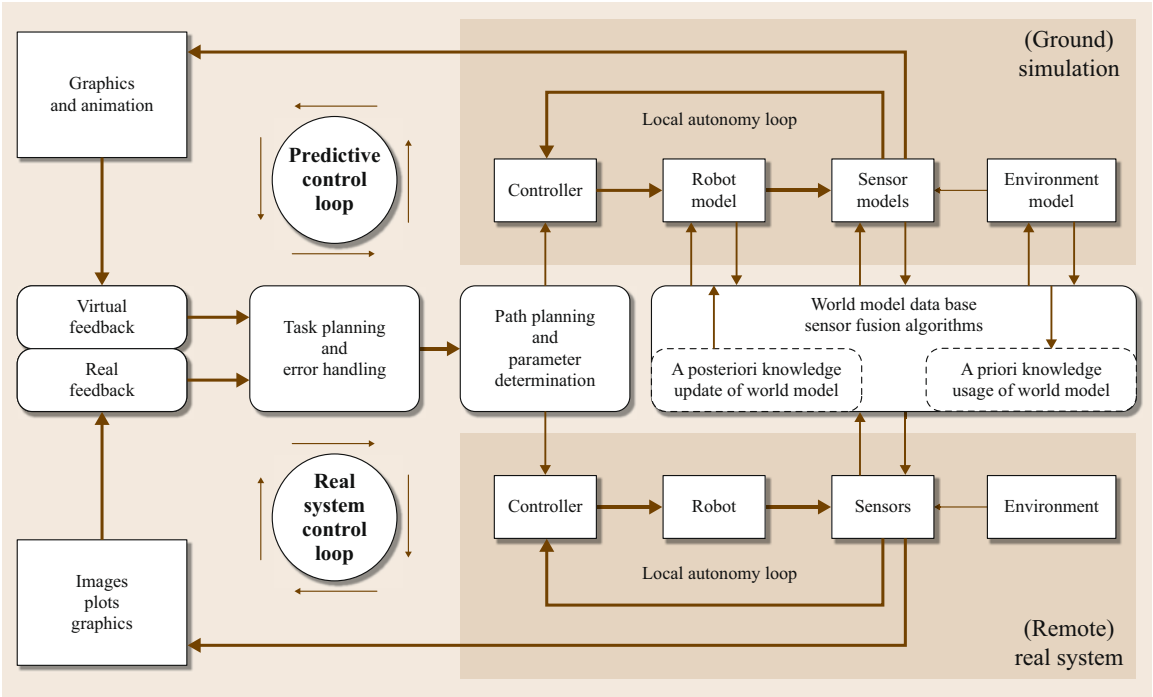


Fig. 43.11 The concept of telesensor programming as demonstrated during the ROTEX mission

ulation, which cannot be observed (measured) in the real system. This gives the operator or task planner more insight into what is happening or may happen in the system in response to commands. Communication between the two loops occurs via a common model data base which delivers a priori knowledge for execution on the remote system and a posteriori knowledge for model updating in the simulated world.

Unique tools are necessary to implement the functionality required for such a telerobotic control system. First a sophisticated simulation system has to be provided to emulate the real robot system. This includes the simulation of sensory perception within the real environment. Also, the operator needs an efficient interface to set up task descriptions, to configure the task control parameters, to decide what kind of sensors and control algorithms should be used, and to debug an entire job execution phase.

For telerobotic systems with large time delays of a few seconds or more, e.g., in space and undersea applications, such a sensor-based task-directed programming approach has advantages. It is not feasible for human operators to handle the robot movements directly under delayed visual feedback. Only a predictive simulation allows the operator to telemanipulate the remote system [43.42]. In addition, the use of force reflecting hand controllers to feed back force signals from the simulated predicted world can improve the operator's performance [43.43]. Finally, an interactive supervisory user interface makes it possible to configure the environmental and control parameters.

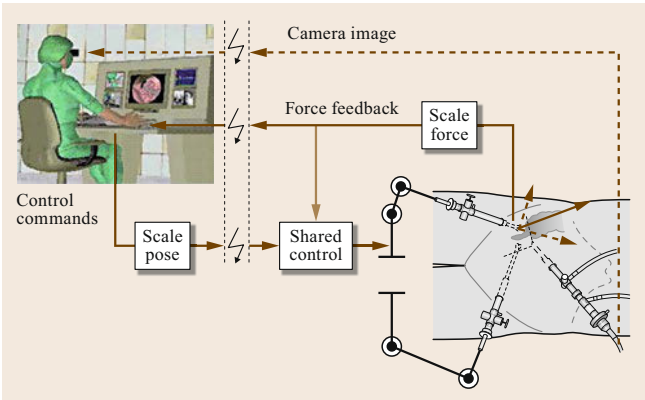


Fig. 43.12 An example for the shared control concept in telerobotic surgery

43.3.2 Shared Control

Shared control tries to combine the basic reliability and sense of presence achievable by direct control with the smarts and possible safety guarantees of autonomous control ([43.44, 45] and VIDEO 299). This may occur in various forms. For example, the slave robot may need

to correct motion commands, regulate subsets of joints or subtasks, or overlay additional commands.

With large communication delays, a human operator may only be able to specify gross path commands, which the slave must fine-tune with local sensory information [43.46]. We may also want the slave to assume control of subtasks, such as maintaining a grasp over long periods of time [43.47]. And in surgical applications, shared control has been proposed to compensate for beating heart movements (Fig. 43.12). The sensed heart motion is overlaid on the user commands, so the surgeon can operate on a virtually stabilized patient [43.48].

A special application of shared control is the use of *virtual fixtures* ([43.49–51] and VIDEO 72). Virtual elements, such as virtual surfaces, virtual velocity field, guide tubes, or other appropriate objects, are superimposed into the visual and/or haptic scene for the user. These fixtures can help the operator perform tasks by limiting movement into restricted regions and/or influencing movement along desired paths. Control is thus shared at the master site, taking advantage of preknowledge of the system or task to modify the user's commands and/or to combine them with autonomously generated signals.

Capitalizing on the accuracy of robotic systems while sharing control with the operator, telerobotic systems with virtual fixtures can achieve safer, faster and more intuitive operation. *Abbott* et al. describe the benefits by comparison to the common physical fixture of a ruler [43.50]:

> *A straight line drawn by a human with the help of a ruler is drawn faster and straighter than a line drawn freehand. Similarly, a [master] robot can apply forces or positions to a human operator to help him or her draw a straight line.*

Based on the nature of the master robot and its controller, the virtual fixtures may apply corrective forces or constrain positions. In both cases, and in contrast to physical fixtures, the level and type of assistance can be programmed and varied.

43.3.3 Direct and Bilateral Teleoperation

To avoid difficulties in creating local autonomy, most telerobotic systems include some form of direct control: they allow the operator to specify the robot's motions. This may involve commanding either position or velocity or acceleration. We begin our discussions with the later two options, which are generally implemented unilaterally without force feedback to the user. We then focus on position control, which is more suited to bilateral operation. We will assume a master–slave system, i. e., the user is holding a joystick or master mechanism serving as an input device.

Unilateral Acceleration or Rate Control

For underwater, airborne, or space applications, a slave robot may be a vehicle actuated by thrusters. Direct control thus requires the user to power the thrusters, which in turn accelerates the vehicle. For other applications, the user may be required to command the rate or velocity of the vehicle or slave robot. In both scenarios, the input device is commonly a joystick, often spring centered, where the acceleration or rate commands are proportional to the joystick displacement. For six degree-of-freedom (DOF) applications, i. e., when the slave needs to be controlled in translation and orientation, a six-dimensional (6-D) space mouse can be used. Alternatively two joysticks may separately command translation and orientation.

Acceleration and rate control are very attractive when the master and slave robots are fundamentally different, for example if the slave robot can reach an effectively unbounded workspace. Unfortunately, basic implementations can require considerable effort for the operator to reach and hold a given target location. As expected, users can more accurately position a system under rate control than under acceleration control [43.52]. Indeed acceleration control necessitates users to regulate a second-order system versus a first-order system for rate control. Assuming the slave has local position feedback available, a control system is often incorporated locally, such that the user may specify position commands and is relieved from the dynamic control problem. We refer to Sect. 43.5.1 for some emerging developments for bilateral control of mobile robots.

Position Control and Kinematic Coupling

Assuming that the slave is under position control, we can consider a kinematic coupling between master and slave, i. e., a mapping between master and slave positions. In particular, we must remember that the master mechanism moves in the master workspace, while the slave robot moves in the slave workspace. The mapping connects these two spaces, which are nearly always somewhat different.

Clutching and Offsets. Before discussing how the two robots are coupled, we must understand that they are not always coupled. For example, before the system is turned on, master and slave robots may, for whatever reason, be placed in some initial position/configuration. We have three options of how to engage the system:

1. First autonomously move one or both robots so they come to the same position

2. Wait until someone (the user) externally moves one robot to match the location of the other, or
3. Connect the two robots with some offset.

Once connected, most systems also allow a temporary disconnection between the two sites. The reason is twofold: to allow the user to rest without affecting the slave state and to allow a shift between the two robots. The later is most important if the workspaces of both robots do not perfectly overlap. This is much like picking up your mouse off your mouse pad to reposition without moving the cursor. In telerobotics the process is called *clutching* or sometimes *indexing*. If clutching is allowed, or both robots are not constrained to start at the same location, the system must allow for offsets between the two robots.

When clutched or disconnected, most systems hold the slave at rest or allow it to float in response to environment forces. It is also possible for the slave to retain its preclutching momentum and continue moving, similar to kinetic scrolling popularized in smartphones [43.53].

Kinematically Similar Mechanisms. The simplest scenario involves a master and slave mechanism that are kinematically equivalent if not entirely identical. In this case, the two robots can be connected at a joint level. With $\boldsymbol{q}$ denoting joint values and subscripts "m" referring to the master, "s" to the slave, "offset" to a shared offset, and "d" to a desired value, we can write

$$\begin{aligned} \boldsymbol{q}_{\mathrm{sd}} &= \boldsymbol{q}_{\mathrm{m}} + \boldsymbol{q}_{\mathrm{offset}} \,, \\ \boldsymbol{q}_{\mathrm{md}} &= \boldsymbol{q}_{\mathrm{s}} - \boldsymbol{q}_{\mathrm{offset}} \,. \end{aligned} \tag{43.1}$$

At the instance the two robots are to be connected or reconnected, the offset is computed as

$$\boldsymbol{q}_{\mathrm{offset}} = \boldsymbol{q}_{\mathrm{s}} - \boldsymbol{q}_{\mathrm{m}} \,. \tag{43.2}$$

Most kinematically similar master–slave systems have the same workspace at both sites and do not allow clutching. By construction the offset is then always zero.

Depending on the controller architecture, the joint velocities may be similarly related, taking derivatives of (43.1). An offset in velocities is not necessary.

Kinematically Dissimilar Mechanisms. In many cases, the master and slave robots differ. Consider that the master is connected to the human user and thus should be designed accordingly. Meanwhile the slave works in some environment and may have a very different joint configuration and different number of joints. As a result, connecting the robots joint by joint may not be feasible or appropriate.

Instead kinematically dissimilar robots are commonly connected at their tips. If $\boldsymbol{x}$ is a robot's tip position, we have

$$\begin{aligned} \boldsymbol{x}_{\mathrm{sd}} &= \boldsymbol{x}_{\mathrm{m}} + \boldsymbol{x}_{\mathrm{offset}} \,, \\ \boldsymbol{x}_{\mathrm{md}} &= \boldsymbol{x}_{\mathrm{s}} - \boldsymbol{x}_{\mathrm{offset}} \,. \end{aligned} \tag{43.3}$$

If orientations are also connected, with $\mathbf{R}$ describing a rotation matrix, we have

$$\begin{aligned} \mathbf{R}_{\mathrm{sd}} &= \mathbf{R}_{\mathrm{m}}\mathbf{R}_{\mathrm{offset}} \,, \\ \mathbf{R}_{\mathrm{md}} &= \mathbf{R}_{\mathrm{s}}\mathbf{R}_{\mathrm{offset}}^{\mathrm{T}} \,, \end{aligned} \tag{43.4}$$

where the orientational offset is defined as slave relative to master

$$\mathbf{R}_{\mathrm{offset}} = \mathbf{R}_{\mathrm{m}}^{\mathrm{T}}\mathbf{R}_{\mathrm{s}} \,. \tag{43.5}$$

Again velocities and angular velocities may be connected if needed and do not require offsets.

Finally note that most telerobotic systems use a video camera at the remote site and a monitor at the local site. To make the connection appear natural, the slave position and orientation should be measured relative to the camera, while the master position and orientation should be measured relative to the user's view.

Scaling and Workspace Mapping. Kinematically dissimilar master–slave robots are commonly also of different size. This means not only do they require clutching to fully map one workspace to another, but they often necessitate motion scaling. And so (43.3) becomes

$$\begin{aligned} \boldsymbol{x}_{\mathrm{sd}} &= \mu\boldsymbol{x}_{\mathrm{m}} + \boldsymbol{x}_{\mathrm{offset}} \,, \\ \boldsymbol{x}_{\mathrm{md}} &= \frac{(\boldsymbol{x}_{\mathrm{s}} - \boldsymbol{x}_{\mathrm{offset}})}{\mu} \,. \end{aligned} \tag{43.6}$$

The orientation, however, typically should not be scaled. The scale μ may be set to either map the two workspaces as best possible, or to provide the most comfort to the user.

If force feedback is provided, as described below, an equivalent force scale may be desired. This will prevent distortion of the remote environmental conditions, such as stiffness or damping, by the scaling. In addition to the motion and force scalings, it is also possible to directly achieve power scaling between master and slave systems [43.51].

Beyond linear scaling, several research efforts have created nonlinear or time-varying mappings, which deform the workspaces. These may effectively change the scale in the proximity of objects [43.54] or drift the offset to best utilize the master workspace [43.55].

Local Position and Advanced Control. By construction we are now assuming that the slave follows a position command. This necessitates a local slave controller to regulate its position. In particular for kinematically dissimilar mechanisms, this will be a Cartesian tip position controller (Chap. 7).

If the slave robot has redundancies or possesses a large number of DOFs, these may be controlled either automatically to optimize some criterion or manually with additional user commands. Indeed some emerging applications are coordinating multiple users to control such complex systems. This is particularly relevant when the kinematic dissimilarity becomes extreme and has received considerable research attention. We refer here to Chap. 11 and especially Sect. 43.5 for appropriate techniques and new developments.

43.4 Bilateral Control and Force Feedback

In pursuit of telepresence and to increase task performance, many master–slave systems incorporate force feedback. That is, the slave robot doubles as a sensor and the master functions as a display device, so that the system provides both forward and feedback pathways from the user to the environment and back. Figure 43.13 depicts the common architecture viewed as a chain of elements from the user to the environment.

The bilateral nature of this setup makes the control architecture particularly challenging: multiple feedback loops form and even without environment contact or user intervention, the two robots form an internal closed loop. The communications between the two sites often inserts delays into the system and this loop, so that stability of the system can be a challenging issue [43.56].

To present force information without stability problems, it is possible to use alternate displays, such as audio or tactile devices [43.57]. Meanwhile, the combination of vibrotactile methods with explicit force feedback can increase high-frequency sensations and provide benefits to the user [43.58]. Tactile shape sensing and display also extends the force information presented to the user [43.59].

In the following we discuss explicit force feedback. We first examine the basic architectures before discussing stability and some advanced techniques.

43.4.1 Position/Force Control

Two basic architectures couple the master and slave robots: position–position and position–force. We assume that the robot tips are to be connected by the equations of Sect. 43.3.3 and give the control laws for translation. Control of orientation or joint motions follows equivalent patterns.

Position–Position Architecture

In the simplest case, both robots are instructed to track each other. Both sites implement a tracking controller, often a proportional–derivative (PD) controller, to fulfill these commands,

$$\begin{aligned} \boldsymbol{F}_\mathrm{m} &= -\boldsymbol{K}_\mathrm{m}(\boldsymbol{x}_\mathrm{m}-\boldsymbol{x}_\mathrm{md})-\boldsymbol{B}_\mathrm{m}(\dot{\boldsymbol{x}}_\mathrm{m}-\dot{\boldsymbol{x}}_\mathrm{md})\,, \\ \boldsymbol{F}_\mathrm{s} &= -\boldsymbol{K}_\mathrm{s}(\boldsymbol{x}_\mathrm{s}-\boldsymbol{x}_\mathrm{sd})-\boldsymbol{B}_\mathrm{s}(\dot{\boldsymbol{x}}_\mathrm{s}-\dot{\boldsymbol{x}}_\mathrm{sd})\,. \end{aligned} \tag{43.7}$$

If the position and velocity gains are the same ($\boldsymbol{K}_\mathrm{m}=\boldsymbol{K}_\mathrm{s}=\boldsymbol{K}$, $\boldsymbol{B}_\mathrm{m}=\boldsymbol{B}_\mathrm{s}=\boldsymbol{B}$), then the two forces are the same and the system effectively provides force feedback. This may also be interpreted as a spring and damper between the tips of each robot, as illustrated in Fig. 43.14. If the two robots are substantially different and require different position and velocity gains, some suitable force, position or power scalings, as explained in Sect. 43.3.3, may be utilized.

Note we have assumed the slave is under impedance control and back-drivable. If the slave is admittance controlled, i. e., it accepts position commands directly, the second part of (43.7) is unnecessary.

Also note that by construction the user feels the slave's controller forces, which include forces associated with the spring–damper and slave inertia in addition to environment forces. Indeed while moving without contact, the user will feel the inertial and other dynamic forces needed to move the slave. Furthermore, if the slave is not back-drivable, i. e., does not easily move under environment forces, the environment force

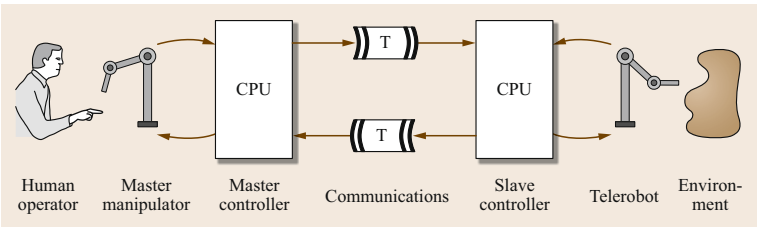


Fig. 43.13 A typical bilateral teleoperator can beviewed as a chain of elements reaching from user to environment (CPU – central processing unit)

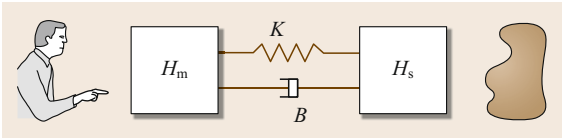


Fig. 43.14 A position–position architecture effectively creates a spring and damper between the two robots

may be entirely hidden from the user. Naturally this defeats the purpose of force feedback. In these cases, a local force control system may be used to render the slave back-drivable. Alternatively, a position–force architecture may be selected.

Position–Force Architecture

In the above position–position architecture, the user was effectively presented with the slave's controller force. While this is very stable, it also means the user feels the friction and inertia in the slave robot, which the controller is actively driving to overcome. In many scenarios this is undesirable. To avoid the issue, position–force architectures place a force sensor at the tip of the slave robot and feedback the force from there. That is, the system is controlled by

$$\begin{aligned} \boldsymbol{F}_{\mathrm{m}} &= \boldsymbol{F}_{\mathrm{sensor}} \,, \\ \boldsymbol{F}_{\mathrm{s}} &= -\boldsymbol{K}_{\mathrm{s}}(\boldsymbol{x}_{\mathrm{s}} - \boldsymbol{x}_{\mathrm{sd}}) - \boldsymbol{B}_{\mathrm{s}}(\dot{\boldsymbol{x}}_{\mathrm{s}} - \dot{\boldsymbol{x}}_{\mathrm{sd}}) \,. \end{aligned} \tag{43.8}$$

This allows the user to only feel the external forces acting between the slave and the environment and presents a more clear sense of the environment. However, this architecture is less stable: the control loop passes from master motion to slave motion to environment forces back to master forces. There may be some lag in the slave's motion tracking not to mention any delay in communications. Meanwhile the loop gain can be very high: a small motion command can turn into a large force if the slave is pressing against a stiff environment. In combination, stability may be compromised in stiff contact and many systems exhibit contact instability in these cases.

43.4.2 Passivity and Stability

The two basic architectures presented in Sect. 43.4.1 clearly illustrate one of the basic tradeoffs and challenges in force feedback: stability versus performance. Stability issues arise because any models of the system depend on the environment as well as the user. Both these elements are difficult to capture and, if we assume we want to explore unknown environments, impossible to predict. This issue makes a stability analysis very difficult. A common tool that avoids some of these issues is the concept of passivity. Although passivity provides only a sufficient (not a necessary) condition for stability, it incorporates the environmental uncertainly very well.

Passivity is an intuitive tool that examines the energy flows in a system and makes stability assertions if energy is dissipated instead of generated. Three rules are of importance here. First, a system is passive if and only if it cannot produce energy. That is the output energy from the system is limited by the initial and accumulated energy in the system. Second, two passive systems can be combined to form a new passive system. Third, the feedback connection of two passive systems is stable.

In the case of telerobotics, we generally assume that the slave robot will only interact with passive environments, that is, that the environments do not contain active motors or the like. Without the human operator, stability can therefore be assured if the system is also passive, without needing an explicit environment model.

On the master side the operator closes a loop and has to be considered in the stability analysis. In general, the master robot will be held by the user's hand and arm. A variety of models and parameters describe the human arm dynamics, mainly in the form of a mass–damper–spring system. In [43.60] we find a summary of model parameters used by different authors. For an impedance-controlled haptic interface, found in many systems, the operator adds some physical damping and a light touch ($\boldsymbol{F}_{\mathrm{human}} \approx 0$) is often one of the more challenging scenarios [43.61, 62]. As such some analyses ignore the human operator entirely. But even in general, humans seem to achieve stable interactions with all passive systems and passivity is usually considered sufficient for stable human–telerobot interactions.

To apply passivity, we take the system originally depicted in Fig. 43.13 and describe it as two-port elements in Fig. 43.15. Each port describes the energy flow between subsystems, where power is the product of a pair

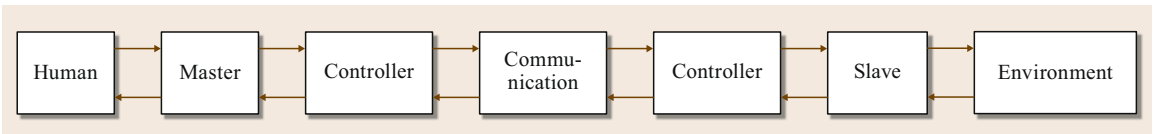


Fig. 43.15 A teleoperator can be analyzed as a chain of-two port elements connecting the one-port operator to the one-port environment

of collocated physical variables. We can choose a sign convention, for example declaring positive power to flow toward the remote environment. Then at the first boundary, the positive power flow is the product of master velocity $\dot{\boldsymbol{x}}_\mathrm{m}$ times applied (human) force $\boldsymbol{F}_\mathrm{human}$

$$P_\mathrm{left} = \dot{\boldsymbol{x}}_\mathrm{m}^\mathrm{T} \boldsymbol{F}_\mathrm{human} \,. \tag{43.9}$$

Meanwhile at the last boundary, the positive power flow is the product of the slave velocity $\dot{\boldsymbol{x}}_\mathrm{s}$ times the environment force $\boldsymbol{F}_\mathrm{env}$ (which ultimately opposes the human force)

$$P_\mathrm{right} = \dot{\boldsymbol{x}}_\mathrm{s}^\mathrm{T} \boldsymbol{F}_\mathrm{env} \,. \tag{43.10}$$

Therefore the entire telerobotic system is passive if

$$\begin{aligned} \int_0^t P_\mathrm{input}\,\mathrm{d}t &= \int_0^t (P_\mathrm{left} - P_\mathrm{right})\,\mathrm{d}t \\ &= \int_0^t \left(\dot{\boldsymbol{x}}_\mathrm{m}^\mathrm{T}\boldsymbol{F}_\mathrm{human} - \dot{\boldsymbol{x}}_\mathrm{s}^\mathrm{T}\boldsymbol{F}_\mathrm{env}\right)\,\mathrm{d}t \\ &> -E_\mathrm{store}(0) \,. \end{aligned} \tag{43.11}$$

Obviously, the ideal teleoperator ($\dot{\boldsymbol{x}}_\mathrm{m} = \dot{\boldsymbol{x}}_\mathrm{s}, \boldsymbol{F}_\mathrm{human} = \boldsymbol{F}_\mathrm{env}$) is passive. These definitions can also be generalized to six dimensional twists and wrenches.

To simplify the analysis, we can examine the passivity of each two-port element or subsystem and then deduce the overall passivity. The master and slave robots are mechanical elements and hence passive. The controllers of a position–position architecture mimic a spring and damper, which are also passive elements. So without delay and ignoring discretization, quantization, and other unmodelled effects, a simple position–position architecture is passive. The following sections will address some of these limitations and thus create controllers that are passive under more circumstances.

While powerful to handle uncertainty, passivity can be overly conservative. Many controllers are overdamped if every subsystem is passive. In contrast, the combination of an active and a passive subsystem may be passive and stable and show less dissipation. This is particularly true for the cascaded arrangement of two-port elements in the telerobotic system of Fig. 43.15. From network theory, the Llewellyn criterion specifies when a possibly active two-port connected with any passive one-port becomes passive. This two-port is then labeled unconditionally stable, as it will be stable in connection to any two passive one-ports. The Llewellyn criterion may hence be used as a more general stability test for telerobotic systems or components [43.63].

Passive controllers are also limited as they cannot hide the dynamics of the slave robot. In the above position–position architecture, the user will feel the forces associated with the slave inertia. In contrast the position–force architecture hides the slave inertia and friction from the user. As such, when the user inserts kinetic energy into the master without feeling any resistance, the system itself creates and injects the kinetic energy for the slave. This violates passivity and provides another insight as to why the architecture suffers from potential stability problems.

43.4.3 Transparency and Multichannel Feedback

Both basic architectures can be captured by the general teleoperator control system described by *Lawrence* [43.13], and later expanded by *Hashtrudi-Zaad* and *Salcudean* [43.63] and shown in Fig. 43.16. Ideally a system will equalize both the operator and environment forces as well as the master and slave motions. Therefore, it is desirable to measure both force and velocity (from which position may be integrated or vice versa) at both sites. With this complete information, the slave may, for example, start moving as soon

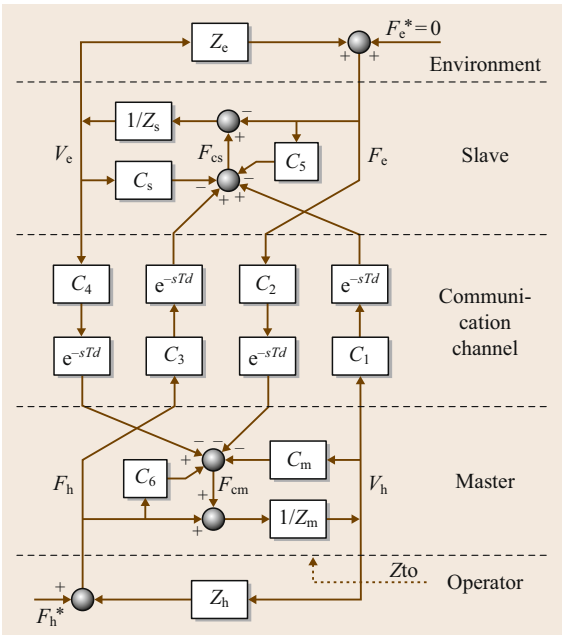


Fig. 43.16 In general, a controller will use both position and force information from both master and slave robot (after [43.63], adapted from [43.13])

as the user applies a force to the master, even before the master itself has moved.

Following these concepts derived in [43.13], we can examine the relationships between velocity and force, in the form of impedances and admittances. Note we do this in a single degree of freedom, assuming that all degrees of freedom may be treated independently. The environment will exhibit some impedance $Z_e(s)$ that is not known in advance and relates the environment force to the slave's velocity

$$\boldsymbol{F}_e(s) = Z_e(s)\boldsymbol{v}_s(s)\,. \tag{43.12}$$

If we describe the teleoperator in whole as a two-port with a hybrid matrix formulation

$$\begin{pmatrix} F_h(s) \\ \boldsymbol{v}_m(s) \end{pmatrix} = \begin{pmatrix} H_{11}(s) & H_{12}(s) \\ H_{21}(s) & H_{22}(s) \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_s(s) \\ -F_e(s) \end{pmatrix}, \tag{43.13}$$

then the user will perceive the impedance

$$Z_{to}(s) = \frac{\boldsymbol{F}_h(s)}{\boldsymbol{v}_m(s)} = (H_{11} - H_{12}Z_e)(H_{21} - H_{22}Z_e)^{-1}\,. \tag{43.14}$$

Transparency describes how close the user's perceived impedance comes to recreating the true environment impedance.

For a detailed treatment of passivity in telerobotics, impedance and admittance interpretations and designs, and transparency, we refer to some of the seminal works in [43.11–13, 63–66].

43.4.4 Time Delay and Scattering Theory

When delays occur in the communications between the local and remote site, even position–position architectures can suffer from serious instabilities [43.67, 68]. This can be traced to the communications block in Fig. 43.15, where the power entering the left side and exiting the right side do not add up. Rather energy may be generated inside the block, which feeds the instability [43.9].

Several approaches to operate under delay have been studied [43.69], in particular shared compliant control [43.70] and the addition of local force loops [43.71]. The use of the Internet for communication, adding variability to the delay, is also an area of interest [43.72, 73]. This further evokes issues of data reduction [43.74].

Here we note that natural wave phenomena are bilateral passive elements that tolerate delay. If the control system is described in the frequency domain and scattering matrices are used in place of impedance and admittance matrices, the system can tolerate delays [43.75]. Scattering matrices relate the sum of velocity and force to their difference, so that passivity becomes a condition on the system gain, which is unaffected by the delay.

43.4.5 Wave Variables

Building on the realization that delay communications can be active and that wave phenomena circumvent the issue, wave variables provide an encoding scheme that is tolerant of delay [43.76]. Consider the power flowing through the system and separate the power moving forward and returning.

$$P = \dot{\boldsymbol{x}}^{\mathrm{T}}\boldsymbol{F} = \frac{1}{2}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u} - \frac{1}{2}\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v} = P_{\text{forward}} - P_{\text{return}}\,, \tag{43.15}$$

where the forward and returning power are by construction nonnegative. This leads to the definition of the wave variables

$$\boldsymbol{u} = \frac{b\dot{\boldsymbol{x}} + \boldsymbol{F}}{\sqrt{2b}}\,, \quad \boldsymbol{v} = \frac{b\dot{\boldsymbol{x}} - \boldsymbol{F}}{\sqrt{2b}}\,, \tag{43.16}$$

where $\boldsymbol{u}$ is the forward-moving and $\boldsymbol{v}$ the returning wave.

If velocity and force signals are encoded into wave variables, transmitted across the delay, and decoded at the far site, the system remains passive regardless of delay. In fact, in the wave domain, passivity corresponds to a wave gain of less than or equal to unity. No requirements are placed on phase and so lag does not destroy stability.

The wave impedance b relates velocity to force and provides a tuning knob to the operator. Large b values mean the system increases force feedback levels at the cost of feeling high inertial forces. Small values of b lower any unwanted sensations, making it easy to move quickly, but also lower the desirable environment forces. Ideally the operator would lower b when there is no risk of contact and raise b when contact is imminent. The concept of scattering has also been extended to a coordinate free context [43.77].

Recent developments are incorporating both position–position and position–force architectures within the wave frame work, so the resulting systems are stable with any environment, stable with any delay, yet maintain the feedback of high-frequency forces that help the operator identify happenings at the remote site [43.78]. To improve performance and assist the operator, especially across the Internet, predictors may also be incorporated [43.79].

43.4.6 Teleoperation with Lossy Communication

The Internet provides an affordable and ubiquitously accessible communication medium. However, it can also introduce nondeterministic effects and lossy connections due to time-varying delays, packet losses, and packet reordering. How to achieve passive bilateral teleoperation over such lossy communication network has been an active research topic in telerobotics.

Many telerobotic systems have relied on the position-position architecture (43.7) with extra damping injection. This leads to the following proportional-derivative (PD) control law, with a simple structure and explicit position feedback,

$$\begin{aligned} \boldsymbol{F}_{\mathrm{m}} &= -\boldsymbol{B}_{\mathrm{d}}\dot{\boldsymbol{x}}_{\mathrm{m}}(t) - \boldsymbol{B}[\dot{\boldsymbol{x}}_{\mathrm{m}}(t) - \dot{\boldsymbol{x}}_{\mathrm{s}}(t-\tau_1)] \\ &\quad - \boldsymbol{K}[\boldsymbol{x}_{\mathrm{m}}(t) - \boldsymbol{x}_{\mathrm{s}}(t-\tau_1)] \\ \boldsymbol{F}_{\mathrm{s}} &= -\boldsymbol{B}_{\mathrm{d}}\dot{\boldsymbol{x}}_{\mathrm{s}}(t) - \boldsymbol{B}[\dot{\boldsymbol{x}}_{\mathrm{s}}(t) - \dot{\boldsymbol{x}}_{\mathrm{m}}(t-\tau_2)] \\ &\quad - \boldsymbol{K}[\boldsymbol{x}_{\mathrm{s}}(t) - \boldsymbol{x}_{\mathrm{m}}(t-\tau_2)]\,, \end{aligned} \tag{43.17}$$

where $\boldsymbol{B}_{\mathrm{d}}, \boldsymbol{K}, \boldsymbol{B}$ are the stabilizing absolute damping and the PD control gains, and $\tau_1, \tau_2 \geq 0$ are the communication delays from slave to master and from master to slave, respectively.

The usage of this controller across lossy communication was yet hampered (or at least reserved) due to the lack of theoretical guarantees for its passivity and stability. Anecdotes say that with sufficiently large damping and small delays and PD gains, the closed-loop system remains stable. This anecdotal observation was justified in [43.80] for the case of constant delay, that is, the PD-like controller (43.17) is passive if the following condition is met,

$$\boldsymbol{B}_{\mathrm{d}} > \frac{\bar{\tau}_1 + \bar{\tau}_2}{2}\boldsymbol{K}\,, \tag{43.18}$$

where $(\bar{\tau}_1 + \bar{\tau}_2)/2$ is the upper-bound of the round-trip delay, which can typically be estimated easily. Without human or environment forcing, master and slave positions will also converge to each other. This result was extended in [43.81] for the case of time-varying delay. Passivity of the PD-like controller (43.17) is guaranteed if

$$\boldsymbol{B}_{\mathrm{d}} > \frac{\sqrt{\bar{\tau}_1^2 + \bar{\tau}_2^2}}{2}\boldsymbol{K} \quad \text{and} \quad |\dot{\tau}_i(t)| < 1\,, \tag{43.19}$$

where the second condition means the delay $\tau_i(t)$ neither grows nor decreases faster than time t. We refer to [43.81] for cases with asymmetric controller gains and to [43.82, 83] for extensions to general digital lossy communication networks.

With a fixed structure, the PD controller (43.17) has to be tuned to the worst-case conditions according to (43.18) and (43.19). It may thus exhibit drastic performance degradation for severely variable communications. To overcome such fixed structure limitation, several passivity-enforcing flexible control techniques have been recently proposed.

The technique of passivity-observer/passivity-controller (PO/PC) was originally devised for haptic device control [43.84] and has been extended for the bilateral teleoperation with digital network with time-varying delay [43.85–87]. Each PO does real-time bookkeeping of energy flows at the master and slave sites. The PC is activated to dissipate energy whenever a passivity violation is detected. The passive set-position modulation (PSPM) framework was proposed in [43.88], where the desired set-position signal received from a general digital lossy communication network is real-time modulated as close to the original set-position signal as possible, yet, only to the extent permissible by the available energy in the system. Passification of a desired control force utilizing the device's physical damping was addressed in [43.89] under the name of energy bounding algorithm (EBA). The idea of modulating a control signal or action under the passivity constraint was also adopted in the two-layer approach [43.90], where the transparency-layer is designed for best performance, while the passivity-layer superimposes constraints to enforce passivity.

Overall, the PO/PC approach may be considered as *corrective* (i. e., detect violation of passivity first and then apply passifying action) whereas the PSPM, EBA and the two-layer approaches are *preventive* (i. e., modulate/bound control action to prevent passivity violation). Interestingly, all the PO/PC, PSPM, EBA and two-layer approaches share some common characteristics: 1) they transmit energy packets along with other information over the communication network to replenish energy levels and enable useful work; and 2) each of these approaches activates their passifying action only when necessary, thus, can significantly improve control performance compared to fixed-structure teleoperation controllers, while also robustly enforcing passivity against a variety of communication imperfectness.

43.4.7 Port-Based Approaches

From Sect. 43.4.2 we know that energy flows provide a great description of telerobotic systems that physically interact with unknown environments and a human user. Indeed all physical interaction dynamics are fundamentally bound to energy exchanges. Passivity is a well suited analysis tool and assures stability if the system energy is bounded. All possible instabilities

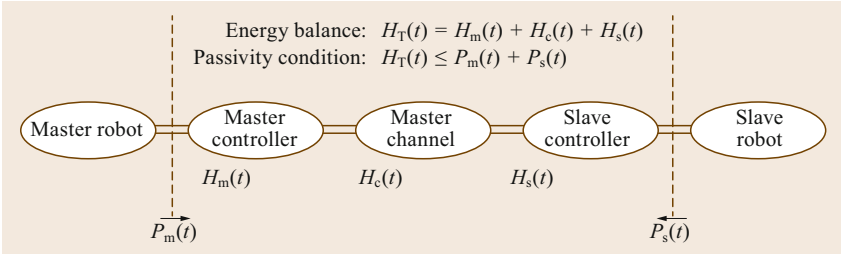


Fig. 43.17 Energy balance of the telemanipulation chain (after [43.90])

can be traced to unsupervised energy injections via the actuators. In Sect. 43.4.6 we also saw methods that explicitly monitor energy flows. Here we describe port-based approaches designed explicitly around energy exchanges. This can be advantageous to handle nonlinear system dynamics, discretization, time-varying delays, and other issues.

The concept of power ports and energy flows (Sect. 43.4.2) allows a precise analytical formulation of physical systems. This approach, which stems from concepts used in bond-graphs, enables a pure energy based analysis of complex nonlinear physical systems [43.91]. It has also provided new perspectives on modeling and robot control [43.92].

To handle sampling issues and prevent instabilities due to limited sampling rates, consider an energy-consistent discretization [43.93]. Rather than measuring power flow at discrete intervals, consider the energy flow that occurs continuously over the entire sampling interval. An exact measure of the energy transfer between sample time kT and the following sample time $(k+1)T$ is

$$\Delta E_k = \int_{kT}^{(k+1)T} \tau(t)\dot{q}(t)\,\mathrm{d}t\,. \tag{43.20}$$

Assume an electric motor with an ideal current amplifier and without any commutation effects is generating the torque as commanded by a zero order hold (ZOH) digital to analog converter. Further assume the position sensor is collocated and synchronized to the ZOH transitions. The torque may then be considered constant during the interval, so that

$$\Delta E_k = \int_{kT}^{(k+1)T} \tau_{k,k+1}\dot{q}(t)\,\mathrm{d}t\,. \tag{43.21}$$

Taking the torque out of the integral, we realize that

$$\Delta E_k = \tau_{k,k+1}[q(k+1) - q(k)]\,. \tag{43.22}$$

This simple result has far reaching consequences for interfacing the digital and physical world and can be easily calculated even if the sampling time varies VIDEO 724. And some of the assumptions may be relaxed with suitable adaptation. Using this more precise measurement of energy flow leads to more consistent energy book-keeping.

As it is shown in Fig. 43.17, we can now track energy in the digital world, associating individual available energy reservoirs H_m and H_s with the master and slave controllers. The communication channel transmits both data and energy packets (EPs), where EPs contain only information about an energy quanta. An EP is only sent if the transmitter has sufficient available energy, which is then decreased by the quanta. An arriving EP injects the quanta into the receiver's available energy. In this way the total *virtual* energy in the system will never increase. If the communication protocol allows an EP to be lost, this will remove energy similar to dissipation. This process is independent of any constant or variable delay.

The port-based paradigm allows any nonlinear control algorithm. But any applied control force will have an energetic consequence and will be allowed if and only if the associated available energy level is sufficiently high. Following the classification of Sect. 43.4.6, this paradigm is preventive. It prevents energy generation without needing to dissipate unexpected energy appearances.

Strategies are needed to address the exchange of energy between master and slave. A simple protocol introduced in [43.90] continually transmits EPs with a percentage of the locally stored energy. It can be shown that this will result in an equal distribution of energy between the master and slave controller. Also, if necessary, a small damper can be superimposed at the master side to extract energy from the human as needed.

With the port-based paradigm, the data communication (related to transparency) and the energy communication (related to passivity) are split: controllers may be nonlinear, energy and data transmission may be independent, and there is basically no restriction on the kind of controller which can be implemented. The fact that energy and data are separated gives the name to the two-layer approach [43.90].

43.5 Emerging Applications of Telerobotics

Historically, telerobotics research has focused on a conventional setup with two fixed-based robotic manipulators serving as the master and slave devices. Recently, there have been substantial efforts to extend the telerobotic theories and frameworks to more unconventional scenarios. Here we summarize some recent results on these emerging applications. The summary is by no means exhaustive and, to be consistent with the chapter, focuses on controls aspects and providing a stable bilateral user interface.

43.5.1 Telerobotics for Mobile Robots

Mobile robots are useful slave devices if the task covers a large spatial area. Flying robots, in particular, can operate in three-dimensional space without being bound to the ground. For mobile robot teleoperation, force feedback may be used to convey proprioceptive information of the slave robot (e.g., velocity), or haptic feedback of virtual (or real) objects in the remote environment.

A key difference of mobile and flying robot teleoperation compared to a conventional setup is *kinematic dissimilarity* [43.94]: the workspace of the master device is bounded while the workspace of the slave robot is unbounded. This suggests to couple the master position to the slave velocity, as with rate-control described in Sect. 43.3.3. A direct coupling between master position and slave velocity, however, cannot be addressed by the standard passivity framework (Sect. 43.4.2). The master position and the slave velocity possess different relative degrees with respect to the torque. One way to circumvent this difficulty is to utilize so called r-variable

$$r := \dot{q} + \lambda q \,. \tag{43.23}$$

That is, by utilizing inverse-dynamics similar to adaptive control design [43.95, 96] or by injecting some PD-type local state feedback with redefined output r [43.97] (Fig. 43.18), it is possible to render the master device to be passive with this r-variable replacing the velocity $\dot{q}$ in its original passivity relation (43.9). This implies that we can couple the r-variable and the slave's velocity passively, just like standard passivity-based controllers.

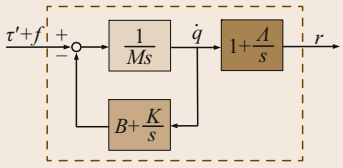


Fig. 43.18 Feedback r-passivation by using PD-type state feedback

Meanwhile the r-variable contains the master position information.

We can also achieve passive mobile robot teleoperation with time-varying delays by extending the port-based ideas presented in Sect. 43.4.7 with the concept of a *virtual vehicle* [43.98]. This virtual vehicle is a simulated slave system, evolving in a gravitation-free field and having a finite energy tank. Commands from the master are only allowed to transfer energy from the tank to accelerate the vehicle or to return energy by decelerating the vehicle. This creates a closed energy system for the slave. A passive behavior can be achieved with a viscoelastic connection between the virtual and real vehicles.

Mobile telerobotics often uses nonholonomically-constrained wheeled mobile robots (WMRs) with pure-rolling wheels. For slave WMRs, it is often possible (with some low-level control as stated below) to split motions (i. e., velocity directions) into those requiring a position-velocity coupling as stated above (e.g., forward velocity of WMR), and others that may be controlled by a standard position-position coupling (e.g., rotation angle of WMR) as explained in Sects. 43.4.1 and 43.4.6.

Other mobile robots, particularly quadrotor or ducted-fan type aerial robots (Chap. 36) or thrust-propelled autonomous aquatic vehicles (AUVs, Chap. 25), are under-actuated with fewer control variables than degrees of freedom and also defy standard teleoperation techniques. To address under-actuation, *abstraction* of the slave robot has been utilized [43.98, 99], that is, human users telecontrol a fully-actuated *virtual system*, assuming that its motion is adequately describing that of the real slave robot, while a certain low-level tracking control is employed to drive the under-actuated slave mobile robot to tightly follow this virtual system. This abstraction leads to a hierarchical control design, composed of: 1) a high-level teleoperation control layer between the virtual vehicle and the master device; and 2) a low-level tracking control layer, into which the issue of the slave's under-actuation is confined.

For the teleoperation layer, we may then use the conventional teleoperation techniques explained in Sect. 43.4 with the (r, v)-coupling as explained above. We may also use the recently proposed control techniques explained in Sect. 43.4.6, which promise sharper haptic feedback with guaranteed stability against lossy communication. Presenting both the slave robot's proprioceptive information and the presence of surrounding objects for obstacle avoidance via the same haptic feedback channel, however, typically results in

a perceptual ambiguity [43.100]. A further perceptual modality (e.g., vision) is usually necessary to resolve this ambiguity. In some applications, the slave mobile robots are required to move around in but to not directly contact any physical environments. In these cases, virtual forces can be generated for any obstacles and the slave robot is interacting only with this precisely known virtual force field. This suggests it would be sufficient to enforce slave stability versus slave passivity ([43.99], VIDEO 71 and VIDEO 72), as a less conservative controller will likely give better system performance.

43.5.2 Multilateral Telerobotics

Many practical telerobotic tasks require dexterous, complicated, and large degree-of-freedom motions, e.g., in surgical training, rehabilitation, or exploration. For such tasks, we may utilize a team of multiple cooperative slave robots or a single slave robot possessing many degrees of freedom. The complexity involved in both cases may require multiple human operators to adequately control and coordinate all degrees of freedom. Following [43.102], we thus consider the following scenarios:

1. Single-master multiple-slave (SMMS) systems
2. Multiple-master multiple-slave (MMMS) systems
3. Multiple-master single-slave (MMSS) systems
4. Single-master single-slave (SMSS) systems, which constitute the conventional telerobotic setup.

Here we introduce some recent results applicable to SMMS and MMSS systems.

It is common for single-master multiple-slave (SMMS) systems to autonomously control simple subtasks among the slaves, e.g., maintaining a grasp, maintaining connectivity, or avoiding collisions (Fig. 43.19). In particular, in [43.101, 104, 105], passive decomposition [43.106] is utilized to decompose the dynamics of multiple slave robots into their shape system, describing the inter-slave formation aspects, and the locked system, abstracting their collective motion and centroid behavior. The locked system can be telecontrolled by a single human user, while an autonomous controller regulates the shape system to maintain the cooperative grasp of an object between the slaves.

Another interesting development in SMMS telerobotics is the combination with the frameworks of multiagent cooperative control (i. e., consensus, flocking, synchronization, and other behaviors). For instance, in [43.103] and VIDEO 73, a single human user directly telecontrols a single leader agent among the slaves, while the behavior of the other slaves is dictated by a leader-follower information graph (Fig. 43.20). A time-varying graph topology with arbitrary split/join operations among the slaves allows for reconfiguration, e.g., for navigation in cluttered environments. And the concept of virtual energy tanks, along with the port-Hamiltonian modeling and port-based approach (Sect. 43.4.7), is utilized to enforce passivity and stability of the total system.

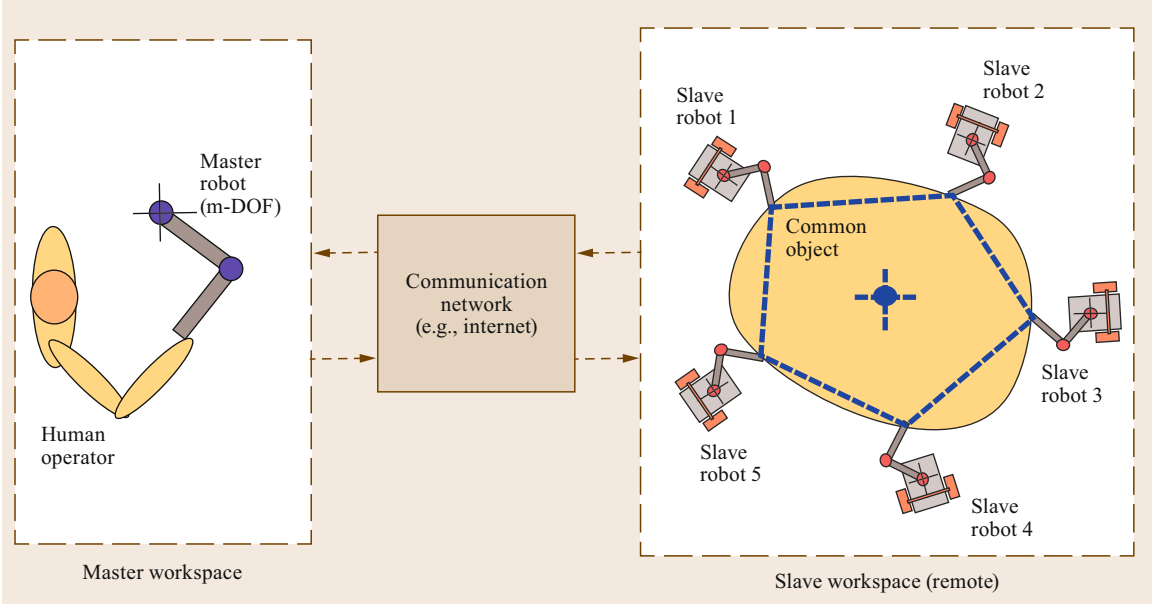


Fig. 43.19 SMMS telerobotic control of multiple slave robots (after [43.101])

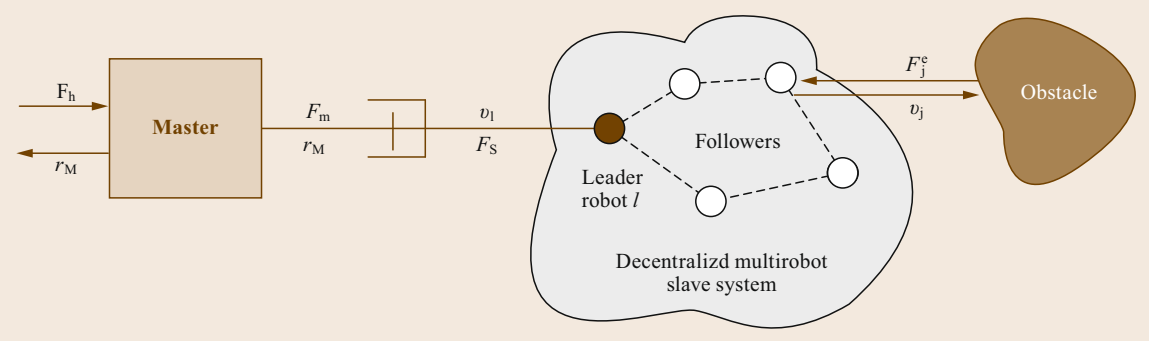


Fig. 43.20 SMMS telerobotic control architecture with possibly time-varying leader-follower information topology (after [43.103])

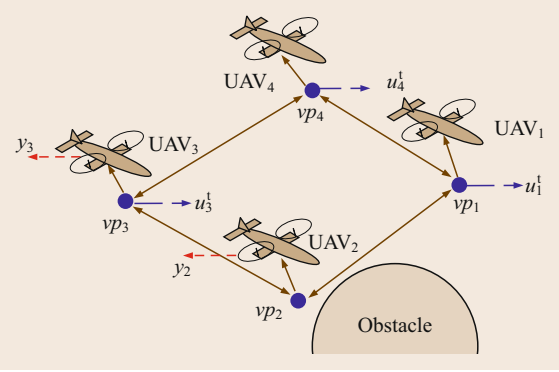


Fig. 43.21 SMMS control architecture for multiple unmanned aerial vehicles (UAVs; after [43.99]), where a single user telecontrols some of the virtual systems (via u_1^t, u_4^t) while telesensing the state of some of real UAVs (i. e., y_2, y_3), which are low-level controlled to follow their respective virtual systems

A distributed SMMS approach was also proposed in [43.99], in VIDEO 71 and VIDEO 72, to enable a single human user to telecontrol some of the slave robots. The inter-slave behaviors are encoded via a distributed artificial potential constructed on a time-invariant undirected information graph (Fig. 43.21). By using kinematic virtual systems to abstract the slave robots, the work achieves the combination of master-passivity and slave-stability. It guarantees no collisions among the slave robots or with obstacles and no separations among the slave robots.

A control approach for MMSS telerobotic systems was proposed in [43.107] and in VIDEO 75, where two human users telecontrol distinct frames on a single large-DOF slave robot. The velocity space of the slave is decomposed according to the two command motions, resolving conflicts between and constraints imposed on them. Priority is given to the primary user's commands

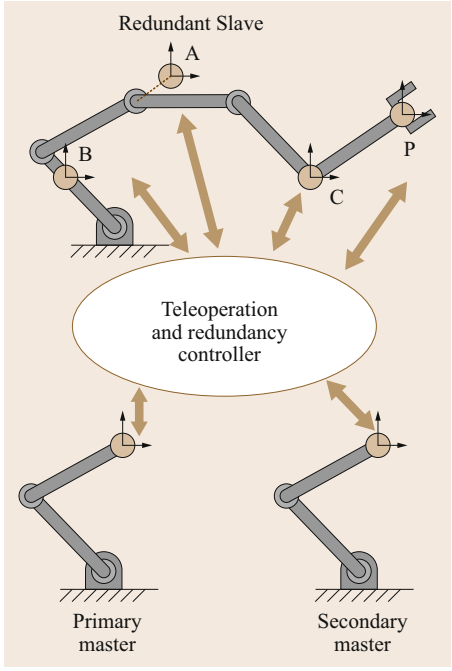


Fig. 43.22 Trilateral teleoperation (after [43.107]), where the primary master controls the end-effector frame P, while the secondary master controls task-space frame A, B or C: if A is chosen, primary and secondary tasks are nonconflicting/nonconstrained; if B is chosen, nonconflicting, yet, the secondary task is constrained; if C is chosen, conflicting, thus, prioritized nullspace control is necessary between primary and secondary tasks

(Fig. 43.22). The kinematics-level prioritized velocity commands are realized by dynamics-level adaptive control for all master and slave robots.

A different shared trilateral MMSS teleoperation framework was developedin [43.108], where two human users teleoperate the *same* point of a single slave

robot. Their control authority is adjusted with a dominance factor $\alpha \in [0, 1]$ set according to the task objective (e.g., $\alpha = 1$ for training or $\alpha = 0$ for evaluation). The MMSS system is optimized for a measure of transparency and Llewellyn's criteria is applied to the equivalent two-port system (with environment impedance Z_e embedded) to establish unconditional stability.

These multilateral and the previous mobile telerobotic approaches are promising to significantly expand the utility and application horizons of conventional telerobotics. Significant technical challenges and research questions remain, for example:

1. How to assign and determine the roles for multiple coordinating human users?
2. How to systematically split control tasks into teleoperative and autonomous control?
3. What are the most suitable performance measures, which are likely different from the conventional metrics?
4. What is the best form of human interface (or haptic feedback) [43.100] for the mobile and multilateral telerobotics and how to complement the interface with other modality (e.g., visual feedback).

43.6 Conclusions and Further Reading

Despite its age, telerobotics remains an exciting and vibrant area of robotics. In many ways, it forms a platform which can utilize the advances in robotic technologies while simultaneously leveraging the proven skills and capabilities of human users. Compare this, for example, with the development of the automobile and its relation to the driver. As cars are gradually becoming more sophisticated with added electronic stability control and navigation systems, they are becoming safer and more useful to their operators, not replacing them. Similarly telerobotics serves as a pathway for gradual progress and, as such, is perhaps best suited to fulfill robotics long-held promise of improving human life. It is seeing use in the challenging area of search and rescue. And with the recent developments and commercializations in telerobotic surgery systems, it is indeed impacting on the lives of tens of thousands of patients in a profound fashion and extending the reach of robotics into our world.

For further reading in the area of supervisory control, we refer to *Sheridan* [43.39]. Though published in 1992, it remains the most complete discussion on the topic. Unfortunately few other books are devoted to or even fully discuss telerobotics. In [43.109] many recent advances, including methods, experiments, applications, and developments, are collected. Beyond this, in the areas of bilateral and shared control, as well as to understand the various applications, we can only refer to the citations provided. Finally, in addition to the standard robotics journals, we note in particular *Presence: Teleoperators and Virtual Environments*, published by the MIT Press. Combined with virtual-reality applications, it focuses on technologies with a human operator.

Video-References

VIDEO 71 Semi-autonomous teleoperation of multiple UAVs: Passing a narrow gap
available from http://handbookofrobotics.org/view-chapter/43/videodetails/71

VIDEO 72 Semi-autonomous teleoperation of multiple UAVs: Tumbing over obstacle
available from http://handbookofrobotics.org/view-chapter/43/videodetails/72

VIDEO 73 Bilateral teleoperation of multiple quadrotors with time-varying topology
available from http://handbookofrobotics.org/view-chapter/43/videodetails/73

VIDEO 74 Passive teleoperation of nonlinear telerobot with tool-dynamics rendering
available from http://handbookofrobotics.org/view-chapter/43/videodetails/74

VIDEO 75 Asymmetric teleoperation of dual-arm mobile manipulator
available from http://handbookofrobotics.org/view-chapter/43/videodetails/75

VIDEO 297 Tele-existence master–slave system for remote manipulation
available from http://handbookofrobotics.org/view-chapter/43/videodetails/297

VIDEO 298 JPL dual-arm telerobot system
available from http://handbookofrobotics.org/view-chapter/43/videodetails/298

VIDEO 299 Single and dual arm supervisory and shared control
available from http://handbookofrobotics.org/view-chapter/43/videodetails/299

VIDEO 318 Teleoperated humanoid robot – HRP
available from http://handbookofrobotics.org/view-chapter/43/videodetails/318

VIDEO 319 Teleoperated humanoid robot – HRP: Tele-driving of lifting vehicle
available from http://handbookofrobotics.org/view-chapter/43/videodetails/319

VIDEO 321 Multi-modal multi-user telepresence and teleaction system
available from http://handbookofrobotics.org/view-chapter/43/videodetails/321

VIDEO 322 Laparoscopic telesurgery workstation
available from http://handbookofrobotics.org/view-chapter/43/videodetails/322

VIDEO 724 Passivity of IPC strategy at 30 Hz sample rate
available from http://handbookofrobotics.org/view-chapter/43/videodetails/724

References

43.1 M. Buss, G. Schmidt: Control problems in multi-modal telepresence systems. In: *Advances in Control*, ed. by P.M. Frank (Springer, London 1999) pp. 65–101

43.2 W.R. Ferell, T.B. Sheridan: Supervisory control of remote manipulation, IEEE Spectrum **4**(10), 81–88 (1967)

43.3 R.C. Goertz: Fundamentals of general-purpose remote manipulators, Nucleonics **10**(11), 36–42 (1952)

43.4 R.C. Goertz: Mechanical master–slave manipulator, Nucleonics **12**(11), 45–46 (1954)

43.5 R.C. Goertz, F. Bevilacqua: A force-reflecting positional servomechanism, Nucleonics **10**(11), 43–45 (1952)

43.6 W.R. Ferell: Remote manipulation with transmission delay, IEEE Trans. Hum. Factors Electron. **6**, 24–32 (1965)

43.7 T.B. Sheridan, W.R. Ferell: Remote manipulative control with transmission delay, IEEE Trans. Hum. Factors Electron. **4**, 25–29 (1963)

43.8 F. Miyazaki, S. Matsubayashi, T. Yoshimi, S. Arimoto: A new control methodology towards advanced teleoperation of master–slave robot systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 997–1002

43.9 R.J. Anderson, M.W. Spong: Asymptotic stability for force reflecting teleoperators with time delay, Int. J. Robotics Res. **11**(2), 135–149 (1992)

43.10 G. Niemeyer, J.-J.E. Slotine: Stable adaptive teleoperation, IEEE J. Ocean. Eng. **16**(1), 152–162 (1991)

43.11 J.E. Colgate: Robust impedance shaping telemanipulation, IEEE Trans. Robotics Autom. **9**(4), 374–384 (1993)

43.12 B. Hannaford: A design framework for teleoperators with kinesthetic feedback, IEEE Trans. Robotics Autom. **5**(4), 426–434 (1989)

43.13 D.A. Lawrence: Stability and transparency in bilateral teleoperation, IEEE Trans. Robotics Autom. **9**(5), 624–637 (1993)

43.14 D. Kuban, H.L. Martin: An advanced remotely maintainable servomanipulator concept, Proc. 1984 Natl. Top. Meet. Robotics Remote Handl. Hostile Environ., Washington (1984)

43.15 J. Vertut, P. Coiffet: *Teleoperation and Robotics: Evolution and Development* (Kogan Page, London 1985)

43.16 J. Vertut: MA23M contained servo manipulator with television camera, PICA and PIADE telescopic supports, with computer-integrated control, Proc. 28th Remote Syst. Technol. Conf. (1980) pp. 13–19

43.17 J. Vertut, P. Coiffet: Bilateral servo manipulator MA23 in direct mode and via optimized computer control, Proc. 2nd Retote Manned Syst. Technol. Conf. (1977)

43.18 A.K. Bejczy: Towards advanced teleoperation in space, Prog. Astronaut. Aeronaut **161**, 107–138 (1994)

43.19 G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl: Sensor-based space robotics – ROTEX and its telerobotic features, IEEE Trans. Robotics Autom. **9**(5), 649–663 (1993)

43.20 T.H. Massie, J.K. Salisbury: The phantom haptic interface: A device for probing virtual objects, Proc. ASME Int. Mech. Eng. Congr. Exhib., Chicago (1994) pp. 295–302

43.21 P.S. Green, J.W. Hill, J.F. Jensen, A. Shah: Telepresence Surgery, IEEE Eng. Med. Bio. Mag. **14**(3), 324–329 (1995)

43.22 R.H. Taylor, J. Funda, B. Eldridge, S. Gomory, K. Gruben, D. LaRose, M. Talamini, L. Kavoussi, J. Anderson: A telerobotic assistant for laparoscopic surgery, IEEE Eng. Med. Bio. Mag. **14**(3), 279–288 (1995)

43.23 A.J. Madhani, G. Niemeyer, J.K. Salisbury: The black falcon: A teleoperated surgical instrument for minimally invasive surgery, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Victoria (1998) pp. 936–944

43.24 S. Charles, H. Das, T. Ohm, C. Boswell, G. Rodriguez, R. Steele, D. Istrate: Dexterity-enhanced telerobotic microsurgery, Proc. Int. Conf. Adv. Robotics (1997) pp. 5–10

43.25 G.S. Guthart, J.K. Salisbury: The Intuitive$^{\ddot{U}}$ telesurgery system: Overview and application, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 618–621

43.26 J.M. Sackier, Y. Wang: Robotically assisted laparoscopic surgery: From concept to development, Surg. Endosc. **8**(1), 63–66 (1994)

43.27 J. Marescaux, J. Leroy, F. Rubino, M. Vix, M. Simone, D. Mutter: Transcontinental robot assisted remote telesurgery: Feasibility and potential applications, Ann. Surg. **235**, 487–492 (2002)

43.28 G.H. Ballantyne: Robotic surgery, telerobotic surgery, telepresence, and telementoring – Review of early clinical results, Surg. Endosc. **16**(10), 1389–1402 (2002)

43.29 D.H. Birkett: Electromechanical instruments for endoscopic surgery, Minim. Invasive Ther. Allied Technol. **10**(6), 271–274 (2001)

43.30 J. Rosen, B. Hannaford: Doc at a distance, IEEE Spectrum **8**(10), 34–39 (2006)

43.31 A.M. Okamura: Methods for haptic feedback in teleoperated robot-assisted surgery, Ind. Robot **31**(6), 499–508 (2004)

43.32 T. Ortmaier, B. Deml, B. Kübler, G. Passig, D. Reintsema, U. Seibold: Robot assisted force feedback surgery, Springer Tracts Adv. Robotics **31**, 361–379 (2007)

43.33 B.M. Yamauchi: PackBot: A versatile platform for military robotics, Proc. SPIE **5422**, 228–237 (2004)

43.34 R.R. Murphy: Trial by fire [rescue robots], IEEE Robotics Autom. Mag. **11**(3), 50–61 (2004)

43.35 J. Wright, A. Trebi-Ollennu, F. Hartman, B. Cooper, S. Maxwell, J. Yen, J. Morrison: Driving a rover on mars using the rover sequencing and visualization program, Int. Conf. Instrumentation, Control Inf. Technol. (2005)

43.36 G. Hirzinger, K. Landzettel, D. Reintsema, C. Preusche, A. Albu-Schäffer, B. Rebele, M. Turk: ROKVISS – Robotics component verification on ISS, Proc. 8th Int. Symp. Artif. Intell. Robotics Autom. Space (iSAIRAS) (2005), Session2B

43.37 C. Preusche, D. Reintsema, K. Landzettel, G. Hirzinger: ROKVISS – Preliminary results for telepresence mode, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 4595–4601

43.38 G. Hirzinger, K. Landzettel, B. Brunner, M. Fischer, C. Preusche, D. Reintsema, A. Albu-Schäffer, G. Schreiber, M. Steinmetz: DLR's robotics technologies for on-orbit servicing, Adv. Robotics **18**(2), 139–174 (2004)

43.39 T.B. Sheridan: *Telerobotics, Automation and Human Supervisory Control* (MIT Press, Cambridge 1992)

43.40 G. Hirzinger, J. Heindl, K. Landzettel, B. Brunner: Multisensory shared autonomy – A key issue in the space robot technology experiment ROTEX, Proc. RSJ/IEEE Int. Conf. Intell. Robots Syst. (IROS) (1992) pp. 221–230

43.41 B. Brunner, K. Arbter, G. Hirzinger: Task directed programming of sensor based robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1080–1087

43.42 A.K. Bejczy, W.S. Kim: Predictive displays and shared compliance control for time-delayed telemanipulation, Proc. IEEE/RSJ Int. Workshop Intell. Robots Syst. (IROS) (1990) pp. 407–412

43.43 P. Backes, K. Tso: UMI: An interactive supervisory and shared control system for telerobotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1990) pp. 1096–1101

43.44 L. Conway, R. Volz, M. Walker: Tele-autonomous systems: Methods and architectures for intermingling autonomous and telerobotic technology, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 1121–1130

43.45 S. Hayati, S.T. Venkataraman: Design and implementation of a robot control system with traded and shared control capability, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 1310–1315

43.46 G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl: ROTEX – The first remotely controlled robot in space, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 2604–2611

43.47 W.B. Griffin, W.R. Provancher, M.R. Cutkosky: Feedback strategies for telemanipulation with shared control of object handling forces, Presence **14**(6), 720–731 (2005)

43.48 T. Ortmaier, M. GrÖger, D.H. Boehm, V. Falk, G. Hirzinger: Motion estimation in beating heart surgery, IEEE Trans. Biomed. Eng. **52**(10), 1729–1740 (2005)

43.49 L. Rosenberg: Virtual fixtures: Perceptual tools for telerobotic manipulation, Proc. IEEE Virtual Real. Int. Symp. (1993) pp. 76–82

43.50 J.J. Abbott, P. Marayong, A.M. Okamura: Haptic virtual fixtures for robot-assisted manipulation, Proc. 12th Int. Symp. Robotics Res. (2007) pp. 49–64

43.51 D.J. Lee, P.Y. Li: Passive bilateral control and tool dynamics rendering for nonlinear mechanical teleoperators, IEEE Trans. Robotics **21**(5), 936–951 (2005)

43.52 M.J. Massimino, T.B. Sheridan, J.B. Roseborough: One handed tracking in six degrees of freedom, Proc. IEEE Int. Conf. Syst. Man Cybern. (1989) pp. 498–503

43.53 A. Ruesch, A.Y. Mersha, S. Stramigioli, R. Carloni: Kinetic scrolling-based position mapping for haptic teleoperation of unmanned aerial vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 3116–3121

43.54 A. Casals, L. Munoz, J. Amat: Workspace deformation based teleoperation for the increase of movement precision, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 2824–2829

43.55 F. Conti, O. Khatib: Spanning large workspaces using small haptic devices, Proc. 1st Jt. Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleoperator Syst. (2005) pp. 183–188

43.56 R.W. Daniel, P.R. McAree: Fundamental limits of performance for force reflecting teleoperation, Int. J. Robotics Res. **17**(8), 811–830 (1998)

43.57 M.J. Massimino, T.B. Sheridan: Sensory substitution for force feedback in teleoperation, Presence Teleoperator Virtual Environ. **2**(4), 344–352 (1993)

43.58 D.A. Kontarinis, R.D. Howe: Tactile display of vibratory information in teleoperation and virtual environments, Presence Teleoperator Virtual Environ. **4**(4), 387–402 (1995)

43.59 D.A. Kontarinis, J.S. Son, W.J. Peine, R.D. Howe: A tactile shape sensing and display system for teleoperated manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 641–646

43.60 J.J. Gil, A. Avello, Á. Rubio, J. Flórez: Stability analysis of a 1 DOF haptic interface using the Routh–Hurwitz criterion, IEEE Trans. Control Syst. Technol. **12**(4), 583–588 (2004)

43.61 N. Hogan: Controlling impedance at the man/machine interface, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 1626–1631

43.62 R.J. Adams, B. Hannaford: Stable haptic interaction with virtual environments,

IEEE Trans. Robotics Autom. **15**(3), 465–474 (1999)
43.63 K. Hashtrudi-Zaad, S.E. Salcudean: Analysis of control architectures for teleoperation systems with impedance/admittance master and slave manipulators, Int. J. Robotics Res. **20**(6), 419–445 (2001)
43.64 Y. Yokokohji, T. Yoshikawa: Bilateral control of master-slave manipulators for ideal kinesthetic coupling – Formulation and experiment, IEEE Trans. Robotics Autom. **10**(5), 605–620 (1994)
43.65 K.B. Fite, J.E. Speich, M. Goldfarb: Transparency and stability robustness in two-channel bilateral telemanipulation, ASME J. Dyn. Syst. Meas. Control **123**(3), 400–407 (2001)
43.66 S.E. Salcudean, M. Zhu, W.-H. Zhu, K. Hashtrudi-Zaad: Transparent bilateral teleoperation under position and rate control, Int. J. Robotics Res. **19**(12), 1185–1202 (2000)
43.67 W.R. Ferrell: Remote manipulation with transmission delay, IEEE Trans. Hum. Factors Electron. **6**, 24–32 (1965)
43.68 T.B. Sheridan: Space teleoperation through time delay: Review and prognosis, IEEE Trans. Robotics Autom. **9**(5), 592–606 (1993)
43.69 A. Eusebi, C. Melchiorri: Force reflecting telemanipulators with time-delay: Stability analysis and control design, IEEE Trans. Robotics Autom. **14**(4), 635–640 (1998)
43.70 W.S. Kim, B. Hannaford, A.K. Bejczy: Force-reflection and shared compliant control in operating telemanipulators with time delays, IEEE Trans. Robotics Autom. **8**(2), 176–185 (1992)
43.71 K. Hashtrudi-Zaad, S.E. Salcudean: Transparency in time-delayed systems and the effect of local force feedback for transparent teleoperation, IEEE Trans. Robotics Autom. **18**(1), 108–114 (2002)
43.72 R. Oboe, P. Fiorini: A design and control environment for internet-based telerobotics, Int. J. Robotics Res. **17**(4), 433–449 (1998)
43.73 S. Munir, W.J. Book: Control techniques and programming issues for time delayed internet based teleoperation, ASME J. Dyn. Syst. Meas. Control **125**(2), 205–214 (2003)
43.74 S. Hirche, M. Buss: Transparent data reduction in networked telepresence and teleaction systems. Part II: Time-delayed communication, Presence Teleoperator Virtual Environ. **16**(5), 532–542 (2007)
43.75 R.J. Anderson, M.W. Spong: Bilateral control of tele-operators with time delay, IEEE Trans. Autom. Control **34**(5), 494–501 (1989)
43.76 G. Niemeyer, J.-J.E. Slotine: Telemanipulation with time delays, Int. J. Robotics Res. **23**(9), 873–890 (2004)
43.77 S. Stramigioli, A. van der Schaft, B. Maschke, C. Melchiorri: Geometric scattering in robotic telemanipulation, IEEE Trans. Robotics Autom. **18**(4), 588–596 (2002)
43.78 N.A. Tanner, G. Niemeyer: High-frequency acceleration feedback in wave variable telerobotics, IEEE/ASME Trans. Mechatron. **11**(2), 119–127 (2006)
43.79 S. Munir, W.J. Book: Internet-based teleoperation using wave variables with prediction, IEEE/ASME Trans. Mechatron. **7**(2), 124–133 (2002)
43.80 D.J. Lee, M.W. Spong: Passive bilateral teleoperation with constant time delay, IEEE Trans. Robotics **22**(2), 269–281 (2006)
43.81 E. Nuno, L. Basanez, R. Ortega, M.W. Spong: Position tracking for non-linear teleoperators with variable time delay, Int. J. Robotics Res. **28**(7), 895–910 (2009)
43.82 K. Huang, D.J. Lee: Consensus-based peer-to-peer control architecture for multiuser haptic interaction over the internet, IEEE Trans. Robotics **29**(2), 417–431 (2013)
43.83 K. Huang, D.J. Lee: Hybrid pd-based control framework for passive bilateral teleoperation over the Internet, Proc. IFAC World Congr. (2011) pp. 1064–1069
43.84 B. Hannaford, J.H. Ryu: Time domain passivity control of haptic interfaces, IEEE Trans. Robotics Autom. **18**(1), 1–10 (2002)
43.85 J.-H. Ryu, C. Preusche, B. Hannaford, G. Hirzinger: Time domain passivity control with reference energy following, IEEE Trans. Control Syst. Technol. **13**(5), 737–742 (2005)
43.86 J. Artigas, C. Preusche, G. Hirzinger: Time domain passivity-based telepresence with time delay, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 4205–4210
43.87 J. Ryu, C. Preusche: Stable bilateral control of teleoperators under time-varying communication delays: Time domain passivity approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 3508–3513
43.88 D.J. Lee, K. Huang: Passive-set-position-modulation framework for interactive robotic systems, IEEE Trans. Robotics Autom. **26**(2), 354–369 (2010)
43.89 J.P. Kim, J. Ryu: Robustly stable haptic interaction control using an energy-bounding algorithm, Int. J. Robotics Res. **29**(6), 666–679 (2010)
43.90 M.C.J. Franken, S. Stramigioli, S. Misra, S. Secchi, A. Macchelli: Bilateral telemanipulation with time delays: A two-layer approach combining passivity and transparency, IEEE Trans. Robotics **27**(4), 741–756 (2011)
43.91 V. Duindam, A. Macchelli, S. Stramigioli, H. Bruyninckx: *Modeling and Control of Complex Physical Systems* (Springer, Berlin, Heidelberg 2009)
43.92 S. Stramigioli: *Modeling and IPC Control of Interactive Mechanical Systems: A Coordinate-Free Approach*, Lecture Notes in Control and Information Sciences, Vol. 266 (Springer, London 2001)
43.93 S. Stramigioli, C. Secchi, A.J. Van der Schaft, C. Fantuzzi: Sampled Data Systems Passivity and Discrete Port-Hamiltonian Systems, IEEE Trans. Robotics **21**(4), 574–587 (2005)
43.94 D.J. Lee, O. Martinez-Palafox, M.W. Spong: Bilateral teleoperation of a wheeled mobile robot over delayed communication networks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 3298–3303

43.95 N. Chopra, M.W. Spong, R. Lozano: Synchronization of bilateral teleoperators with time delay, Automatica **44**, 2142–2148 (2008)

43.96 J.-J.E. Slotine, W. Li: On the adaptive control of robot manipulators, Int. J. Robotics Res. **6**(3), 49–59 (1987)

43.97 D.J. Lee, D. Xu: Feedback *r*-passivity of lagrangian systems for mobile robot teleoperation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 2118–2123

43.98 S. Stramigioli, R. Mahony, P. Corke: A novel approach to haptic tele-operation of aerial robot vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 5302–5308

43.99 D.J. Lee, A. Franchi, H.-I. Son, C. Ha, H.H. Bülthoff, P.R. Giordano: Semi-autonomous haptic teleoperation control architecture of multiple unmanned aerial vehicles, IEEE/ASME Trans. Mech. **18**, 1334–1345 (2013)

43.100 H.I. Son, A. Franchi, L.L. Chuang, J. Kim, H.H. Bülthoff, P.R. Giordano: Human-centered design and evaluation of haptic cueing for teleoperation of multiple mobile robots, IEEE Trans. Cybern. **43**(2), 597–609 (2013)

43.101 D.J. Lee, M.W. Spong: Bilateral teleoperation of multiple cooperative robots over delayed communication networks: Theory, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 362–367

43.102 P.F. Hokayem, M.W. Spong: Bilateral teleoperation: An historical survey, Automatica **42**, 2035–2057 (2006)

43.103 A. Franchi, C. Secchi, H.I. Son, H.H. Bülthoff, P.R. Giordano: Bilateral teleoperation of groups of mobile robots with time-varying topology, IEEE Trans. Robotics **28**(5), 1019–1033 (2012)

43.104 G. Hwang, H. Hashimoto: Development of a human-robot-shared controlled teletweezing system, IEEE Trans. Control Sys. Technol. **15**(5), 960–966 (2007)

43.105 E.J. Rodriguez-Seda, J.J. Troy, C.A. Erignac, P. Murray, D.M. Stipanovic, M.W. Spong: Bilateral teleoperation of multiple mobile agents: Coordinated motion and collision avoidance, IEEE Trans. Control Sys. Technol. **18**(4), 984–992 (2010)

43.106 D.J. Lee, P.Y. Li: Passive decomposition of multiple mechanical systems under motion coordination requirements, IEEE Trans. Autom. Control **58**(1), 230–235 (2013)

43.107 P. Malysz, S. Sirouspour: Trilateral teleoperation control of kinematically redundant robotic manipulators, Int. J. Robotics Res. **30**(13), 1643–1664 (2011)

43.108 B. Khademian, K. Hashtrudi-Zaad: Shared control architectures for haptic training: performance and coupled stability analysis, Int. J. Robotics Res. **30**(13), 1627–1642 (2011)

43.109 M. Ferre, M. Buss, R. Aracil, C. Melchiorri, C. Balague (Eds.): *Advances in Telerobotics*, Springer Tracts in Advanced Robotics, Vol. 31 (Springer, Berlin, Heidelberg 2007)

Multimedia Contents

44. Networked Robots

Dezhen Song, Ken Goldberg, Nak-Young Chong

As of 2013, almost all robots have access to computer networks that offer extensive computing, memory, and other resources that can dramatically improve performance. The underlying enabling framework is the focus of this chapter: networked robots. Networked robots trace their origin to telerobots or remotely controlled robots. Telerobots are widely used to explore undersea terrains and outer space, to defuse bombs and to clean up hazardous waste. Until 1994, telerobots were accessible only to trained and trusted experts through dedicated communication channels. This chapter will describe relevant network technology, the history of networked robots as it evolves from teleoperation to cloud robotics, properties of networked robots, how to build a networked robot, example systems. Later in the chapter, we focus on the recent progress on cloud robotics, and topics for future research.

44.1 **Overview and Background** 1109
44.2 **A Brief History** 1110
44.2.1 Networked Teleoperation 1110
44.2.2 Cloud Robotics and Automation .. 1111
44.3 **Communications and Networking** 1112
44.3.1 The Internet 1113
44.3.2 Wired Communication Links 1113
44.3.3 Wireless Links 1114
44.3.4 Video and Audio Transmission Standards 1114
44.4 **Properties of Networked Robots** 1115
44.4.1 Overall Structure 1115
44.4.2 Building a Networked Robot System 1116
44.4.3 State-Command Presentation 1117
44.4.4 Command Execution/State Generation 1119
44.4.5 Virtual Fixtures 1119
44.4.6 Collaborative Control and Crowd Sourcing 1120
44.5 **Cloud Robotics** 1121
44.5.1 Big Data 1122
44.5.2 Cloud Computing 1122
44.5.3 Collective Robot Learning 1123
44.5.4 Open Source and Open Access 1123
44.5.5 Crowdsourcing and Call Centers .. 1124
44.6 **Conclusion and Future Directions** 1125
Video-References 1126
References 1126

44.1 Overview and Background

As illustrated in Fig. 44.1, the field of networked robots locates at the intersection between two exciting fields: robotics and networking. Similarly, teleoperation (Chap. 43) and multiple mobile robot systems (Chap. 51) also find their overlaps in the intersection. The primary concerns of the teleoperation are stability and time delay. Multiple mobile robot systems concerns coordination and planning of autonomous robots and sensors communicating over local networks. *The subfield of Networked Robots focuses on the robot system architectures, interfaces, hardware, software, and applications that use networks (primarily the Internet/Cloud).*

By 2012, several hundred networked robots have been developed and put online for public use. Many papers have been published describing these systems and a book on this subject by *Goldberg* and *Siegwart* is available [44.1]. Updated information about new research and an archive/survey of networked robots is available on the website of the IEEE technical committee on networked robots, which fosters research in this area [44.2].

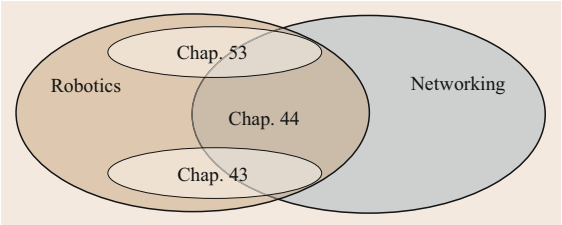


Fig. 44.1 Relationship between the subjects of networked robots (Chap. 44, this chapter), teleoperation (Chap. 43), and multiple mobile robot systems (Chap. 53) ►

The rest of the chapter is organized as follows: we first review the history and related work in Sect. 44.2. In Sect. 44.3, we review network and communication technology to provide necessary background for the following two main Sects. 44.4 and 44.5. Section 44.4 focuses on traditional networked robots while Sect. 44.5 summarize the new development in cloud robotics. Section 44.6, we conclude the chapter with recent applications and future directions.

44.2 A Brief History

Networked robots have their root in teleoperation systems, which started as remotely controlled devices. However, thanks to the recent evolution of the Internet and wireless networks, networked robots quickly expand their scope from the traditional master–slave teleopereation relationship to an integration of robots, human, agents, off-board sensors, databases, and clouds over the globe. To review the history of networked robots, we trace back to the root: remotely controlled devices.

44.2.1 Networked Teleoperation

Like many technologies, remotely controlled devices were first imagined in science fiction. In 1898, *Nicola Tesla* [44.3] demonstrated a radio-controlled boat in New York's Madison Square Garden. The first major experiments in teleoperation were motivated by the need to handle radioactive materials in the 1940s. *Goertz* and *Thompson* demonstrated one of the first bilateral simulators in the 1950s at the Argonne National Laboratory [44.4]. Remotely operated mechanisms have been designed for use in inhospitable environments such as undersea [44.5] and space exploration [44.6]. At General Electric, *Mosher* [44.7] developed a two-arm teleoperator with video cameras. Prosthetic hands were also applied to teleoperation [44.8]. More recently, teleoperation is being considered for medical diagnosis [44.9], manufacturing [44.10] and micromanipulation [44.11]. See Chap. 43 and the book from *Sheridan* [44.12] for excellent reviews on teleoperation and telerobotics research.

The concept of hypertext (linked references) was proposed by Vannevar Bush in 1945 and was made possible by the subsequent developments in computing and networking. In the early 1990s, Berners-Lee introduced the hypertext transmission protocol (HTTP). A group of students led by Marc Andreessen developed an open source version of the first graphical user interface, the *Mosaic* browser, and put it online in 1993. The first networked camera, the predecessor of today's *webcam*, went online in November 1993 [44.13]

Approximately nine months later, the first networked telerobot went online. The *Mercury Project* combined an IBM industrial robot arm with a digital camera and used the robot's air nozzle to allow remote users to excavate for buried artifacts in a sandbox [44.14, 15]. Working independently, a team led by *Taylor* and *Trevelyan* at the University of Western Australia demonstrated a remotely controlled six-axis telerobot in September 1994 [44.16, 17]. These early projects pioneered a new field of networked telerobots. See [44.18–26] for other examples.

Networked telerobots are a special case of *supervisory control* telerobots, as proposed by *Sheridan* and his colleagues [44.12]. Under supervisory control, a local computer plays an active role in closing the feedback loop. Most networked robotics are type (c) supervisory control systems (Fig. 44.2).

Although a majority of networked telerobotic systems consist of a single human operator and a single robot [44.27–34], *Chong* et al. [44.35] propose a useful taxonomy: single operator single robot (SOSR), single operator multiple robot (SOMR) [44.36, 37], multiple operator single robot (MOSR), and multiple operator multiple robot (MOMR) [44.38, 39] (VIDEO 81, VIDEO 84). These frameworks greatly extend system architecture of networked robots. In fact, human operators can often be replaced with autonomous agents, off-board sensors, expert systems, and programmed logics, as demonstrated by *Xu* and *Song* [44.40] and *Sanders* et al. [44.41]. The extended networked connectivity also allows us to employ techniques such as crowd sourcing and collaborative control for demanding applications such as nature observation and environment monitoring [44.42, 43]. Hence, networked teler-

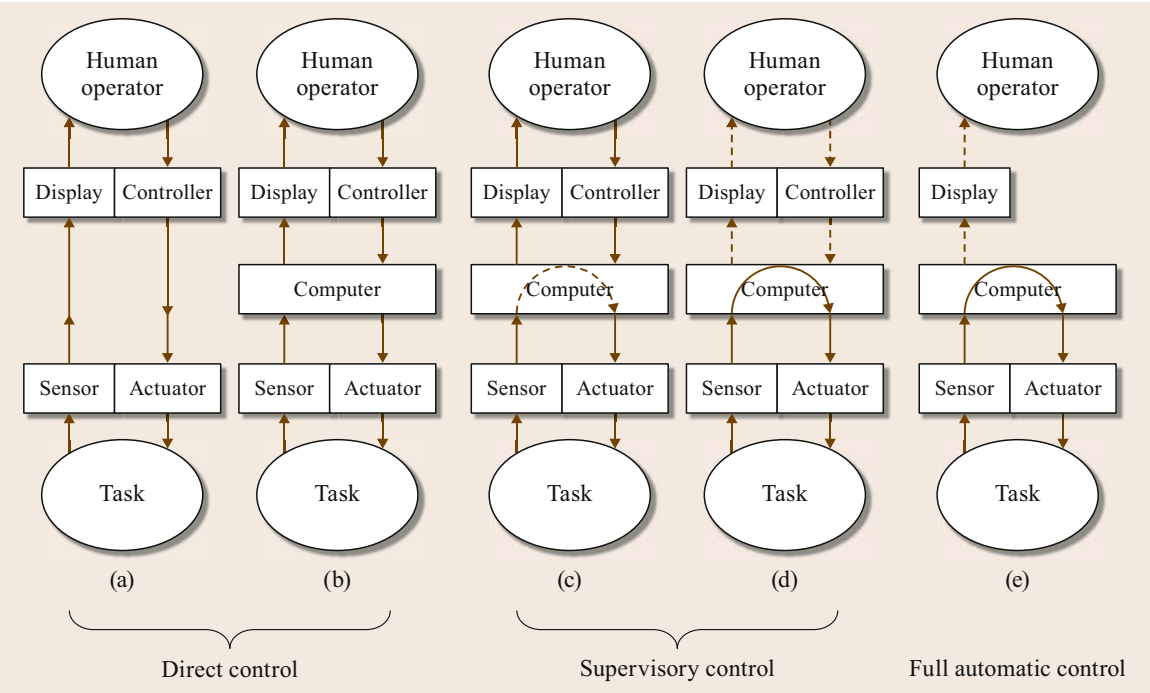


Fig. 44.2 A spectrum of teleoperation control modes adapted from *Sheridan*'s text (after [44.12]). We label them (a–e), in order of increasing robot autonomy. At the far left would be a mechanical linkage where the human directly operates the robot from another room through sliding mechanical bars, and on the far right is the system where the human role is limited to observation/monitoring. In (c–e), the dashed lines indicated that communication may be intermittent

obots fully evolute into networked robots: an integration of robots, humans [44.44], computing power, off-board sensing, and databases over the Internet.

The last 18 years (1994–2012) witnessed the extensive development in networked robots. New systems, new experiments, and new applications go well beyond traditional fields such as defense, space, and nuclear material handing [44.12] that motivated teleoperation in early 1950s. As the Internet introduces universal access to every corner of life, the impact of networked robots becomes broader and deeper in modern society. Recent applications range from education, industry, commercial, health care, geology, environmental monitoring, to entertainment, and arts.

Networked robots provide a new medium for people to interact with remote environment. A networked robot can provide more interactivity beyond what a normal videoconferencing system. The physical robot not only represents the remote person but also transmits multimodal feedback to the person, which is often referred as *telepresence* in the literature [44.30]. *Paulos* et al.'s Personal ROving Presence (PRoP) robot [44.45], *Jouppi* and *Thomas*' Surrogate robot [44.30], *Takayama* et al.'s Texai [44.46], and *Lazewatsky* and *Smart*'s inexpensive platform [44.47] are representative work.

Networked robots have great potential for education and training. In fact, one of the earliest networked telerobot systems [44.48] originates from the idea of a remote laboratory. Networked telerobots provide universal access to the general public, who may have little to no knowledge of robots, with opportunities to understand, learn, and operate robots, which were expensive scientific equipment limited to universities and large corporate laboratories before. Built on networked telerobots, online remote laboratories [44.49, 50] greatly improves distance learning by providing an interactive experience. For example, teleoperated telescopes help students to understand astronomy [44.51]. Teleoperated microscope [44.52] helps student to observe micro-organisms. The Tele-Actor project [44.53] allows a group of students to remotely control a human tele-actor to visit environments that are normally not accessible to them such as clean-room environments for semiconductor manufactory facility and deoxyribonucleic acid (DNA) analysis laboratories.

44.2.2 Cloud Robotics and Automation

Recent development of cloud computing provide new means and platform for networked robots. In

2010, *Kuffner* at Google introduced the term *Cloud robotics* [44.54] to describe a new approach to robotics that takes advantage of the Internet as a resource for massively parallel computation and real-time sharing of vast data resources. The Google autonomous driving project exemplifies this approach: the system indexes maps and images that are collected and updated by satellite, Streetview, and crowdsourcing from the network to facilitate accurate localization. Another example is Kiva Systems new approach to warehouse automation and logistics using large numbers of mobile platforms to move pallets using a local network to coordinate planforms and update tracking data. These are just two new projects that build on resources from the Cloud. Steve Cousins of Willow Garage aptly summarized the idea: *No robot is an island*. Cloud robotics recognizes the wide availability of networking, incorporates elements of open-source, open-access, and crowdsourcing to greatly extend earlier concepts of *Online Robots* [44.1] and *Networked Robots* [44.55, 56].

The Cloud has been used as a metaphor for the Internet since the inception of the World Wide Web in the early 1990s. As of 2012, researchers are pursuing a number of cloud robotics and automation projects [44.57, 58]. New resources range from software architectures [44.59–62] to computing resources [44.63]. The RoboEarth project [44.64] aims to develop [44.65]:

> *a World Wide Web for robots: a giant network and database repository where robots can share information and learn from each other about their behavior and their environment.*

Cloud robotics and automation is related to concepts of the *Internet of Things* [44.66] and the *Industrial Internet*, which envision how radio-frequency identification (RFID) and inexpensive processors can be incorporated into a vast array of objects from inventory items to household appliances to allow them to communicate and share information.

44.3 Communications and Networking

Below is a short review of relevant terminologies and technologies on networking. For details, see the texts by [44.67].

A communication network includes three elements: *links*, *routers/switchers*, and *hosts*. Links refer to the physical medium that carry bits from one place to another. Examples of links include copper or fiber-optic cables and wireless (radio frequency or infrared) channels. Switches and routers are hubs that direct digital information between links. Hosts are communication end points such as browsers, computers, and robots.

Networks can be based in one physical area (local-area network, or LAN), or distributed over wide distances (wide-area network, or WAN). Access control is a fundamental problem in networking. Among a variety of methods, the *ethernet* protocol is the most popular. Ethernet provides a broadcast-capable multi-access LAN. It adopts a carrier-sense multiple-access (CSMA) strategy to address the multiple-access problem. Defined in the IEEE 802.x standard, CSMA allows each host to send information over the link at any time. Therefore, collisions may happen between two or more simultaneous transmission requests. Collisions can be detected either by directly sensing the voltage in the case of wired networks, which is referred to as collision detection (CSMA/CD), or by checking the time-out of an anticipated acknowledgement in wireless networks, which is referred to as collision avoidance (CSMA/CA). If a collision is detected, both/all senders randomly back off a short period of time before retransmitting. CSMA has a number of important properties: (1) it is a completely decentralized approach, (2) it does not need clock synchronization over the entire network, and (3) it is very easy to implement. However, the disadvantages of CSMA are: (1) the efficiency of the network is not very high and (2) the transmission delay can change drastically.

As mentioned previously, LANs are interconnected with each other via routers/switchers. The information transmitted is in packet format. A packet is a string of bits and usually contains the source address, the destination address, content bits, and a checksum. Routers/switchers distribute packets according to their routing table. Routers/switchers have no memory of packets, which ensures scalability of the network. Packets are usually routed according to a first-in first-out (FIFO) rule, which is independent of the application. The packet formats and addresses are independent of the host technology, which ensures extensibility. This routing mechanism is referred to as packet switching in the networking literature. It is quite different from a traditional telephone network, which is referred to as circuit switching. A telephone network is designed to guarantee a dedicated circuit between a sender and a receiver once a phone call is established. The dedicated circuitry ensures communication quality. However, it requires a large number of circuits to ensure the quality of service (QOS), which leads to poor utilization of the overall network. A packet-switching network cannot guarantee dedicated bandwidth for each indi-

vidual pair of transmissions, but it improves overall resource utilization. The Internet, which is the most popular communication media and the infrastructure of networked telerobots, is a packet-switching network.

44.3.1 The Internet

The creation of the Internet can be traced back to US Department of Defense's (DOD) APRA NET network in the 1960s. There are two features of the APRA NET network that enabled the successful evolution of the Internet. One feature is the ability for information (packets) to be rerouted around failures. Originally, this was designed to ensure communication in the event of a nuclear war. Interestingly, this dynamic routing capability also allows the topology of the Internet to grow easily. The second important feature is the ability for heterogeneous networks to interconnect with one another. Heterogeneous networks, such as X.25, G.701, ethernet, can all connect to the Internet as long as they can implement the Internet protocol (IP). The IP is media, operating system (OS), and data rate independent. This flexible design allows a variety of applications and hosts to connect to the Internet as long as they can generate and understand IP.

Figure 44.3 illustrates a four-layer model of the protocols used in the Internet. On the top of the IP, we have two primary transport layer protocols: the transmission control protocol (TCP) and the user data protocol (UDP). TCP is an end-to-end transmission control protocol. It manages packet ordering, error control, rate control, and flow control based on packet round-trip time. TCP guarantees the arrival of each packet. However, excessive retransmission of TCP in a congested network may introduce undesirable time delays in a networked telerobotic system. UDP behaves differently; it is a broadcast-capable protocol and does not have a retransmission mechanism. Users must take care of error control and rate control themselves. UDP has a lot less overhead compared to TCP. UDP packets are transmitted at the sender's preset rate and the rate is changed based on the congestion of a network. UDP has great potential, but it is often blocked by firewalls because of a lack of a rate control mechanism. It is also worth mentioning that the widely accepted term TCP/IP refers to the family of protocols that build on IP, TCP, and UDP.

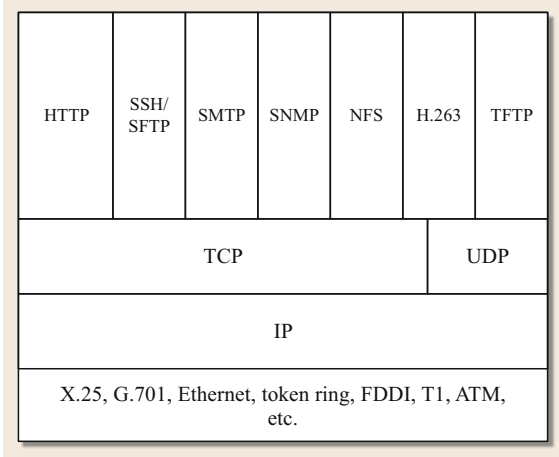


Fig. 44.3 A four-layer model of Internet protocols (after [44.67])

In the application layer of the Internet protocols, the HTTP is one of the most important protocols. HTTP is the protocol for the World Wide Web (WWW). It allows the sharing of multimedia information among heterogeneous hosts and OSs including text, image, audio, and video. The protocol has significantly contributed to the boom of the Internet. It also changes the traditional client/server (C/S) communication architecture to a browser/server (B/S) architecture. A typical configuration of the B/S architecture consists of a web server and clients with web browsers. The web server projects the contents in hypertext markup language (HTML) format or its variants, which is transmitted over the Internet using HTTP. User inputs can be acquired using the common gateway interface (CGI) or other variants. The B/S architecture is the most accessible because no specialized software is needed at the client end.

44.3.2 Wired Communication Links

Even during peak usage, the network backbones of the Internet often run at less than 30% of their overall capacity. The average backbone utilization is around 15–20%. The primary speed limitation for the Internet

Table 44.1 *Last-mile* Internet speed by wired connection type (if not specified, the downstream transmission and the upstream transmission share the same bandwidth)

Types	Bits per second
Dialup modem (V.92)	Up to 56 K
Integrated services digital network (ISDN)	64–160 K for BRI, up to 2048 K for PRI
High data rate digital subscriber line (HDSL)	Up to 2.3 M duplex on two twisted-pair lines
Asymmetric digital subscriber line (ADSL)	1.544–24.0 M downstream, 0.5–3.3 M upstream
Cable modem	2–400 M downstream, 0.4–108 M upstream
Fiber to the home (FTTH)	0.005–1 G downstream, 0.002–1 G upstream
Direct Internet II node	1.0–10.0 G

Table 44.2 Survey of wireless technologies in terms of bit rate and range

Types	Bit rate (bps)	Band (Hz)	Range (m)
Zigbee (802.15.4)	20–250 K	868–915 M/2.4 G	50
Bluetooth	732 K–3.0 M	2.4 G	100
3G HSPA	400 K–14.0 M	$\leq$ 3.5 G	N/A
HSPA+	5.76 M–44.0 M	$\leq$ 3.5 G	N/A
LTE	10 M–300 M	$\leq$ 3.5 G	N/A
WiFi (802.11a,b,g,n)	11–600 M	2.4 G/5 G	100

is the *last mile*, the link between clients and their local Internet service providers (ISP).

Table 44.1 lists typical bit rates for different connection types. It is interesting to note the asymmetric speeds in many cases, where upstream bit rate (from the client to the Internet), are far slower than downstream bit rates (from the Internet to the client). These asymmetries introduce complexity into the network model for teleoperation. Since the speed difference between the slowest modem link and the fastest Internet II node is over 10 000, designers of a networked telerobotic system should anticipate a large variance of communication speeds.

44.3.3 Wireless Links

Table 44.2 compares the speed, band, and range of wireless standards as of 2012. Increasing bit rate and communication range requires increasing power. The amount of radio frequency (RF) transmission power required over a distance d is proportional to d^k, where $2 \leq k \leq 4$ depending on the antenna type. In Table 44.2, Bluetooth and Zigbee are typical low-power transmission standards that are good for short distances. HSPA+ and LTE are commercially marketed as the 4G cellphone network.

By providing high-speed connectivity at low cost, WiFi is the most popular wireless standard in 2012. Its range is approximate 100 m line of sight and the WiFi wireless network usually consists of small-scale interconnected access points. The coverage range usually limits these networks to an office building, home, and other indoor environments. WiFi is a good option for indoor mobile robots and human operators. If the robot needs to navigate in the outdoor environment, the 3G or 4G cellphone network can provide the best coverage available. Although obvious overlap exists among wireless standards in coverage and bandwidth, there are two import issues that have not been covered by Table 44.2. One is mobility. We know that, if an RF source or receiver is moving, the corresponding Doppler effect causes a frequency shift, which could cause problems in communication. WiFi is not designed for fast-moving hosts. 3G HSPA cellphone allows the host to move at a vehicle speed under 120 km/h. However, LTE allows the host to move at a speed of 350 or 500 km/h, which even works for high-speed trains.

Long range wireless links often suffer from latency problem, which may drastically decreases system performer as discussed in Chap. 43. One may notice that we did not list satellite wireless in Table 44.2 because the long latency (0.5–1.7 s) and high price makes it difficult to be useful for robots. The large antenna size and high power consumption rate also limits its usage in mobile robots. In fact, the best option for long range wireless is LTE. LTE is designed with a transmission latency of less that 4 ms whereas 3G HSPA cellphone networks have a variable latency of 10–500 ms.

44.3.4 Video and Audio Transmission Standards

In networked robots systems, the representation of the remote environment is often needed to be delivered to online users in video and audio format. To deliver video and audio over the Internet, raw video and audio data from camera optical sensor and microphone must be compressed according to different video and audio compression standards to fit in the limited network bandwidth. Due to lack of bandwidth and computing power to encode streaming video, most early systems only transmit periodic snapshots of the remote scene in JPEG format at limited frame rate, i. e., 1–2 frames per second or less. Audio was rarely considered in the early system design. The rudimentary video delivery methods in the early system were mostly implemented using HTML and JavaScript to reload the JPEG periodically.

Today, the expansion of HTML standards allow web browsers to employ plug-ins as the client end of streaming video. HTML5 even natively supports video decoding. Therefore, the server end of recent systems often employs streaming server software, such as Adobe Flash Media Encoder, Apple Quick Time Streaming Server, Oracle Java Media Framework, Helix Media Delivery Platform, Microsoft DirectX, SkypeKit, etc., to encode and deliver video. These streaming video sever packages often provide easy-to-use software development kit (SDK) to facilitate system integration.

It is worth noting that these different software packages are just different implementations of video/audio

Table 44.3 A comparison of existing videostreaming standards for the same resolution under the same fixed bandwidth (FMBT represents buffering time settings that would not significantly decrease compression ratio or video quality.)

Standards	Feasible minimum buffering time (FMBT)	Framerate
MJPEG	Zero (< 10 ms)	Low
MPEG2	Variable (i. e., 50 ms – video length), 2–10 s are common	Moderate
H.263+	< 300 ms	High
H.264/ MPEG4-AVC	Zero (< 10 ms)	Highest

streaming protocols. Not every protocol is suitable for networked robots. Some protocols are designed to deliver video on demand while others are designed for live streaming for videoconferencing purposes. Networked robots use real-time video as feedback information, which imposes strict requirements in latency and bandwidth similar to those of videoconferencing. One way latency of more than 150 ms can significantly degrade telepresence and hence the performance of the human operator.

Latency is often caused by bandwidth and video encoding/decoding time. Since audio data amount is negligible when comparing to that video data. We will focus the discussion on video compression standards. There is always a tradeoff between frame rate and resolution for a given bandwidth. There is also a tradeoff between compression ratio and computation time for a given central processing unit (CPU). The computation time includes both CPU time and data-buffering time at both client and server ends. Video encoding is a very computationally intensive task. A long computation period introduces latency and significantly impair the system performance. It is possible to use hardware to cut down the computation time but not the data-buffering time, which are controlled by the video encoder.

There are many standards and protocols available but most of them are just variations of MJPEG, MPEG2, H.263, and MPEG4/AVC/H.264. We compare those standards in Table 44.3. Note that the comparison is qualitative and may not be the most accurate due to the fact that each video encoding standard has many parameters that affect the overall buffering time.

From networked robot point of view, the buffering time determines the latency and the frame rate determines the responsiveness of the system. An ideal videostream should have both high frame rate and low buffering time. But if both cannot be achieved at the same time, low latency is preferred. From Table 44.3, H.264/MPEG4-AVC clearly outperforms other competitors and is the most popular video compression method.

44.4 Properties of Networked Robots

Networked robots have the following properties:

- The physical world is affected by a device that is locally controlled by a network *server*, which connects to the Internet to communicate with remote human users, databases, agents, and off-board sensors, which are referred to as *clients* of the system.
- Human decision making capability is often an integral part of the system. If so, humans often access the robot via web browsers, such as Internet Explorer or Firefox, or apps in mobile device. As of 2012, the standard protocol for network browsers is the hypertext transfer protocol (HTTP), a stateless transmission protocol.
- Most networked robots are continuously accessible (online), 24 h a day, 7 days a week.
- Networks may be unreliable or have different speed for clients with different connections.
- Since hundreds of millions of people now have access to the Internet, mechanisms are needed to handle client authentication and contention. System security and privacy of users are important in the networked robots.
- Input and output for human users for networked robots are usually achieved with the standard computer screen, mouse, and keyboard.
- Clients may be inexperienced or malicious, so online tutorials and safeguards are generally required.
- Additional sensing, databases, and computing resources may be available over the network.

44.4.1 Overall Structure

As defined by *Mason*, *Peshkin*, and others [44.68, 69], in *quasistatic* robot systems, accelerations, and inertial forces are negligible compared to dissipative forces. In quasistatic robot systems, motions are often modeled as transitions between discrete atomic *configurations*.

We adopt a similar terminology for networked telerobots. In quasistatic telerobotics (QT), robot dynamics and stability are handled locally. After each atomic motion, a new state report is presented to the remote user,

who sends back an atomic command. The atomic state describes the status of the robot and its corresponding environment. Atomic commands refer to human directives, which are desired robotic actions.

Several issues arise:

- *State-command presentation*: How should state and available commands be presented to remote human operators using the two-dimensional (2-D) screen display?
- *Command execution/state generation*: How should commands be executed locally to ensure that the desired state is achieved and maintained by the robot?
- *Command coordination*: How should commands be resolved when there are multiple human operators and/or agents? How to synchronize and aggregate commands issued by users/agents with different network connectivity, background, responsiveness, error rate, etc., to achieve the best possible system performance?
- *Virtual Fixture: Error prevention and state correction*: How should the system prevent the wrong commands that may lead the robot to collision or other undesirable states?

Before we detail these issues, let us walk through how to build a minimum networked robot system.

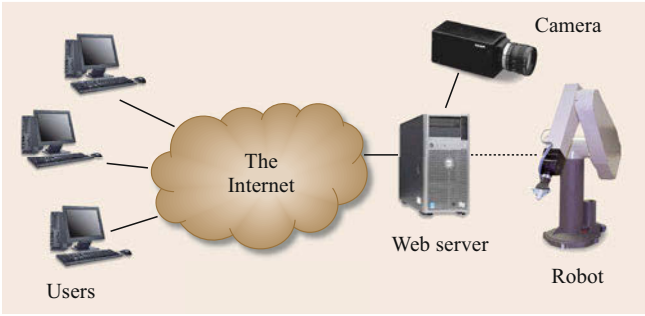


Fig. 44.4 Typical system architecture for a networked telerobot

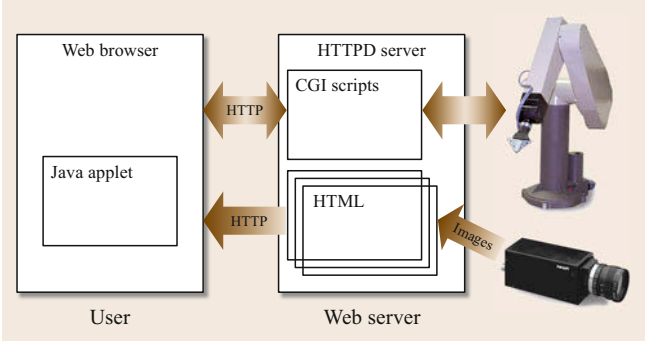


Fig. 44.5 A sample software architecture of a networked telerobot

A reader can follow the below example to build his/her owner networked robot system as well as understand challenges in the issues.

44.4.2 Building a Networked Robot System

This minimal system is a networked telerobotic system which allows a group of users to access a robot via web browsers. As illustrated in Fig. 44.4, a typical or minimal networked telerobotic system typically includes three components:

- *Users*: Anyone with an Internet connection and a web browser or equivalent apps that understand HTTP.
- *Web server*: A computer running a web server software.
- *Robot*: A robot manipulator, a mobile robot, or any device that can modify or affect its environment.

Users access the system via their web browsers. Any web browser that is compatible with W3C's HTML standard can access a web server. In 2012, the most popular web browsers are Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Apple Safari, and Opera. New browsers and updated versions with new features are introduced periodically. All of these popular browsers issue the corresponding mobile apps to support mobile devices such as Apple iPads, Apple iPhones, and Google Andriod-based Tablets and smart phones.

A web server is a computer that responds to HTTP requests over the Internet. Depending upon the operating system of the web server, popular server software packages include Apache and Microsoft Internet Information Services (IIS). Most servers can be freely downloaded from the Internet.

To develop a networked telerobot, one needs a basic knowledge of developing, configuring, and maintaining web servers. As illustrated in Fig. 44.5, the development requires knowledge of HTML and at least one local programming languages such as C, C#, CGI, Javascript, Perl, PHP, .Net, or Java.

It is important to consider compatibility with the variety of browsers. Although HTML is designed to be compatible with all browsers, there are exceptions. For example, Javascript, which is the embedded scripting language of web browsers, is not completely compatible between Internet Explorer and Firefox. One also needs to master the common HTML components such as forms that are used to accept user inputs, frames that are used to divide the interface into different functional regions, etc. An introduction to HTML can be found in [44.70].

User commands are usually processed by the web server using CGI, the common gateway interface. Most sophisticated methods such as PHP, Java Server Pages (JSP), and socket-based system programming can also be used. CGI is invoked by the HTTP server when the CGI script is referred in the uniform resource locator (URL). The CGI program then interprets the inputs, which is often the next robot motion command, and sends commands to the robot via a local communication channel. CGI scripts can be written in almost any programming language. The most popular ones are Perl and C.

A simple networked telerobotic system can be constructed using only HTML and CGI. However, if the robot requires a sophisticated control interface, advanced plug-ins such as Java Applet, Silver Light, or Flash, are recommended. These plug-ins run inside the web browser on the client's computer. Information about these plug-ins can be found at home pages of Oracle, Microsoft, and Adobe, respectively. Java applet is highly recommended because it is the most widely supported by different browsers. Recently, the fast adoption of HTML5 also provide a new long term solution to solve the compatibility issue.

Most telerobotic systems also collect user data and robot data. Therefore, database design and data processing program are also needed. The most common used databases include MySQL and PostgresSQL. Both are open-source databases and support a variety of platforms and operation systems. Since a networked telerobotic system is online 24 h a day, reliability is also an important consideration in system design. Website security is critical. Other common auxiliary developments include online documentation, online manual, and user feedback collection.

It is not difficult to expand this minimal networked telerobotic system into a full-fledged networked robot system. For example, some users can be replaced by agents that runs 24 h a day and 7 days a week to monitor system states and co-perform tasks with humans or take over the system when nobody is online. These agents can be implemented using cloud computing. Such extensions are usually based on the need of the task.

44.4.3 State-Command Presentation

To generate a correct and high-quality command depends on how effectively the human operator understands the state feedback. The state-command presentation contains three subproblems: the 2-D representation of the true robot state (state display), the assistance provided by the interface to generate new commands (spatial reasoning), and the input mechanism.

State Displays

Unlike traditional point-to-point teleoperation, where specialized training and equipment are available to operators, networked telerobots offer wide access to the general public. Designers cannot assume that operators

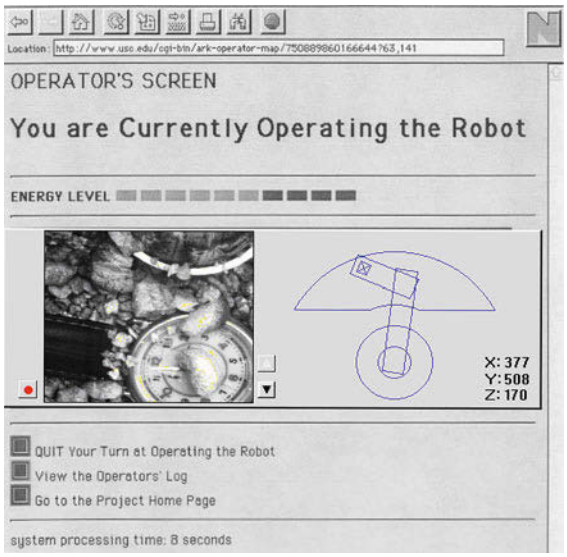


Fig. 44.6 Browser's view of the first networked telerobot interface (after [44.55]). The schematic at *lower right* gives an overhead view of position of the four-axis robot arm (with the camera at the end marked with X), and the image at the *lower left* indicates the current view of the camera. The *small button marked with a dot at the left* directs a 1 s burst of compressed air into the sand below the camera. The Mercury Project was online from August 1994 to March 1995

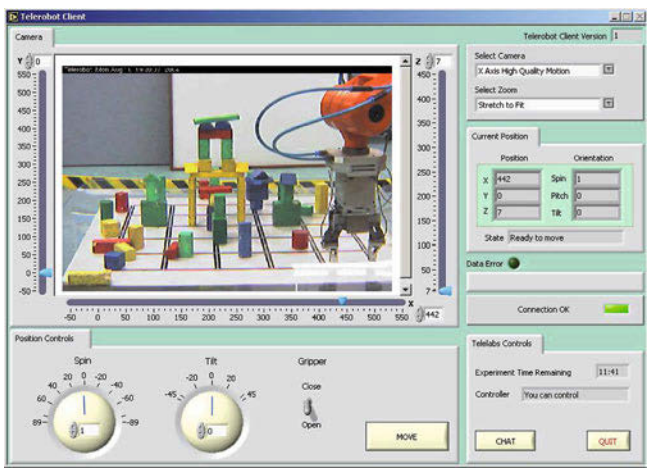


Fig. 44.7 Browser interface to the Australian networked telerobot which was a six-axis arm that could pick up and move blocks (after [44.17])

have any prior experience with robots. As illustrated in Fig.44.6, networked telerobotic systems must display the robot state on a 2-D screen display.

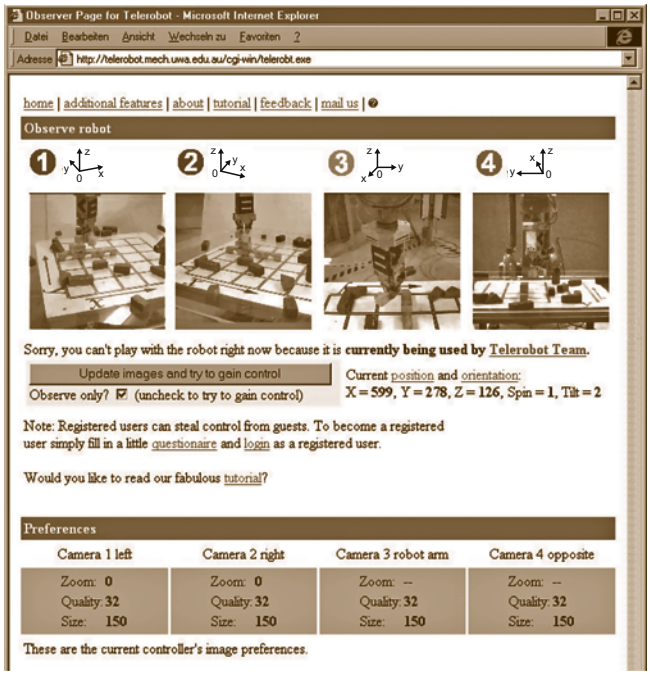


Fig. 44.8 Use of a multicamera system for multiviewpoint state feedback (after [44.71])

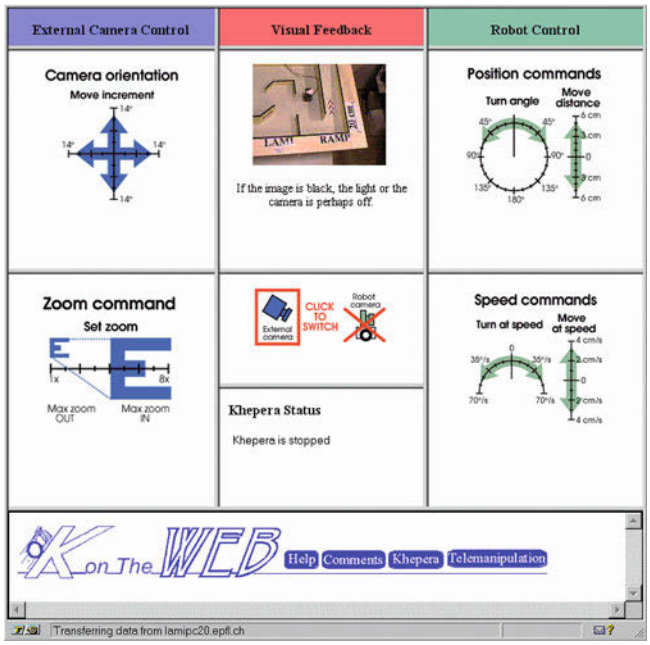


Fig. 44.9 Camera control and mobile robot control in Patrick Saucy and Francesco Mondada's Khep on the web project

The states of the teleoperated robot are often characterized in either world coordinates or robot joint configuration, which are either displayed in numerical format or through a graphical representation. Figure 44.6 lists robot *XYZ* coordinates on the interface and draws a simple 2-D projection to indicate joint configurations. Figure 44.7 illustrates another example of teleoperation interface that was developed by *Taylor* and *Trevelyan* [44.48]. In this interface, *XYZ* coordinates are presented in a sliding bar near the video window.

The state of the robot is usually displayed in a 2-D view as shown in Figs. 44.6 and 44.7. In some systems, multiple cameras can help the human operator to understand the spatial relationship between the robot and the objects in the surrounding environment. Figure 44.8 shows an example with four distinct camera views for a six-degree-of-freedom industrial robot.

Figure 44.9 demonstrate an interface with a pan–tilt–zoom robotic camera. The interface in Fig. 44.9 is designed for a mobile robot.

More sophisticated spatial reasoning can eliminate the need for humans to provide low-level control by automatically generating a sequence of commands after it receives task-level commands from the human operator. This is particularly important when the robotic system is highly dynamic and requires a very fast response. In this case, it is impossible to ask the human to generate intermediate steps in the robot control; for example, *Belousov* et al. adopt a shared autonomy model to direct a robot to capture a moving rod [44.28] as shown in Fig. 44.10. *Fong* and *Thorpe* [44.72] summarize vehicle teleoperation systems that utilize these supervisory control techniques. *Su* and *Luo* developed an incremental algorithm for better translation of the intention and motion of operators into remote robot action commands [44.33].

The fast development of sensing and display technology makes it possible to visualize robot and environment states in three-dimensional (3-D) displays or generate synthetic eco-centric views (a.k.a. third person views) (VIDEO 82(. To achieve that, it often requires the robot to be equipped with multiple cameras and laser range finders to quickly reconstruct the remote environment [44.73, 74]. Sometimes, the reconstructed sensory information can be superimposed on priorly known 3-D information to form an augmented reality. This kind of display can drastically increase telepresence and performance.

Human Operator Input

Most networked telerobotic systems only rely on mouses and keyboards for input. The design problem is what to click on in the interface. Given the fact that user commands can be quite different, we need to adopt

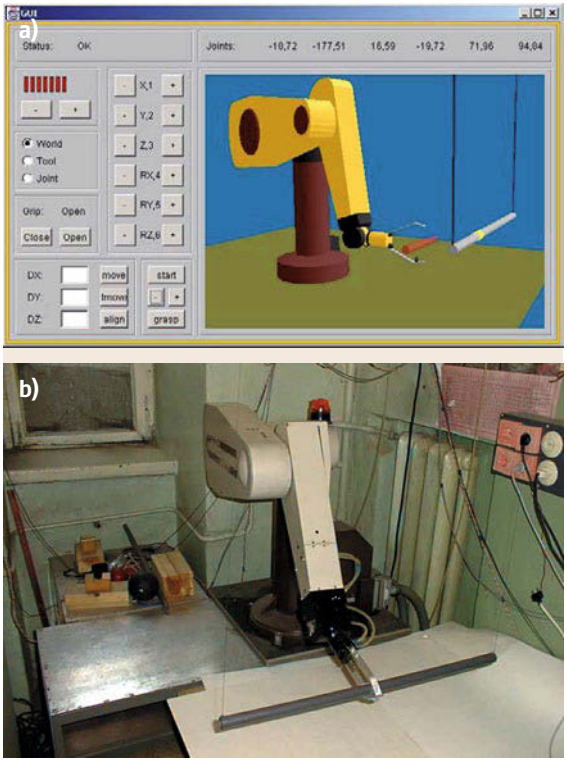


Fig.44.10a,b A web-based teleoperation system that allows a robot to capture a fast-moving rod. **(a)** User interface, and **(b)** system setup (after [44.28])

an appropriate interface for inputs; for example, inputs could be Cartesian *XYZ* coordinates in world coordinate system or robot configurations in angular joint configurations.

For angular inputs, it is often suggested to use a round dial as a control interface, as illustrated in bottom left of Fig. 44.7 and the right-hand side of Fig. 44.9. For linear motion in Cartesian coordinate, arrows operated by either mouse clicks or the keyboard are often suggested. Position and speed controls are often needed, as illustrated in Fig. 44.9. Speed control is usually controlled by mouse clicks on a linear progress bar for translation and a dial for rotation.

The most common control type is the position control. The most straightforward way is to click on the video image directly. To implement the function, the software needs to translate the 2-D click inputs into three-dimensional (3-D) world coordinates. To simplify the problem, the system designer usually assumes that the clicked position is on a fixed plane; for example, a mouse click on the interface of Fig. 44.6 assumes the robot moves on the *X*–*Y* plane. The combination of a mouse click on the image can also allow abstract task-level command. The example in Fig. 44.12 uses mouse clicks to place votes on an image to generate a command that directs a robot to pick up a test agent at the task level.

44.4.4 Command Execution/State Generation

When a robot receives a command, it executes the command and a new state is generated and transmitted back to the human operator. However, commands may not arrive in time or may get lost in transmission. Also, because users are often inexperienced, their commands may contain errors. Over the limited communication channel, it is impossible to ask the human to control the manipulator directly. Computer vision, laser range finder, local intelligence, and augmented-reality-based displays [44.74] are required to assist the human operator.

Belousov et al. demonstrated a system that allowed a web user to capture a fast rod that is thrown at a robot manipulator [44.28]. The rod is on bifilar suspension, performing complicated oscillations. Belousov et al. designed a shared-autonomy control to implement the capture. First, an operator chooses the desired point for capture on the rod and the capture instant using a 3-D online virtual model of the robot and the rod. Then, the capturing operation is performed automatically using a motion prediction algorithm that is based on the rod's motion model and two orthogonal camera inputs, which perceive the rod's position locally in real time.

This *shared autonomy* approach is often required when the task execution requires much faster response than the Internet can allow. Human commands have to remain at the task level instead of directing the movements of every actuators. The root of this approach can be traced back to the *Tele-Autonomous* concept proposed by *Conway* et al. [44.75] in 1990. In this paper, two important notions including time clutch and position clutches are introduced to illustrate the shared autonomy approach. The time clutch disengages the time synchronization between the human operator and the robot. The human operator verifies his/her commands on a predictive display before sending a set of verified commands to remote robots. The robot can then optimize the intermediate trajectory proposed by the human operator and disengage the position correspondence, which is referred to as the position clutch. Recent work [44.76] uses the similar idea to guide load-haul-dump vehicles in the underground mines by combining human inputs with tunnel following behavior.

44.4.5 Virtual Fixtures

Due to time delay, lack of background, and possible malicious behavior, human errors are inevitably introduced

to system from time to time. Erroneous states may be generated from the incorrect commands. If unchecked, robots or objects in the environment may be damaged. Sometimes, users may have good intention but are not able to generate accurate commands to control the robot remotely. For example, it is hard to generate a set of commands to direct a mobile robot to move along the wall and maintain a distance of 1 m to the wall at the same time.

Virtual fixtures are designed to cope with these challenges in teleoperation tasks. Proposed by *Rosenberg* [44.77], virtual fixtures are defined as an overlay of abstract sensory information on a robot workspace in order to improve the telepresence in a telemanipulation task. To further explain the definition, Rosenberg uses a ruler as an example. It is very difficult for a human to draw a straight line using bare hands. However, if a ruler, which is a physical fixture, is provided, then the task becomes easy. Similar to a physical fixture, a virtual fixture is designed to guide robot motion through some fictitious boundaries or force fields, such as virtual tubes or surface, generated according to sensory data. The virtual fixtures are often implemented using control laws [44.78, 79] based on a *virtual contact* model.

Virtual fixtures serve for two main purposes: avoiding operation mistakes and guide robots along the designable trajectories. This is also a type of shared autonomy that is similar to that in Sect. 44.4.4 where both the robot and the human share control in the system. Chapter 43 details the shared control scheme. It is worth noting that virtual fixtures should be visualized in the display to help operators understand the robot state to maintain situation awareness. This actually turns the display to *augmented reality* [44.80].

44.4.6 Collaborative Control and Crowd Sourcing

When more than one human is sharing control of the device, command coordination is needed. According to [44.81], multiple human operators can reduce the chance of errors, cope with malicious inputs, utilize operators' different expertise, and train new operators. In [44.82, 83], a collaboratively controlled networked robot is defined as a telerobot simultaneously controlled by many participants, where input from each participant is combined to generate a single control stream.

When group inputs are in the form of direction vectors, averaging can be used as an aggregation mechanism [44.84]. When decisions are distinct choices or at the abstract task level, voting is a better choice [44.53]. As illustrated in Fig. 44.11, *Goldberg* and *Song* (VIDEO 83) develop the Tele-Actor system using spatial dynamic voting. The Tele-Actor is a human equipped with an audio/video device and controlled by a group of online users. Users indicate their intensions by positioning their votes on a 320×320 pixel voting image during the voting interval. Votes are collected at the server and used to determine the Tele-Actor's next action based on the most requested region on the voting image [44.85].

Fig. 44.11 Spatial dynamic voting interface for the Tele-Actor system: the spatial dynamic voting (SDV) interface as viewed by each user. In the remote environment, the Tele-Actor takes images with a digital camera, which are transmitted over the network and displayed to all participants with a relevant question. With a mouse click, each user places a color-coded marker (a *votel* or voting element) on the image. Users view the position of all votels and can change their votel positions based on the group's response. Votel positions are then processed to identify a *consensus region* in the voting image that is sent back to the Tele-Actor. In this manner, the group collaborates to guide the actions of the Tele-Actor (after [44.53])

Another approach to collaboratively control a networked robot is the employ a optimization framework. *Song* et al. [44.86, 87] developed a collaboratively controlled camera that allowed many clients to share control of its camera parameters, as illustrated in Fig. 44.12. Users indicate the area they want to view by drawing rectangles on a panoramic image. The algorithm computes an optimal camera frame with respect to the user satisfaction function, which is defined as the frame selection problem [44.88, 89].

Recent work by *Xu* et al. [44.40, 90] further the optimization framework to *p*-frames that allows multiple cameras to be controlled and coordinated whereas

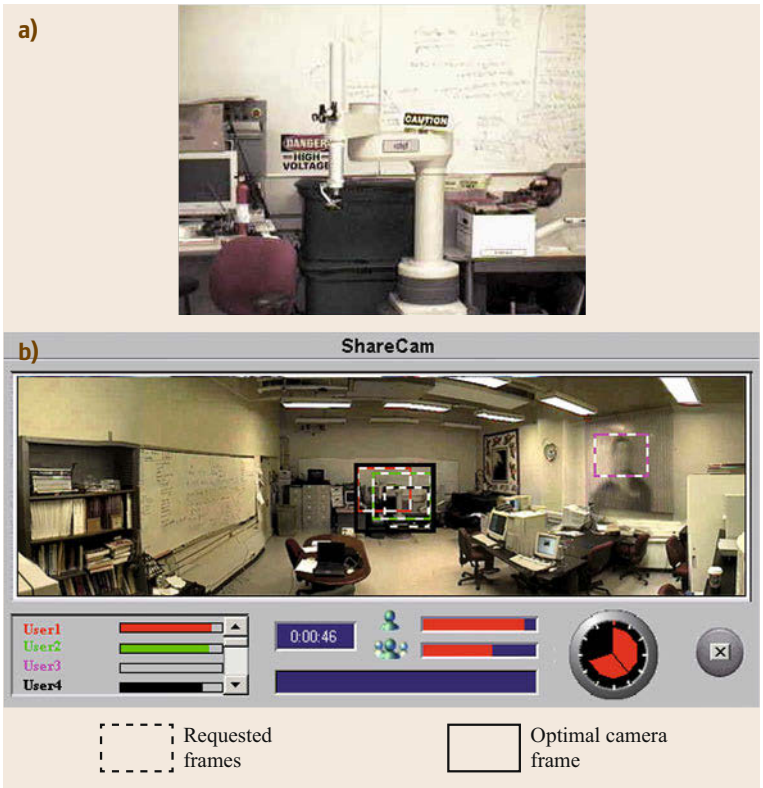


Fig.44.12a,b Frame selection interface (after [44.86]). The user interface includes two image windows. The lower window **(b)** displays a fixed panoramic image based on the camera's full workspace (reachable field of view). Each user requests a camera frame by positioning a *dashed rectangle* in **(b)**. Based on these requests, the algorithm computes an optimal camera frame (shown with a *solid rectangle*), moves the camera accordingly, and displays the resulting live streaming video image in the upper window **(a)**

human inputs can also be replaced by autonomous agents and other sensory inputs. These developments have been applied to a recent project, the Collaborative Observatory for Nature Environments (CONE) project [44.91], which aims to design a networked robotic camera system to collect data from the wilderness for natural scientists.

One important issue in collaborative control is the disconnection between individual commands and the robot action, which may lead to the loss of situation awareness, less participation, and eventual system failure. Inspired by engaging power in scoring systems in computer games, *Goldberg* et al. [44.92] design scoring mechanism for the collaborative control architecture by evaluating the individual leadership level. The early results show great improvement in the group performance. Furthermore, the recent development of social media, such as Blog and Twitter, can also be employed in the collaborative control to facilitate user interaction in real time, which can make the system more engaging and effective. The resulting new architecture can be viewed as a crowd sourcing [44.42, 93] type approach to networked robots that combines human recognition and decision making capabilities to robot execution at a different scale and depth than a regular teleoperation system.

44.5 Cloud Robotics

As noted earlier, the term *Cloud Robotics* is increasingly common based on advances in what is now called *Cloud Computing*. Cloud Robotics extends what were previously called *Online Robots* [44.1] and *Networked Robots* [44.55, 56]. Cloud computing provides robots with vast resources in computation, memory, programming.

Here we review five ways that cloud robotics and automation can potentially improve robots and automation performance:

1. Providing access to global libraries of images, maps, and object data, eventually annotated with geometry and mechanical properties.

2. Massively parallel computation on demand for demanding tasks like optimal motion planning and sample-based statistical modeling
3. Robot sharing of outcomes, trajectories, and dynamic control policies.
4. Human sharing of *open-source* code, data, and designs for programming, experimentation, and hardware construction.
5. On-demand human guidance (*call centers*) for exception handling and error recovery. Updated information and links are available at [44.94].

44.5.1 Big Data

The term *Big Data* describes data sets that are beyond the capabilities of standard relational database systems, which describes the growing library of images, maps, and many other forms of data relevant to robotics and automation on the Internet. One example is grasping, where online datasets can be consulted to determine appropriate grasps. The Columbia Grasp dataset [44.95] and the MIT KIT object dataset [44.96] are available online and have been widely used to evaluate grasping algorithms [44.97–100].

Related work explores how computer vision can be used with Cloud resources to incrementally learn grasp strategies [44.102, 103] by matching sensor data against 3-D computer-aided drafting (CAD) models in an online database. Examples of sensor data include 2-D image features [44.104], 3-D features [44.105], and 3-D point clouds [44.106]. Google Goggles [44.107], a free network-based image recognition service for mobile devices, has been incorporated into a system for robot grasping [44.101] as illustrated in Fig. 44.13.

Dalibard et al. attach *manuals* of manipulation tasks to objects [44.108]. The RoboEarch project stores data related to objects maps, and tasks, for applications ranging from object recognition to mobile navigation to grasping and manipulation (Fig. 44.14) [44.64].

As noted below, online datasets are effectively used to facilitate learning in computer vision. By leveraging Google's 3-D warehouse, [44.109] reduced the need for manually labeled training data. Using community photo collections, [44.110] created an augmented reality application with processing in the cloud.

44.5.2 Cloud Computing

As of 2012, Cloud Computing services like Amazon's EC2 elastic computing engine provide massively parallel computation on demand [44.111]. Examples include Amazon Web Services [44.112] Elastic Compute Cloud, known as EC2 [44.113], Google Compute Engine [44.114], Microsoft Azure [44.115]. These provide a large pool of computing resources that can be rented by the public for short-term computing tasks. These services were originally used primarily by web application developers, but have increasingly been used in scientific and technical high-performance computing (HPC) applications [44.116–119].

Cloud computing is challenging when there are real-time constraints [44.120]; this is an active area of research. However, there are many robotics applications that are not time sensitive such as decluttering a room or precomputing grasp strategies.

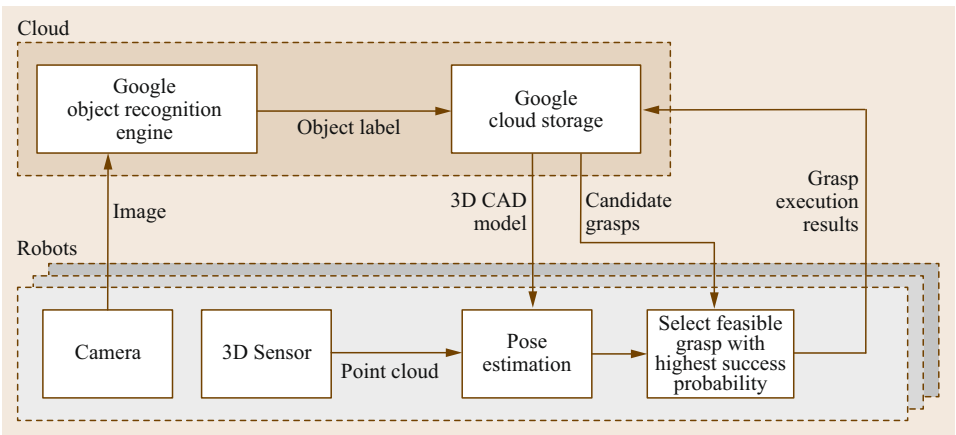


Fig. 44.13 System architecture for cloud-based object recognition for grasping. The robot captures an image of an object and sends via the network to the Google object recognition server. The server processes the image and returns data for a set of candidate objects, each with precomputed grasping options. The robot compares the returned CAD models with the detected point cloud to refine identification and to perform pose estimation, and selects an appropriate grasp. After the grasp is executed, data on the outcome is used to update models in the cloud for future reference (after [44.101])

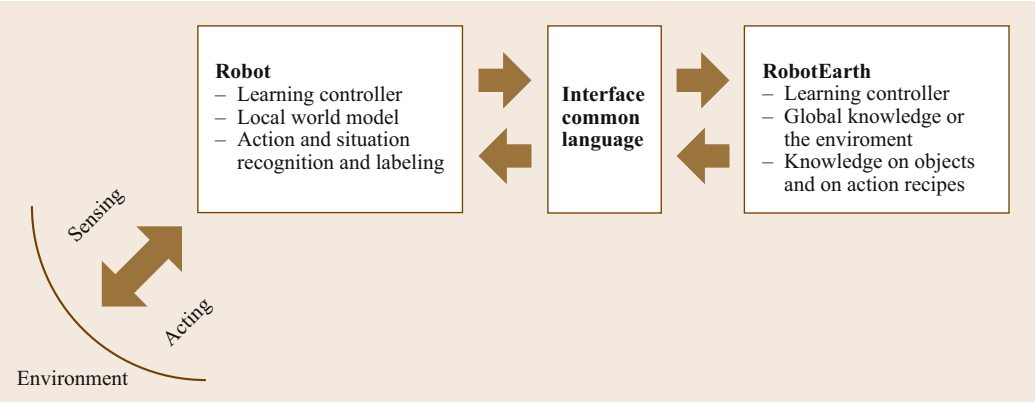


Fig. 44.14 RoboEarth architecture (after [44.64])

There are many sources of uncertainty in robotics and automation [44.121]. Cloud computing allows massive sampling over error distributions and Monte Carlo sampling is *embarrassingly parallel*; recent research in fields as varied as medicine [44.122] and particle physics [44.123] have taken advantage of the cloud. Real-time video and image analysis can be performed in the Cloud [44.109, 124, 125]. Image processing in the cloud has been used for assistive technology for the visually impaired [44.126] and for senior citizens [44.127]. Cloud computing is ideal for sample-based statistical motion planning under uncertainty, where it can be used to explore many possible perturbations in object and environment pose, shape, and robot response to sensors and commands [44.128]. Cloud-based sampling is also being investigated for grasping objects with shape uncertainty [44.129, 130] (Fig. 44.15). A grasp planning algorithm accepts as input a nominal polygonal outline with Gaussian uncertainty around each vertex and the center of mass to compute a grasp quality metric based on a lower bound on the probability of achieving force closure.

44.5.3 Collective Robot Learning

The Cloud allows robots and automation systems to *share* data from physical trials in a variety of environments, for example, initial and desired conditions, associated control policies and trajectories, and importantly: data on performance and outcomes. Such data are a rich source for robot learning.

One example is for path planning, where previously generated paths are adapted to similar environments [44.131] and grasp stability of finger contacts can be learned from previous grasps on an object [44.98].

The MyRobots project [44.132] from RobotShop proposes a *social network* for robots [44.133]:

> *In the same way humans benefit from socializing, collaborating and sharing, robots can benefit from those interactions too by sharing their sensor information giving insight on their perspective of their current state.*

44.5.4 Open Source and Open Access

The Cloud facilitates sharing by humans of designs for hardware, data, and code. The success of open-source software [44.134–136] is now widely accepted in the robotics and automation community. A primary example is ROS, the Robot Operating System, which provides libraries and tools to help software developers create robot applications [44.137, 138]. ROS has also been ported to Android devices [44.139]. ROS has be-

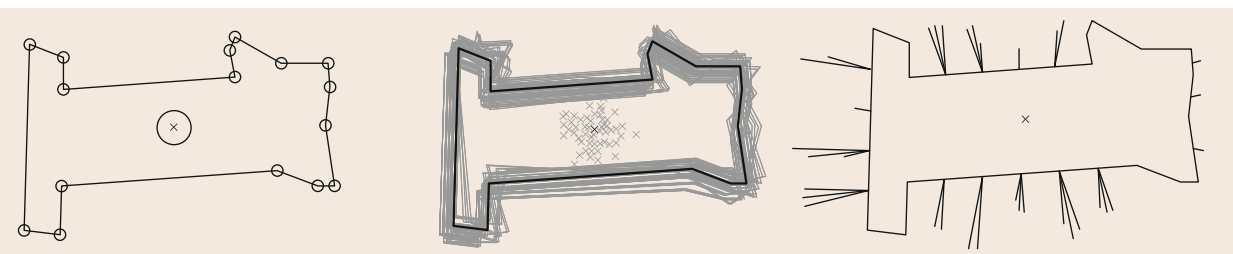


Fig. 44.15 A cloud-based approach to geometric shape uncertainty for grasping (after [44.129, 130])

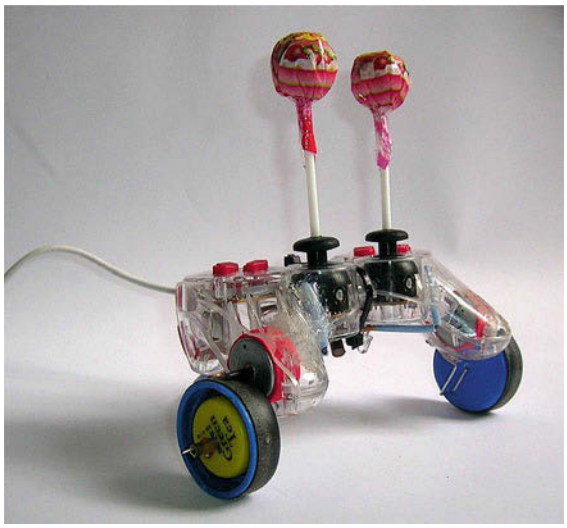


Fig. 44.16 Suckerbot, designed by Tom Tilley of Thailand, a winner of the $10 Robot Design Challenge (after [44.140])

come a standard akin to Linux and is now used by almost all robot developers in research and many in industry.

Additionally, many simulation libraries for robotics are now open-source, which allows students and researchers to rapidly set up and adapt new systems and share the resulting software. Open-source simulation libraries include Bullet [44.141], a physics simulator originally used for video games, OpenRAVE [44.142] and Gazebo [44.143], simulation environments geared specifically toward robotics, OOPSMP, a motion-planning library [44.144], and GraspIt!, a grasping simulator [44.145].

Another exciting trend is in open-source hardware, where CAD models and the technical details of construction of devices are made freely available [44.147, 148]. The Arduino project [44.149] is a widely used open-source microcontroller platform, and has been used in many robotics projects. The Raven [44.150] is an open-source laparoscopic surgery robot developed as a research platform an order of magnitude less expensive than commercial surgical robots [44.151].

The Cloud can also be used to facilitate open challenges and design competitions. For example, the African Robotics Network with support from IEEE Robotics and Automation Society hosted the *$10 Robot* Design Challenge in the summer of 2012. This open competition attracted 28 designs from around the world including a winning entry from Thailand (Fig. 44.16) that modified a surplus Sony game controller, adapting its embedded vibration motors to drive wheels and adding lollipops to the thumb switches as inertial counterweights for contact sensing, which can be built from surplus parts for US $8.96 [44.140].

44.5.5 Crowdsourcing and Call Centers

In contrast to automated telephone reservation and technical support systems, consider a future scenario where errors and exceptions are detected by robots and automation systems, which then access human guidance on-demand at remote call centers. Human skill, experience, and intution is being tapped to solve a number of problems such as image labeling for computer vision [44.54, 62, 102, 152]. Amazon's Mechanical Turk is pioneering on-demand *crowdsourcing* that can draw on *human computation* or *social computing systems*. Research projects are exploring how

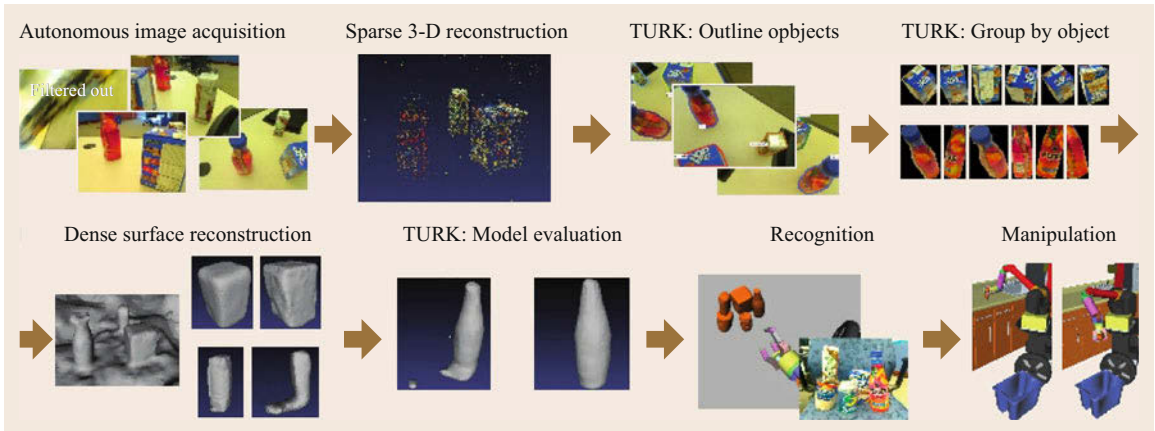


Fig. 44.17 A cloud robot system that incorporates Amazon's Mechanical Turk to *crowdsource* object identification to facilitate robot grasping (after [44.146])

this can be used for path planning [44.153], to determine depth layers, image normals, and symmetry from images [44.154], and to refine image segmentation [44.155]. Researchers are working to understand pricing models [44.156] and apply crowdsourcing to grasping [44.146] (Fig. 44.17).

44.6 Conclusion and Future Directions

As this technology matures, networked robots will gradually go beyond university laboratories and find application in the real world.

As mentioned earlier in Sects. 44.2.2 and 44.5, the new efforts in cloud robotics lead by Google and RoboEarth naturally bridge research and applications. The open source nature and ready-to-use APIs can quickly spread and deploy research results. Japan's Advanced Telecommunications Research Institute International (ATR) Intelligent Robotics and Communication Laboratory has also announced its networked robot project led by Norihiro Hagita (ATR). Its mission is to develop network-based intelligent robots for applications such as service, medical, and safety. Hideyuki Tokuda (Keio University) chaired the Networked Robot Forum in Spring 2005, which promotes research and development (R&D) and standardization on network robots through activities to support awareness campaigns and verification experiments in collaboration among wide-ranging parties, which includes over 100 industry and academic members. Korea's Ministry of Information and Communication has also announced the Ubiquitous Robotic Companion (URC) project to develop network-based intelligent robots.

Networked robots have allowed tens of thousands of nonspecialists around the world to interact with robots. The design of networked robots presents a number of engineering challenges to build reliable systems that can be operated by nonspecialists 24 h a day, 7 days a week, and remain online for years. Many new research challenges remain:

- *New interfaces*: As portable devices such as cellphones and tablet computers increase in computation power, networked robotics should be able to adopt them as new interfaces. As computers becomes increasingly powerful, they become capable of visualizing more sophisticated sensor inputs. Designers of new interfaces should also keep track of new developments in hardware such as haptic interfaces and voice recognition systems. New software standards such as flash, extensible markup language (XML), extensible hyper text markup language (XHTML), virtual reality modeling language (VRML), and wireless markup language (WML) will also change the way we design interface.
  New interface technology arises as human computer interaction technology, mobile computing, and computer graphics areas progress. Recent progresses on brain–machine interaction explore the possibility of using brain wave, such as electroencephalography (EEG) signals, to control robot movements for ground robots [44.157] and unmanned aerial vehicles (UAV)s [44.158]. Gesture [44.159] and multitouch [44.160] are also used to generate control commands. Unlike the traditional mouse and keyboard interfaces, the new interfaces facilitate more natural interaction but suffers from precision issues, because these methods have large noise and require more research efforts in improving robustness and accuracy.
- *New algorithms*: Algorithms determine performance. Scalable algorithms that are capable of handing large amounts of data such as video/sensor network inputs and utilize fast-evolving hardware capability such as distributed and parallel computation will become increasingly important in the networked robotics, especially in cloud robotics.
- *New protocols*: Although we have listed some pioneering work in changing the network environment to improve teleoperation, there are still a large number of open problems such as new protocols, appropriate bandwidth allocation [44.161], QoS [44.162], security, routing mechanisms [44.29, 163], and many more. Network communication is a very fast-evolving field. The incorporation/modification of network communication ideas into networked telerobotic system design will continue to be an active research area. The common object request broker architecture (CORBA) or real-time CORBA [44.20, 21, 39, 164, 165] have great potential for networked robots.
- *New performance metrics*: As more and more robots enter service, it is important to develop metrics to quantify the performance of the robot–human team. As we are more familiar with metrics developed to assess robot performance or task performance [44.162], recent progresses on using the robot to assess human performance [44.166,

167] shed light on new metrics. Standardizing these metrics will also be an important direction.

- *Video for robotics*: Another interesting observation is that all of existing video compression and transmission standards try to rebuild a true and complete representation of camera field of view. However, it might not be necessary or infeasible due to bandwidth limit for a networked robot [44.168]. Sometimes, a high level abstraction is sufficient. For example, when a mobile robot is avoiding an moving obstacle, all the robot needs to know is the speed and bounding box of the moving object instead of knowledge that whether this object is human or other robots. We might want to control the level of details in video perception and transmission. This actually imposes an interesting problem: we need a new streaming standard that serves for networked robots.
- *Applications*: Recent successful applications include environment monitoring [44.43, 169], manufacturing [44.170, 171], and infrastructure inspection and maintenance [44.172, 173]. The fast development of networked robot systems is worldwide. Many new applications are emerging in areas such as security, inspection, education, and entertainment. Application requirements such as reliability, security, and modularity will continue to pose new challenges for system design.

Video-References

VIDEO 81 A heterogeneous multiple-operator-multiple-robot system
available from http://handbookofrobotics.org/view-chapter/44/videodetails/81

VIDEO 82 Teleoperation of a mini-excavator
available from http://handbookofrobotics.org/view-chapter/44/videodetails/82

VIDEO 83 Tele-Actor
available from http://handbookofrobotics.org/view-chapter/44/videodetails/83

VIDEO 84 A multi-operator-multi-robot teleoperation system
available from http://handbookofrobotics.org/view-chapter/44/videodetails/84

References

44.1 K. Goldberg, R. Siegwart (Eds.): *Beyond Webcams: An Introduction to Online Robots* (MIT Press, Cambridge 2002)

44.2 IEEE Technical Committee on Networked Robots: http://tab.ieee-ras.org/

44.3 N. Tesla: Method of and apparatus for controlling mechanism of moving vessels or vehicles, US Patent 613809 A (1898)

44.4 R. Goertz, R. Thompson: Electronically controlled manipulator, Nucleonics **12**(11), 46–47 (1954)

44.5 R.D. Ballard: A last long look at titanic, Nat. Geogr. **170**(6), 698–727 (1986)

44.6 A.K. Bejczy: Sensors, controls, and man-machine interface for advanced teleoperation, Science **208**(4450), 1327–1335 (1980)

44.7 R.S. Mosher: Industrial manipulators, Sci. Am. **211**(4), 88–96 (1964)

44.8 R. Tomovic: On man-machine control, Automatica **5**, 401–404 (1969)

44.9 A. Bejczy, G. Bekey, R. Taylor, S. Rovetta: A research methodology for tele-surgery with time delays, 1st Int. Symp. Med. Robotics Comput. Assist. Surg. (MRCAS) (1994)

44.10 M. Gertz, D. Stewart, P. Khosla: A human-machine interface for distributed virtual laboratories, IEEE Robotics Autom. Mag. **1**, 5–13 (1994)

44.11 T. Sato, J. Ichikawa, M. Mitsuishi, Y. Hatamura: A new micro-teleoperation system employing a hand-held force feedback pencil, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994)

44.12 T.B. Sheridan: *Telerobotics, Automation, and Human Supervisory Control* (MIT Press, Cambridge 1992)

44.13 FirstWebcam: http://www.cl.cam.ac.uk/coffee/qsf/timeline.html. (1993)

44.14 K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, J. Wiegley: Robot teleoperation via WWW, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995)

44.15 K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, J. Wiegley: Beyond the web: Manipulating the physical world via the WWW, Comput. Netw. ISDN Syst. J. **28**(1), 209–219 (1995)

44.16 B. Dalton, K. Taylor: A framework for internet robotics, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998)

44.17 K. Taylor, J. Trevelyan: The telelabs project, http://telerobot.mech.uwa.edu.au (1994)

44.18 H. Hu, L. Yu, P.W. Tsui, Q. Zhou: Internet-based robotic systems for teleoperation, Assem. Automat. **21**(2), 143–151 (2001)

44.19 R. Safaric, M. Debevc, R. Parkin, S. Uran: Telerobotics experiments via internet, IEEE Trans. Ind. Electron. **48**(2), 424–431 (2001)

44.20 S. Jia, K. Takase: A corba-based internet robotic system, Adv. Robotics **15**(6), 663–673 (2001)

44.21 S. Jia, Y. Hada, G. Ye, K. Takase: Distributed telecare robotic systems using corba as a communication architecture, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

44.22 J. Kim, B. Choi, S. Park, K. Kim, S. Ko: Remote control system using real-time MPEG-4 streaming technology for mobile robot, IEEE Int. Conf. Consum. Electron. (2002)

44.23 T. Mirfakhrai, S. Payandeh: A delay prediction approach for teleoperation over the internet, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

44.24 K. Han, Y. Kim, J. Kim, S. Hsia: Internet control of personal robot between KAIST and UC Davis, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

44.25 L. Ngai, W.S. Newman, V. Liberatore: An experiment in internet-based, human-assisted robotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

44.26 R.C. Luo, T.M. Chen: Development of a multibehavior-based mobile robot for remote supervisory control through the internet, IEEE/ASME Trans. Mechatron. **5**(4), 376–385 (2000)

44.27 D. Aarno, S. Ekvall, D. Kragi: Adaptive virtual fixtures for machine-assisted teleoperation tasks, IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1151–1156

44.28 I. Belousov, S. Chebukov, V. Sazonov: Web-based teleoperation of the robot interacting with fast moving objects, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 685–690

44.29 Z. Cen, A. Goradia, M. Mutka, N. Xi, W. Fung, Y. Liu: Improving the operation efficiency of supermedia enhanced internet based teleoperation via an overlay network, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 691–696

44.30 N.P. Jouppi, S. Thomas: Telepresence systems with automatic preservation of user head height, local rotation, and remote translation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 62–68

44.31 B. Ricks, C.W. Nielsen, M.A. Goodrich: Ecological displays for robot interaction: A new perspective, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 3 (2004) pp. 2855–2860

44.32 D. Ryu, S. Kang, M. Kim, J. Song: Multi-modal user interface for teleoperation of ROBHAZ-DT2 field robot system, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 1 (2004) pp. 168–173

44.33 J. Su, Z. Luo: Incremental motion compression for telepresent walking subject to spatial constraints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 69–74

44.34 I. Toshima, S. Aoki: Effect of driving delay with an acoustical tele-presence robot, telehead, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 56–61

44.35 N. Chong, T. Kotoku, K. Ohba, K. Komoriya, N. Matsuhira, K. Tanie: Remote coordinated controls in multiple telerobot cooperation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (2000) pp. 3138–3343

44.36 P. Cheng, V. Kumar: An almost communicationless approach to task allocation for multiple unmanned aerial vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 1384–1389

44.37 X. Ding, M. Powers, M. Egerstedt, S. Young, T. Balch: Executive decision support, IEEE Robotics Autom. Mag. **16**(2), 73–81 (2009)

44.38 J. Liu, L. Sun, T. Chen, X. Huang, C. Zhao: Competitive multi-robot teleoperation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

44.39 Z. Zhang, Q. Cao, L. Zhang, C. Lo: A CORBA-based cooperative mobile robot system, Ind. Robot Int. J. **36**(1), 36–44 (2009)

44.40 Y. Xu, D. Song: Systems and algorithms for autonomous and scalable crowd surveillance using robotic PTZ cameras assisted by a wide-angle camera, Auton. Robots **29**(1), 53–66 (2010)

44.41 D. Sanders, J. Graham-Jones, A. Gegov: Improving ability of tele-operators to complete progressively more difficult mobile robot paths using simple expert systems and ultrasonic sensors, Ind. Robot Int. J. **37**(5), 431–440 (2010)

44.42 S. Faridani, B. Lee, S. Glasscock, J. Rappole, D. Song, K. Goldberg: A networked telerobotic observatory for collaborative remote observation of avian activity and range change, IFAC Workshop Netw. Robots (2009)

44.43 R. Bogue: Robots for monitoring the environment, Ind. Robot Int. J. **38**(6), 560–566 (2011)

44.44 R. Murphy, J. Burke: From remote tool to shared roles, IEEE Robotics Autom. Mag. **15**(4), 39–49 (2008)

44.45 E. Paulos, J. Canny, F. Barrientos: Prop: Personal roving presence, SIGGRAPH Vis. Proc. (1997) p. 99

44.46 L. Takayama, E. Marder-Eppstein, H. Harris, J. Beer: Assisted driving of a mobile remote presence system: System design and controlled user evaluation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 1883–1889

44.47 D. Lazewatsky, W. Smart: An inexpensive robot platform for teleoperation and experimentation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 1211–1216

44.48 K. Taylor, J.P. Trevelyan: Australia's telerobot on the web, 26th Symp. Ind. Robotics (1995) pp. 39–44

44.49 A. Khamis, D.M. Rivero, F. Rodriguez, M. Salichs: Pattern-based architecture for building mobile robotics remote laboratories, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3284–3289

44.50 C. Cosma, M. Confente, D. Botturi, P. Fiorini: Laboratory tools for robotics and automation education, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3303–3308

44.51 K.W. Dorman, J.L. Pullen, W.O. Keksz, P.H. Eismann, K.A. Kowalski, J.P. Karlen: The servicing aid tool: A teleoperated robotics system for space applications, 7th Annu. Workshop Space Operat. Appl. Res. (SOAR), Vol. 1 (1994)

44.52 C. Pollak, H. Hutter: A webcam as recording device for light microscopes, J. Comput.-Assist. Microsc. **10**(4), 179–183 (1998)

44.53 K. Goldberg, D. Song, A. Levandowski: Collaborative teleoperation using networked spatial dynamic voting, Proceedings IEEE **91**(3), 430–439 (2003)

44.54 J.J. Kuffner: Cloud-Enabled Robots, IEEE-RAS Int. Conf. Humanoid Robots (2010)

44.55 K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, J. Wiegley: Desktop teleoperation via the World Wide Web, Proc. IEEE Int. Conf. Robotics Autom., Vol. 1 (1995) pp. 654–659

44.56 G. McKee: What is networked robotics?, Inf. Control Autom. Robotics **15**, 35–45 (2008)

44.57 E. Guizzo: Cloud robotics: Connected to the cloud, robots get smarter, IEEE Spectrum http://spectrum.ieee.org/automaton/robotics/robotics-software/cloud-robotics (2011)

44.58 M. Tenorth, A.C. Perzylo, R. Lafrenz, M. Beetz: The RoboEarth language: Representing and exchanging knowledge about actions, objects, and environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1284–1289

44.59 R. Arumugam, V.R. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F.F. Kong, A.S. Kumar, K.D. Meng, G.W. Kit: DAvinCi: A cloud computing framework for service robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 3084–3089

44.60 Z. Du, W. Yang, Y. Chen, X. Sun, X. Wang, C. Xu: Design of a robot cloud center, Int. Symp. Auton. Decentralized Syst. (2011) pp. 269–275

44.61 G. Hu, W.P. Tay, Y. Wen: Cloud robotics: Architecture, challenges and applications, IEEE Network **26**(3), 21–28 (2012)

44.62 K. Kamei, S. Nishio, N. Hagita, M. Sato: Cloud networked robotics, IEEE Network **26**(3), 28–34 (2012)

44.63 D. Hunziker, M. Gajamohan, M. Waibel, R. D'Andrea: Rapyuta: The RoboEarth cloud engine, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

44.64 M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Gálvez-López, K. Häussermann, R. Janssen, J.M.M. Montiel, A. Perzylo, B. le Schieß, M. Tenorth, O. Zweigle, R. De Molengraft: RoboEarth, IEEE Robotics Autom. Mag. **18**(2), 69–82 (2011)

44.65 RoboEarth: What is RoboEarth?, http://www.roboearth.org/what-is-roboearth

44.66 L. Atzori, A. Iera, G. Morabito: The internet of things: A survey, Comput. Netw. **54**(15), 2787–2805 (2010)

44.67 J. Walrand, P. Varaiya: *High-Performance Communication Networks*, 2nd edn. (Morgan Kaufmann Press, San Francisco 2000)

44.68 M.A. Peshkin, A.C. Sanderson: Minimization of energy in quasi-static manipulation, IEEE Trans. Robotics Autom. **5**(1), 53–60 (1989)

44.69 M.T. Mason: On the scope of quasi-static pushing, 3rd Int. Symp. Robotics Res. (1986)

44.70 E. Ladd, J. O'Donnell: *Using Html 4, Xml, and Java 1.2* (QUE Press, Indianapolis 1998)

44.71 H. Friz: Design of an Augmented Reality User Interface for an Internet based Telerobot using Multiple Monoscopic Views, Ph.D. Thesis (Technical Univ. Clausthal, Clausthal-Zellerfeld 2000)

44.72 T. Fong, C. Thorpe: Vehicle teleoperation interfaces, Auton. Robots **11**, 9–18 (2001)

44.73 A. Birk, N. Vaskevicius, K. Pathak, S. Schwertfeger, J. Poppinga, H. Buelow: 3-D perception and modeling, IEEE Robotics Autom. Mag. **16**(4), 53–60 (2009)

44.74 A. Kellyo, N. Chan, H. Herman, D. Huber, R. Meyers, P. Rander, R. Warner, J. Ziglar, E. Capstick: Real-time photorealistic virtualized reality interface for remote mobile robot control, Int. J. Robotics Res. **30**(3), 384–404 (2011)

44.75 L. Conway, R.A. Volz, M.W. Walker: Teleautonomous systems: Projecting and coordinating intelligent action at a distance, IEEE Trans. Robotics Autom. **6**(20), 146–158 (1990)

44.76 J. Larsson, M. Broxvall, A. Saffiotti: An evaluation of local autonomy applied to teleoperated vehicles in underground mines, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 1745–1752

44.77 L.B. Rosenberg: Virtual fixtures: Perceptual tools for telerobotic manipulation, IEEE Virtual Real. Annu. Int. Symp. (VRAIS) (1993) pp. 76–82

44.78 P. Marayong, M. Li, A. Okamura, G. Hager: Spatial motion constraints: Theory and demonstrations for robot guidance using virtual fixtures, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2003) pp. 1954–1959

44.79 A. Bettini, P. Marayong, S. Lang, A. Okamura, G. Hager: Vision-assisted control for manipulation using virtual fixtures, IEEE Trans. Robotics **20**(6), 953–966 (2004)

44.80 R. Azuma: A survey of augmented reality, Presence **6**(4), 355–385 (1997)

44.81 K. Goldberg, B. Chen, R. Solomon, S. Bui, B. Farzin, J. Heitler, D. Poon, G. Smith: Collaborative teleoperation via the internet, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2000) pp. 2019–2024

44.82 D. Song: Systems and Algorithms for Collaborative Teleoperation, Ph.D. Thesis (Univ. California, Berkeley 2004)

44.83 D. Song: *Sharing a Vision: Systems and Algorithms for Collaboratively-Teleoperated Robotic Cameras* (Springer, Berlin, Heidelberg 2009)

44.84 K. Goldberg, B. Chen: Collaborative teleoperation via the internet, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001)

44.85 K. Goldberg, D. Song: Tele-Actor http://www.tele-actor.net, Univ. of California, Berkeley

44.86 D. Song, A. Pashkevich, K. Goldberg: Sharecam part II: Approximate and distributed algorithms for a collaboratively controlled robotic webcam, Proc. IEEE/RSJ Int. Conf. Intell. Robots (IROS), Vol. 2 (2003) pp. 1087–1093

44.87 D. Song, K. Goldberg: Sharecam part I: Interface, system architecture, and implementation of a collaboratively controlled robotic webcam, Proc.

IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 2 (2003) pp. 1080–1086
44.88 D. Song, K. Goldberg: Approximate algorithms for a collaboratively controlled robotic camera, IEEE Trans. Robotics **23**(5), 1061–1070 (2007)
44.89 D. Song, A.F. van der Stappen, K. Goldberg: Exact algorithms for single frame selection on multi-axis satellites, IEEE Trans. Autom. Sci. Eng. **3**(1), 16–28 (2006)
44.90 Y. Xu, D. Song, J. Yi: An approximation algorithm for the least overlapping p-frame problem with non-partial coverage for networked robotic cameras, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008)
44.91 D. Song, N. Qin, K. Goldberg: Systems, control models, and codec for collaborative observation of remote environments with an autonomous networked robotic camera, Auton. Robots **24**(4), 435–449 (2008)
44.92 K. Goldberg, D. Song, I.Y. Song, J. McGonigal, W. Zheng, D. Plautz: Unsupervised scoring for scalable internet-based collaborative teleoperation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)
44.93 J. Rappole, S. Glasscock, K. Goldberg, D. Song, S. Faridani: Range change among new world tropical and subtropical birds, Bonn. Zool. Monogr. **57**, 151–167 (2011)
44.94 http://goldberg.berkeley.edu/cloud-robotics/, UC Berkeley
44.95 C. Goldfeder, M. Ciocarlie, P.K. Allen: The Columbia grasp database, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 1710–1716
44.96 A. Kasper, Z. Xue, R. Dillmann: The KIT object models database: An object model database for object recognition, localization and manipulation in service robotics, Int. J. Robotics Res. **31**(8), 927–934 (2012)
44.97 H. Dang, J. Weisz, P.K. Allen: Blind grasping: Stable robotic grasping using tactile feedback and hand kinematics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5917–5922
44.98 H. Dang, P.K. Allen: Learning grasp stability, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 2392–2397
44.99 J. Weisz, P.K. Allen: Pose error robust grasping from contact wrench space metrics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 557–562
44.100 M. Popovic, G. Kootstra, J.A. Jorgensen, D. Kragic, N. Kruger: Grasping unknown objects using an Early Cognitive Vision system for general scene understanding, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 987–994
44.101 B. Kehoe, A. Matsukawa, S. Candido, J. Kuffner, K. Goldberg: Cloud-based robot grasping with the Google object recognition engine, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)
44.102 M. Ciocarlie, C. Pantofaru, K. Hsiao, G. Bradski, P. Brook, E. Dreyfuss: A side of data with my robot, IEEE Robotics Autom. Mag. **18**(2), 44–57 (2011)
44.103 M.A. Moussa, M.S. Kamel: An experimental approach to robotic grasping using a connectionist architecture and generic grasping functions, IEEE Trans. Syst. Man Cybern. C **28**(2), 239–253 (1998)
44.104 K. Huebner, K. Welke, M. Przybylski, N. Vahrenkamp, T. Asfour, D. Kragic: Grasping known objects with humanoid robots: A box-based approach, Int. Conf. Adv. Robotics (2009)
44.105 C. Goldfeder, P.K. Allen: Data-driven grasping, Auton. Robots **31**(1), 1–20 (2011)
44.106 M. Ciocarlie, K. Hsiao, E.G. Jones, S. Chitta, R.B. Rusu, I.A. Sucan: Towards reliable grasping and manipulation in household environments, Intl. Symp. Exp. Robotics (2010) pp. 1–12
44.107 Google Goggles, http://www.google.com/mobile/goggles/
44.108 S. Dalibard, A. Nakhaei, F. Lamiraux, J.-P. Laumond: Manipulation of documented objects by a walking humanoid robot, IEEE-RAS Int. Conf. Humanoid Robots (2010) pp. 518–523
44.109 K. Lai, D. Fox: Object recognition in 3-D point clouds using web data and domain adaptation, Int. J. Robotics Res. **29**(8), 1019–1037 (2010)
44.110 S. Gammeter, A. Gassmann, L. Bossard, T. Quack, L. Van Gool: Server-side object recognition and client-side object tracking for mobile augmented reality, IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (2010) pp. 1–8
44.111 M. Armbrust, I. Stoica, M. Zaharia, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin: A view of cloud computing, Communication ACM **53**(4), 50 (2010)
44.112 Amazon Web Services, http://aws.amazon.com
44.113 Amazon Elastic Cloud (EC2), http://aws.amazon.com/ec2/
44.114 Google Compute Engine, https://cloud.google.com/products/compute-engine
44.115 Microsoft Azure, http://www.windowsazure.com
44.116 G. Juve, E. Deelman, G.B. Berriman, B.P. Berman, P. Maechling: An evaluation of the cost and performance of scientific workflows on Amazon EC2, J. Grid Comput. **10**(1), 5–21 (2012)
44.117 P. Mehrotra, J. Djomehri, S. Heistand, R. Hood, H. Jin, A. Lazanoff, S. Saini, R. Biswas: Performance evaluation of Amazon EC2 for NASA HPC applications, Proc. 3rd Workshop Sci. Cloud Comput. Date (ScienceCloud) (2012)
44.118 R. Tudoran, A. Costan, G. Antoniu, L. Bougé: A performance evaluation of Azure and Nimbus clouds for scientific applications, Proc. 2nd Int. Workshop Cloud Comput. Platf. (CloudCP) (2012) pp. 1–6
44.119 TOP500 List, http://www.top500.org/list/2012/06/100 (June 2012)
44.120 N.K. Jangid: Real time cloud computing, Proc. 1st Natl. Conf. Data Manag. Secur., Jaipur (2011)
44.121 J. Glover, D. Rus, N. Roy: Probabilistic models of object geometry for grasp planning. In: *Robotics: Science and Systems IV*, ed. by O. Brock, J. Trinkle, F. Ramos (MIT Press, Cambridge 2008)
44.122 H. Wang, Y. Ma, G. Pratx, L. Xing: Toward real-time Monte Carlo simulation using a commercial cloud computing infrastructure, Phys. Med. Biol. **56**(17), 175–181 (2011)

44.123 M. Sevior, T. Fifield, N. Katayama: Belle monte-carlo production on the Amazon EC2 cloud, J. Phys. **219**(1), 012003 (2010)

44.124 D. Nister, H. Stewenius: Scalable recognition with a vocabulary tree, IEEE Comput. Soc. Conf. Comp. Vis. Pattern Recognit., Vol. 2 (2006) pp. 2161–2168

44.125 J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman: Object retrieval with large vocabularies and fast spatial matching, IEEE Conf. Comput. Vis. Pattern Recognit. (2007) pp. 1–8

44.126 B. Bhargava, P. Angin, L. Duan: A mobile-cloud pedestrian crossing guide for the blind, Int. Conf. Adv. Comput. Commun. (2011)

44.127 J.J.S. García: Using cloud computing as a HPC platform for embedded systems, http://www.atc.us.es/descargas/tfmHPCCloud.pdf (2011)

44.128 J. van den Berg, P. Abbeel, K. Goldberg: LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information, Int. J. Robotics Res. **30**(7), 895–913 (2011)

44.129 B. Kehoe, D. Berenson, K. Goldberg: Estimating part tolerance bounds based on adaptive cloud-based grasp planning with slip, Proc. IEEE Int. Conf. Automat. Sci. Eng. (2012)

44.130 B. Kehoe, D. Berenson, K. Goldberg: Toward cloud-based grasping with uncertainty in shape: Estimating lower bounds on achieving force closure with zero-slip push grasps, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 576–583

44.131 D. Berenson, P. Abbeel, K. Goldberg: A robot path planning framework that learns from experience, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 3671–3678

44.132 MyRobots.com, http://myrobots.com

44.133 What is MyRobots?, http://myrobots.com/wiki/About

44.134 L. Dabbish, C. Stuart, J. Tsay, J. Herbsleb: Social coding in GitHub: transparency and collaboration in an open software repository, Proc. ACM Conf. Comp. Support. Coop. Work (2012) pp. 1277–1286

44.135 A. Hars: Working for free? Motivations of participating in open source projects, Proc. 34th Annu. Hawaii Int. Conf. Syst. Sci. (2001)

44.136 D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, D. Zagorodnov: The Eucalyptus open-source cloud-computing system, IEEE/ACM Int. Symp. Clust. Comput. Grid (2009) pp. 124–131

44.137 ROS (Robot Operating System), http://ros.org.

44.138 M. Quigley, B. Gerkey: ROS: An open-source robot operating system, ICRA Workshop Open Source Softw. (2009)

44.139 rosjava, an implementation of ROS in pure Java with Android support, http://cloudrobotics.com

44.140 The African Robotics Network (AFRON): *Ten Dollar Robot* design challenge winners, http://robotics-africa.org/design_challenge.html (2012)

44.141 Bullet Physics Library, http://bulletphysics.org

44.142 OpenRAVE, http://openrave.org/

44.143 Gazebo, http://gazebosim.org

44.144 E. Plaku, K.E. Bekris, L.E. Kavraki: OOPS for motion planning: An online, open-source, programming system, Proc. IEEE Int. Conf. Robotics Autom. (2007) pp. 3711–3716

44.145 A.T. Miller, P.K. Allen: GraspIt! A versatile simulator for robotic grasping, IEEE Robotics Autom. Mag. **11**(4), 110–122 (2004)

44.146 A. Sorokin, D. Berenson, S.S. Srinivasa, M. Hebert: People helping robots helping people: Crowdsourcing for grasping novel objects, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 2117–2122

44.147 S. Davidson: Open-source hardware, IEEE Des. Test Comput. **21**(5), 456–456 (2004)

44.148 E. Rubow: Open Source Hardware, Tech. Rep. http://cseweb.ucsd.edu/classes/fa08/cse237a/topicresearch/erubow_tr_report.pdf (2008)

44.149 Arduino: http://www.arduino.cc

44.150 H.H. King, L. Cheng, P. Roan, D. Friedman, S. Nia, J. Ma, D. Glozman, J. Rosen, B. Hannaford: Raven II: Open platform for surgical robotics research, Hamlyn Symp. Med. Robotics (2012)

44.151 An open-source robo-surgeon, The Economist, http://www.economist.com/node/21548489 (2012)

44.152 L. von Ahn: Human computation, Des. Autom. Conf. (2009) pp. 418–419

44.153 J.C. Gamboa Higuera, A. Xu, F. Shkurti, G. Dudek: Socially-driven collective path planning for robot missions, 9th Conf. Comput. Robot Vis. (2012) pp. 417–424

44.154 Y. Gingold, A. Shamir, D. Cohen-Or: Micro perceptual human computation for visual tasks, ACM Trans. Graphics **31**(5), 1–12 (2012)

44.155 M. Johnson-Roberson, J. Bohg, G. Skantze, J. Gustafson, R. Carlson, B. Rasolzadeh, D. Kragic: Enhanced visual scene understanding through human-robot dialog, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 3342–3348

44.156 A. Sorokin, D. Forsyth: Utility data annotation with Amazon Mechanical Turk, IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops (2008) pp. 1–8

44.157 C. Escolano, J. Antelis, J. Minguez: Human brain-teleoperated robot between remote places, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 4430–4437

44.158 A. Akce, M. Johnson, T. Bretl: Remote teleoperation of an unmanned aircraft with a brain-machine interface: Theory and preliminary results, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 5322–5327

44.159 J. Roselln, R. Suárez, C. Rosales, A. Pérez: Autonomous motion planning of a hand-arm robotic system based on captured human-like hand postures, Auton. Robots **31**(1), 87–102 (2011)

44.160 K. Onda, F. Arai: Parallel teleoperation of holographic optical tweezers using multi-touch user interface, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1069–1074

44.161 P.X. Liu, M. Meng, S.X. Yang: Data communications for internet robots, Auton. Robots **15**, 213–223 (2003)

44.162 W. Fung, N. Xi, W. Lo, B. Song, Y. Sun, Y. Liu, I.H. Elhajj: Task driven dynamic QOS based bandwidth allcoation for real-time teleoperation via the internet, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003)

44.163 F. Zeiger, N. Kraemer, K. Schilling: Commanding mobile robots via wireless ad-hoc networks – A comparison of four ad-hoc routing protocol implementations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 590–595

44.164 M. Amoretti, S. Bottazzi, M. Reggiani, S. Caselli: Evaluation of data distribution techniques in a corba-based telerobotic system, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003)

44.165 S. Bottazzi, S. Caselli, M. Reggiani, M. Amoretti: A software framework based on real time COBRA for telerobotics systems, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002)

44.166 J. Chen, E. Haas, M. Barnes: Human performance issues and user interface design for teleoperated robots, IEEE Trans. Syst. Man Cybern. C **37**(6), 1231–1245 (2007)

44.167 Y. Jia, N. Xi, Y. Wang, X. Li: Online identification of quality of teleoperator (QoT) for performance improvement of telerobotic operations, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 451–456

44.168 S. Livatino, G. Muscato, S. Sessa, C. Koffel, C. Arena, A. Pennisi, D. Di Mauro, F. Malkondu: Mobile robotic teleguide based on video images, IEEE Robotics Autom. Mag. **15**(4), 58–67 (2008)

44.169 G. Podnar, J. Dolan, A. Elfes, S. Stancliff, E. Lin, J. Hosier, T. Ames, J. Moisan, T. Moisan, J. Higinbotham, E. Kulczycki: Operation of robotic science boats using the telesupervised adaptive ocean sensor fleet system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 1061–1068

44.170 Y. Kwon, S. Rauniar: E-quality for manufacturing (EQM) within the framework of internet-based systems, IEEE Trans. Syst. Man Cybern. C **37**(6), 1365–1372 (2007)

44.171 L. Wang: Wise-shopfloor: An integrated approach for web-based collaborative manufacturing, IEEE Trans. Syst. Man Cybern. C **38**(4), 562–573 (2008)

44.172 P. Debenest, M. Guarnieri, K. Takita, E. Fukushima, S. Hirose, K. Tamura, A. Kimura, H. Kubokawa, N. Iwama, F. Shiga: Expliner – Robot for inspection of transmission lines, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 3978–3984

44.173 N. Pouliot, S. Montambault: Linescout technology: From inspection to robotic maintenance on live transmission power lines, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 1034–1040

Multimedia Contents

Part E Moving

Part E Moving in the Environment

Ed. by Raja Chatila

45 World Modeling
Wolfram Burgard, Freiburg, Germany
Martial Hebert, Pittsburgh, USA
Maren Bennewitz, Bonn, Germany

46 Simultaneous Localization and Mapping
Cyrill Stachniss, Bonn, Germany
John J. Leonard, Cambridge, USA
Sebastian Thrun, Mountain View, USA

47 Motion Planning and Obstacle Avoidance
Javier Minguez, Zaragoza, Spain
Florant Lamiraux, Toulouse, France
Jean-Paul Laumond, Toulouse, France

48 Modeling and Control of Legged Robots
Pierre-Brice Wieber, Grenoble, France
Russ Tedrake, Cambridge, USA
Scott Kuindersma, Cambridge, USA

49 Modeling and Control of Wheeled Mobile Robots
Claude Samson, Sophia-Antipolis, France
Pascal Morin, Paris, France
Roland Lenain, Aubiere, France

50 Modeling and Control of Robots on Rough Terrain
Keiji Nagatani, Sendai, Japan
Genya Ishigami, Yokohama, Japan
Yoshito Okada, Sendai, Japan

51 Modeling and Control of Underwater Robots
Gianluca Antonelli, Cassino, Italy
Thor I. Fossen, Trondheim, Norway
Dana R. Yoerger, Woods Hole, USA

52 Modeling and Control of Aerial Robots
Robert Mahony, Canberra, Australia
Randal W. Beard, Provo, USA
Vijay Kumar, Philadelphia, USA

53 Multiple Mobile Robot Systems
Lynne E. Parker, Knoxville, USA
Daniela Rus, Cambridge, USA
Gaurav S. Sukhatme, Los Angeles, USA

Until the mid 1960s, robots were only able to move in a predetermined workspace, the one they could reach from their firmly fixed base. Part E is about their conquest of the whole space. Mobile Robotics started as a research domain in its own right in the late 1960s with the Shakey project at SRI. 2015 marks the 50th anniversary of this seminal project that had a lasting legacy such as A^*. The seminal paper by N.J. Nilsson *A Mobile Automaton: An Application of Artificial Intelligence Techniques* at the International Joint Conference on Artificial Intelligence (IJCAI) 1969, already addressed perception, mapping, motion planning, and the notion of control architecture. Those issues would indeed be at the core of mobile robotics research for the following decades. The 1980s boomed with mobile robot projects, and as soon as it was necessary to cope with the reality of the real physical world, problems appeared that fostered novel research directions, actually moving away from the original concept in which the robot was just an application of artificial intelligence (AI) techniques. This part addresses all the issues that, put together, are necessary to build and control a mobile robot, except for the mechanical design itself.

Navigation is the capacity of moving from one location to another arbitrarily distant one. To move efficiently, a robot needs to use appropriate representations of its environment in order to plan and control its motions according to the presence of obstacles and terrain difficulties, and also to use environment features as landmarks for its localization. This is the topic of Chapter 45 which addresses the different representations used for indoors or outdoors environments, including topological maps and semantic attributes.

However, building environment maps is usually achieved incrementally, as the robot discovers its environment while navigating in it. Hence partial perceptions built from different positions, need to be fused together – taking into account sensing errors and uncertainties (see Chapter 5, Part A) to construct a consistent global map. This requires the robot to know these positions, and, because of motion inaccuracies, the transforms between them are uncertain. This requires to reference the positions to environment features which are only defined in the environment map itself. As a result, localization needs the map and mapping needs localization. Hence, localization and mapping are two interwound problems that must be solved simultaneously. The solution to this problem is the topic of Chapter 46.

Once environment maps are available, or during their construction, the robot has to plan its path as optimally as possible to reach its targets while avoiding obstacles. Chapter 7 in Part A overviews the techniques for solving this central problem, which requires geometrical reasoning in the configuration space (CS) whose construction is complex. To avoid an explicit construction of the CS, probabilistic techniques proved the most efficient way to compute paths – at the price of optimality. However, the kinematic constraints of the robot's locomotion system such as non-holonomy have to be taken into account. We see in Chapter 47 in this part how control problems cannot be separated from geometry. This chapter addresses motion planning from the viewpoint of mobile robotics and introduces the tools from control theory and differential geometry that enable to tackle kinematics constraints. In addition, it provides an overview of local sensor-based methods that are used when the robot encounters unknown or mobile obstacles while moving.

The rest of this part essentially focusses on various means of locomotion used to move in the environment: legged locomotion (Chapter 48), robots with wheels (Chapter 49), with tracks (Chapter 50), and underwater and aerial robots (Chapters 51 and 52 respectively) which face the specific problems of three-dimensional (3-D) motion with environment perturbations. In these two last cases the close links between Robotics and Control are of course central.

Finally, after explaining how one single robot moves in different environments, Part E concludes with Chapter 53 on the interaction and coordination of multiple mobile robot systems.

Multimedia Contents

45. World Modeling

Wolfram Burgard, Martial Hebert, Maren Bennewitz

In this chapter we describe popular ways to represent the environment of a mobile robot. For indoor environments, which are often stored using two-dimensional representations, we discuss occupancy grids, line maps, topological maps, and landmark-based representations. Each of these techniques has its own advantages and disadvantages. Whilst occupancy grid maps allow for quick access and can efficiently be updated, line maps are more compact. Also landmark-based maps can efficiently be updated and maintained, however, they do not readily support navigation tasks such as path planning like topological representations do.

Additionally, we discuss approaches suited for outdoor terrain modeling. In outdoor environments, the flat-surface assumption underling many mapping techniques for indoor environments is no longer valid. A very popular approach in this context are elevation and variants maps, which store the surface of the terrain over a regularly spaced grid. Alternatives to such maps are point clouds, meshes, or three-dimensional grids, which provide a greater flexibility but have higher storage demands.

45.1 **Historical Overview** 1136
45.2 **Models for Indoors and Structured Environments** 1137
45.2.1 Occupancy Grids 1137
45.2.2 Line Maps 1138
45.2.3 Topological Maps 1140
45.2.4 Landmark-Based Maps 1141
45.3 **World and Terrain Models for Natural Environments** 1141
45.3.1 Elevation Grids 1141
45.3.2 3-D Grids and Point Sets 1143
45.3.3 Meshes 1145
45.3.4 Cost Maps 1145
45.3.5 Semantic Attributes 1146
45.3.6 Heterogeneous and Hierarchical Models 1148
45.4 **Dynamic Environments** 1149
45.5 **Summary and Further Reading** 1149
Video-References 1150
References 1150

The construction of models of the environment is crucial to the development of several applications of mobile robot systems. It is through these environment models that the robot can adapt its decisions to the current state of the world. The models are constructed from sensor data as the robot discovers its environment. There are three challenges in constructing environment models from sensor data. First, the models must be compact so that they can be used efficiently by other components of the system, such as path planners. Second, the models must be adapted to the task and to the type of environment. For example, modeling the environment as a set of planes is not relevant to a robot operating in natural terrain. In particular, this implies that a *universal* representation for mobile robots in not possible and that we must instead choose from an array of different approaches. Third, the representation must accommodate the uncertainty inherent to both sensor data and to the robot's state estimation system. The latter point is particular important since environment models typically accumulate sensor readings over substantial distances into a common reference frame. Drift in position estimates are unavoidable and must be accounted for in the model representation and in its construction.

45.1 Historical Overview

Historically, work in this area concentrated first on robots operating in indoor environments. In that case, the models take advantage of the fact that the world can be represented as vertical structures on reference ground planes. This simplification can be used for representing the world as a two-dimensional (2-D) grid. Uncertainty in the measurements and in the robot's pose can then be modeled by using probabilities of occupancy in the grid rather than binary occupied/empty flags. Another feature of indoor environments is that they are highly structured in that they contain primarily linear structures such as lines and planes. This observations led to a second class of representations based on collections of points, lines, and planes to represent the environment. Here again, a lot of attention was paid to representing the uncertainty on the relative poses of these geometric elements, including a considerable amount of work in the 1980s on using Kalman filters and other probabilistic techniques.

With continued progress on sensing (e.g., three-dimensional laser range scanners and stereo vision), and on mechanical and controls aspects of mobile robots systems, it became possible to develop mobile robot systems for operations in unstructured, natural terrain, motivated in part by planetary exploration and military applications. In these scenarios, it is no longer appropriate to project the data in a 2-D grid, and environments cannot be described adequately by a small vocabulary of geometric elements. Since, in most cases, it is still appropriate to assume (at least locally) that there is a reference ground plane, a natural representation is a two-and-a half-dimensional 2.5-D grid, in which each cell contains the elevation (and possibly other features) of the terrain at that location. Although they have been used extensively, the main challenges with such *elevation maps* are that they are not compact representations and that it is difficult to incorporate uncertainty in the representation. Accordingly, much of the research has focused on designing efficient data structures and algorithms for elevation maps, for example, hierarchical representations. Recently, the uncertainty challenge has been tackled by using probabilistic variants of elevation maps.

While elevation maps provide a natural representation for many types of natural terrains, they cannot represent environments with vertical or overhanging structures. This limitation has become more evident in recent years as applications of mobile robots are increasingly demanding operation in urban environment (in which building walls and other structures cannot be represented by elevation maps) and the use of aerial data, which involves overhanging structures such as tree canopies. This has led to the development of representations that are truly three-dimensional (3-D), such as point clouds, 3-D grids, and meshes. The two challenges described above apply here as well, except that the situation is more complicated because of the increased complexity introduced by the third dimension. Current research includes efficient computation over 3-D structures and probabilistic representations of 3-D data.

Because of the large volume of data involved in all of these representations, it is important to be able to group the data into larger chunks corresponding to semantically meaningful parts of the environment. This can be done at different levels, depending on the application the environment. At the lowest level, work in this area involves classifying the points into classes that are relevant to navigation tasks (e.g., distinguishing between vegetation and ground, extracting walls, tree surfaces, etc.). At an intermediate level of representation, this part of the work involves extracting parts of the environments that are considered landmarks of interest for the navigation task (e.g., roads). Finally, the highest representation level involves extracting and representing objects in the environments (e.g., natural obstacles, or specific objects such as cars for operation in urban environments).

All of these representations assume a static environment. In fact, many of the current applications of mobile robots call for operation in mixed environments in which the robot shares its environments with other moving agents, including other robots, people, and vehicles. Assuming that perception algorithms are able to detect and track individual moving objects in the environment, the challenge here is to insert this information into a representation that can be used by a planner. In this case, any one of the previous representations can be used as a foundation, but it needs to be extended to handle the temporal dimension by storing a history of the locations and states of the detected objects. Such representations include the trajectories of the detected objects and, in some cases, information on the future predicted trajectories of the objects.

45.2 Models for Indoors and Structured Environments

45.2.1 Occupancy Grids

Occupancy grid maps, which were introduced in the 1980s by *Moravec* and *Elfes* [45.1], are a popular, probabilistic approach to represent the environment. They are an approximative technique in which we calculate for each cell of a discrete grid the posterior probability that the corresponding area in the environment is occupied by an obstacle. The advantage of occupancy grid maps lies in the fact that they do not rely on any predefined features. Additionally, they offer a constant-time access to grid cells and provide the ability to represent unknown (unobserved) areas, which can be important, for example, in exploration tasks. Their disadvantage lies in potential discretization errors and the high memory requirements.

Throughout this section we assume that the map m consists of a discrete, two-dimensional grid of L cells denoted as $m_1, \ldots, m_L$. Given the sensory input $z_{1:t}$ obtained by the robot at the corresponding positions $x_{1:t}$, the occupancy grid mapping approach calculates a posterior probability $p(m \mid x_{1:t}, z_{1:t})$.

To keep the calculations tractable, the overall approach assumes that the individual cells of the grid are independent, i. e., that the following equation holds

$$p(m \mid x_{1:t}, z_{1:t}) = \prod_{l=1}^{L} p(m_l \mid x_{1:t}, z_{1:t}) \,. \tag{45.1}$$

Note that this assumption is rather strong. It basically states that any information about the occupancy of one cell does not tell us anything about its neighboring cells. In practice, we often find objects that are larger than the individual grid cells, like doors, cabinets, chairs etc. Thus, if we know that one cell is occupied, for each of its neighboring cells the probability raises that it is occupied. Despite this fact, occupancy grid maps have been successfully applied in numerous installations of mobile robots and have been proven to be a powerful tool that supports various navigation tasks such as localization and path planning.

Because of the independence assumption expressed by (45.1) we can concentrate on the estimation of the occupancy probability of the individual cells m_l in m. Under additional independence assumptions one can finally arrive at the following formula for calculating the occupancy probability $p(m_l \mid x_{1:t}, z_{1:t})$ that cell m_l is occupied given the prior probability $p(m_l \mid x_{1:t-1}, z_{1:t-1})$ and the new observation z_t observed at position x_t

$$\begin{aligned} &p(m_l \mid x_{1:t}, z_{1:t}) \\ &= \left[1 + \frac{(1-p(m_l \mid x_t, z_t))}{p(m_l \mid x_t, z_t)} \cdot \frac{p(m_l)}{(1-p(m_l))} \right. \\ &\quad \left. \times \frac{1-p(m_l \mid x_{1:t-1}, z_{1:t-1})}{p(m_l \mid x_{1:t-1}, z_{1:t-1})} \right]^{-1} . \end{aligned} \tag{45.2}$$

In practice, one often assumes that the prior $p(m_l)$ is 0.5 so that the second factor in the product becomes 1 and therefore vanishes from the equation.

Additionally, if we define

$$\mathrm{Odds}(x) = \frac{p(x)}{1-p(x)} \,, \tag{45.3}$$

the incremental updates can be computed using the following equation

$$\begin{aligned} &\mathrm{Odds}(m_l \mid x_{1:t}, z_{1:t}) \\ &= \mathrm{Odds}(m_l \mid x_t, z_t) \cdot \mathrm{Odds}(m_l)^{-1} \\ &\quad \times \mathrm{Odds}(m_l \mid x_{1:t-1}, z_{1:t-1}) \,. \end{aligned} \tag{45.4}$$

To recover the occupancy probability from the Odds representation given in (45.4), one can use the following formula, which can easily be derived using (45.3)

$$p(x) = \frac{\mathrm{Odds}(x)}{1 + \mathrm{Odds}(x)} \,. \tag{45.5}$$

It remains to describe how to compute the occupancy probability $p(m_l \mid x_t, z_t)$ of a grid cell given a *single* observation z_t and the corresponding pose x_t of the robot. This quantity strongly depends on the sensor of the robot and has to be defined individually for each type of sensor. Additionally, the parameters of these models have to be adapted according to the properties of each sensor. Let us assume that the function $\mathrm{dist}(x_t, m_l)$ refers to the distance between the sensor at pose x_t and the center of the cell m_l. Let us first assume that we only need to consider the optical axis of the sensor cone as is, for example, the case for laser beams sent out by a laser range finder. Then, $p(m_l \mid x_t, z_t)$ can be formulated as

$$p(m_l \mid x_t, z_t) = \begin{cases} p_{\text{prior}}, & z_t \text{ is a maximum range reading} \\ p_{\text{prior}}, & m_l \text{ is not covered by } z_t \\ p_{\text{occ}}, & |z_t - \mathrm{dist}(x_t, m_l)| < r/2 \\ p_{\text{free}}, & z_t \geq \mathrm{dist}(x_t, m_l) \end{cases} , \tag{45.6}$$

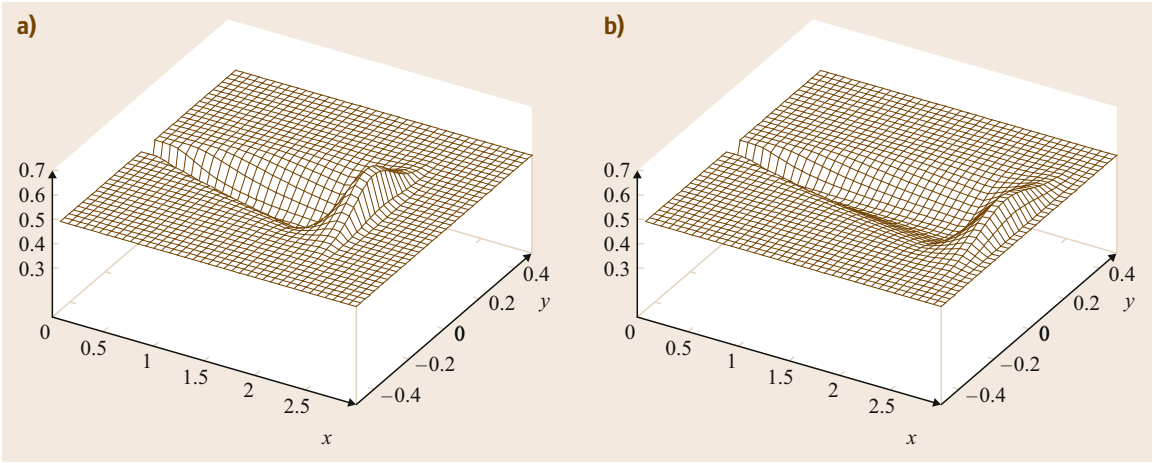


Fig.45.1a,b Occupancy probability introduced by a single ultrasound measurement of **(a)** $z = 2.0\,\mathrm{m}$ and **(b)** $z = 2.5\,\mathrm{m}$ (after [45.2])

where r is the resolution of the grid map. Obviously, it must hold $0 \leq p_{\mathrm{free}} < p_{\mathrm{prior}} < p_{\mathrm{occ}} \leq 1$.

If a sonar sensor is used, the sensor model is slightly more complicated, since the sensor is not a beam sensor and the observations are more noisy than the ones of a laser range finder. To deal with these properties, one can use a mixture of three functions to express the model. First, the influence of an observation (which is represented by the difference between p_{prior} and p_{occ} as well as between p_{prior} and p_{free}) decreases with the measured distance. Second, the proximity information of a sonar is substantially affected by noise. This can be considered by employing a piecewise linear function to model a smooth transition from p_{free} to p_{occ}. Finally, the sonar sensor should not be modeled as a beam sensor, since it sends out a conic signal. The accuracy of an observation decreases with the angular distance between the cell under consideration and the optical axis of the observation. This is expressed by the derivation from the prior and is typically modeled using a Gaussian with zero mean. Therefore, it is maximal along the optical axis and decreases the bigger the angular distance form the optical axis is [45.2].

Two examples of resulting models are depicted in Fig. 45.1, which shows two three-dimensional plots of the resulting occupancy probabilities for a measurement of 2 m (Fig. 45.1a) and 2.5 m (Fig. 45.1b). In this figure, the optical axis of the sensor cone was identical with the x-axis and the sensor was placed in the origin of the coordinate frame. As can be seen, the occupancy probability is high for cells whose distance to x_t is close to z_t. It decreases for cells with shorter distance than z_t as well as with increasing values of the angular distance.

Figure 45.2 depicts the mapping process for a sequence of observations recorded with an iRobot B21r robot. The first row shows how a map was built from a sequence of previous ultrasound scans. Afterwards the robot perceived a series of 18 ultrasound scans, each consisting of 24 measurements. The occupancy probabilities for these 18 scans are depicted in rows 2–7. The occupancy probability grid obtained by integrating the individual observations into the map is shown in the last row of this figure. As can be seen, the belief converges to a representation of the corridor structure in which the scans where recorded. A typical resulting map obtained for an indoor environment is depicted in Fig. 45.3.

45.2.2 Line Maps

The representation of the environment by line models is a popular alternative to the grid-based approximations described above. Line models have several advantages over these nonparametric representations. They require substantially less memory than grids and therefore scale better with the size of the environment. They furthermore are more accurate since they do not suffer from discretization problems. In this section, we consider the problem of calculating lines that provide the approximation of a set of range points by a line. If the data points are given by n pairs (x_i, y_i) of Cartesian coordinates, the line that minimizes the squared distances to all points can be calculated in closed form according to

$$\tan 2\phi = \frac{-2\sum_i (\bar{x} - x_i)(\bar{y} - y_i)}{\sum_i [(\bar{y} - y_i)^2 - (\bar{x} - x_i)^2]}\,, \tag{45.7}$$

$$r = \bar{x}\cos\phi + \bar{y}\sin\phi\,, \tag{45.8}$$

where $\bar{x} = \frac{1}{n}\sum_i x_i$ and $\bar{y} = \frac{1}{n}\sum_i y_i$. In these equations, r is the normal distance of the line from the origin and ϕ is angle of the normal.

Unfortunately, there is no closed-form solution for situations in which the data points are generated by multiple linear structures. In such a situation two problems arise. First one has to answer the question how many lines there are, and second one has to solve a data association problem, which is to find the assignment of the data points to the individual lines. Once the number of lines and the associations are known, we can apply (45.7) and (45.8) to calculate the individual line parameters.

There is a popular approach to solve this problem for range scans with multiple linear structures. This approach, which goes back to the work of *Douglas* and *Peucker* [45.3], is also known as the *split-and-merge* algorithm. Its key idea is to recursively subdivide the point set into subsets that can be more accurately approximated by a line. The approach starts with the line calculated from all data points and determines the data point with maximum distance from this line. If this distance is below a given threshold, the algorithm terminates and provides the fitted line as output. Otherwise, it calculates the point with the largest distance from the line that connects the first and the last point. This point is the so-called splitting point. It then divides the point set into two subsets, one containing all points from the first up to the splitting point and one containing all points after the splitting point. The algorithm is then recursively applied to the two subsets. Figure 45.4 shows a visualization of the procedure.

Whereas the split-and-merge algorithm is quite effective and efficient, it is not guaranteed that the resulting model is actually optimal, i. e., is the model that minimizes the squared distances of all data points. One approach that is able to find such a model is based on the expectation maximization (EM) algorithm, which in the application described here can be regarded as a variant of the fuzzy k-means clustering algorithm. Let us suppose that the number m of lines in the model θ is known. Let us furthermore suppose that the likelihood of a data point $z = (x, y)$ given the model θ consisting of the line set $\{\theta_1, \ldots, \theta_m\}$ is defined as

$$p(z \mid \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\frac{d(z, \theta_k)^2}{\sigma^2}\right), \qquad (45.9)$$

where σ is the standard deviation of the measurement noise and θ_k is the line for which Euclidean distance $d(z, \theta_k)$ to z is minimal.

The goal of the EM algorithm is to generate an iterated sequence of models of increased likelihood. To achieve this, one introduces so-called correspondence variables $c_{ij} \in \{0, 1\}$ that specify to which linear component of the model each point belongs. Since the correct values of these assignment variables are unknown, one estimates a posterior about their values. Let θ_j be a component of the model and z_i be a measurement. Then the expectation about c_{ij}, i. e., that measurement i belongs

Fig. 45.2 Incremental mapping in a corridor environment. The *upper left* image shows the initial map and the *lower* one shows the resulting map. The maps *in between* are the local maps built from the individual ultrasound scans perceived by the robot (after [45.2])

Fig. 45.3 Occupancy grid map obtained from ultrasound data (after [45.2])

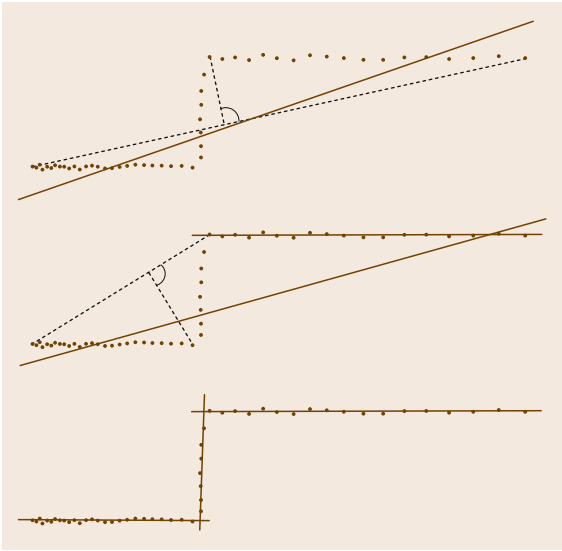


Fig. 45.4 Split-and-merge algorithm applied to a point set

to line j is computed in the E-step as

$$E[c_{ij} \mid \theta_j, z_i] = p(c_{ij} \mid \theta_j, z_i) \,, \tag{45.10}$$

$$= \alpha p(z_i \mid c_{ij}, \theta_j) p(c_{ij} \mid \theta_j) \,, \tag{45.11}$$

$$= \alpha' p(z_i \mid \theta_j) \,. \tag{45.12}$$

In the M-step, the algorithm then computes the parameters of the model by taking into account the expectations computed in the E-step

$$\theta_j^* = \arg\min_{\theta_j'} \sum_i \sum_j E[c_{ij} \mid \theta_j, z_i] d^2(z_i, \theta_j') \,. \tag{45.13}$$

Given a fixed variance σ for all data points in z we can determine the most likely model in closed form according to the following equations, which are probabilistic variants of (45.7) and (45.8) that take into account the data association uncertainty given by the expectations calculated in the E-step

$$\tan 2\phi_j = \frac{-2 \sum_i E[c_{ij}|\theta_j, z_i](\bar{x} - x_i)(\bar{y} - y_i)}{\sum_i E[c_{ij}|\theta_j, z_i][(\bar{y} - y_i)^2 - (\bar{x} - x_i)^2]} \,, \tag{45.14}$$

$$r_j = \bar{x} \cos\phi_j + \bar{y} \sin\phi_j \,. \tag{45.15}$$

Here $\bar{x}$ and $\bar{y}$ are computed as

$$\bar{x} = \frac{\sum_i E[c_{ij}|\theta_j, z_i] x_i}{\sum_i E[c_{ij}|\theta_j, z_i]} \,, \quad \bar{y} = \frac{\sum_i E[c_{ij}|\theta_j, z_i] y_i}{\sum_i E[c_{ij}|\theta_j, z_i]} \,. \tag{45.16}$$

Figure 45.5 depicts a line map extracted from 311 823 data points using the EM-based approach. In this example, the model consists of 94 lines. One approach to determine the optimal number of lines is to utilize the Bayesian information criterion [45.4].

Fig. 45.5 Line map consisting of 94 lines generated for 311 823 range points generated with the EM-based approach (after [45.4])

45.2.3 Topological Maps

In contrast to the previously discussed representations, which mainly focus on the geometric structure of the environment, topological representations have also received significant attraction. One of the pioneering approaches to topological mapping, which has been presented in 1988, is the work by *Kuipers* and *Byun* [45.7]. In this approach, the environment is represented by a graph-like structure, in which the nodes are locally distinguishable places and the nodes are travel edges along which the robot can move between the places (VIDEO 270). Here, distinctive places are identified according to the distance to nearby objects. The distinctive places were been defined by *Choset* and *Nagatani* [45.8] as the meet-points in the generalized Voronoi diagrams, i. e., the points with an out-degree of three or more. A generalized Voronoi diagram is the set of points equidistant from the closest two or more obstacle boundaries. Generalized Voronoi diagrams are a very popular representation as they can be considered as road-maps with a high correspondence to the topological structure of the environment. They

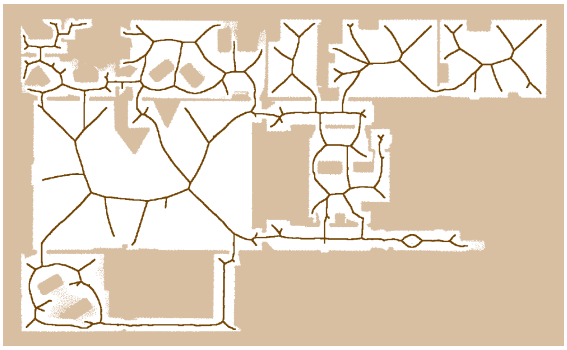


Fig. 45.6 Example of a generalized Voronoi graph (after [45.5])

Fig. 45.7 (a) Mobile robot mapping a set of stone rocks. The image (b) depicts the path and the estimated landmark positions. The manually determined positions of the rocks are marked by *circles* (after [45.6])

have been used extensively for path planning. To plan a path from a starting position to a goal point in the environment, all the robot has to do is to first plan a path to the generalized Voronoi graph, than along the generalized Voronoi graph, and then from the generalized Voronoi graph to the goal point [45.8]. Figure 45.6 shows an example generalized Voronoi graph for an indoor environment. Note that in this figure only those parts of the graph that can be traversed by the robot without colliding with objects are displayed.

45.2.4 Landmark-Based Maps

For environments with locally distinguishable features, landmark-based maps have been extensively used. If we assume that the position of the robot is always known, it simply remains to maintain an estimate about the positions of the individual landmarks over time. In a planar environment, m is represented by K two-dimensional Gaussians with mean μ_k and covariance Σ_k, one for each landmark. If a linearized version of the perception problem is given, the individual Gaussians can be updated using the equations for the extended Kalman filter. Note that, compared to the use of the extended Kalman filter EKF for landmark-based simultaneous localization and mapping (SLAM), where we needed a $2K+3$-dimensional state vector (3 dimensions for the position of the robot and $2K$ dimensions for the positions of the individual landmarks), we only need K two-dimensional Gaussians to represent the entire map since the robot pose is known. This property, for example, has been utilized in the FastSLAM algorithm [45.6]. Over the last few years, the graph-based SLAM paradigm has become a popular approach for estimating landmark maps due to their ability of relinearizing the motion and the measurement functions (for more details see Chap. 46).

Figure 45.7a shows a robot equipped with a SICK laser range finder that maps the position of rocks. The right image depicts the path of the vehicle as well as the manually determined and automatically estimated landmark positions.

45.3 World and Terrain Models for Natural Environments

From a survey of environment modeling [45.9] with an emphasis on probabilistic techniques for indoor environments, a taxonomy can be created along several axis: metric versus topological versus semantic, robot-centric versus world-centric, or application-based. We choose to first review purely geometric models (elevation grid, 3-D grid, meshes), then geometric models with low-level attributes (cost maps), and then models with richer semantic attributes, to finish with heterogeneous and hierarchical models.

45.3.1 Elevation Grids

Assuming that the terrain can be represented as a function $h = f(x, y)$, where x and y are the coordinates of a reference place and h is the corresponding elevation. A natural representation is a digital elevation map which stores the value of h at discrete locations (x_i, y_i). Because they are simple data structures and can be generated from sensor data in a relatively straightforward manner, elevation maps have been used extensively for mobile robots operating in natural environments with no vertical surfaces or overhangs (e.g., for planetary exploration scenarios [45.10]). Several issues need to be addressed when using elevation maps for mobile robots.

A regularly sampled grid is appropriate when the sensor data is roughly uniformly distributed on the reference plane. This is the case, for example, with aerial data. For ground robots, however, the distribution of the data on the reference ground plane varies dramatically

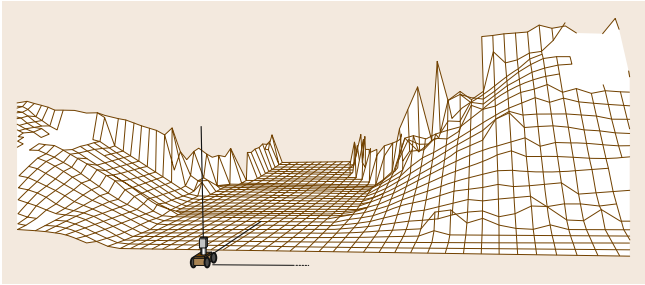


Fig. 45.8 Example elevation map built by accumulating 3-D data from a range sensor

because of the small incidence angles with the reference plane. This can be addressed by using variable-size cells instead of regularly sampled cells. In that case, the distribution of the cells on the reference plane is designed to approximate the distribution of points that would be measured by the sensor on this reference plane. Such nonuniform representations are useful primarily in cases in which the map is referenced to the current position of the robot [45.11] (Fig. 45.9). Frequent resampling is needed if the map is expressed in a global reference frame.

Irrespective of the sampling scheme used to construct the reference grid, the density of data in the grid varies due to local self-occlusions of the terrain surface and to mismatches between the resolution of the projected sensor data and the resolution of the grid. This can be problematic because, from the point of view of a planner, the elevation map would appear as a scattering of cells with elevation data and cells with no evidence (Fig. 45.8). Various schemes have been proposed to remedy this problem. The basic idea is to estimate the elevation values of the empty cells by interpolating across the cells with know elevation values. This has to be done with great care to avoid filling up regions that correspond to parts of the environment occluded by the terrain (*range shadows*) in which elevation data should not be inferred. Such mistakes can be catastrophic since a planner may generate paths through areas that are entirely unknown. One general approach to this problem is to use surface interpolation techniques including a term that allows for discontinuities in the resulting surfaces [45.12]. An alternative approach is to use the visibility constraints induced by the known geometry of the sensor in order to estimate plausible values of elevation at each cell [45.13, 14].

A difficulty in interpolating techniques is to explicitly take into account the sensor uncertainty. In particular, it is difficult to account for the varying resolution of the sensor as a function of range in a fixed-resolution grid. An alternative is to use multiple grids at different resolutions. The appropriate resolution to be used at a particular point (x, y) can be decided based on the range from (x, y) to the closest sensor location to retrieve the value at (x, y) from the appropriate map for that resolution. Generally, coarser map resolutions are used for longer ranges. This approach eliminates the gaps in the map due to undersampling at long range from the sensor by using the optimal resolution based on sensor geometry [45.15].

Uncertainty in the sensor measurement is expressed most naturally with respect to the sensor frame, for example, by expressing the uncertainty in the direction of the measurement. As a result, converting such uncertainty models to elevation maps is difficult because a distribution in the direction of the measurement maps to a distribution in the reference plane, not a distribution on the elevation value at a particular grid point. When updating a cell based on sensory input, one has to take into account that the uncertainty in a measurement increases with the distance measured due to errors in the tilting angle. A popular approach to estimate the height h at location (x, y) is to use a Kalman filter. If we assume that σ is the standard deviation of the current measurement h in the vertical direction in (x, y) and σ_{t-1} is the standard deviation of the current estimate or h_{t-1}, we can apply the following equations to

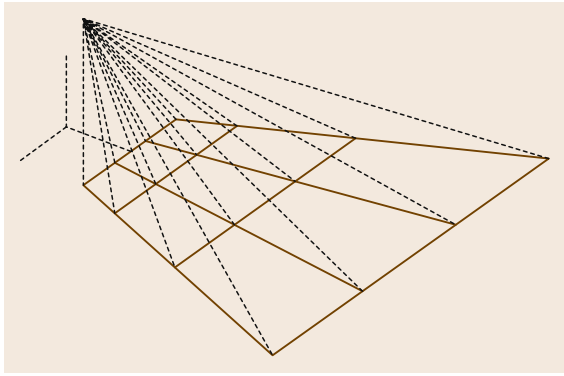


Fig. 45.9 Example of a robot-centric map with variable size (after [45.11])

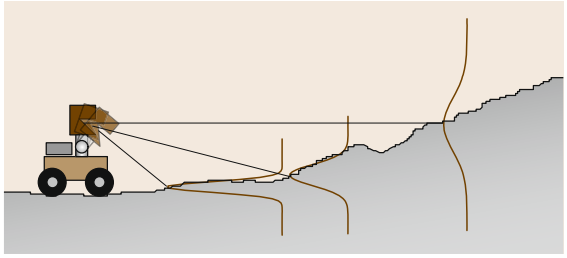


Fig. 45.10 The standard deviation of a height measurements can be modeled to depend linearly on the distance of the grid cell to the sensor

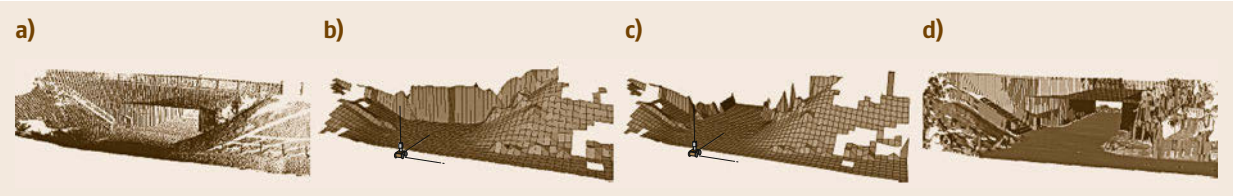


Fig.45.11a–d Different versions of digital elevation maps (after [45.16]): **(a)** scan (point set) of a bridge, **(b)** a standard elevation map computed from this data set, **(c)** an extended elevation map, which correctly represents the underpass under the bridge, and **(d)** a multilevel surface map that correctly represents the height of the vertical objects

obtain the new estimate h_t with standard deviation σ_t,

$$h_t = \frac{\sigma^2 h_{t-1} + \sigma_{t-1}^2 h}{\sigma_{t-1}^2 + \sigma^2} , \tag{45.17}$$

$$\sigma_t^2 = \frac{\sigma_{t-1}^2 \sigma^2}{\sigma_{t-1}^2 + \sigma^2} . \tag{45.18}$$

One possible solution to deal with varying uncertainties is to use a model in which the standard deviation of the height of a measurement increases linearly with the length of the range measurement, as indicated in Fig. 45.10.

Vegetation cover is another source of error in the recovery of terrain elevation by totally or partially occluding the ground surface from the vehicle sensors. Online learning techniques can address this issue by predicting ground elevation ahead of the vehicle based on past appearance observations of similar terrain and vehicle state over the terrain [45.17].

Intermediate representations between 2-D elevation maps and full 3-D representations, addressed below, are extended elevation maps [45.18] and so-called multilevel surface maps [45.16]. Such approaches are specifically useful in structured terrain with vertical objects or overhangs such as bridges (Fig. 45.11).

45.3.2 3-D Grids and Point Sets

Elevation maps as described above assume a reference direction. In many cases, that assumption is violated. An alternative is to represent the data directly in 3-D without projecting it on a reference 2-D plane. The advantage of doing this is that all the sensor data can be preserved in its original distribution and that there is no restriction on the geometry of the environment. Figure. 45.12 shows an example representation using 3-D points. The drawbacks of SLAM systems operating on such point clouds [45.19–21] is that neither free space nor unknown areas can be modeled and that sensor noise and dynamic objects cannot be dealt with directly. Another problem is that very large sets of 3-D points are difficult to handle efficiently. The point cloud library [45.22] addresses this problem by providing an octree-based implementation to store and address large point clouds in an efficient manner.

Data structures based on dynamic 3-D grids can also be used for this purpose [45.26]. An interesting characteristic of these representations is that they enable the computation of cost evaluated from the true local 3-D distribution of the data (as opposed to costs

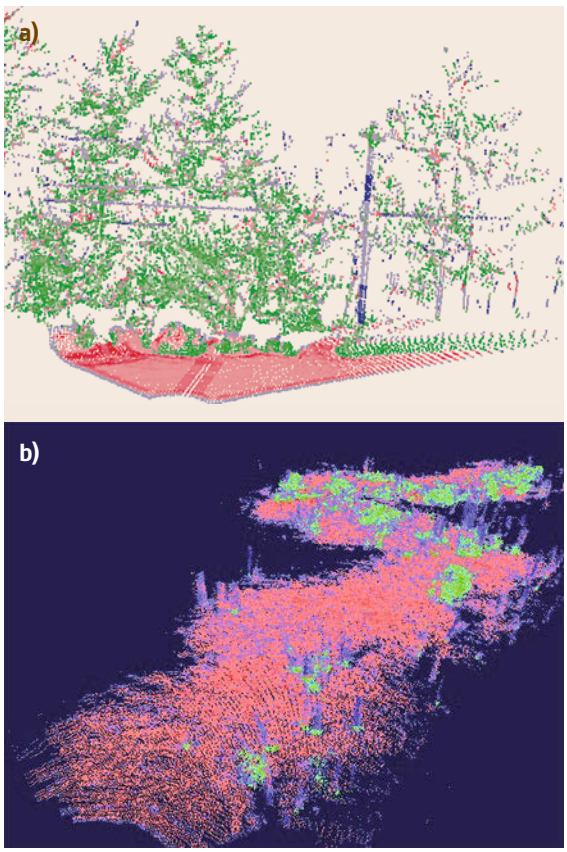


Fig.45.12a,b Example of mapping and classification of 3-D point sets: **(a)** 3-D data and classification result (*green* = vegetation, *red* = surfaces, *blue* = lines; the saturation of the color is proportional to the confidence in the classification result); **(b)** map accumulated over a large number of scans

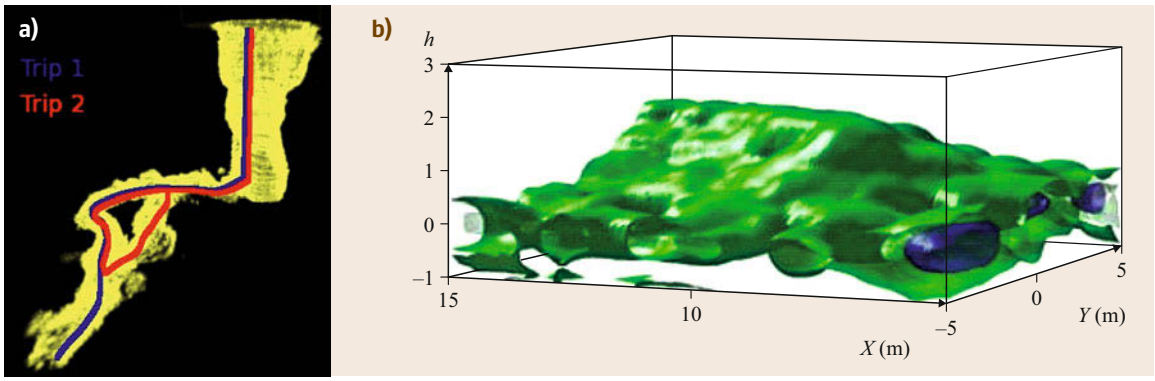


Fig.45.13a,b Examples of volumetric maps for a natural environment. Whereas **(a)** shows a map of an underwater cave (after [45.23]), **(b)** depicts a terrain map obtained from a radar (after [45.24])

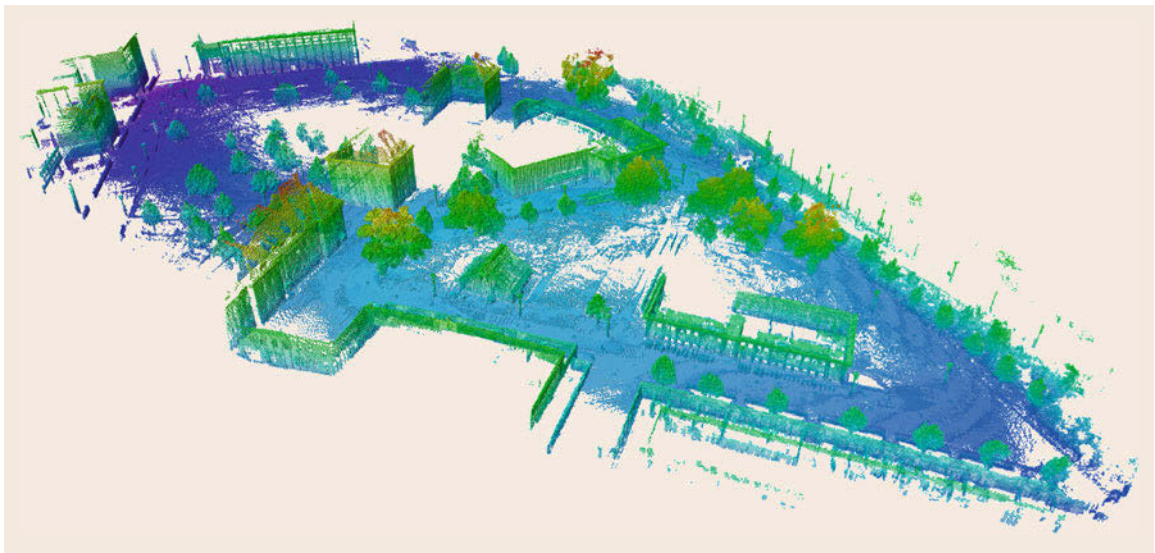


Fig. 45.14 Resulting octree-based representation of an outdoor environment at 0.2 m resolution (size of the scene: 292 m × 167 m × 28 m) (after [45.25]). For clarity, only occupied volumes are shown with height visualized by a color (*gray scale*) coding

computed from a local surface $z = f(x, y)$). This is important, for example, in environments with vegetation scatter, which cannot be modeled as a surface.

Extensions of occupancy grids to dense 3-D occupancy grids have been used successfully to model natural environments using a radar [45.24]. For more efficiently maintaining 3-D representations of even large-scale environments, also tree-based representations can been employed. In particular, octree-based 3-D representations have been employed, for example, to map an underwater cave from several sonars mounted on an underwater vehicle [45.23]. Example maps for these methods are presented in Fig. 45.13.

The OctoMap approach [45.27] uses octrees and supports a compact memory representation, multiresolution queries, and probabilistic occupancy estimates. The volumetric representation of space allows for explicitly representing free space and unknown areas. To exploit hierarchical dependencies in the environment it can be extended to maintain a collection of submaps in a tree-structure, where each node represents a subspace of the environment [45.28]. Figure. 45.14 and VIDEO 79 show an example outdoor map acquired from laser data and built by OctoMap. The OctoMap framework has been used successfully for various tasks including autonomous navigation with air vehicles [45.29], humanoid navigation in multilevel indoor environments [45.30], and mobile manipulation in cluttered environments [45.31, 32] where volumetric information is necessary to plan paths.

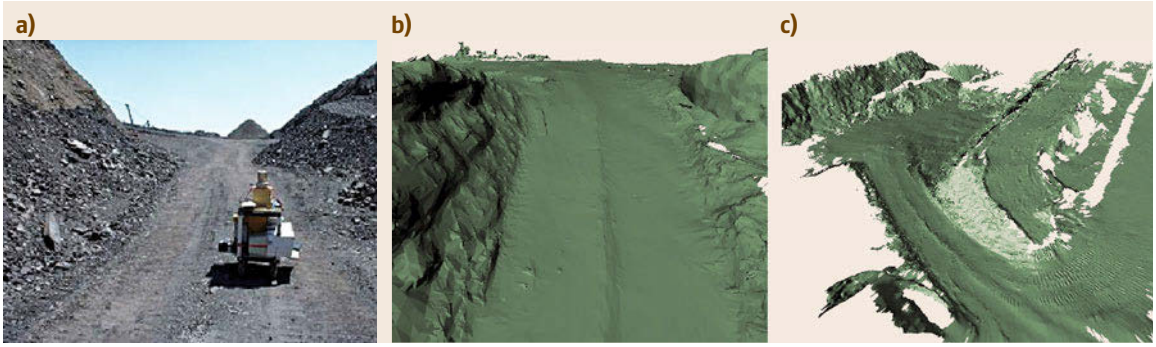


Fig.45.15a–c Two views (**b–c**) of the mesh representation of a map (**a**) obtained by scanning an environment.The meshes are reduced by a factor of 10 from the initial data

In the approaches mentioned above, the data sets are accumulated, sometimes probabilistically, into discrete environment volumes. A different approach is to represent the environment with discrete sensor samples integrated probabilistically into a metric map [45.33]. Such an approach is different from occupancy grids in several aspects: the storage requirement and resolution is adaptive instead of fixed and the potential for extensibility and scalability is higher.

45.3.3 Meshes

As described above, elevation maps are compact and easy to implement but they are restricted to a particular class of terrain; at the other extreme, using 3-D representations directly is more general but it is also more expensive computationally, and it does not represent explicitly surface continuity. A compromise is to represent the map by a mesh. This approach is attractive because it can, in principle, represent any combination of surfaces. It is also a compact representation in that, even though the size of the mesh may be initially very large, efficient mesh simplification algorithms exist in the literature [45.34] and can be used to reduce the map to a small number of vertices.

The key issue with meshes is that extraction of the correct surfaces from raw data can be difficult in complex environments. In practice, the data may be corrupted by sensor noise and by random clutter from other sources, such as vegetation, which cannot be represented as a continuous surface. In addition, it is necessary to precisely detect discontinuities in the data so that disconnected pieces of surface do not become accidentally linked in the mesh formation process [45.18]. Figure 45.15 shows typical mesh representations of an urban environment.

In some applications, the objective is to produce terrain models not to support autonomous navigation but to produce a model to be visualized to a human. City planning is a typical application. Examples include producing 3-D texture models of urban environments using videos from a camera mounted on a moving vehicle equipped with an inertial navigation system [45.35] (VIDEO 269), using two laser scanners in push-broom mode, one for mapping and the other for localization [45.36], with a camera coregistered with the mapping laser, and collecting high-resolution laser data and imagery to produce geometric and photometric correct 3-D models of buildings [45.37]. Figure 45.16 depicts examples of textured meshes representing urban terrain.

45.3.4 Cost Maps

The most direct use of elevation maps is to compute traversability costs at each cell of the grid. The costs are computed by comparing the local terrain shape with a kinematic model of the robot. Several approaches have been proposed for computing the costs depending on the exact vehicle model; for example, a cost can be computed by considering the local slope and 3-D texture of the terrain [45.38]. More involved models take into account a detailed model of the robot dynamics. In that case, different costs are generated for different possible robot speeds and path curvatures. The cost grid is used in a minimum-cost planner (VIDEO 271). Since the terrain map is updated continuously as the robot traverses the environment, it is also necessary to update the costs continuously as new data arrives. Consequently, it is important that the planner is able to support changing costs without needing to process the entire grid every time new data is inserted.

An example of such a combination of grid representation and dynamic planner is the D* system, which uses a version of A* that fully supports dynamic grid updates [45.39–41]. Whenever a grid cell (or a group of grid cells) is updated, the planner makes minimal updates to its internal representation so that the optimal path can be updated quickly.

Fig.45.16a–c Examples of urban terrain modeled as a textured mesh **(a)** (after [45.35]), **(b)** (after [45.37]), and **(c)** (after [45.36])

Defining the exact relation between the costs and the elevation values stored in the grid can be quite difficult. In fact, beyond the limiting cases of terrain with large elevation gradient or large slope, there is little guidance as to why a particular part of the terrain may be more traversable than another as it depends critically on the exact configuration of the robot. For this reason, recent work has focused on deriving cost maps directly from observations, rather than by using handcrafted algorithms. One approach involves learning the best weights to combine a set of predefined costs. Another approach uses costs computed from other sources for inferring how to determine the costs on the terrain currently seen by the robot; for example, data from overhead imagery can be used to predict the traversability costs that should be used for a ground robot [45.42] (Fig. 45.17) or for learning to map local elevation distributions to costs values by analyzing actual robot trajectories through the terrain. Similar online learning approaches are used to predict terrain roughness [45.43], terrain slippage [45.44], or traversability in general [45.45, 46] to be used in a cost map.

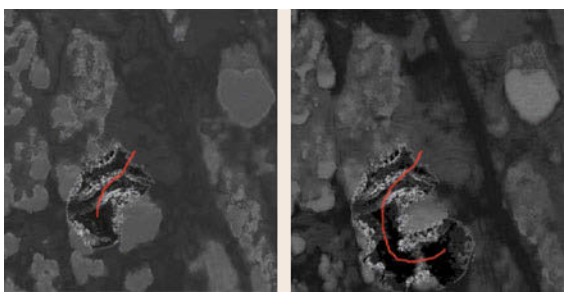


Fig. 45.17 Example of online learning of cost (after [45.42]). The vehicle trajectory is represented in *red*. As the vehicle progresses, note the change in cost over all the map. The *darker* the appearance, the *lower* the cost

45.3.5 Semantic Attributes

The representations described above are concerned only with storing the data in a way that is compact and that can be used to estimate drivability costs. Often it is necessary to reason with high-level knowledge about the environment, e.g., the location and types of objects around the robot or the type of terrain (vegetation, mud, wall) in the environment. For convenience we will refer to this type of information as *semantic attributes* attached to different parts of the environment. There are several different ways to approach the problem of generating and representing semantic attributes. One possible direction is to approach the problem as

one of extracting *landmarks* from the environment. As for their indoor counterparts, outdoor environments can also be modeled using landmarks. Landmarks are defined broadly as scene elements easy to detect, salient in their surroundings, and easy to recognize. They can be specific objects, such as rocks [45.47], tree trunks in forest [45.48], or locations with a distinct appearance signature in 2-D or 3-D, for urban terrain [45.49, 50] or natural environments [45.51]. Landmarks can be used to produce topological representations of the environment or used in conjunction with metric maps. Recent efforts involve learning a compact representation of environments without defining a priori what a landmark should be [45.52, 53] or using statistical learning techniques [45.54] to extract new features from images. Nonlinear dimension-reduction techniques are used to obtain the representation and the results were demonstrated onboard ground and aerial vehicles (Fig. 45.18).

Another view of the problem is as a *classification and grouping* problem. Indeed, the map representations described in the earlier sections are low-level representations in that they do not attempt to group the data points into larger structures that are coherent with respect to geometry, terrain type, or semantic content. In practice, one would like to abstract the data into larger units that can be used by a planner or transmitted to another robot or an operator; for example, in an urban environment, one would like to group the parts of the data that belong to planes corresponding to pieces of walls. One such example is presented in Fig. 45.19.

One approach is to group 3-D points into components based on low-level classification and feature detection [45.58]. The shape of these component is analyzed further to discriminate between various natural object components (tree trunks, branches) and man-made obstacle (wires). Geometric primitives are then fitted to the components (mesh for the ground surface, ball tree for the vegetation, and cylinder to branches) to produce a compact high-level geometric description of the terrain.

Local terrain classification can be achieved by computing local features from the map and classifying each element of the map in different classes of terrain. This can be done from an elevation map, in which case the *elements* are the cells of the elevation map, or from a point cloud representation in which case the elements are 3-D locations. Given a location in the map x, a feature vector is $\boldsymbol{V}(x)$ is computed in a neighborhood $N(x)$ of x and a classifier $f(V)$ returns the type of terrain at x. Features that have been used in the past are statistics of the distribution of the map data around each element, such as the slope and distribution of elevation [45.59], and the second-order moment of the distribution of the 3-D points in a neighborhood [45.58]. The terrain classes depend on the application. The most direct classification approach uses a binary classifier that separates the terrain into obstacle regions and traversable regions. More involved classifiers segment the map into more classes such as vegetation, solid surfaces, and linear structures [45.13, 58, 60]; an example is shown in Fig. 45.12. In some cases, it is possible to store, with each data element x in the map, the direction in which a measurement was taken for this element. In this case, it is possible to refine the classification by rea-

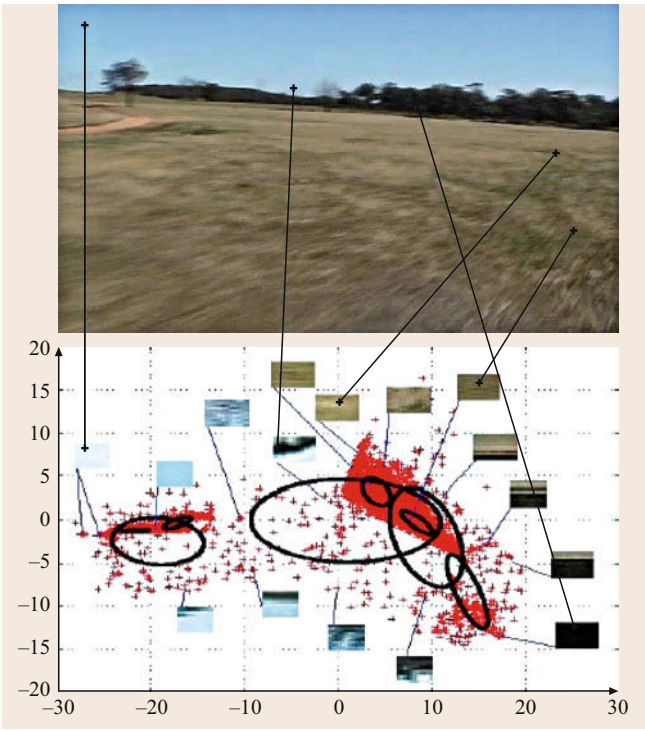


Fig. 45.18 Sample image and low-dimensional embedding of randomly sampled high-dimensional image patches (after [45.53])

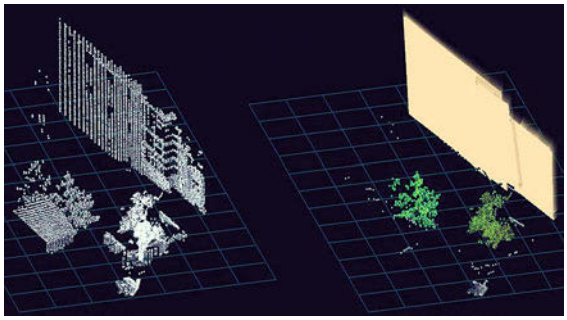


Fig. 45.19 Terrain classification and extraction of geometric features.Whereas the 3-D data is shown on the *left*, the feature map with extracted planes and vegetation regions is shown on the *right* (after [45.55])

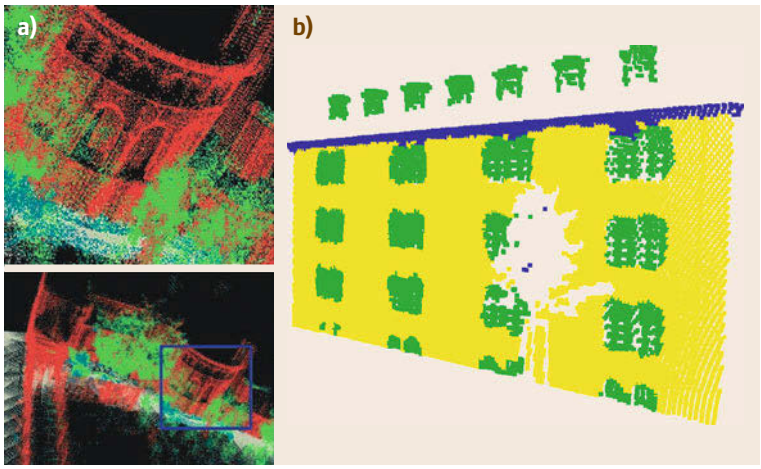


Fig. 45.20 (a) Structure learning for terrain classification (after [45.56]) and (b) building features extraction (after [45.57])

soning about the intersection of the measurement ray $d(x)$ with the rest of map; for example, this type of geometric reasoning [45.61] has been exploited to recover the load-bearing surface obscured by vegetation [45.51, 62, 63] and for extracting *negative* obstacles (such as ditches) [45.64]. In all of these cases, the classification information cannot be recovered directly from local statistics and must be inferred from longer-range geometric reasoning.

This class of approaches to local feature classification is similar to approaches for extracting features from images and it suffers from similar limitations. Specifically, these representations are sensitive to the choice of neighborhood used for computing the features. If it is too large, information over a large area is averaged out, leading to poor classification performance. If it is too small, there is not enough information in the neighborhood for reliable classification. The situation is complicated by the fact that the resolution of the map, or more precisely the density of data points in the map, may vary drastically as the distance from the sensor changes. This problem is addressed by using different neighborhood sizes at different locations in the map, or by tuning the classifier differently depending on the map location, typically based on distance from the sensor [45.15, 65]. A second problem is to generate the classifier f. This can be done by using a physical model that predicts the statistics of the local data distribution in the map assuming different terrain types [45.66]. This is generally difficult and a preferred approach is to train the classifier on training data.

This level of classification provides information about the local type of terrain, which can be used in planning, but it does not extract the extended geometric structures that may exist in the environment (Fig. 45.19). Geometric structures such as planar patches can in principle be extracted by using techniques similar to the ones described in the context of indoor environments, such as EM, with the added difficulty that there is a larger amount of clutter that complicates the extraction of the patches. Robust techniques that can handle this level of clutter are typically used for extracting the planes [45.67, 68].

Classification based on local attributes can be improved significantly by taking into account contextual information; for example, hidden Markov models have been successfully used for laser data analysis to determine terrain traversability [45.69]. Another structure learning approach based on Markov random fields in conjunction with margin-maximization criterion has been demonstrated for a wide class of applications, including 3-D terrain classification and object segmentation [45.56] or building structures extraction [45.57] (Fig. 45.20). Finally there are approaches to detect, select, model, and recognize natural landmarks automatically for robot localization and environment mapping [45.47].

45.3.6 Heterogeneous and Hierarchical Models

For long-range navigation, an autonomous vehicle must perform numerous tasks, including absolute and relative localization, path planning, and reactive obstacle avoidance. In addition, it must perform some tasks to fulfill the mission requirements, detecting and modeling objects, for example. To achieve this, a hierarchical framework has been presented that accounts for the different scales and granularities of the representation needed [45.59]. Furthermore, models built from heterogeneous imagery sources (overhead, descent, and ground) have been used to produce 3-D multiresolution terrain models that support Mars exploration [45.70]. The hybrid metric map [45.71], enhances feature maps

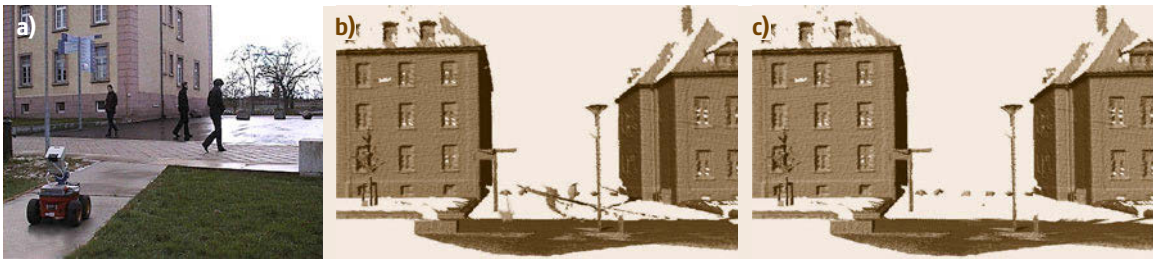


Fig. 45.21 (a) A mobile robot acquiring a three-dimensional scan of an urban scene. (b) The people in the scene cause spurious data points in the resulting meshes. (c) The same scene after filtering people

with metric maps to provide a dense but compact environment representation. Another representation extracts landmarks based on laser/imagery appearance as well as a metric map [45.72].

45.4 Dynamic Environments

The majority of techniques for mapping have been developed for static environments. Certain approaches like occupancy grids or elevation maps can in principle deal with dynamic environments in which objects move. Their drawback lies in the fact that the time to unlearn that a cell is free or that the elevation has changed can require as many observations as the robot received with the same area being occupied or with a different elevation. To resolve this problem, several alternative approaches have been proposed in the past. One very popular technique is to track moving objects using feature-based tracking algorithms [45.73, 74]. Such approaches are especially useful when the type of the dynamic objects is known in advance. They have been successfully applied in the context of learning three-dimensional city maps from range data. Figures 45.21 and 45.22 show applications in which dynamic objects were removed from the 3-D data acquired with mobile robots in urban scenes. Alternative approaches to such tracking techniques include those that learn maps on different time scales [45.75], that explicitly learn different states of dynamic environments [45.76, 77], or that only map the static aspects [45.78].

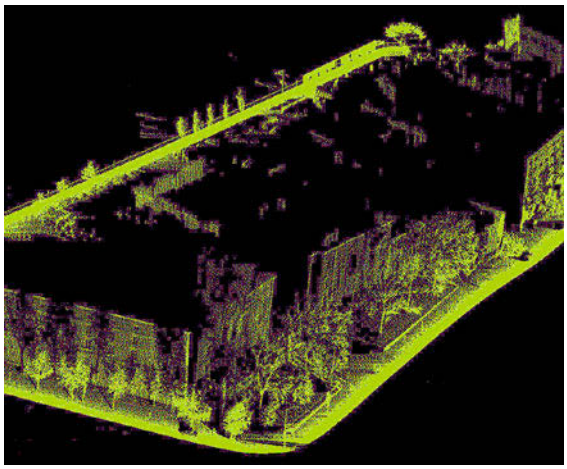


Fig. 45.22 Complex three-dimensional scene acquired with a mobile robot after filtering dynamic objects (after [45.74])

45.5 Summary and Further Reading

Further reading about typical representations and how they can be utilized for mobile robot navigation can be found in recent textbooks on mobile robotics [45.2, 79, 80]. A comprehensive survey of the fundamentals of spatial data structures and their applications is available in *Samet* [45.81].

Video-References

VIDEO 79 OctoMap visualization
available from http://handbookofrobotics.org/view-chapter/45/videodetails/79

VIDEO 269 3-D textured model of urban environments
available from http://handbookofrobotics.org/view-chapter/45/videodetails/269

VIDEO 270 Service robot navigation in urban environments
available from http://handbookofrobotics.org/view-chapter/45/videodetails/270

VIDEO 271 Learning navigation cost grids
available from http://handbookofrobotics.org/view-chapter/45/videodetails/271

References

45.1 H.P. Moravec, A.E. Elfes: High resolution maps from wide angle sonar, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1985)

45.2 H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, S. Thrun: *Principles of Robot Motion: Theory, Algorithms and Implementation* (MIT Press, Cambridge 2005)

45.3 D.H. Douglas, T.K. Peucker: Algorithms for the reduction of the number of points required to represent a line or its caricature, Cdn. Cartogr. **10**(2), 112–122 (1973)

45.4 D. Sack, W. Burgard: A comparison of methods for line extraction from range data, Proc. IFAC Symp. Intell. Auton. Veh. (IAV) (2004)

45.5 P. Beeson, N.K. Jong, B. Kuipers: Towards autonomous topological place detection using the extended Voronoi graph, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

45.6 M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit: FastSLAM: A factored solution to the simultaneous localization and mapping problem, Proc. Nat. Conf. Artif. Intell. (AAAI) (2002)

45.7 B.J. Kuipers, Y.-T. Byun: A robust qualitative method for spatial learning in unknown environments, Proc. Nat. Conf. Artif. Intell. (AAAI) (1988)

45.8 H. Choset, K. Nagatani: Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization, IEEE Trans. Robotics Autom. **17**(2), 125–137 (2001)

45.9 S. Thrun: Robotic mapping: A survey. In: *Exploring Artificial Intelligence in the New Millenium*, ed. by G. Lakemeyer, B. Nebel (Morgan Kaufmann, San Diego 2003)

45.10 M. Maimone, P. Leger, J. Biesiadecki: Overview of the Mars exploration rovers' autonomous mobility and vision capabilities, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)

45.11 S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury, M. Herrb, R. Chatila: Autonomous rover navigation on unknown terrains: functions and integration, Int. J. Robotics Res. **21**(10-11), 917–942 (2002)

45.12 R. Olea: *Geostatistics for Engineers and Earth Scientists* (Kluwer, Boston 1999)

45.13 A. Kelly, A. Stentz, O. Amidi, M. Bode, D. Bradley, A. Diaz-Calderon, M. Happold, H. Herman, R. Mandelbaum, T. Pilarki, P. Rander, S. Thayer, N. Vallidi, R. Warner: Toward reliable off road autonomous vehicles operating in challenging environments, Int. J. Robotics Res. **25**(5-6), 449–483 (2006)

45.14 I.S. Kweon, T. Kanade: High-resolution terrain map from multiple sensor data, IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 278–292 (1992)

45.15 M. Montemerlo, S. Thrun: A multi-resolution pyramid for outdoor robot terrain perception, Proc. AAAI Nat. Conf. Artif. Intel., San Jose (2004)

45.16 R. Triebel, P. Pfaff, W. Burgard: Multi-level surface maps for outdoor terrain mapping and loop closing, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2006)

45.17 C. Wellington, A. Courville, A. Stentz: A generative model of terrain for autonomous navigation in vegetation, Int. J. Robotics Res. **25**(12), 1287–1304 (2006)

45.18 P. Pfaff, R. Triebel, W. Burgard: An efficient extension to elevation maps for outdoor terrain mapping and loop closing, Int. J. Robotics Res. **26**(2), 217–230 (2007)

45.19 D.M. Cole, P.M. Newman: Using laser range data for 3D SLAM in outdoor environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

45.20 A. Nüchter, K. Lingemann, J. Hertzberg, H. Surmann: 6D SLAM – 3D mapping outdoor environments: Research articles, J. Field Robotics **24**(8-9), 699–722 (2007)

45.21 J. Elseberg, D. Borrmann, A. Nüchter: Efficient processing of large 3D point clouds, Proc. 23rd Int. Symp. Infor. Commun. Autom. Technol. (ICAT) (2011)

45.22 R.B. Rusu, S. Cousins: 3D is here: Point cloud library (PCL), Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

45.23 N. Fairfield, G. Kantor, D. Wettergreen: Real-time SLAM with octree evidence grids for exploration in underwater tunnels, J. Field Robotics **24**(1), 3–21 (2007)

45.24 A. Foessel: Scene Modeling from Motion-Free Radar Sensing, Ph.D. Thesis (Carnegie Mellon Univ., Pittsburgh 2002)

45.25 K.M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, W. Burgard: OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems, Proc. ICRA Workshop Best Pract. 3D Percept. Model. Mob. Manip. (2010)

45.26 J.-F. Lalonde, N. Vandapel, M. Hebert: Data structure for efficient processing in 3-D, Proc. Robotics Sci. Syst., Vol. I (2005) p. 48

45.27 A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard: OctoMap: An efficient probabilistic 3D mapping framework based on octrees, Auton. Robots **34**(3), 189–206 (2013)

45.28 K.M. Wurm, D. Hennes, D. Holz, R.B. Rusu, C. Stachniss, K. Konolige, W. Burgard: Hierarchies of octrees for efficient 3D mapping, Proc. IEEE/RSJ Int Conf. Intell. Robots Syst. (IROS) (2011)

45.29 L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, M. Pollefeys: Autonomous obstacle avoidance and maneuvering on a vision-guided MAV using onboard processing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

45.30 S. Oßwald, A. Hornung, M. Bennewitz: Improved proposals for highly accurate localization using range and vision data, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

45.31 M. Ciocarlie, K. Hsiao, E.G. Jones, S. Chitta, R.B. Rusu, I.A. Sucan: Towards reliable grasping and manipulation in household environments, Int. Symp. Exp. Robotics (ISER) (2010)

45.32 A. Hornung, M. Phillips, E.G. Jones, M. Bennewitz, M. Likhachev, S. Chitta: Navigation in three-dimensional cluttered environments for mobile manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

45.33 J. Leal: Stochastic Environment Representation, Ph.D. Thesis (Univ. of Sydney, Sydney 2003)

45.34 P. Heckbert, M. Garland: Optimal triangulation and quadric-based surface simplification, J. Comput. Geom. Theory Appl. **14**(1-3), 49–65 (1999)

45.35 A. Akbarzadeh: Towards urban 3d reconstruction from video, Proc. Int. Symp. 3D Data Vis. Transm. (2006)

45.36 C. Frueh, S. Jain, A. Zakhor: Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images, Int. J. Comput. Vis. **61**(2), 159–184 (2005)

45.37 I. Stamos, P. Allen: Geometry and texture recovery of scenes of large scales, Comput. Vis. Image Underst. **88**, 94–118 (2002)

45.38 D. Gennery: Traversability analysis and path planning for a planetary rover, Auton. Robotics **6**, 131–146 (1999)

45.39 D. Ferguson, A. Stentz: The delayed D* algorithm for efficient path replanning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

45.40 D. Ferguson, A. Stentz: Field D*: An interpolation-based path planner and replanner, Proc. Int. Symp. Robotics Res. (ISRR) (2005)

45.41 M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, S. Thrun: Anytime dynamic A*: An anytime, replanning algorithm, Proc. Int. Conf. Autom. Plan. Sched. (ICAPS) (2005)

45.42 B. Sofman, E. Lin, J. Bagnell, J. Cole, N. Vandapel, A. Stentz: Improving robot navigation through self-supervised online learning, J. Field Robotics **23**(12), 1059–1075 (2006)

45.43 D. Stavens, S. Thrun: A self-supervised terrain roughness estimator for off-road autonomous driving, Uncertainty Artif. Intell., Boston (2006)

45.44 A. Angelova, L. Matthies, D. Helmick, P. Perona: Slip prediction using visual information, Robotics Sci. Syst., Philadelphia (2006)

45.45 D. Kim, J. Sun, S. Oh, J. Rehg, A. Bobick: Traversability classification using unsupervised online visual learning for outdoor robot navigation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

45.46 S. Thrun, M. Montemerlo, A. Aron: Probabilistic terrain analysis for high-speed desert driving, Robotics Sci. Syst. (2005)

45.47 R. Murrieta-Cid, C. Parra, M. Devy: Visual navigation in natural environments: From range and color data to a landmark-based model, Auton. Robotics **13**(2), 143–168 (2002)

45.48 D. Asmar, J. Zelek, S. Abdallah: Tree trunks as landmarks for outdoor vision SLAM, Proc. Conf. Comp. Vis. Pattern Recogn. Workshop (2006)

45.49 I. Posner, D. Schroeter, P. Newman: Using scene similarity for place labelling, Int. Symp. Exp. Robotics (2006)

45.50 A. Torralba, K.P. Murphy, W.T. Freeman, M.A. Rubin: Context-based vision system for place and object recognition, Proc. IEEE Int. Conf. Comput. Vis. (ICCV) (2003)

45.51 D. Bradley, S. Thayer, A. Stentz, P. Rander: *Vegetation Detection for Mobile Robot Navigation*, *Tech. Rep. CMU-RI-TR-04-12* (Carnegie Mellon Univ., Pittsburgh 2004)

45.52 S. Kumar, J. Guivant, H. Durrant-Whyte: Informative representations of unstructured environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

45.53 S. Kumar, F. Ramos, B. Douillard, M. Ridley, H. Durrant-Whyte: A novel visual perception framework, Proc. 9th Int. Conf. Control Autom. Robotics Vis. (ICARCV) (2006)

45.54 F. Ramos, S. Kumar, B. Upcroft, H. Durrant-Whyte: Representing natural objects in unstructured environments, NIPS Workshop Mach. Learn. Robotics (2005)

45.55 C. Pantofaru, R. Unnikrishnan, M. Hebert: Toward generating labeled maps from color and range data for robot navigation, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (2003)

45.56 D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, A. Ng: Discriminative learning of Markov random fields for segmentation of 3D scan data, Proc. Conf. Comp. Vis. Pattern Recogn. (CVPR) (2005)

45.57 R. Triebel, K. Kersting, W. Burgard: Robust 3D scan point classification using associative Markov networks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

45.58 J.F. Lalonde, N. Vandapel, D. Huber, M. Hebert: Natural terrain classification using three-dimensional ladar data for ground robot mobility, J. Field Robotics **23**(10), 839–861 (2006)

45.59 M. Devy, R. Chatila, P. Fillatreau, S. Lacroix, F. Nashashibi: On autonomous navigation in a natural environment, Robotics Auton. Syst. **16**(1), 5–16 (1995)

45.60 R. Manduchi, A. Castano, A. Talukder, L. Matthies: Obstacle detection and terrain classification for autonomous off-road navigation, Auton. Robotics **18**(1), 81–102 (2005)

45.61 D. Huber, M. Hebert: 3D modeling using a statistical sensor model and stochastic search, Proc. IEEE Conf. Comput. Vision Pattern Recogn. (CVPR) (2003) pp. 858–865

45.62 S. Balakirsky, A. Lacaze: World modeling and behavior generation for autonomous ground vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000)

45.63 A. Lacaze, K. Murphy, M. Delgiorno: Autonomous mobility for the demo III experimental unmanned vehicles, Proc. AUVSI Int. Conf. Unmanned Veh. (2002)

45.64 P. Bellutta, R. Manduchi, L. Matthies, K. Owens, A. Rankin: Terrain perception for demo III, Proc. Intell. Veh. Symp. (2000)

45.65 J.F. Lalonde, R. Unnikrishnan, N. Vandapel, M. Hebert: Scale selection for classification of point-sampled 3D surfaces, Proc. 5th Int. Conf. 3-D Digital Imaging Model. (3DIM) (2005)

45.66 J. Macedo, R. Manduchi, L. Matthies: Ladar-based discrimination of grass from obstacles for autonomous navigation, Proc. 7th Int. Symp. Exp. Robotics (ISER) (2000)

45.67 H. Chen, P. Meer, D. Tyler: Robust regression for data with multiple structures, Proc. IEEE Int. Conf. Comput. Vis. Pattern Recogn. (CVPR) (2001)

45.68 R. Unnikrishnan, M. Hebert: Robust extraction of multiple structures from non-uniformly sampled data, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2003)

45.69 D. Wolf, G. Sukhatme, D. Fox, W. Burgard: Autonomous terrain mapping and classification using hidden Markov models, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

45.70 C. Olson, L. Matthies, J. Wright, R. Li, K. Di: Visual terrain mapping for Mars exploration, Comput. Vis. Underst. **105**, 73–85 (2007)

45.71 J. Nieto, J. Guivant, E. Nebot: The hybrid metric maps (hymms): A novel map representation for denseSLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

45.72 F. Ramos, J. Nieto, H. Durrant-Whyte: Recognising and modelling landmarks to close loops in outdoor SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)

45.73 D. Hähnel, D. Schulz, W. Burgard: Mobile robot mapping in populated environments, Adv. Robotics **17**(7), 579–598 (2003)

45.74 C.-C. Wang, C. Thorpe, S. Thrun: Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

45.75 P. Biber, T. Duckett: Dynamic maps for long-term operation of mobile service robots, Proc. Robotics Sci. Syst. (2005)

45.76 D. Meyer-Delius, J. Hess, G. Grisetti, W. Burgard: Temporary maps for robust localization in semi-static environments, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Taipei (2010)

45.77 C. Stachniss, W. Burgard: Mobile robot mapping and localization in non-static environments, Proc. Nat. Conf. Artif. Intell. (AAAI), Pittsburgh (2005)

45.78 D. Hähnel, R. Triebel, W. Burgard, S. Thrun: Map building with mobile robots in dynamic environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

45.79 R. Siegwart, I. Nourbakhsh: *Introduction to Autonomous Mobile Robots* (MIT Press, Cambridge 2001)

45.80 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)

45.81 H. Samet: *Foundations of Multidimensional and Metric Data Structures* (Elsevier, Amsterdam 2006)

Multimedia Contents

46. Simultaneous Localization and Mapping

Cyrill Stachniss, John J. Leonard, Sebastian Thrun

This chapter provides a comprehensive introduction in to the *simultaneous localization and mapping problem*, better known in its abbreviated form as SLAM. SLAM addresses the main perception problem of a robot navigating an unknown environment. While navigating the environment, the robot seeks to acquire a map thereof, and at the same time it wishes to localize itself using its map. The use of SLAM problems can be motivated in two different ways: one might be interested in detailed environment models, or one might seek to maintain an accurate sense of a mobile robot's location. SLAM serves both of these purposes.

We review the three major paradigms from which many published methods for SLAM are derived: (1) the extended Kalman filter (EKF); (2) particle filtering; and (3) graph optimization. We also review recent work in three-dimensional (3-D) SLAM using visual and red green blue distance-sensors (RGB-D), and close with a discussion of open research problems in robotic mapping.

46.1 **SLAM: Problem Definition** 1154
46.1.1 Mathematical Basis 1154
46.1.2 Example: SLAM in Landmark Worlds 1155
46.1.3 Taxonomy of the SLAM Problem .. 1156
46.2 **The Three Main SLAM Paradigms** 1157
46.2.1 Extended Kalman Filters 1157
46.2.2 Particle Methods 1159
46.2.3 Graph-Based Optimization Techniques..... 1162
46.2.4 Relation of Paradigms..... 1166
46.3 **Visual and RGB-D SLAM** 1166
46.4 **Conclusion and Future Challenges** 1169
Video-References..... 1170
References..... 1171

This chapter provides a comprehensive introduction into one of the key enabling technologies of mobile robot navigation: *simultaneous localization and mapping*, or in short *SLAM*. SLAM addresses the problem of acquiring a spatial map of an environment while simultaneously localizing the robot relative to this model. The SLAM problem is generally regarded as one of the most important problems in the pursuit of building truly autonomous mobile robots. It is of great practical importance; if a robust, general-purpose solution to SLAM can be found, then many new applications of mobile robotics will become possible.

While the problem is deceptively easy to state, it presents many challenges, despite significant progress made in this area. At present, we have robust methods for mapping environments that are mainly static, structured, and of limited size. Robustly mapping unstructured, dynamic, and large-scale environments in an online fashion remains largely an open research problem.

The historical roots of methods that can be applied to address the SLAM problem can be traced back to Gauss [46.1], who is largely credited for inventing the least squares method. In the Twentieth Century, a number of fields outside robotics have studied the making of environment models from a moving sensor platform, most notably in *photogrammetry* [46.2–4] and *computer vision* [46.5]. Strongly related problems in these fields are bundle adjustment and structure from motion. SLAM builds on this work, often extending the basic paradigms into more scalable algorithms. Modern SLAM systems often view the estimation problem as solving a sparse graph of constraints and applying

nonlinear optimization to compute the map and the trajectory of the robot. As we strive to enable long-lived autonomous robots, an emerging challenge is to handle massive sensor data streams.

This chapter begins with a definition of the SLAM problem, which shall include a brief taxonomy of different versions of the problem. The centerpiece of this chapter is a layman introduction into the three major paradigms in this field, and the various extensions that exist. As the reader will quickly recognize, there is no single best solution to the SLAM method. The method chosen by the practitioner will depend on a number of factors, such as the desired map resolution, the update time, and the nature of the features in the map, and so on. Nevertheless, the three methods discussed in this chapter cover the major paradigms in this field.

For more a detailed treatment of SLAM, we refer the reader to Durrant-Whyte and Bailey [46.6, 7], who provide an in-depth tutorial for SLAM, Grisetti et al. for a tutorial on graph-based SLAM [46.8], and Thrun et al., which dedicates a number of chapters to the topic of SLAM [46.9].

46.1 SLAM: Problem Definition

The SLAM problem is defined as follows: A mobile robot roams an unknown environment, starting at an initial location x_0. Its motion is uncertain, making it gradually more difficult to determine its current pose in global coordinates. As it roams, the robot can sense its environment with a noisy sensor. The SLAM problem is the problem of building a map of the environment while simultaneously determining the robot's position relative to this map given noisy data.

46.1.1 Mathematical Basis

Formally, SLAM is best described in probabilistic terminology. Let us denote time by t, and the robot location by x_t. For mobile robots on a flat ground, x_t is usually a three-dimensional vector, comprising its two-dimensional (2-D) coordinate in the plane plus a single rotational value for its orientation. The sequence of locations, or *path*, is then given as

$$X_T = \{x_0, x_1, x_2, \ldots, x_T\} \,. \tag{46.1}$$

Here T is some terminal time (T might be ∞). The initial location x_0 often serves as a point of reference for the estimation algorithm; other positions cannot be sensed.

Odometry provides relative information between two consecutive locations. Let u_t denote the odometry that characterized the motion between time $t-1$ and time t; such data might be obtained from the robot's wheel encoders or from the controls given to those motors. Then the sequence

$$U_T = \{u_1, u_2, u_3 \ldots, u_T\} \tag{46.2}$$

characterizes the relative motion of the robot. For noise-free motion, U_T would be sufficient to recover the poses from the initial location x_0. However, odometry measurements are noisy, and path integration techniques inevitably diverge from the truth.

Finally, the robot senses objects in the environment. Let m denote the *true* map of the environment. The environment may be comprised of landmarks, objects, surfaces, etc., and m describes their locations. The environment map m is often assumed to be time-invariant, i. e., static.

The robot measurements establish information between features in m and the robot location x_t. If we, without loss of generality, assume that the robot takes exactly one measurement at each point in time, the sequence of measurements is given as

$$Z_T = \{z_1, z_2, z_3, \ldots, z_T\} \,. \tag{46.3}$$

Figure 46.1 illustrates the variables involved in the SLAM problem. It shows the sequence of locations and sensor measurements, and the causal relationships between these variables. This diagram represents a *graph-*

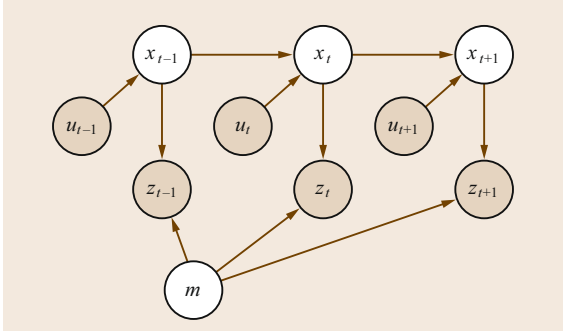


Fig. 46.1 Graphical model of the SLAM problem. Arcs indicate causal relationships, and shaded nodes are directly observable to the robot. In SLAM, the robot seeks to recover the unobservable variables

ical model for SLAM. It is useful in understanding the dependencies in the problem at hand.

The SLAM problem is now the problem of recovering a model of the world m and the sequence of robot locations X_T from the odometry and measurement data. The literature distinguishes two main forms of the SLAM problem, which are both of equal practical importance. One is known as the *full SLAM problem*: it involves estimating the posterior over the entire robot path together with the map

$$p(X_T, m \mid Z_T, U_T) \,. \tag{46.4}$$

Written in this way, the full SLAM problem is the problem of calculating the joint posterior probability over X_T and m from the available data. Notice that the variables right of the conditioning bar are all directly observable to the robot, whereas those on the left are the ones that we want. As we shall see, algorithms for the full SLAM problem are often batch, that is, they process all data at the same time.

The second, equally important SLAM problem is the *online SLAM problem*. This problem is defined via

$$p(x_t, m \mid Z_t, U_t) \,. \tag{46.5}$$

Online SLAM seeks to recover the present robot location, instead of the entire path. Algorithms that address the online problem are usually incremental and can process one data item at a time. In the literature, such algorithms are typically called *filters*.

To *solve* the SLAM problem, the robot needs to be endowed with two more models: a mathematical model that relates odometry measurements u_t to robot locations x_{t-1} and x_t; and a model that relates measurements z_t to the environment m and the robot location x_t. These models correspond to the arcs in Fig. 46.1.

In SLAM, it is common to think of those mathematical models as probability distributions: $p(x_t \mid x_{t-1}, u_t)$ characterizes the probability distribution of the location x_t assuming that a robot started at a known location x_{t-1} and measured the odometry data u_t. And likewise, $p(z_t \mid x_t, m)$ is the probability for measuring z_t if this measurement is taken at a known location x_t in a known environment m. Of course, in the SLAM problem we do *not* know the robot location, and neither do we know the environment. As we shall see, Bayes rule takes care of this, by transforming these mathematical relationships into a form where we can *recover* probability distributions over those latent variables from the measured data.

46.1.2 Example: SLAM in Landmark Worlds

One common setting of SLAM involves an assumption that the environment is populated by point-landmarks. When building 2-D maps, point-landmarks may correspond to door posts and corners of rooms, which, when projected into a 2-D map are characterized by a point coordinate. In a 2-D world, each point-landmark is characterized by two coordinate values. Hence the world is a vector of size $2N$, where N is the number of point-landmarks in the world. In a commonly studied setting, the robot can sense three things: the relative range to nearby landmarks, their relative bearing, and the identity of these landmarks. The range and bearing may be noisy, but in the most simple case the identity of the sensed landmarks is known perfectly. Determining the identity of the sensed landmarks is also known as the *data association problem*. In practice, it is one of the most difficult problems in SLAM.

To model the above described setup, one begins with defining the *exact*, noise-free measurement function. The measurement function h describes the workings of the sensors: it accepts as input a description of the environment m and a robot location x_t, and it computes the measurement

$$h(x_t, m) \,. \tag{46.6}$$

Computing h is straightforward in our simplified landmark setting; it is a simple exercise in trigonometry. The probabilistic measurement model can be derived from this measurement function by adding a noise term. It is a probability distribution that peaks at the noise-free value $h(x_t, m)$ but allows for measurement noise, for example,

$$p(z_t \mid x_t, m) = \mathcal{N}(h(x_t, m), \mathbf{Q}_t) \,. \tag{46.7}$$

Here $\mathcal{N}$ denotes the 2-D normal distribution, which is centered at $h(x_t, m)$. The 2-by-2 matrix $\mathbf{Q}_t$ is the noise covariance, indexed by time.

The motion model is derived from a kinematic model of robot motion. Given the location vector x_{t-1} and the motion u_t, textbook kinematics tells us how to calculate x_t. Let this function be denoted by g

$$g(x_{t-1}, u_t) \,. \tag{46.8}$$

The motion model may then be defined by a normal distribution centered at $g(x_{t-1}, u_t)$ but subject to Gaussian noise

$$p(x_t \mid x_{t-1}, u_t) = \mathcal{N}(g(x_{t-1}, u_t), \mathbf{R}_t) \,. \tag{46.9}$$

Here $\mathbf{R}_t$ is a covariance. It is of size 3-by-3, since the location is a three-dimensional 3-D vector.

With these definitions, we have all we need to develop a SLAM algorithm. While in the literature,

point-landmark problems with range-bearing sensing are by far the most studied, SLAM algorithms are not confined to landmark worlds. But no matter what the map representation and the sensor modality, any SLAM algorithm needs a similarly crisp definition of the features in m, the measurement model $p(z_t \mid x_t, m)$, and the motion model $p(x_t \mid x_{t-1}, u_t)$. Note that none of those distributions has to be restricted to Gaussian noise as done in the example above.

46.1.3 Taxonomy of the SLAM Problem

SLAM problems are distinguished along a number of different dimensions. Most important research papers identify the type of problems addressed by making the underlying assumptions explicit. We already encountered one such distinction: full versus online. Other common distinctions are as follows:

Volumetric Versus Feature-Based

In volumetric SLAM, the map is sampled at a resolution high enough to allow for photo-realistic reconstruction of the environment. The map m in volumetric SLAM is usually quite high-dimensional, with the result that the computation can be quite involved. Feature-based SLAM extracts sparse features from the sensor stream. The map is then only comprised of features. Our point-landmark example is an instance of feature-based SLAM. Feature-based SLAM techniques tend to be more efficient, but their results may be inferior to volumetric SLAM due to the fact that the extraction of features discards information in the sensor measurements.

Topological Versus Metric

Some mapping techniques recover only a qualitative description of the environment, which characterizes the relation of basic locations. Such methods are known as topological. A topological map might be defined over a set of distinct places and a set of qualitative relations between these places (e.g., place A is adjacent to place B). Metric SLAM methods provide metric information between the relation of such places. In recent years, topological methods have fallen out of fashion, despite ample evidence that humans often use topological information for navigation.

Known Versus Unknown Correspondence

The correspondence problem is the problem of relating the identity of sensed things to other sensed things. In the landmark example above, we assumed that the identity of landmarks is known. Some SLAM algorithms make such an assumption, others do not. The ones that do not provide special mechanisms for estimating the correspondence of measured features to previously observed landmarks in the map. The problem of estimating the correspondence is known as *data association problem*. It is one of the most difficult problems in SLAM.

Static Versus Dynamic

Static SLAM algorithms assume that the environment does not change over time. Dynamic methods allow for changes in the environment. The vast literature on SLAM assumes static environments. Dynamic effects are often treated just as measurement outliers. Methods that reason about motion in the environment are more involved, but they tend to be more robust in most applications.

Small Versus Large Uncertainty

SLAM problems are distinguished by the degree of location uncertainty that they can handle. The most simple SLAM algorithms allow only for small errors in the location estimate. They are good for situations in which a robot goes down a path that does not intersect itself, and then returns along the same path. In many environments it is possible to reach the same location from multiple directions. Here the robot may accrue a large amount of uncertainty. This problem is known as the *loop closing problem*. When closing a loop, the uncertainty may be large. The ability to close loops is a key characteristic of modern-day SLAM algorithms. The uncertainty can be reduced if the robot can sense information about its position in some absolute coordinate frame, e.g., through the use of a satellite-based global positioning system (GPS) receiver.

Active Versus Passive

In passive SLAM algorithms, some other entity controls the robot, and the SLAM algorithm is purely observing. The vast majority of algorithms are of this type; they give the robot designer the freedom to implement arbitrary motion controllers, and pursue arbitrary motion objectives. In active SLAM, the robot actively explores its environment in the pursuit of an accurate map. Active SLAM methods tend to yield more accurate maps in less time, but they constrain the robot motion. There exist hybrid techniques in which the SLAM algorithm controls only the pointing direction of the robot's sensors, but not the motion direction.

Single-Robot Versus Multi-Robot

Most SLAM problems are defined for a single robot platform, although recently the problem of multi-robot exploration has gained in popularity. Multi-robot SLAM problems come in many flavors. In some, robots get to observe each other, in others, robots are told their relative initial locations. Multirobot SLAM problems

are also distinguished by the type of communication allowed between the different robots. In some, the robots can communicate with no latency and infinite bandwidth. More realistic are setups in which only nearby robots can communicate, and the communication is subject to latency and bandwidth limitations.

Any-Time and Any-Space

Robots that do all computations onboard have limited resources in memory and computation power. Anytime and any-space SLAM systems are an alternative to traditional methods. They enable the robot to compute a solution given the resource constraints of the system. The more resources available, the better the solution.

As this taxonomy suggests, there exists a flurry of SLAM algorithms. Most modern-day conferences dedicate multiple sessions to SLAM. This chapter focuses on the very basic SLAM setup. In particular it assumes a static environment with a single robot. Extensions are discussed towards the end of this chapter, in which the relevant literature is discussed.

46.2 The Three Main SLAM Paradigms

This section reviews three basic SLAM paradigms, from which most others are derived. The first, known as EKF SLAM, is in robotics historically the earliest but has become less popular due to its limiting computational properties and issues resulting from performing single linearizations only. The second approach uses nonparametric statistical filtering techniques known as particle filters. It is a popular method for online SLAM and provides a perspective on addressing the data association problem in SLAM. The third paradigm is based on graphical representations and successfully applies sparse nonlinear optimization methods to the SLAM problem. It is the main paradigm for solving the full SLAM problem and recently also incremental techniques are available.

46.2.1 Extended Kalman Filters

Historically, the EKF formulation of SLAM is the earliest, and perhaps the most influential, SLAM algorithm. EKF SLAM was introduced in [46.10, 11] and [46.12, 13], which were the first papers to propose the use of a single state vector to estimate the locations of the robot and a set of features in the environment, with an associated error covariance matrix representing the uncertainty in these estimates, including the correlations between the vehicle and feature state estimates. As the robot moves through its environment taking measurements, the system state vector and covariance matrix are updated using the extended Kalman filter [46.14, 15]. As new features are observed, new states are added to the system state vector; the size of the system covariance matrix grows quadratically.

This approach assumes a metrical, feature-based environmental representation, in which objects can be effectively represented as points in an appropriate parameter space. The position of the robot and the locations of features form a network of uncertain spatial relationships. The development of appropriate representations is a critical issue in SLAM, and intimately related to the topics of sensing and world modeling discussed in Chap. 36 and in Part C.

The EKF algorithm represents the robot estimate by a multivariate Gaussian

$$p(x_t, m \mid Z_t, U_t) = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \,. \tag{46.10}$$

The high-dimensional vector $\boldsymbol{\mu}_t$ contains the robot's best estimate of its own current location x_t and the location of the features in the environment. In our point-landmark example, the dimension of $\boldsymbol{\mu}_t$ would be $3 + 2N$, since we need three variables to represent the robot location and $2N$ variables for the N landmarks in the map.

The matrix $\boldsymbol{\Sigma}_t$ is the covariance of the robot's assessment of its expected error in the guess $\boldsymbol{\mu}_t$. The matrix $\boldsymbol{\Sigma}_t$ is of size $(3 + 2N) \times (3 + 2N)$ and it is positive semi-definite. In SLAM, this matrix is usually dense. The off-diagonal elements capture the correlations in the estimates of different variables. Nonzero correlations come along because the robot's location is uncertain, and as a result the locations of the landmarks in the maps are uncertain.

The EKF SLAM algorithm is easily derived for our point-landmark example. Suppose, for a moment, the motion function g and the measurement function h were *linear* in their arguments. Then, the vanilla Kalman filter, as described in any textbook on Kalman filtering, would be applicable. EKF SLAM linearizes the functions g and h using Taylor series expansion. In its most basic form and in the absence of any data association problems, EKF SLAM is basically the application of the EKF to the online SLAM problem.

Figure 46.2 illustrates the EKF SLAM algorithm for an artificial example. The robot navigates from a start pose that serves as the origin of its coordinate system. As it moves, its own pose uncertainty increases, as indicated by uncertainty ellipses of growing diameter. It

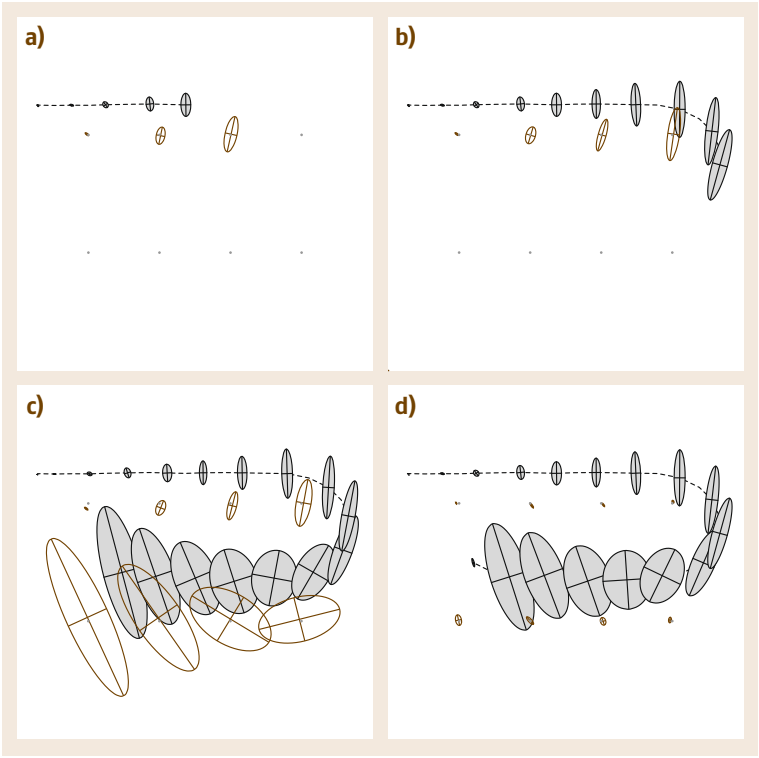


Fig.46.2a–d EKF applied to the online SLAM problem. The robot's path is a *dotted line*, and its estimates of its own position are *shaded ellipses*. Eight distinguishable landmarks of unknown location are shown as *small dots*, and their location estimates are shown as *white ellipses*. In **(a–c)** the robot's positional uncertainty is increasing, as is its uncertainty about the landmarks it encounters. In **(d)** the robot senses the first landmark again, and the uncertainty of *all* landmarks decreases, as does the uncertainty of its current pose (image courtesy of Michael Montemerlo, Stanford University)

also senses nearby landmarks and maps them with an uncertainty that combines the fixed measurement uncertainty with the increasing pose uncertainty. As a result, the uncertainty in the landmark locations grows over time. The interesting transition happens in Fig. 46.2d: Here the robot observes the landmark it saw in the very beginning of mapping, and whose location is relatively well known. Through this observation, the robot's pose error is reduced, as indicated in Fig. 46.2d – notice the very small error ellipse for the final robot pose. This observation also reduces the uncertainty for other landmarks in the map. This phenomenon arises from a correlation that is expressed in the covariance matrix of the Gaussian posterior. Since most of the uncertainty in earlier landmark estimates is caused by the robot pose, and since this very uncertainty persists over time, the location estimates of those landmarks are correlated. When gaining information on the robot's pose, this information spreads to previously observed landmarks. This effect is probably the most important characteristic of the SLAM posterior [46.16]. Information that helps localize the robot is propagated through the map, and as a result improves the localization of other landmarks in the map.

With a few adaptations, EKF SLAM can also be applied in the presence of *uncertain data association*. If the identity of observed features is unknown, the basic EKF idea becomes inapplicable. The solution here is to reason about the most likely data association when a landmark is observed. This is usually done based on proximity: *which of the landmarks in the map corresponds most likely to the landmark just observed?* The proximity calculation considers the measurement noise and the actual uncertainty in the poster estimate, and the metric used in this calculation is known as a Mahalanobis distance, which is a weighted quadratic distance. To minimize the chances of false data associations, many implementations use visible features to distinguish individual landmarks and associate groups of landmarks observed simultaneously [46.17, 18], although distinct features can also be computed from laser data [46.19, 20]. Typical implementations also maintain a *provisional landmark list* and only add landmarks to the internal map when they have been observed sufficiently frequently [46.16, 21]. With an appropriate landmark definition and careful implementation of the data association step, EKF SLAM has been applied successfully in a wide range of environments, using airborne, underwater, indoor, and various other platforms.

The basic formulation of EKF SLAM assumes that the location of features in the map is fully observable from a single position of the robot. The method has been extended to situations with partial observability,

with range-only [46.22] or angle-only [46.23, 24] measurements. The technique has also been utilized using a feature-less representation, in which the state consists of current and past robot poses, and measurements take the form of constraints between the poses (derived for example from laser scan matching or from camera measurements) [46.25, 26].

A key concern of the EKF approach to SLAM lies in the quadratic nature of the covariance matrix. A number of researchers have proposed extensions to the EKF SLAM algorithms that achieve scalability, for example through submap decomposition [46.27–30]. A related family of approaches [46.31–34] employs the Extended Information Filter, which operates on the inverse of the covariance matrix. A key insight is that whereas the EKF covariance is densely populated, the information matrix is sparse when the full robot trajectory is maintained, leading to the development of efficient algorithms and providing a conceptual link to the pose graph optimization methods described in Sect. 46.2.3.

The issues of consistency and convergence in EKF SLAM have been investigated in [46.35, 36]. Observability-based rules for designing consistent EKF SLAM estimators are presented in [46.37].

46.2.2 Particle Methods

The second principal SLAM paradigm is based on particle filters. Particle filters can be traced back to [46.38], but they have become popular only in the last two decades. Particle filters represent a posterior through a set of *particles*. For the novice in SLAM, each particle is best thought as a concrete guess as to what the true value of the state may be. By collecting many such guesses into a set of guesses, or set of particles, the particle filter approximates the posterior distribution. Under mild conditions, the particle filter has been shown to approach the true posterior as the particle set size goes to infinity. It is also a nonparametric representation that represents multimodal distributions with ease.

The key problem with the particle filter in the context of SLAM is that the space of maps and robot paths is huge. Suppose we have a map with 100 features. How many particles would it take to populate that space? In fact, particle filters scale exponentially with the dimension of the underlying state space. Three or four dimensions are thus acceptable, but 100 dimensions are generally not.

The trick to make particle filters amenable to the SLAM problem goes back to [46.39, 40] and is known as Rao–Blackwellization. It has been introduced into the SLAM literature in [46.41], followed by [46.42], who coined the name fastSLAM (fast simultaneous localization and mapping). Let us first explain the basic FastSLAM algorithm on the simplified point-landmark example, and then discuss the justification for this approach.

At any point in time, FastSLAM maintains K particles of the type

$$X_t^{[k]}, \boldsymbol{\mu}_{t,1}^{[k]}, \ldots, \boldsymbol{\mu}_{t,N}^{[k]}, \boldsymbol{\Sigma}_{t,1}^{[k]}, \ldots, \boldsymbol{\Sigma}_{t,N}^{[k]}. \tag{46.11}$$

Here $[k]$ is the index of the sample. This expression states that a particle contains:

- A sample path $X_t^{[k]}$, and
- A set of N 2-D Gaussians with means $\boldsymbol{\mu}_{t,n}^{[k]}$ and variances $\boldsymbol{\Sigma}_{t,n}^{[k]}$), one for each landmark in the environment.

Here n is the index of the landmark (with $1 \leq n \leq N$). From that it follows that K particles possess K path samples. It also possesses KN Gaussians, each of which models exactly one landmark for one of the particles.

Initializing FastSLAM is simple: just set each particle's robot location to the starting coordinates, typically $(0, 0, 0)^{\mathrm{T}}$, and zero the map. The particle update then proceeds as follows:

- When an odometry reading is received, new location variables are generated stochastically, one for each of the particles. The distribution for generating those location particles is based on the motion model

$$x_t^{[k]} \approx p(x_t \mid x_{t-1}^{[k]}, u_t) \,. \tag{46.12}$$

  Here $x_{t-1}^{[k]}$ is the previous location, which is part of the particle. This probabilistic sampling step is easily implemented for any robot whose kinematics can be computed.
- When a measurement z_t is received, two things happen: first, FastSLAM computes for each particle the probability of the new measurement z_t. Let the index of the sensed landmark be n. Then the desired probability is defined as follows

$$w_t^{[k]} = \mathcal{N}(z_t \mid x_t^{[k]}, \boldsymbol{\mu}_{t,n}^{[k]}, \boldsymbol{\Sigma}_{t,n}^{[k]}) \,. \tag{46.13}$$

  The factor $w_t^{[k]}$ is called the *importance weight*, since it measures how *important* the particle is in the light of the new sensor measurement. As before, $\mathcal{N}$ denotes the normal distribution, but this time it is calculated for a specific value, z_t. The importance weights of all particles are then normalized so that they sum to 1.

  Next, FastSLAM draws with replacement from the set of existing particles a set of new particles. The probability of drawing a particle is its normalized

importance weight. This step is called *resampling*. The intuition behind resampling is that particles for which the measurement is more plausible have a higher chance of surviving the resampling process.

Finally, FastSLAM updates for the new particle set the mean $\boldsymbol{\mu}_{t,n}^{[k]}$ and covariance $\boldsymbol{\Sigma}_{t,n}^{[k]}$, based on the measurement z_t. This update follows the standard EKF update rules – note that the extended Kalman filters maintained in FastSLAM are, in contrast to EKF SLAM, all low-dimensional (typically 2-D).

This all may sound complex, but FastSLAM is quite easy to implement. Sampling from the motion model usually involves simple kinematic calculations. Computing the importance of a measurement is often straightforward too, especially for Gaussian measurement noise. And updating a low-dimensional particle filter is also not complicated.

FastSLAM has been shown to approximate the full SLAM posterior. The derivation of FastSLAM exploits three techniques: Rao–Blackwellization, conditional independence, and resampling. Rao–Blackwellization is the following concept. Suppose we would like to compute a probability distribution $p(a, b)$, where a and b are arbitrary random variables. The vanilla particle filter would draw particles from the joint distributions, that is, each particle would have a value for a and one for b. However, if the conditional $p(b \mid a)$ can be described in closed form, it is equally legitimate to just draw particles from $p(a)$, and attach to each particle a closed-form description of $p(b \mid a)$. This trick is known as Rao–Blackwellization, and it yields better results than sampling from the joint. FastSLAM applies this technique, in that it samples from the path posterior $p(X_t^{[k]} \mid U_t, Z_t)$ and represents the map $p(m \mid X_t^{[k]}, U_t, Z_t)$ in Gaussian form.

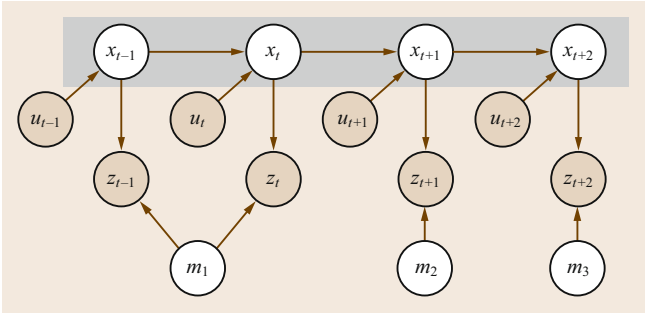


Fig. 46.3 The SLAM problem depicted as Bayes network graph. The robot moves from location x_{t-1} to location x_{t+2}, driven by a sequence of controls. At each location x_t it observes a nearby feature in the map $m = \{m_1, m_2, m_3\}$. This graphical network illustrates that the location variables *separate* the individual features in the map from each other. If the locations are known, there remains no other path involving variables whose value is not known, between any two features in the map. This lack of a path renders the posterior of any two features in the map conditionally independent (given the locations)

FastSLAM also breaks down the posterior over maps (conditioned on paths) into sequences of low-dimensional Gaussians. The justification for this decomposition is subtle. It arises from a specific conditional independence assumption that is native to SLAM. Fig. 46.3 illustrates the concept graphically. In SLAM, knowledge of the robot path renders all landmark estimates independent. This is easily shown for the graphical network in Fig. 46.3: we find that if we remove the path variables from Fig. 46.3, then the landmark variables are all disconnected [46.43]. Thus, in SLAM *any dependence between multiple landmark estimates is mediated through the robot path.* This subtle but important observation implies that even if we used a large, monolithic Gaussian for the entire map (one per particle, of course), the off-diagonal element between different landmarks would simply remain zero. It is therefore legitimate to implement the map more efficiently, using N small Gaussians, one for each landmark. This explains the efficient map representation in FastSLAM.

Figure 46.4 shows results for a point-feature problem; here the point features are the centers of tree trunks as observed by an outdoor robot. The dataset used here is known as the Victoria Park dataset [46.44]. Fig. 46.4a shows the path of the vehicle obtained by integrating the vehicle controls, without perception. As can be seen, controls are a poor predictor of location for this vehicle; after 30 min of driving, the estimated position of the vehicle is well over 100 m away from its GPS position.

The FastSLAM algorithm has a number of interesting properties. First, it solves both full and online SLAM problems. Each particle has a sample of an entire path but the actual update equation only uses the most recent pose. This makes FastSLAM a filter. Second, FastSLAM can maintain multiple data association hypotheses. It is straightforward to make data association decisions on a per-particle basis, instead of having to adopt the same hypothesis for the entire filter. While we will not give any mathematical justification, we note that the resulting FastSLAM algorithm can even deal with unknown data association – something that the extended Kalman filter cannot claim. And third, FastSLAM can be implemented very efficiently using advanced tree methods to represent the map estimates, the update can be performed in time logarithmic in the size of the map N, and linear in the number of particles M.

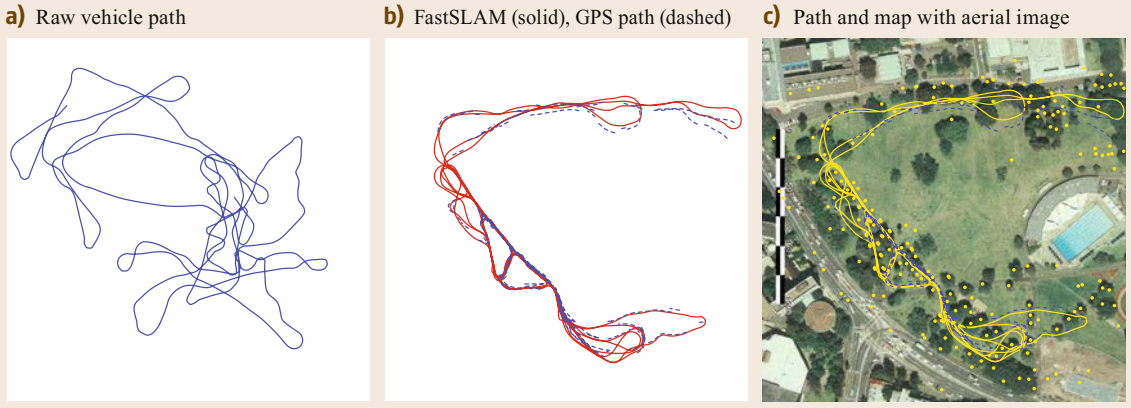


Fig. 46.4 (a) Vehicle path predicted by the odometry; (b) True path (*dashed line*) and FastSLAM 1.0 path (*solid line*); (c) Victoria Park results overlaid on aerial imagery with the GPS path in *blue* (*dashed*), average FastSLAM 1.0 path in *yellow* (*solid*), and estimated features as *yellow dots* (data and aerial image courtesy of José Guivant and Eduardo Nebot, Australian Centre for Field Robotics)

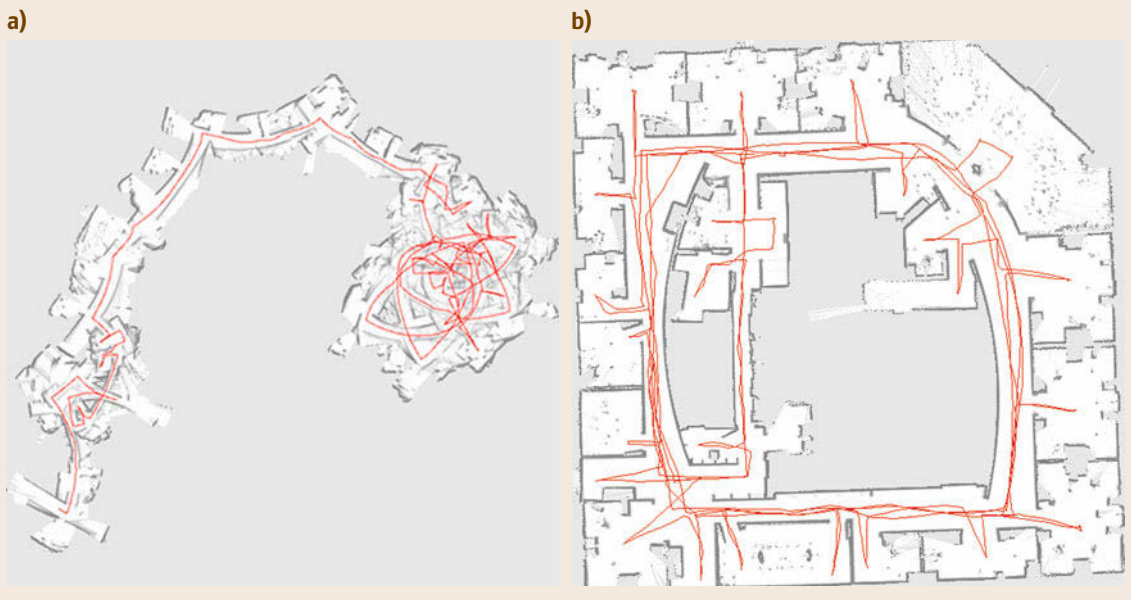


Fig. 46.5 Occupancy grid map generated from laser range data and based on pure odometry (image courtesy of Dirk Hähnel, University of Freiburg)

FastSLAM has been extended in several ways. One set of variants are grid-based versions of FastSLAM, in which the Gaussians used to model point landmarks are replaced by an occupancy grid map [46.45–47]. The variant of [46.46] is illustrated in Fig. 46.5.

Figure 46.6 illustrates a simplified situation with three particles just before closing a large loop. The three different particles each stand for different paths, and they also posses their own local maps. When the loop is closed importance resampling selects those particles whose maps are most consistent with the measurement. A resulting large-scale map is shown in Fig. 46.5. Further extensions can be found in [46.48, 49], whose methods are called DP-SLAM and operate on ancestry trees to provide efficient tree update methods for grid-based maps. Related to that, approximations to FastSLAM in which particles share their maps have been proposed [46.50].

The works in [46.45, 47, 51] provide ways to incorporate new observations into the location sampling

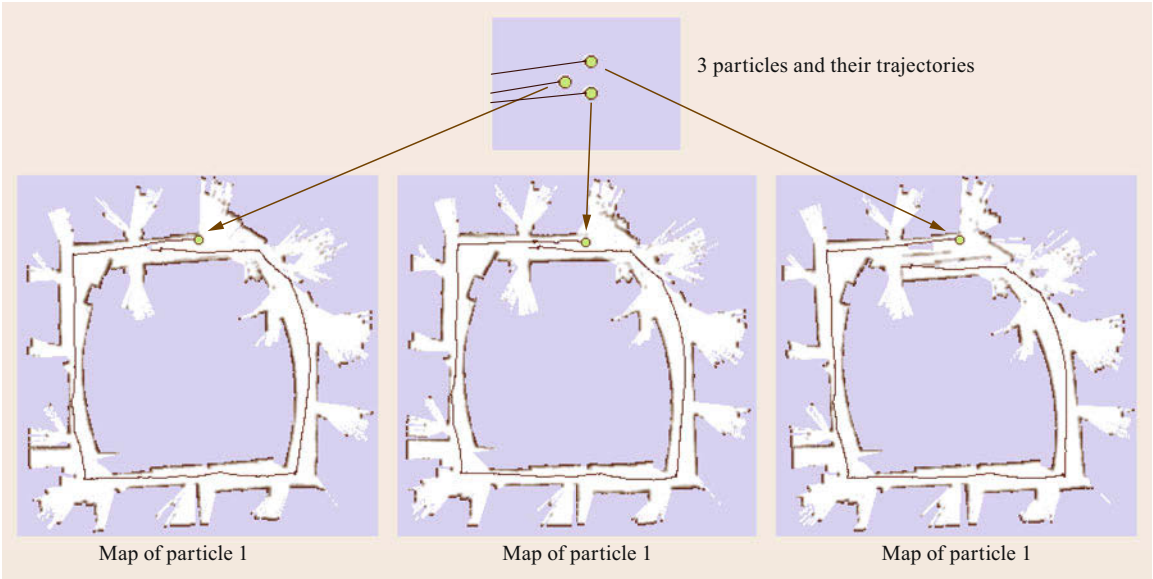


Fig. 46.6 Application of the grid-based variant of the FastSLAM algorithm. Each particle carries its own map and the importance weights of the particles are computed based on the likelihood of the measurements given the particle's own map

process for landmarks and grid maps, based on prior work in [46.52]. This leads to an improved sampling process

$$x_t^{[k]} \approx \frac{p(z_t \mid m_{t-1}^{[k]}, x_t)\, p(x_t \mid x_{t-1}^{[k]}, u_t)}{p(z_t \mid m_{t-1}^{[k]}, x_{t-1}^{[k]}, u_t)} \,, \qquad (46.14)$$

which incorporates the odometry and the observation at the same time. Using an improved proposal distribution leads to more accurately sampled locations. This in turn leads to more accurate maps and requires a smaller number of particles compared to approaches using the sampling process given in (46.12). This extension makes FastSLAM and especially its grid-based variants robust tools for addressing the SLAM problem.

Finally, there are approaches that aim to overcome the assumption that the observations show Gaussian characteristics. As shown in [46.47], there are several situations in which the model is nonGaussian and also multimodal. A sum of Gaussians model on a per-particle bases, however, can be efficiently considered in the particle filter and it eliminates this problem in practice without introducing additional computational demands.

The so-far developed particle filters-based SLAM systems suffer from two problems. First, the number of samples that are required to compute consistent maps is often set manually by making an *educated guess*. The larger the uncertainty that the filter needs to represent during mapping, the more critical becomes this parameter. Second, nested loops combined with extensive re-visits of previously mapped areas can lead to particle depletion, which in turn may prevent the system from estimating a consistent map. Adaptive resampling strategies [46.45], particles sharing maps [46.50], or filter backup approaches [46.53] improve the situation but cannot eliminate this problem in general.

46.2.3 Graph-Based Optimization Techniques

A third family of algorithms solves the SLAM problem through nonlinear sparse optimization. They draw their intuition from a graphical representation of the SLAM problem and the first working solution in robotics was proposed in [46.54]. The graph-based representation used here is closely related to a series of papers [46.55–64]. We note that most of the earlier techniques are offline and address the full SLAM problem. In more recent years, new incremental versions that effectively re-use the previously computed solution have been proposed such as [46.65–67].

The basic intuition of graph-based SLAM is a follows. Landmarks and robot locations can be thought of as nodes in a graph. Every consecutive pair of locations x_{t-1}, x_t is tied together by an edge that represents the information conveyed by the odometry reading u_t. Further edges exist between the nodes that correspond to locations x_t and landmarks m_i, assuming that at time t the robot sensed landmark i. Edges in this graph are

soft constraints. Relaxing these constraints yields the robot's best estimate for the map and the full path.

The construction of the graph is illustrated in Fig. 46.7. Suppose at time $t = 1$, the robot senses landmark m_1. This adds an arc in the (yet highly incomplete) graph between x_1 and m_1. When caching the edges in a matrix format (which happens to correspond to a quadratic equation defining the resulting constraints), a value is added to the elements between x_1 and m_1, as shown on the right hand side of Fig. 46.7a.

Now suppose the robot moves. The odometry reading u_2 leads to an arc between nodes x_1 and x_2, as shown in Fig. 46.7b. Consecutive application of these two basic steps leads to an graph of increasing size, as illustrated in Fig. 46.7c. Nevertheless this graph is *sparse*, in that each node is only connected to a small number of other nodes (assuming a sensor with limited sensing range). The number of constraints in the graph is (at worst) *linear* in the time elapsed and in the number of nodes in the graph.

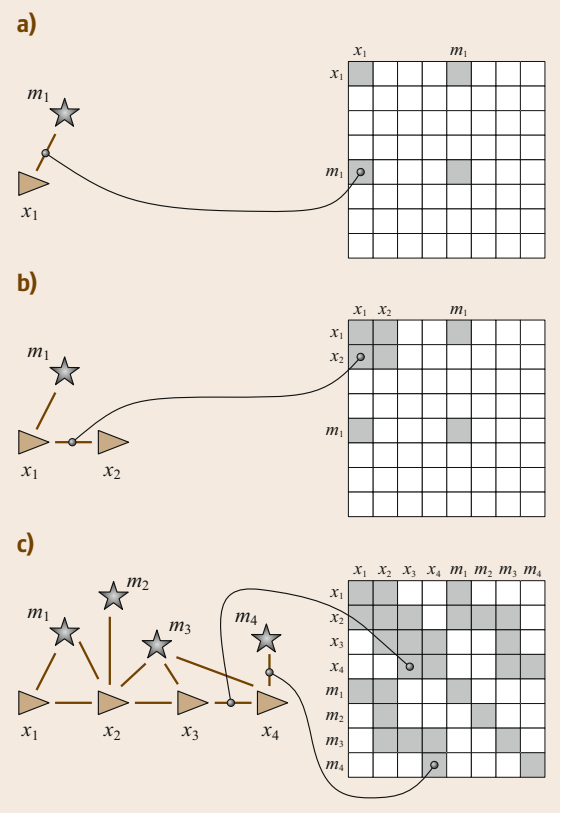


Fig.46.7a–c Illustration of the graph construction. The **(a)** diagram shows the graph, the **(b)** the constraints in matrix form. **(a)** Observation ls landmark m_1. **(b)** Robot motion from x_1 to x_2. **(c)** Several steps later

If we think of the graph as a spring-mass model [46.60], computing the SLAM solution is equivalent to computing the state of minimal energy of this model. To see, we note that the graph corresponds to the log-posterior of the full SLAM problem (46.4)

$$\log p(X_T, m \mid Z_T, U_T) \,. \tag{46.15}$$

Without derivation, we state that this logarithm is of the form

$$\begin{aligned} &\log p(X_T, m \mid Z_T, U_T) \\ &= \text{const} + \sum_t \log p(x_t \mid x_{t-1}, u_t) \\ &\quad + \sum_t \log p(z_t \mid x_t, m) \,, \end{aligned} \tag{46.16}$$

assuming independence between the individual observations and odometry readings. Each constraint of the form $\log p(x_t \mid x_{t-1}, u_t)$ is the result of exactly one robot motion event, and it corresponds to an edge in the graph. Likewise, each constraint of the form $\log p(z_t \mid x_t, m)$ is the result of one sensor measurement, to which we can also find a corresponding edge in the graph. The SLAM problem is then simply to find the mode of this equation, i. e.,

$$X_T^*, m^* = \underset{X_T, m}{\text{argmax}} \log p(X_T, m \mid Z_T, U_T) \,. \tag{46.17}$$

Without derivation, we note that under the Gaussian noise assumptions, which was made in the point-landmark example, this expression resolves to the following quadratic form

$$\begin{aligned} &\log p(X_T, m \mid Z_T, U_T) = \text{const} \\ &\quad + \sum_t \underbrace{[x_t - g(x_{t-1}, u_t)]^{\mathrm{T}} \, R_t^{-1} \, [x_t - g(x_{t-1}, u_t)]}_{\text{odometry reading}} \\ &\quad + \sum_t \underbrace{[z_t - h(x_t, m)]^{\mathrm{T}} \, Q_t^{-1} \, [z_t - h(x_t, m)]}_{\text{feature observation}} . \end{aligned} \tag{46.18}$$

This quadratic form yields a sparse system of equations and a number of efficient optimization techniques can be applied. Common choices include direct methods such as sparse Cholesky and QR decomposition, or iterative ones such as gradient descent, conjugate gradient, and others. Most SLAM implementations rely on iteratively linearizing the functions g and h, in which case the objective in (46.18) becomes quadratic in all of its variables.

Extensions to support an effective correction of large-scale graphs are hierarchical methods. One of

the first is the ATLAS framework [46.25], which constructs a two-level hierarchy combining a Kalman filter that operates in the lower level and a global optimization at the higher level. Similar to that, Hierarchical SLAM [46.68] is a technique for using independent local maps, which are merged in case of re-visiting a place. A fully hierarchical approach has been presented in [46.65]. It builds a multilevel pose-graph and employs an incremental, lazy optimization scheme that allows for optimizing large graphs and at the same time can be executed at each step during mapping. An alternative hierarchical approach is [46.69], which recursively partitions the graph into multiple-level submaps using the nested dissection algorithm.

When it comes to computing highly accurate environment reconstructions, approaches that do not only optimize the poses of the robot but also each individual measurement of a dense sensor often provide better results. In the spirit of bundle adjustment [46.4], approaches for laser scanners [46.70] and Kinect cameras [46.71] have been proposed.

The graphical paradigm can be extended to handle the data association problems as we can integrate additional knowledge on data association into (46.18). Suppose some oracle informed us that landmarks m_i and m_j in the map corresponded to one and the same physical landmark in the world. Then, we can either remove m_j from the graph and attach all adjacent edges to m_i, or we can add a soft correspondence constraint [46.72] of the form

$$(m_j - m_i)^{\mathrm{T}} \, \mathbf{\Gamma} \, (m_j - m_i) \; . \tag{46.19}$$

Here $\mathbf{\Gamma}$ is 2-by-2 diagonal matrix whose coefficients determine the penalty for *not* assigning identical locations to two landmarks (hence we want $\mathbf{\Gamma}$ to be large). Since graphical methods are usually used for the full SLAM problem, the optimization can be interleaved with the search for the optimal data association.

Data association errors typically have a strong impact in the resulting map estimate. Even a small number of wrong data associations is likely to result in inconsistent map estimates. Recently, novel approaches have been proposed that are robust under a certain number of false associations. For example, [46.73, 74] propose an iterative procedure that allows for disabling constraints, an action that is associated with a cost. A generalization of this method introduced in [46.75] formulates [46.74] as a robust cost function also reducing the computational requirements. Such approaches can deal with a significant number of false associations and still provide high-quality maps. Consistency checks for loop closure hypotheses can be found in other approaches as well, both in the front-end [46.76] and in the optimizer [46.77]. There has also been an extension that can deal with multimodal constraints [46.78], proposing a max-mixture representation for maintaining efficiency of the log likelihood optimizing in (46.16). As a result of that, the multimodal extension has only little impact on the runtime and can easily be incorporated in most optimizers. Also robust cost function are used for SLAM, for example pseudo Huber and several alternatives [46.75, 79–81].

Graphical SLAM methods have the advantage that they scale to much higher-dimensional maps than EKF SLAM, exploiting the sparsity of the graph. The key limiting factor in EKF SLAM is the covariance matrix, which takes space (and update time) quadratic in the size of the map. No such constraint exists in graphical methods. The update time of the graph is constant, and the amount of memory required is linear (under some mild assumptions). A further advantage of graph-based methods over the EKF is their ability to constantly relinearize the error function which often leads to better results. Performing the optimization can be expensive, however. Technically, finding the optimal data association is suspected to be an NP-hard problem, although in practice the number of plausible assignments is usually small. The continuous optimization of the log likelihood function in (46.18) depends among other things on the number and size of loops in the map. Also the initialization can have a strong impact on the result and a good initial guess can simplify the optimization substantially [46.8, 82, 83].

We note that the graph-based paradigm is very closely linked to information theory, in that the soft constraints constitute the information the robot has on the world (in an information-theoretic sense [46.92]). Most methods in the field are offline and they optimize for the entire robot path. If the robot path is long, the optimization may become cumbersome. Over the last five years, however, incremental optimization techniques have been proposed that aim at providing a sufficient but not necessarily perfect model of the environment at every point in time. This allows a robot to make decisions based on the current model, for example, to determine exploration goals. In this context, incremental variants [46.93, 94] of stochastic gradient descent techniques [46.8, 91] have been proposed that estimate which part of the graph requires re-optimization given new sensor data. Incremental methods [46.66, 79, 95] in the smoothing and mapping framework can be execute at each step of the mapping process and achieve the performance by variable ordering and selective relinearization. As also evaluated in [46.96] for the SLAM problem, variable ordering impacts the performance of the optimization. Others use hierarchical data structures [46.89] and pose-graphs [46.97] combined with

Table 46.1 Recent open-source graph-based SLAM implementations

Name	Comment
Dynamic covariance scaling (DCS) [46.75]	Optimization with a robust cost function for dealing with outliers Integrated into g^{2}o
g^{2}o [46.80]	Flexible and easily extendable optimization framework for SLAM Comes with different optimization approaches and error functions Supports external plugins
GTSAM2.1 [46.79]	Flexible optimization framework for SLAM and SFMstructure from motion Implements direct and iterative optimization techniques Implements smoothing and mapping (SAM), iSAM, and iSAM2 Implements bundle adjustment for Visual SLAM and SFM
HOG-Man [46.65]	Incremental optimization approach via hierarchical pose graphs and lazy optimization Requires pose-graphs with full rank constraints
iSAM2 [46.66]	General incremental nonlinear optimization with variable elimination Variable re-ordering to retain sparsity On-demand re-linearization of selected variables
KinFu (KinectFusion reimplemented)	Open source reimplementation of KinectFusion [46.84] within the point cloud library (PCL) Dense and highly accurate reconstruction using a Kinect camera Currently limited to medium sized rooms
MaxMixture [46.78]	Optimization for multimodal constraints and outliers Robust to outliers Plugin for g^{2}o
Parallel tracking and mapping (PTAM) [46.85]	System for tracking a hand-held monocular camera and observed features Operates on comparably small workspaces
RGBD-SLAM [46.86]	Kinect-frontend for HOG-Man and g^{2}o Fairly standard combination of SURF matching and RANSAC
ScaViSLAM [46.87]	SLAM system for stereo and Kinect-style cameras Combines local bundle adjustment with sparse global optimization for on-the-fly processing
SLAM6-D [46.88]	SLAM system that operates on point clouds from 3-D laser data Applies iterative closest point algorithm (ICP) and global relaxation
Sparse surface adjustment (SSA) [46.70, 71]	Optimizes robot poses and proximity sensor data jointly Provides smooth surface estimates Assumes a range sensor (e.g., laser scanner, Kinect, or similar)
TreeMap [46.89]	Incremental optimization approach Update in $O(\log N)$ time Provides only a mean estimate
TORO [46.90]	Optimization approach that extends stochastic gradient descent (SGD) [46.91] Robust under bad initial guesses Recovers quickly from large errors but slow convergence at minimum Assumes that constraints have roughly spherical covariance matrices Provides only a mean estimate
Vertigo [46.74]	Switchable constraints for robust optimization Plugin for g^{2}o

a lazy optimization for on-the-fly mapping [46.65]. As an alternative to global methods, relative optimization approaches [46.98] aim at computing locally consistent geometric maps but only topological maps on the global scale. Hybrid approaches [46.87] seek to combine the best of both worlds.

There also exists a number of cross-overs that manipulate the graph online so as to factor out past robot location variables. The resulting algorithms are filters [46.25, 33, 99, 100], and they tend to be intimately related to information filter methods. Many of the original attempts to decompose EKF SLAM representations into smaller submaps to scale up are based on motivations that are not dissimilar to the graphical approach [46.27, 28, 101].

Recently, researchers addressed the problem of long-term operation and frequent revisits of already mapped terrain. To avoid densely connected pose-graphs that lead to slow convergence behavior, the robot can switch between SLAM and localization, can merge nodes to avoid a growth of the graph [46.90, 102], or can discard nodes or edges [46.32, 103–105].

Graphical and optimization-based SLAM algorithm are still subject of intense research and the paradigm scales to maps large numbers of nodes [46.25, 55, 57, 59, 63–65, 89, 90, 106, 107]. Arguably, the graph-based

paradigm has generated some the largest SLAM maps ever built. Furthermore, the SLAM community started to release flexible optimization frameworks and SLAM implementations under open source licenses to support further developments and to allow for efficient comparisons, (Table 46.1). Especially the optimization frameworks [46.66, 79, 80] are flexible and powerful state of the art tools for developing graph-based SLAM systems. They can be either used as a *black box* or can be easily extended though plugins.

46.2.4 Relation of Paradigms

The three paradigms just discussed cover the vast majority of work in the field of SLAM. As discussed, EKF SLAM comes with a computational hurdle that poses serious scaling limitations and the linearization may lead to inconsistent maps. The most promising extensions of EKF SLAM are based on building local submaps; however, in many ways the resulting algorithms resemble the graph-based approach.

Particle filter methods sidestep some of the issues arising from the natural inter-feature correlations in the map – which hindered the EKF. By sampling from robot poses, the individual landmarks in the map become independent, and hence are decorrelated. As a result, FastSLAM can represent the posterior by a sampled robot pose, and many local, independent Gaussians for its landmarks. The particle representation offers advantages for SLAM as it allows for computationally efficient updates and for sampling over data associations. On the negative side, the number of necessary particles can grow very large, especially for robots seeking to map multiple nested loops.

Graph-based methods address the full SLAM problem, hence are in the standard formulation not online. They draw their intuition form the fact that SLAM can be modeled by a sparse graph of soft constraints, where each constraint either corresponds to a motion or a measurement event. Due to the availability of highly efficient optimization methods for sparse nonlinear optimization problems, graph-based SLAM has become the method of choice for building large-scale maps. Recent developments have brought up several graph-based methods for incremental map building that can be executed at every time step during navigation. Data association search can be incorporated into the basic mathematical framework and different approaches that are even robust under wrong data associations are available today.

46.3 Visual and RGB-D SLAM

A popular and important topic in recent years has been Visual SLAM – the challenge of building maps and tracking the robot pose in full 6-DOF using data from cameras [46.108] or RGB-D (Kinect) sensors [46.86, 109]. Visual sensors offer a wealth of information that enables the construction of rich 3-D models of the world. They also enable difficult issues such as loop-closing to be addressed in novel ways using appearance information [46.110]. Visual SLAM is anticipated to be a critical area for future research in perception for robotics, as we seek to develop low-cost systems that are capably of intelligent physical interaction with the world.

Attempting SLAM with monocular, stereo, omnidirectional, or RGB-D cameras raises the level-of-difficulty of many of the SLAM components, such as data association and computational efficiency, described above. A key challenge is robustness. Many visual SLAM applications of interest, such as augmented reality [46.85], entail handheld camera motions, which present greater difficulties for state estimation, in comparison to the motion of a wheeled robot across a flat floor.

Visual navigation and mapping was a key early goal in the mobile robotics community [46.111, 112], but early approaches were hampered by the lack of sufficient computational resources to handle massive video data streams. Early approaches were typically based on extended Kalman filters [46.113–116], but did not compute the full covariance for the feature poses and camera trajectory, resulting in a loss of consistency. Visual SLAM is closely related to the structure from motion (SFM) problem in computer vision [46.4, 5]. Historically, SFM was primarily concerned with off-line batch processing, whereas SLAM seeks to achieve a solution for online operation, suitable for closed-loop interaction of a robot or user with its environment. In comparison to laser scanners, cameras provide a fire hydrant of information, making online processing nearly impossible until recent increases in computation have become available.

Davison was an early pioneer in developing complete visual SLAM systems, initially using a real-time active stereo head [46.121] that tracked distinctive visual features with a full covariance EKF approach. Subsequent work developed the first real-time SLAM system that operated with a single freely moving camera as the only data source [46.23, 122]. This system could build sparse, room-size maps of indoor scenes at 30 Hz frame-rate in real-time, a notable historical achievement

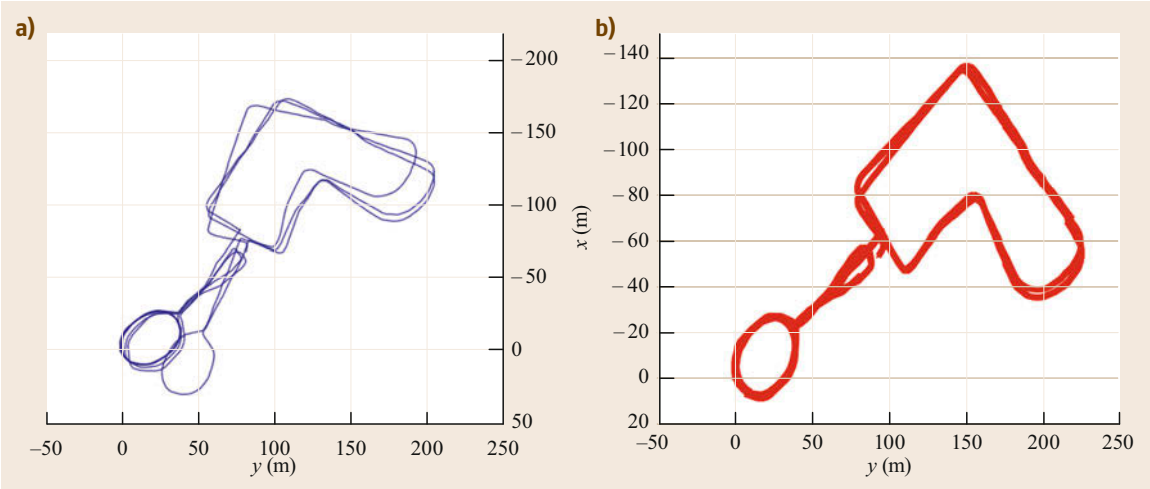


Fig. 46.8 **(a)** A 2-km path and 50 000 frames estimated for the New College Dataset (after [46.117]) using relative bundle adjustment (after [46.98]). **(b)** the relative bundle adjustment solution is easily improved by taking FAB-MAP (after [46.118]) loop-closures into account – this is achieved without global optimization (after [46.119]). Sibley et al. advocate that relative metric accuracy and topological consistency are the requirements for autonomous navigation, and these are better achieved using a relative manifold representation instead of using a conventional single Euclidean representation [46.120]

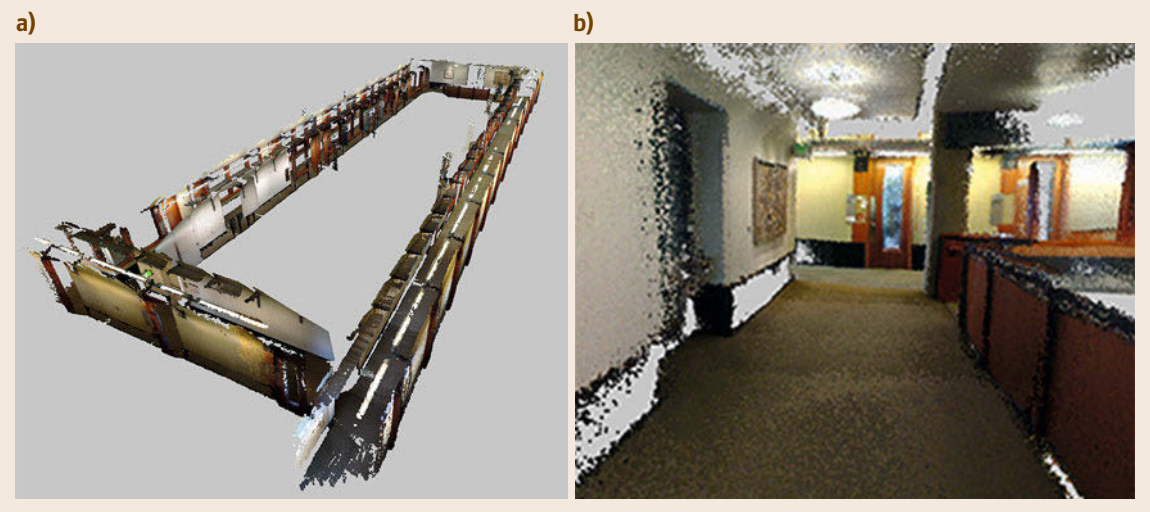


Fig. 46.9 **(a)** 3-D Model and **(b)** close-up view of a corridor environment in the Paul G. Allen building at University of Washington built from Kinect data (after [46.109]; image courtesy of Peter Henry, University of Washington)

in visual SLAM research. A difficulty encountered with initial monocular SLAM [46.23] was coping with the initialization of points that were far away from the camera, due to nonGaussian distributions of such feature locations resulting from poor depth information. This limitation was overcome in [46.24], introducing an inverse depth parameterization for monocular SLAM, a key development for enabling a unified treatment of initialization and tracking of visual features in real-time.

A milestone in creating robust visual SLAM systems was the introduction of keyframes in parallel tracking and mapping (PTAM) [46.85], which separated the tasks of keyframe mapping and localization into parallel threads, improving robustness and performance for online processing. Keyframes are now a mainstream concept for complexity reduction in visual SLAM systems. Related approaches using keyframes include [46.87, 123–126]. The work

in [46.108] analyzes the tradeoffs between filtering and keyframe-based bundle adjustment in visual SLAM, and concluded that keyframe bundle adjustment outperforms filtering, as it provides the most accuracy per unit of computing time.

As pointed out by Davison and other researchers, an appealing aspect of visual SLAM is that camera measurements can provide odometry information, and indeed visual odometry is a key component of modern SLAM systems [46.127]. Here, [46.128] and [46.129] provide an extensive tutorial of techniques for visual odometry, including feature detection, feature matching, outlier rejection, and constraint estimation, and trajectory optimization. Finally, a publicly available visual odometry library [46.130] that is optimized for efficient operation on small unmanned aerial vehicles is available today.

Visual information offers a tremendous source of information for loop closing, not present in the canonical 2-D laser SLAM systems developed in the early 2000s. The work in [46.110] was one of the first to employ techniques for visual object recognition [46.131] to location recognition. More recently FAB-MAP [46.118, 132] has demonstrated appearance-only place recognition at large scale, mapping trajectories with a length of 1000 km. Combining a bag-of-features approach with a probabilistic place model and Chow–Liu tree inference leads to place recognition that is robust against perceptual aliasing while remaining computationally efficient. Other work on place recognition includes [46.133], which combines bag-of-words loop closing with tests of geometrical consistency based on conditional random fields (Fig. 46.8).

The techniques described above have formed the basis for a number of notable large-scale SLAM systems developed in recent years. A 2008 special issue of the IEEE Transactions on Robotics provides a good snapshot of recent state-of-the-art SLAM techniques [46.135]. Other notable recent examples include [46.98, 119, 126, 136–138]. The idea of employing relative bundle adjustment [46.98] to compute a full maximum likelihood solution in an online fashion, even for loop closures, by employing a manifold representation that does not attempt to enforce Euclidean constraints results in maps that can be computed at high frame rate (see also Fig. 46.8). Finally, view-based mapping systems [46.126, 136, 137] aim at large-scale and/or life-long visual mapping based on the pose graph optimization techniques described above in Section 46.2.3.

Several compelling 3-D mapping and localization have been created in recent years with RGB-D (Kinect) sensors. The combination of direct range measurements with dense visual imagery can enable dramatic improvements in mapping and navigation systems for indoor environments. State-of-the-art RGB-D SLAM systems include [46.109] and [46.86]. Figure 46.9 shows examples of the output of these systems.

Other researchers aim at exploiting the surface properties of scanned environments to correct for sen-

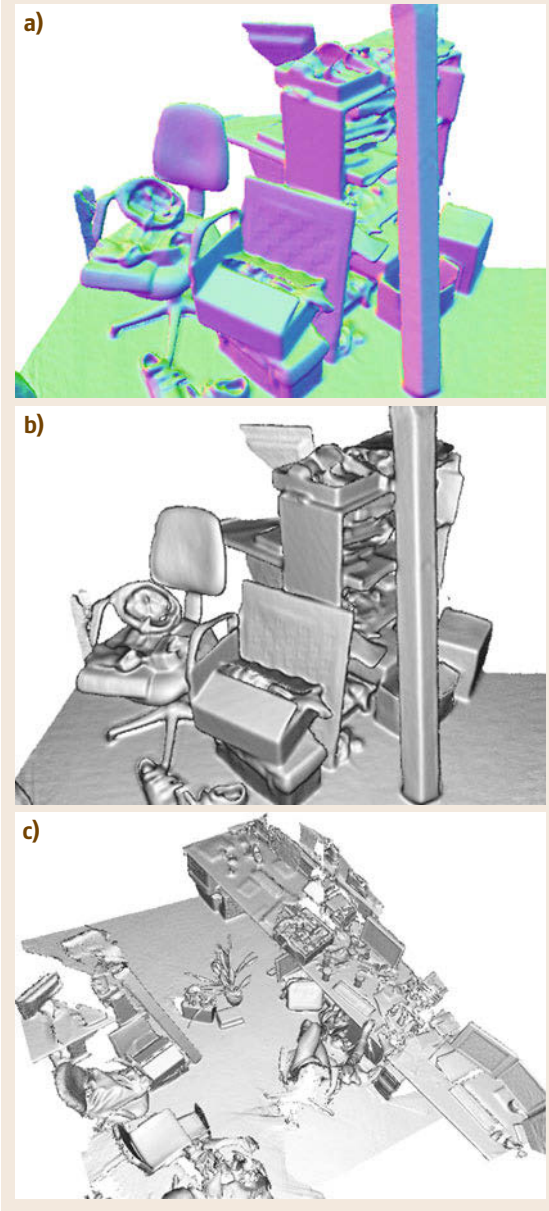


Fig.46.10a–c Results obtained with KinectFusion (after [46.134]). **(a)** A local scene as a normal map and **(b)** as a Phong-shaded rendering. The **(c)** image depicts a larger scene (image courtesy of Richard Newcombe, Imperial College London)

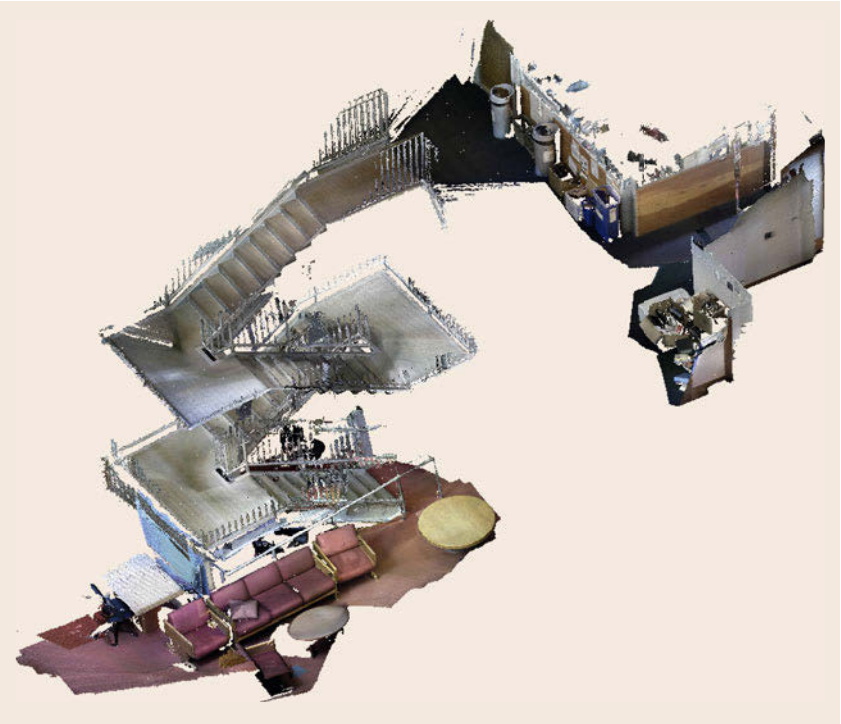


Fig. 46.11 Spatially extended KinectFusion output produced in real-time with Kintinuous (after [46.139])

sor noise of range sensors such as the Kinect [46.71]. They jointly optimize the poses of the sensor and the positions of the surface points measured and iteratively refine the structure of the error function by recomputing the data associations after each optimization, resulting in accurate smooth models of the environment.

An emerging area for future research is the development of fully dense processing methods that exploit recent advances in commodity graphical processing unit (GPU) technology. Kinect-based dense tracking and mapping, a fully-dense method for small-scale visual tracking and reconstruction is described in [46.140]. Dense modeling and tracking are achieved via highly parallelized operations on commodity GPU hardware to yield a system that outperforms previous methods such as PTAM for challenging camera trajectories. Dense methods offer an interesting perspective from which to address long-standing problems, such as visual odometry, from a fresh perspective, without requiring explicit feature detection and matching [46.141]. KinectFusion [46.84, 134] is a dense modeling system that tracks the 3-D pose of a handheld Kinect while concurrently reconstructing high-quality scene 3-D models in real-time. See Fig. 46.10 for an example. KinectFusion has been applied to spatially extended environments in [46.139, 142]. An example is shown in Fig. 46.11.

46.4 Conclusion and Future Challenges

This chapter has provided an introduction into SLAM, which is defined as the problem faced by a mobile platform roaming an unknown environment, and seeking to localize itself while concurrently building a map of the environment. The chapter discussed three main paradigms in SLAM, which are based on the extended Kalman filter, particle filters, and graph-based sparse optimization techniques, and then described recent progress in Visual/Kinect SLAM.

The following references provide an in-depth tutorial on SLAM and much greater depth of coverage on the details of popular SLAM algorithms. Furthermore, several implementations of popular SLAM systems, including most of the approaches listed in Table 46.1, can be found in online resources such as http://www.openslam.org or in the references [46.6, 9, 21, 62].

The considerable progress in SLAM in the past decade is beyond doubt. The core state estimation at

the heart of SLAM is now quite well understood, and a number of impressive implementations have been developed, including several widely used open source software implementations and some commercial projects. None-the-less, a number of open research challenges remain for the general problem of robotic mapping in complex and dynamic environments over extended periods of time, including robots sharing, extending, and revising previously built models, efficient failure recovery, zero user intervention, and operation on resource-constrained systems. Another exciting area for the future is the further development of fully dense visual mapping systems exploiting the latest advances in GPU hardware development.

An ultimate goal is to realize the challenge of *persistent* navigation and mapping – the capability for a robot to perform SLAM robustly for days, weeks, or months at a time with minimal human supervision, in complex and dynamic environments. Taking the limit as $t \to \infty$ poses difficult challenges to most current algorithms; in fact, most robot mapping and navigation algorithms are doomed to fail with the passage of time, as errors inevitably accrue. Despite recent encouraging solutions [46.102, 105], more research is needed for techniques that can recover from mistakes and enable robots to deal with changes in the environment and enabling a long-term autonomous existence.

Video-References

VIDEO 439 Deformation-based loop closure for Dense RGB-D SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/439

VIDEO 440 Large-scale SLAM using the Atlas framework
available from http://handbookofrobotics.org/view-chapter/46/videodetails/440

VIDEO 441 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/441

VIDEO 442 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/442

VIDEO 443 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/443

VIDEO 444 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/444

VIDEO 445 Graph-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/445

VIDEO 446 Graph-based SLAM using TORO
available from http://handbookofrobotics.org/view-chapter/46/videodetails/446

VIDEO 447 Sparse pose adjustment
available from http://handbookofrobotics.org/view-chapter/46/videodetails/447

VIDEO 449 Pose graph compression for laser-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/449

VIDEO 450 Pose graph compression for laser-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/450

VIDEO 451 Pose graph compression for laser-based SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/451

VIDEO 452 DTAM: Dense tracking and mapping in real-time
available from http://handbookofrobotics.org/view-chapter/46/videodetails/452

VIDEO 453 MonoSLAM: Real-time single camera SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/453

VIDEO 454 SLAM++: Simultaneous localisation and mapping at the level of objects
available from http://handbookofrobotics.org/view-chapter/46/videodetails/454

VIDEO 455 Extended Kalman filter SLAM
available from http://handbookofrobotics.org/view-chapter/46/videodetails/455

References

46.1 C.F. Gauss: *Theoria Motus Corporum Coelestium (Theory of the Motion of the Heavenly Bodies Moving about the Sun in Conic Sections)* (Perthes and Bessen, Hamburg 1809), Republished in 1857 and by Dover in 1963

46.2 D.C. Brown: The bundle adjustment – Progress and prospects, Int. Arch. Photogramm. **21**(3), 3:3–3:35 (1976)

46.3 G. Konecny: *Geoinformation: Remote Sensing, Photogrammetry and Geographical Information Systems* (Taylor Francis, London 2002)

46.4 B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon: Bundle adjustment – A modern synthesis, Lect. Notes Comput. Sci. **62**, 298–372 (2000)

46.5 R. Hartley, A. Zisserman: *Multiple View Geometry in Computer Vision* (Cambridge Univ. Press, Cambridge 2003)

46.6 T. Bailey, H.F. Durrant-Whyte: Simultaneous localisation and mapping (SLAM): Part II, Robotics Autom. Mag. **13**(3), 108–117 (2006)

46.7 H.F. Durrant-Whyte, T. Bailey: Simultaneous localisation and mapping (SLAM): Part I, Robotics Autom. Mag. **13**(2), 99–110 (2006)

46.8 G. Grisetti, C. Stachniss, W. Burgard: Nonlinear constraint network optimization for efficient map learning, IEEE Trans. Intell. Transp. Syst. **10**(3), 428–439 (2009)

46.9 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge, 2005)

46.10 R. Smith, M. Self, P. Cheeseman: A stochastic map for uncertain spatial relationships, Proc. Int. Symp. Robotics Res. (ISRR) (MIT Press, Cambridge 1988) pp. 467–474

46.11 R. Smith, M. Self, P. Cheeseman: Estimating uncertain spatial relationships in robotics. In: *Autonomous Robot Vehicles*, ed. by I.J. Cox, G.T. Wilfong (Springer Verlag, Berlin, Heidelberg 1990) pp. 167–193

46.12 P. Moutarlier, R. Chatila: Stochastic multisensory data fusion for mobile robot location and environment modeling, 5th Int. Symp. Robotics Res. (ISRR) (1989) pp. 207–216

46.13 P. Moutarlier, R. Chatila: An experimental system for incremental environment modeling by an autonomous mobile robot, 1st Int. Sym. Exp. Robotics (ISER) (1990)

46.14 R. Kalman: A new approach to linear filtering and prediction problems, J. Fluids **82**, 35–45 (1960)

46.15 A.M. Jazwinsky: *Stochastic Processes and Filtering Theory* (Academic, New York 1970)

46.16 M.G. Dissanayake, P.M. Newman, S. Clark, H.F. Durrant-Whyte, M. Csorba: A solution to the simultaneous localization and map building (SLAM) Problem, IEEE Trans. Robotics Autom. **17**(3), 229–241 (2001)

46.17 J. Neira, J. Tardos, J. Castellanos: Linear time vehicle relocation in SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 427–433

46.18 J. Neira, J.D. Tardos: Data association in stochastic mapping using the joint compatibility test, IEEE Trans. Robotics Autom. **17**(6), 890–897 (2001)

46.19 G.D. Tipaldi, M. Braun, K.O. Arras: Flirt: interest regions for 2D range data with applications to robot navigation, Proc. Int. Symp. Exp. Robotics (ISER) (2010)

46.20 G.D. Tipaldi, L. Spinello, W. Burgard: Geometrical flirt phrases for large scale place recognition in 2D range data, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

46.21 T. Bailey: Mobile Robot Localisation and Mapping in Extensive Outdoor Environments, Ph.D. Thesis (Univ. of Sydney, Sydney 2002)

46.22 J.J. Leonard, R.R. Rikoski, P.M. Newman, M. Bosse: Mapping partially observable features from multiple uncertain vantage points, Int. J. Robotics Res. **21**(10), 943–975 (2002)

46.23 A.J. Davison: Real-time simultaneous localisation and mapping with a single camera, Proc. IEEE 9th Int. Conf. Comput. Vis. (2003) pp. 1403–1410

46.24 J.M.M. Montiel, J. Civera, A.J. Davison: Unified inverse depth parametrization for monocular SLAM, Robotics Sci. Syst., Vol. 1 (2006)

46.25 M. Bosse, P.M. Newman, J. Leonard, S. Teller: Simultaneous localization and map building in large-scale cyclic environments using the Atlas Framework, Int. J. Robotics Res. **23**(12), 1113–1139 (2004)

46.26 J. Nieto, T. Bailey, E. Nebot: Scan-SLAM: Combining EKF-SLAM and scan correlation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

46.27 J. Guivant, E. Nebot: Optimization of the simultaneous localization and map building algorithm for real time implementation, IEEE Trans. Robotics. Autom. **17**(3), 242–257 (2001)

46.28 J.J. Leonard, H. Feder: A computationally efficient method for large-scale concurrent mapping and localization, Proc. 9th Int. Symp. Robotics Res. (ISRR), ed. by J. Hollerbach, D. Koditschek (1999) pp. 169–176

46.29 J.D. Tardós, J. Neira, P.M. Newman, J.J. Leonard: Robust mapping and localization in indoor environments using sonar data, Int. J. Robotics Res. **21**(4), 311–330 (2002)

46.30 S.B. Williams, G. Dissanayake, H.F. Durrant-Whyte: Towards multi-vehicle simultaneous localisation and mapping, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 2743–2748

46.31 R.M. Eustice, H. Singh, J.J. Leonard: Exactly sparse delayed-state filters for view-based SLAM, IEEE Trans. Robotics **22**(6), 1100–1114 (2006)

46.32 V. Ila, J.M. Porta, J. Andrade-Cetto: Information-based compact pose SLAM, IEEE Trans. Robotics **26**(1), 78–93 (2010)

46.33 S. Thrun, D. Koller, Z. Ghahramani, H.F. Durrant-Whyte, A.Y. Ng: Simultaneous mapping and localization with sparse extended information filters, Proc.5th Int. Workshop Algorithmic Found. Robotics, ed. by J.-D. Boissonnat, J. Burdick, K. Goldberg, S. Hutchinson (2002)

46.34 M.R. Walter, R.M. Eustice, J.J. Leonard: Exactly sparse extended information filters for feature-based SLAM, Int. J. Robotics Res. **26**(4), 335–359 (2007)

46.35 T. Bailey, J. Nieto, J. Guivant, M. Stevens, E. Nebot: Consistency of the EKF-SLAM algorithm, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 3562–3568

46.36 S. Huang, G. Dissanayake: Convergence and consistency analysis for extended Kalman filter based SLAM, IEEE Trans. Robotics **23**(5), 1036–1049 (2007)

46.37 G.P. Huang, A.I. Mourikis, S.I. Roumeliotis: Observability-based Rules for Designing Consistent EKF SLAM Estimators, Int. J. Robotics Res. **29**, 502–528 (2010)

46.38 N. Metropolis, S. Ulam: The Monte Carlo method, J. Am. Stat. Assoc. **44**(247), 335–341 (1949)

46.39 D. Blackwell: Conditional expectation and unbiased sequential estimation, Ann. Math. Stat. **18**, 105–110 (1947)

46.40 C.R. Rao: Information and accuracy obtainable in estimation of statistical parameters, Bull. Calcutta Math. Soc. **37**(3), 81–91 (1945)

46.41 K. Murphy, S. Russel: Rao–Blackwellized particle filtering for dynamic Bayesian networks. In: *Sequential Monte Carlo Methods in Practice*, ed. by A. Docout, N. de Freitas, N. Gordon (Springer, Berlin 2001) pp. 499–516

46.42 M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit: FastSLAM: A factored solution to the simultaneous localization and mapping problem, Proc. AAAI Natl. Conf. Artif. Intell. (2002)

46.43 J. Pearl: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Morgan Kaufmann, San Mateo 1988)

46.44 J. Guivant, E. Nebot, S. Baiker: Autonomous navigation and map building using laser range sensors in outdoor applications, J. Robotics Syst. **17**(10), 565–583 (2000)

46.45 G. Grisetti, C. Stachniss, W. Burgard: Improved techniques for grid mapping with Rao-Blackwellized particle filters, IEEE Trans. Robotics **23**, 34–46 (2007)

46.46 D. Hähnel, D. Fox, W. Burgard, S. Thrun: A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003)

46.47 C. Stachniss, G. Grisetti, W. Burgard, N. Roy: Evaluation of gaussian proposal distributions for mapping with rao-blackwellized particle filters, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007)

46.48 A. Eliazar, R. Parr: DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks, Proc. 16th Int. Jt. Conf. Artif. Intell. (IJCAI) (2003) pp. 1135–1142

46.49 A. Eliazar, R. Parr: DP-SLAM 2.0, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2004) pp. 1314–1320

46.50 G. Grisetti, G.D. Tipaldi, C. Stachniss, W. Burgard, D. Nardi: Fast and accurate SLAM with Rao-Blackwellized particle filters, J. Robotics Auton. Syst. **55**(1), 30–38 (2007)

46.51 D. Roller, M. Montemerlo, S. Thrun, B. Wegbreit: FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges, Int. Jt. Conf. Artif. Intell. (IICAI) (Morgan Kaufmann, San Francisco 2003) pp. 1151–1156

46.52 R. van der Merwe, N. de Freitas, A. Doucet, E. Wan: The unscented particle filter, Proc. Adv. Neural Inform. Process. Syst. Conf. (2000) pp. 584–590

46.53 C. Stachniss, G. Grisetti, W. Burgard: Recovering particle diversity in a Rao-Blackwellized particle filter for SLAM after actively closing loops, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 655–660

46.54 F. Lu, E. Milios: Globally consistent range scan alignment for environmental mapping, Auton. Robots **4**, 333–349 (1997)

46.55 F. Dellaert: Square root SAM, Robotics Sci. Syst., ed. by S. Thrun, G. Sukhatme, S. Schaal, O. Brock (MIT Press, Cambridge 2005)

46.56 T. Duckett, S. Marsland, J. Shapiro: Learning globally consistent maps by relaxation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 3841–3846

46.57 T. Duckett, S. Marsland, J. Shapiro: Fast, online learning of globally consistent maps, Auton. Robots **12**(3), 287–300 (2002)

46.58 J. Folkesson, H.I. Christensen: Graphical SLAM: A self-correcting map, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 383–390

46.59 U. Frese, G. Hirzinger: Simultaneous localization and mapping – A discussion, Proc. IJCAI Workshop Reason. Uncertain. Robotics (2001) pp. 17–26

46.60 M. Golfarelli, D. Maio, S. Rizzi: Elastic correction of dead-reckoning errors in map building, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998) pp. 905–911

46.61 J. Gutmann, K. Konolige: Incremental mapping of large cyclic environments, Proc. IEEE Int. Symp. Comput. Intell. Robotics Autom. (CIRA) (2000) pp. 318–325

46.62 G. Grisetti, R. Kümmerle, C. Stachniss, W. Burgard: A Tutorial on Graph-based SLAM, IEEE Trans. Intell. Transp. Syst. Mag. **2**, 31–43 (2010)

46.63 K. Konolige: Large-scale map-making, Proc. AAAI Natl. Conf. Artif. Intell. (MIT Press, Cambridge 2004) pp. 457–463

46.64 M. Montemerlo, S. Thrun: Large-scale robotic 3-D mapping of urban structures, Springer Tract. Adv. Robotics **21**, 141–150 (2005)

46.65 G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, C. Hertzberg: Hierarchical optimization on manifolds for online 2D and 3D mapping, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

46.66 M. Kaess, H. Johannsson, R. Roberts, V. Ila, J.J. Leonard, F. Dellaert: iSAM2: Incremental smoothing and mapping using the Bayes tree, Int. J. Robotics Res. **31**, 217–236 (2012)

46.67 M. Kaess, A. Ranganathan, F. Dellaert: iSAM: Incremental Smoothing and Mapping, IEEE Trans. Robotics **24**(6), 1365–1378 (2008)

46.68 C. Estrada, J. Neira, J.D. Tardós: Hierarchical SLAM: Real-time accurate mapping of large environments, IEEE Trans. Robotics **21**(4), 588–596 (2005)

46.69 K. Ni, F. Dellaert: Multi-level submap based SLAM using nested dissection, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010)

46.70 M. Ruhnke, R. Kümmerle, G. Grisetti, W. Burgard: Highly accurate maximum likelihood laser mapping by jointly optimizing laser points and robot poses, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

46.71 M. Ruhnke, R. Kümmerle, G. Grisetti, W. Burgard: Highly accurate 3D surface models by sparse surface adjustment, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

46.72 Y. Liu, S. Thrun: Results for outdoor-SLAM using sparse extended information filters, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

46.73 N. Sünderhauf, P. Protzel: BRIEF-Gist – Closing the loop by simple means, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 1234–1241

46.74 N. Sünderhauf, P. Protzel: Switchable constraints for robust pose graph SLAM, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

46.75 P. Agarwal, G.D. Tipaldi, L. Spinello, C. Stachniss, W. Burgard: Robust map optimization using dynamic covariance scaling, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

46.76 E. Olson: Recognizing places using spectrally clustered local matches, J. Robotics Auton. Syst. **57**(12), 1157–1172 (2009)

46.77 Y. Latif, C. Cadena Lerma, J. Neira: Robust loop closing over time, Robotics Sci. Syst. (2012)

46.78 E. Olson, P. Agarwal: Inference on networks of mixtures for robust robot mapping, Robotics Sci. Syst. (2012)

46.79 F. Dellaert: *Factor graphs and GTSAM: A hands-on introduction*, Tech. Rep. GT-RIM-CP & R-2012-002 (Georgia Tech, Atlanta 2012)

46.80 R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, W. Burgard: G^2o: A general framework for graph optimization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011)

46.81 D.M. Rosen, M. Kaess, J.J. Leonard: An incremental trust-region method for robust online sparse least-squares estimation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 1262–1269

46.82 L. Carlone, R. Aragues, J. Castellanos, B. Bona: A linear approximation for graph-based simultaneous localization and mapping, Robotics Sci. Syst. (2011)

46.83 G. Grisetti, R. Kümmerle, K. Ni: Robust optimization of factor graphs by using condensed measurements, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

46.84 S. Izadi, R.A. Newcombe, D. Kim, O. Hilliges, D. Molyneaux, S. Hodges, P. Kohli, J. Shotton, A.J. Davison, A. Fitzgibbon: Kinectfusion: Real-time dynamic 3D surface reconstruction and interaction, ACM SIGGRAPH Talks (2011) p. 23

46.85 G. Klein, D. Murray: Parallel tracking and mapping for small AR workspaces, IEEE ACM Int. Symp. Mixed Augment. Real. (ISMAR) (2007) pp. 225–234

46.86 F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, W. Burgard: An evaluation of the RGB-D SLAM system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

46.87 H. Strasdat, A.J. Davison, J.M.M. Montiel, K. Konolige: Double window optimisation for constant time visual SLAM, Int. Conf. Computer Vis. (ICCV) (2011)

46.88 A. Nüchter: 3D robot mapping, Springer Tract. Adv. Robotics **52** (2009)

46.89 U. Frese: Treemap: An $O(\log n)$ algorithm for indoor simultaneous localization and mapping, Auton. Robots **21**(2), 103–122 (2006)

46.90 G. Grisetti, C. Stachniss, S. Grzonka, W. Burgard: A tree parameterization for efficiently computing maximum likelihood maps using gradient descent, Robotics Sci. Syst. (2007)

46.91 E. Olson, J.J. Leonard, S. Teller: Fast iterative alignment of pose graphs with poor initial estimates, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2262–2269

46.92 T.M. Cover, J.A. Thomas: *Elements of Information Theory* (Wiley, New York 1991)

46.93 E. Olson, J.J. Leonard, S. Teller: Spatially-adaptive learning rates for online incremental SLAM, Robotics Sci. Syst. (2007)

46.94 G. Grisetti, D. Lordi Rizzini, C. Stachniss, E. Olson, W. Burgard: Online constraint network optimization for efficient maximum likelihood map learning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008)

46.95 M. Kaess, A. Ranganathan, F. Dellaert: Fast incremental square root information smoothing, Int. Jt. Conf. Artif. Intell. (ISCAI) (2007)

46.96 P. Agarwal, E. Olson: Evaluating variable reordering strategies for SLAM, Proc. IEEE/RSJ Int. Conf. Intel Robots Syst. (IROS) (2012)

46.97 K. Ni, D. Steedly, F. Dellaert: Tectonic SAM: exact, out-of-core, submap-based SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 1678–1685

46.98 G. Sibley, C. Mei, I. Reid, P. Newman: Adaptive relative bundle adjustment, Robotics Sci. Syst. (2009)

46.99 P.M. Newman, J.J. Leonard, R. Rikoski: Towards constant-time SLAM on an autonomous underwater vehicle using synthetic aperture sonar, 11th Int. Symp. Robotics Res. (2003)

46.100 M.A. Paskin: Thin junction tree filters for simultaneous localization and mapping, Int. Jt. Conf. Artif. Intell. (IJCAI) (Morgan Kaufmann, New York 2003) pp. 1157–1164

46.101 S.B. Williams: Efficient Solutions to Autonomous Mapping and Navigation Problems, Ph.D. Thesis (ACFR Univ. Sydney, Sydney 2001)

46.102 H. Johannsson, M. Kaess, M.F. Fallon, J.J. Leonard: Temporally scalable visual SLAM using a reduced pose graph, RSS Workshop Long-term Oper. Auton. Robotic Syst. Chang. Environ. (2012)

46.103 N. Carlevaris-Bianco, R.M. Eustice: Generic factor-based node marginalization and edge sparsification for pose-graph SLAM, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

46.104 M. Kaess, F. Dellaert: Covariance recovery from a square root information matrix for data association, J. Robotics Auton. Syst. **57**(12), 1198–1210 (2009)

46.105 H. Kretzschmar, C. Stachniss: Information-theoretic compression of pose graphs for laser-based SLAM, Int. J. Robotics Res. **31**(11), 1219–1230 (2012)

46.106 U. Frese, L. Schröder: Closing a million-landmarks loop, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006)

46.107 J. McDonald, M. Kaess, C. Cadena, J. Neira, J.J. Leonard: Real-time 6-DOF multi-session visual SLAM over large scale environments, J. Robotics Auton. Syst. **61**(10), 1144–1158 (2012)

46.108 H. Strasdat, J.M.M. Montiel, A.J. Davison: Real-time monocular SLAM: Why filter?, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

46.109 P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox: RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments, Int. J. Robotics Res. **31**(5), 647–663 (2012)

46.110 D. Nister, H. Stewenius: Scalable recognition with a vocabulary tree, Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (ICCVPR) (2006) pp. 2161–2168

46.111 R.A. Brooks: Aspects of mobile robot visual map making, Proc. Int. Symp. Robotics Res. (ISRR) (MIT Press, Cambridge 1984) pp. 287–293

46.112 H. Moravec: Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover, Ph.D. Thesis (Stanford Univ., Stanford 1980)

46.113 N. Ayache, O. Faugeras: Building, registrating, and fusing noisy visual maps, Int. J. Robotics Res. **7**(6), 45–65 (1988)

46.114 D. Kriegman, E. Triendl, T. Binford: Stereo vision and navigation in buildings for mobile robots, IEEE Trans. Robotics Autom. **5**(6), 792–803 (1989)

46.115 L. Matthies, S. Shafer: Error modeling in stereo navigation, IEEE J. Robotics Autom. **3**(3), 239–248 (1987)

46.116 S. Pollard, J. Porrill, J. Mayhew: Predictive feedforward stereo processing, Alvey Vis. Conf. (1989) pp. 97–102

46.117 M. Smith, I. Baldwin, W. Churchill, R. Paul, P. Newman: The new college vision and laser data set, Int. J. Robotics Res. **28**(5), 595–599 (2009)

46.118 M. Cummins, P.M. Newman: Appearance-only SLAM at large scale with FAB-MAP 2.0, Int. J. Robotics Res. **30**(9), 1100–1123 (2010)

46.119 P.M. Newman, G. Sibley, M. Smith, M. Cummins, A. Harrison, C. Mei, I. Posner, R. Shade, D. Schroter, L. Murphy, W. Churchill, D. Cole, I. Reid: Navigating, recognising and describing urban spaces with vision and laser, Int, J. Robotics Res. **28**, 11–12 (2009)

46.120 G. Sibley, C. Mei, I. Reid, P. Newman: Vast-scale outdoor navigation using adaptive relative bundle adjustment, Int. J. Robotics Res. **29**(8), 958–980 (2010)

46.121 A. Davison, D. Murray: Mobile robot localisation using active vision, Eur. Conf. Comput. Vis. (ECCV) (1998) pp. 809–825

46.122 A.J. Davison, I. Reid, N. Molton, O. Stasse: MonoSLAM: Real-time single camera SLAM, IEEE Trans., Pattern Anal. Mach. Intell. **29**(6), 1052–1067 (2007)

46.123 R.O. Castle, G. Klein, D.W. Murray: Wide-area augmented reality using camera tracking and mapping in multiple regions, Comput. Vis. Image Underst. **115**(6), 854–867 (2011)

46.124 E. Eade, T. Drummond: Unified loop closing and recovery for real time monocular SLAM, Br. Mach. Vis. Conf. (2008)

46.125 E. Eade, P. Fong, M.E. Munich: Monocular graph SLAM with complexity reduction, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 3017–3024

46.126 K. Konolige, M. Agrawal: FrameSLAM: From bundle adjustment to real-time visual mapping, IEEE Trans. Robotics **24**(5), 1066–1077 (2008)

46.127 D. Nister, O. Naroditsky, J. Bergen: Visual odometry for ground vehicle applications, J. Field Robotics **23**(1), 3–20 (2006)

46.128 D. Scaramuzza, F. Fraundorfer: Visual odometry. Part I: The first 30 years and fundamentals, IEEE Robotics Autom. Mag. **18**(4), 80–92 (2011)

46.129 F. Fraundorfer, D. Scaramuzza: Visual odometry. Part II: Matching, robustness, optimization, and applications, IEEE Robotics Autom. Mag. **19**(2), 78–90 (2012)

46.130 A.S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, N. Roy: Visual odometry and mapping for autonomous flight using an RGB-D camera, Proc. Int. Symp. Robotics Res. (ISRR) (2011)

46.131 J. Sivic, A. Zisserman: Video Google: A text retrieval approach to object matching in videos, Int. Conf. Computer Vis. (ICCV) (2003) p. 1470

46.132 M. Cummins, P.M. Newman: Probabilistic appearance based navigation and loop closing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2042–2048

46.133 C. Cadena, D. Gálvez, F. Ramos, J.D. Tardós, J. Neira: Robust place recognition with stereo cameras, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010)

46.134 R.A. Newcombe, A.J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, A. Fitzgibbon: Kinectfusion: Real-time dense surface mapping and tracking, IEEE/ACM Int. Sym. Mixed Augment. Real. (ISMAR) (2011) pp. 127–136

46.135 J. Neira, A.J. Davison, J.J. Leonard: Guest editorial special issue on visual SLAM, IEEE Trans. Robotics **24**(5), 929–931 (2008)

46.136 K. Konolige, J. Bowman: Towards lifelong visual maps, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 1156–1163

46.137 K. Konolige, J. Bowman, J.D. Chen, P. Mihelich, M. Calonder, V. Lepetit, P. Fua: View-

based maps, Int. J. Robotics Res. **29**(8), 941–957 (2010)
46.138 G. Sibley, C. Mei, I. Reid, P. Newman: Planes, trains and automobiles – Autonomy for the modern robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 285–292
46.139 T. Whelan, H. Johannsson, M. Kaess, J.J. Leonard, J.B. McDonald: Robust real-time visual odometry for dense RGB-D mapping, IEEE Int. Conf. Robotics Autom. (ICRA (2013)
46.140 R.A. Newcombe, S.J. Lovegrove, A.J. Davison: DTAM: Dense tracking and mapping in real-time, Int. Conf. Computer Vis. (ICCV) (2011) pp. 2320–2327
46.141 F. Steinbruecker, J. Sturm, D. Cremers: Real-time visual odometry from dense RGB-D images, Workshop Live Dense Reconstr. Mov. Cameras Int. Conf. Comput. Vis. (ICCV) (2011)
46.142 H. Roth, M. Vona: Moving volume kinectfusion, Br. Mach. Vis. Conf. (2012)

Multimedia Contents

47. Motion Planning and Obstacle Avoidance

Javier Minguez, Florant Lamiraux, Jean-Paul Laumond

This chapter describes motion planning and obstacle avoidance for mobile robots. We will see how the two areas do not share the same modeling background. From the very beginning of motion planning, research has been dominated by computer sciences. Researchers aim at devising well-grounded algorithms with well-understood completeness and exactness properties.

The challenge of this chapter is to present both nonholonomic motion planning (Sects. 47.1–47.6) and obstacle avoidance (Sects. 47.7–47.10) issues. Section 47.11 reviews recent successful approaches that tend to embrace the whole problem of motion planning and motion control. These approaches benefit from both nonholonomic motion planning and obstacle avoidance methods.

47.1 **Nonholonomic Mobile Robots: Where Motion Planning Meets Control Theory** 1178
47.2 **Kinematic Constraints and Controllability** 1179
47.2.1 Definitions 1179
47.2.2 Controllability 1179
47.2.3 Example: The Differentially-Driven Mobile Robot 1180
47.3 **Motion Planning and Small-Time Controllability** 1180
47.3.1 The Decision Problem 1180
47.3.2 The Complete Problem 1181
47.4 **Local Steering Methods and Small-Time Controllability** 1181
47.4.1 Local Steering Methods Accounting for Small-Time Controllability 1182
47.4.2 Equivalence Between Chained-Formed and Feedback-Linearizable Systems 1184
47.5 **Robots and Trailers** 1184
47.5.1 Differentially-Driven Mobile Robots 1184
47.5.2 Differentially-Driven Mobile Robots Towing One Trailer 1184
47.5.3 Car-Like Mobile Robots 1185
47.5.4 Bi-Steerable Mobile Robots 1185
47.5.5 Differentially-Driven Mobile Robots Towing Trailers 1185
47.5.6 Open Problems 1186
47.6 **Approximate Methods** 1186
47.6.1 Forward Dynamic Programming 1186
47.6.2 Discretization of the Input Space 1186
47.6.3 Input-Based Rapidly Exploring Random Trees 1187
47.7 **From Motion Planning to Obstacle Avoidance** 1187
47.8 **Definition of Obstacle Avoidance** 1187
47.8.1 Obstacle Avoidance Problem 1187
47.9 **Obstacle Avoidance Techniques** 1188
47.9.1 Potential Field Methods 1189
47.9.2 Vector Field Histogram 1190
47.9.3 The Obstacle Restriction Method 1191
47.9.4 Dynamic Window Approach 1192
47.9.5 Velocity Obstacles 1192
47.9.6 Nearness Diagram Navigation 1193
47.10 **Robot Shape, Kinematics, and Dynamics in Obstacle Avoidance** 1194
47.10.1 Techniques that Abstract Vehicle Aspects 1194
47.10.2 Techniques of Decomposition in Subproblems 1195

47.11 Integration Planning – Reaction 1196
47.11.1 Systems of Path Deformation 1196
47.11.2 Systems of Tactical Planning 1197
47.12 Conclusions, Future Directions, and Further Reading 1198
Video-References 1199
References 1199

The introduction of nonholonomic constraints has entailed revisiting these algorithms via the introduction of differential geometry materials. Such a combination has been made possible for certain classes of systems, the so-called small-time controllable ones. It remains that the underlying hypothesis of motion planning algorithms is the knowledge of a global and accurate map of the environment. More than that the considered system is a formal system of equations that does not account for the entire physical system: uncertainties in the world or system modeling are not considered. Such hypotheses are too strong in practice. This is why other complementary research has been done in parallel in a more pragmatic but realistic manner. Such research deals with obstacle avoidance. The problem here is not to deal with complicated systems like a car with multiple trailers. The considered systems are much simpler with respect to their geometric shape. The problem considers sensor-based motions to face the physical issues of a real system navigating in a real world better than motion planning algorithms. How can we *navigate* toward a goal in a cluttered environment when the obstacles to avoid have just been discovered in real time? This is the question that obstacle avoidance addresses.

The appearance of mobile robots in the late 1960s early 1970s initiated a new research domain: autonomous navigation. It is interesting to note that the first navigation systems were published at the very first International Joint Conferences on Artificial Intelligence (IJCAI 1969). These systems were based on seminal ideas, which have been very fruitful in the development of robot motion planning algorithms. For instance, in 1969, the mobile robot Shakey used a grid-based approach to model and explore the environment [47.1], in 1977 Jason used a visibility graph built from the corners of the obstacles [47.2], and in 1979 Hilare decomposed the environment into collision-free convex cells [47.3].

In the late 1970s, studies of robot manipulators popularized the notion of the configuration space of a mechanical system [47.4]. In the configuration space the *piano* becomes a point. The motion planning problem for a mechanical system was thus reduced to finding a path for a point in the configuration space. The way was open to extending the seminal ideas and to developing new and well-grounded algorithms (see *Latombe*'s book [47.5]).

One decade later, the notion of nonholonomic systems (also borrowed from mechanics) appeared in the literature [47.6] on robot motion planning through the problem of car parking. This problem had not been solved by the pioneering works on mobile robot navigation. Nonholonomic motion planning then became an attractive research field [47.7].

Besides this research effort in path planning, work was initiated in order to make robots move out of their initially artificial environments where the world was cylindrical and composed of wooden vertical boards. Robots started to move in laboratory buildings, with people walking around. Inaccurate localization, uncertain and incomplete maps of the world, and unexpected moving or static obstacles made roboticists aware of the gap between planning a path and executing a motion. Since then the domain of obstacle avoidance has been very active.

In the 2000s a lot of effort has been made toward the integration of motion planning and obstacle avoidance. This effort was stimulated by the DARPA Urban challenge competition. The most successful integration was done by the Carnegie Mellon University team, who won the competition [47.8].

47.1 Nonholonomic Mobile Robots: Where Motion Planning Meets Control Theory

Nonholonomic constraints are nonintegrable linear constraints over the velocity space of a system. For instance, the rolling without slipping constraint of a differentially-driven mobile robot (Fig. 47.1) is linear with respect to the velocity vector (vector of linear and angular velocities) of the differentially-driven robot and is non integrable (it cannot be integrated into a constraint over the configuration variables). As a consequence, a differentially-driven mobile robot can go anywhere but not following any trajectory. Other types of nonholonomic constraints arise when considering second-order differential equations such as the conser-

vation of inertial momentum. A number of papers investigate the famous falling cat problem or free-floating robots in space [47.7]. This chapter is devoted to nonholonomic constraints for mobile robots with wheels.

While the constraints due to obstacles are expressed directly in the configuration space, that is a manifold, nonholonomic constraints are expressed in the tangent space. In the presence of a linear kinematic constraint, the first question that naturally arises is: *does this constraint reduce the space* reachable *by the system?* This question can be answered by studying the structure of the distribution spanned by the Lie algebra of the control system.

Even in the absence of obstacles, planning admissible motions (i. e., that satisfy the kinematic constraints) for a nonholonomic system between two configurations is not an easy task. Exact solutions have been proposed only for some classes of systems but a lot of systems remain without exact solution. In the general case, however, approximate solutions can be used.

The motion planning problem for a nonholonomic system can be stated as follows: given a map of the environment with obstacles in the workspace, a robot subject to nonholonomic constraints, an initial configuration and a goal configuration, find an admissible collision-free path between the initial and goal configurations. Solving this problem requires taking into account both the configuration space constraints due to obstacles and the nonholonomic constraints. The tools developed to address this issue thus combine motion planning and control theory techniques. Such a combination is possible for the class of so-called small-time controllable systems due to topological arguments (Theorem 47.2 in Sect. 47.3).

47.2 Kinematic Constraints and Controllability

In this section, we give the main definition of controllability, using *Sussman*'s terminology [47.9].

47.2.1 Definitions

Let us denote by CS the configuration space of dimension n of a given mobile robot and by $\boldsymbol{q}$ the configuration of this robot. If the robot is mounted on wheels, it is subject to kinematic constraints, linear in the velocity vector

$$\omega_i(\boldsymbol{q})\dot{\boldsymbol{q}} = 0\,, \quad i \in \{1,\ldots,k\}\,.$$

We assume that these constraints are linearly independent for any $\boldsymbol{q}$. Equivalently, for each $\boldsymbol{q}$, there exists $m = n-k$ linearly independent vectors $f_1(\boldsymbol{q}),\ldots,f_m(\boldsymbol{q})$ such that the above constraints are equivalent to

$$\exists (u_1,\ldots,u_m) \in \mathbb{R}^m\,, \quad \dot{\boldsymbol{q}} = \sum_{i=1}^{m} u_i f_i(\boldsymbol{q})\,. \tag{47.1}$$

Let us note that the choice of vectors $f_i(\boldsymbol{q})$ is not unique. Fortunately, all the following developments are valid whatever choice we make. Moreover, if the linear constraints are smooth, vector fields $f_1,\ldots,f_m$ can be chosen smooth with respect to $\boldsymbol{q}$. We assume this condition from now on.

Let us define by $\mathcal{U}$ a compact subset of $\mathbb{R}^m$. We denote by Σ the control system defined by (47.1), with $(u_1,\ldots,u_m) \in \mathcal{U}$.

Definition 47.1
A local and small-time controllability:

1. Σ is *locally controllable about configuration* $\boldsymbol{q}$ iff the set of configurations reachable from $\boldsymbol{q}$ by an admissible trajectory contains a neighborhood of $\boldsymbol{q}$.
2. Σ is *small-time controllable about configuration* $\boldsymbol{q}$ iff the set of configurations reachable from $\boldsymbol{q}$ by an admissible trajectory in time less than T contains a neighborhood of $\boldsymbol{q}$ for any T.

$f_1,\ldots,f_m$ are called *control vector fields* of Σ. A system's small-time controllable about each configuration is said to be *small-time controllable*.

47.2.2 Controllability

Checking the controllability properties of a system requires the analysis of the control Lie algebra associated with the system. Let us illustrate in an informal way what the Lie bracket of two vector fields is. Consider two basic motions: *go along a straight line* and *turn on the spot*, supported by vector fields denoted by f and g, respectively. Now consider the following combination: go forward during a time t, turn clockwise during the same time t, go backward during the same time t and then turn counterclockwise during the time t. The system reaches a configuration that is not the starting one. Of course when t tends to 0 the goal configuration is very close to the starting one. The direction indicated by such goal configurations when t tends to 0 correspond to a new vector field, which is the Lie bracket of f and g. In a more mathematical formulation, the Lie

bracket $[f, g]$ of two vector fields f and g is defined as being the vector field $\partial f.g - \partial g.f$. The k-th coordinate of $[f, g]$ is

$$[f, g][k] = \sum_{i=1}^{n} \left(g[i] \frac{\partial}{\partial x_i} f[k] - f[i] \frac{\partial}{\partial x_i} g[k] \right) .$$

The following theorem [47.10] gives a powerful result for symmetric systems (a system is said to be symmetric when $\mathcal{U}$ is symmetric with respect to the origin).

Theorem 47.1
A symmetric system is small-time controllable about configuration $\boldsymbol{q}$ iff the rank of the vector space spanned by the family of vector fields f_i together with all their brackets is n at $\boldsymbol{q}$.

Checking the Lie algebra rank condition (LARC) on a control system consists in trying to build a basis of the tangent space from a basis (e.g., a P. Hall family) of the free Lie algebra spanned by the control vector fields. An algorithm is proposed in [47.11, 12].

47.2.3 Example: The Differentially-Driven Mobile Robot

To illustrate the notions developed in this section, we consider the differentially-driven mobile robot displayed in Fig. 47.1. The configuration space of this robot is $\mathbb{R}^2 \times \mathrm{S}^1$, and a configuration can be represented by $\boldsymbol{q} = (x, y, \theta)$ where (x, y) is the position in the horizontal plane of the center of the wheel axis of the robot and θ the orientation with respect to the x-axis. The rolling without slipping kinematic constraint

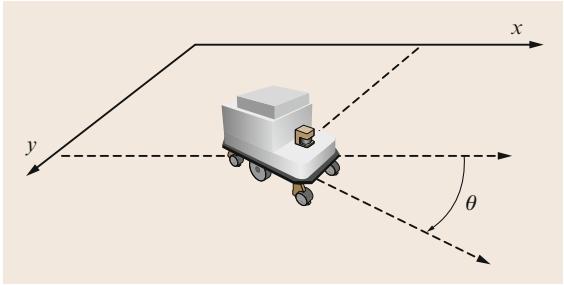


Fig. 47.1 A differentially-driven mobile robot is subject to one linear kinematic constraint, due to the rolling without slipping constraint of the wheel axis

$$-\dot{x} \sin\theta + \dot{y} \cos\theta = 0$$

is linear with respect to the velocity vector $(\dot{x}, \dot{y}, \dot{\theta})$. Therefore, the subspace of admissible velocities is spanned by two vector fields, for instance,

$$f_1(\boldsymbol{q}) = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix} \quad \text{and} \quad f_2(\boldsymbol{q}) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} . \qquad (47.2)$$

The Lie bracket of these two vector fields is

$$f_3(\boldsymbol{q}) = \begin{pmatrix} \sin\theta \\ -\cos\theta \\ 0 \end{pmatrix} .$$

This implies that about any configuration $\boldsymbol{q}$, the rank of the vector space spanned by $f_1(\boldsymbol{q}), f_2(\boldsymbol{q}), f_3(\boldsymbol{q})$ is 3 and, therefore, that the differentially-driven mobile robot is small-time controllable.

47.3 Motion Planning and Small-Time Controllability

Motion planning raises two problems: the first one addresses the *existence* of a collision-free admissible path; this is the decision problem. The second one addresses the *computation* of such a path; this is the complete problem.

47.3.1 The Decision Problem

From now on, we assume that the set of collision-free configurations is an open subset. This implies that contact configurations are assumed in collision.

Theorem 47.2
The existence of a collision-free admissible path for a symmetric small-time controllable mobile robot between two configurations is equivalent to the existence of a collision-free (not necessarily admissible) path between these configurations.

Proof: Let us consider a not necessarily admissible collision-free path between two configurations $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ as a continuous mapping Γ from interval $[0, 1]$ into the configuration space CS such that:

1. $\Gamma(0) = \boldsymbol{q}_1$, $\Gamma(1) = \boldsymbol{q}_2$,
2. for any $t \in [0, 1]$, $\Gamma(t)$ is collision free.

Point 2 implies that for any t there exists a neighborhood $U(t)$ of $\Gamma(t)$ included in the collision-free subset of the configuration space.

Let us denote by $\varepsilon(t)$ the bigger lower bound of the time to collision of all the trajectories starting from

$\Gamma(t)$. As the control vector $(u_1, \ldots, u_m)$ remains in the compact set $\mathcal{U}$, $\varepsilon(t) > 0$.

As the system is small-time controllable about $\Gamma(t)$, the set reachable from $\Gamma(t)$ in time less than $\varepsilon(t)$ is a neighborhood of $\Gamma(t)$, which we denote by $V(t)$.

The collection $\{V(t), t \in [0,1]\}$ is an open covering of the compact set $\{\Gamma(t), t \in [0,1]\}$. Therefore, we can extract a finite covering, $\{V(t_1), \ldots, V(t_l)\}$, where $t_1 = 0 < t_2 < \ldots < t_{l-1} < t_l = 1$ such that for any i between 1 and $l-1$, $V(t_i) \cap V(t_{i+1}) \neq \emptyset$. For each i between 1 and $l-1$, we choose one configuration $\boldsymbol{r}_i$ in $V(t_i) \cap V(t_{i+1})$. As the system is symmetric, there exists an admissible collision-free path between $\boldsymbol{q}(t_i)$ and $\boldsymbol{r}_i$ and between $\boldsymbol{r}_i$ and $\boldsymbol{q}(t_{i+1})$. The concatenation of these paths is a collision-free admissible path between $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$. ■

47.3.2 The Complete Problem

In the former section, we have established that the decision problem, i. e., determining whether there exists a collision-free admissible path between two configurations, is equivalent to determining whether the configurations lie in the same connected component of the collision-free configuration space. In this section we present the tools necessary to solve the complete problem. These tools blend ideas from the classical motion planning problem addressed in Chap. 7 and from open loop control theory, but require specific developments that we are going to present in the next section. Two main approaches have been devised in order to plan admissible collision-free motions for nonholonomic systems. The first one proposed by [47.13] exploits the idea of the proof of Theorem 47.2 by recursively approximating a not necessarily admissible collision-free path by a sequence of feasible paths. The second approach replaces the local method of probabilistic roadmap method (PRM) algorithms (Chap. 7) by a local steering method that connects configuration pairs by admissible paths (VIDEO 707).

Both approaches use a steering method. Before briefly describing them, we give the definition of a local steering method.

Definition 47.2

A local steering method for system Σ is a mapping

$$S_{\text{loc}} : CS \times CS \to C^1_{\text{pw}}([0,1], CS)$$
$$(\boldsymbol{q}_1, \boldsymbol{q}_2) \mapsto S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)$$

5 where $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)$ is a piecewise continuously differentiable curve in CS satisfying the following properties:

1. $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)$ satisfies the kinematic constraints associated to Σ,
2. $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)$ connects $\boldsymbol{q}_1$ to $\boldsymbol{q}_2$: $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)(0) = \boldsymbol{q}_1$, $S_{\text{loc}}(\boldsymbol{q}_1, \boldsymbol{q}_2)(1) = \boldsymbol{q}_2$.

Approximation of a Not Necessarily Admissible Path

A not necessarily admissible collision-free path $\Gamma(t), t \in [0,1]$ connecting two configurations and a local steering method S_{loc} being given, the approximation algorithm proceeds recursively, by calling function `approximation` defined by Algorithm 47.1 with input Γ, 0 and 1.

Algorithm 47.1

`approximation` function: inputs are a path Γ and two abscissas $t1$ and $t2$ along this path

if $S_{\text{loc}}(\Gamma(t1), \Gamma(t2))$ collision free **then**
  return $S_{\text{loc}}(\Gamma(t1), \Gamma(t2))$
else
  return concat(`approximation`(Γ,t1,(t1+t2)/2),
    `approximation`(Γ,(t1+t2)/2,t2))
end if

Sampling-Based Roadmap Methods

Most sampling-based roadmap methods as described in Chap. 6 can be adapted to nonholonomic systems by replacing the connection method between pairs of configurations by a local steering method. This strategy is rather efficient for PRM algorithms. For the rapidly-exploring random tree (RRT) method, the efficiency strongly depends on the metric used to choose the nearest neighbor. The distance function between two configurations needs to account for the length of the path returned by the local steering method to connect these configurations [47.14].

47.4 Local Steering Methods and Small-Time Controllability

The approximation algorithm described in the former section is recursive and raises the completeness question: does the algorithm finish in finite time or may it fail to find a solution?

A sufficient condition for the approximation algorithm to find a solution in a finite number of iterations is that the local steering method accounts for small-time controllability.

Definition 47.3

A local steering method S_{loc} accounts for the small-time controllability of system Σ iff

For any $\boldsymbol{q} \in CS$, for any neighborhood U of $\boldsymbol{q}$, there exists a neighborhood V of $\boldsymbol{q}$ such that for any $\boldsymbol{r} \in V$, $S_{\text{loc}}(\boldsymbol{q}, \boldsymbol{r})([0, 1]) \subset U$.

In other words, a local steering method accounts for the small-time controllability of a system if it produces paths getting closer to the configurations it connects when these configurations get closer to each other.

This property is also sufficient for probabilistic completeness of roadmap sampling-based methods.

47.4.1 Local Steering Methods Accounting for Small-Time Controllability

Constructing a local steering method that accounts for small-time controllability is a difficult task that has been achieved only for a few classes of systems. Most mobile robots studied in the domain of motion planning are wheeled mobile robots towing trailers or not.

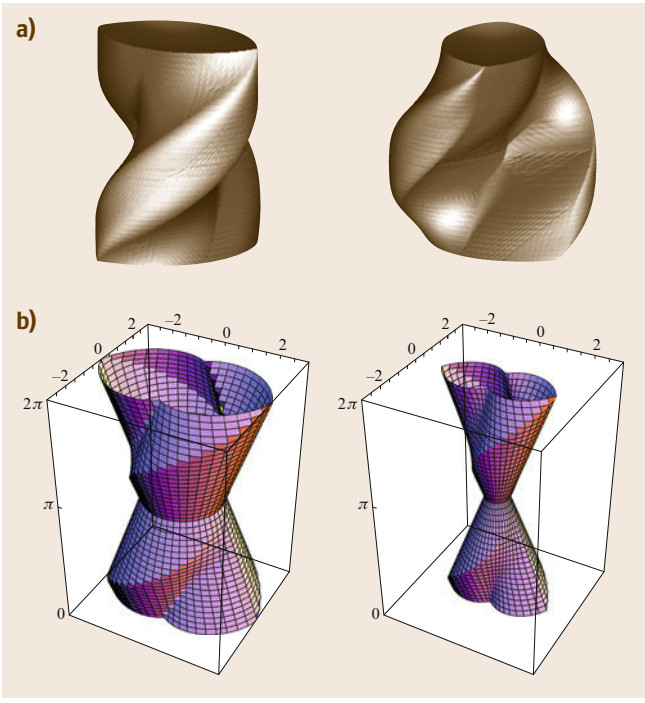


Fig. 47.2 **(a)** Two perspective views of an RS (Reeds and Shepp) ball: set of configurations reachable by a path of length less than a given distance for the Reeds and Shepp car. **(b)** Perspective views of two DD (differentially driven) balls: set of configurations reachable by a path of a length less than a given distance for the differentially-driven robot. Orientation θ is represented on z-axis

Steering Using Optimal Control

The simplest system, namely the differentially-driven robot presented in Sect. 47.2.3 with bounded velocities or the so-called *Reeds* and *Shepp* [47.15] car with bounded curvature have the same control vector fields (47.2). The difference lies in the domain of the control variables:

- $-1 \leq u_1 \leq 1$, $|u_2| \leq |u_1|$ for RS car,
- $|u_1| + b|u_2| \leq 1$ for the differentially-driven robot,

where b is half of the distance between the right and left wheels. For these systems, a local steering method accounting for small-time controllability can be constructed using optimal control theory. For any admissible path defined over an interval I of one of these systems, we define a length as follows

$$\int_I |u_1| \quad \text{for the RS car ,}$$

$$\int_I |u_1| + b|u_2| \quad \text{for the differentially-driven robot .}$$

The length of the shortest path between two configurations in both cases defines a metric over the configuration space. The synthesis of the shortest paths, i. e., the determination of the shortest path between any pair of configurations was achieved by [47.16] for the RS car and later by [47.17] for the differentially-driven robot. Figure 47.2 shows a representation of the balls corresponding to these metrics.

Optimal control naturally defines a local steering method that associates a shortest path between these configurations to any pair of configurations. Let us note that the shortest path is unique between most pairs of configurations. A general result states that the collection of balls of radius $r > 0$ centered about $\boldsymbol{q}$ induced by nonholonomic metrics constitutes an increasing collection of neighborhoods of $\boldsymbol{q}$, the intersection of which is $\{\boldsymbol{q}\}$. This property directly implies that local steering methods based on shortest paths account for small-time controllability.

The main advantage of optimal control is that it provides both a local steering method and a distance metric consistent with the steering method. This makes the steering methods well suited for path planning algorithms designed for holonomic systems and using a distance function, such as RRT (Chap. 7), for instance.

Unfortunately, the synthesis of the shortest paths has been realized only for the two simple systems described in this section. For more complex systems, the problem remains open.

The main drawback of the shortest path-based steering methods described in this section is that input

functions are not continuous. This requires an additional step to compute a time-parameterization of the paths before motion execution. Along this time-parameterization, input discontinuities force the robot to stop. For instance, to follow two successive arcs of circles of opposite curvature a mobile robot needs to stop between the arcs of circle in order to ensure continuity of the linear and angular velocities u_1 and u_2.

Steering Chained-Form Systems

Some classes of systems can be put into the form called the *chained-form* system by a change of variable

$$\dot{z}_1 = u_1 \,, \tag{47.3}$$

$$\dot{z}_2 = u_2 \,, \tag{47.4}$$

$$\dot{z}_3 = z_2 u_1 \,, \tag{47.5}$$

$$\vdots \quad \vdots$$

$$\dot{z}_n = z_{n-1} u_1 \,. \tag{47.6}$$

Let us consider the following inputs [47.18]

$$\begin{cases} u_1(t) &= a_0 + a_1 \sin \omega t \,, \\ u_2(t) &= b_0 + b_1 \cos \omega t \\ &\quad + \ldots + b_{n-2} \cos(n-2)\omega t \,. \end{cases} \tag{47.7}$$

Let $Z^{\text{start}} \in \mathbb{R}^n$ be a starting configuration. Each $z_i(1)$ can be computed from the coordinates of Z^{start} and parameters $(a_0, a_1, b_0, b_1, \ldots, b_{n-2})$. For a given $a_1 \neq 0$ and a given configuration Z^{start}, the mapping from $(a_0, b_0, b_1, b_2, b_3)$ to $Z(1)$ is a C^1-diffeomorphism at the origin; the system is then invertible. For n smaller or equal to 5, parameters $(a_0, b_0, b_1, \ldots, b_{n-2})$ can be analytically computed from the coordinates of the two configurations Z^{start} and Z^{goal}. The corresponding sinusoidal inputs steer the system from Z^{start} to Z^{goal}. The shape of the path only depends on parameter a_1. Each value of a_1 thus defines a local steering method denoted by $S^{a_1}_{\sin}$. None of these steering methods account for small-time controllability since for any $Z \in \mathbb{R}^n$, $S^{a_1}_{\sin}(Z, Z)([0, 1])$ is not reduced to $\{Z\}$. To construct a local steering method accounting for small-time controllability from the collection of $S^{a_1}_{\sin}$, we need to make a_1 depend on the configurations Z_1 and Z_2 that we want to connect

$$\lim_{Z^2 \to Z^1} a_1(Z^1, Z^2) = 0 \,,$$

$$\lim_{Z^2 \to Z^1} a_0\left[Z^1, Z^2, a_1(Z^1, Z^2)\right] = 0 \,,$$

$$\lim_{Z^2 \to Z^1} b_i\left[Z^1, Z^2, a_1(Z^1, Z^2)\right] = 0 \,.$$

Such a construction is achieved in [47.19].

Steering Feedback-Linearizable Systems

The concept of *feedback linearizability* (or differential flatness) was introduced by *Fliess* et al. [47.20, 21].

A system is said to be *feedback linearizable* if there exists an output (i. e., function of the state, input and input derivatives) called the *linearizing output*, such that the state and the input of the system is a function of the linearizing output and its derivatives. The dimension of the linearizing output is the same as the dimension of the input.

Let us illustrate this notion with a simple example. We consider a differentially-driven mobile robot towing a trailer hitched on top of the wheel axis of the robot, as displayed in Fig. 47.3. The tangent to the curve followed by the center of the wheel axis of the trailer gives the orientation of the trailer. From the orientation of the trailer along the curve, we can deduce the curve followed by the center of the robot. The tangent to the curve followed by the center of the robot gives the orientation of the robot. Thus the linearizing output of this system is the center of the wheel axis of the trailer. By differentiating the linearizing output twice, we can reconstruct the configuration of the system.

Feedback linearizability is very interesting for steering purposes. Indeed, the linearizing output is not subject to any kinematic constraints. Therefore, if we know the relation between the state and the linearizing output, planning an admissible path between two configurations simply consists in building a curve in $\mathbb{R}^m$,

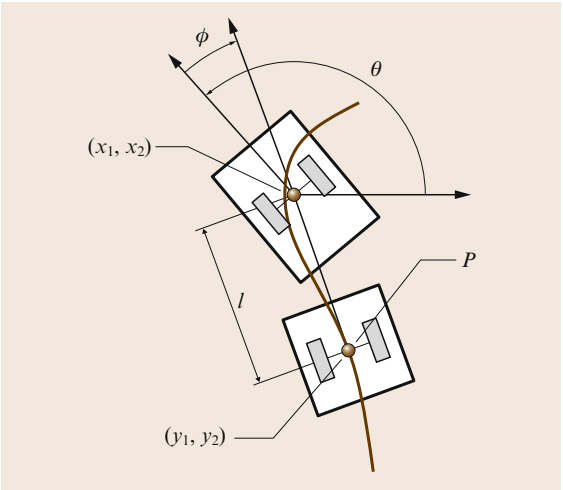


Fig. 47.3 A differentially-driven robot towing a trailer. Hitched on top of the wheel axis of the robot is a feedback-linearizable system. The linearizing output is the center of the wheel axis of the trailer. The configuration of the system can be reconstructed by differentiating the curve $\boldsymbol{y}(s)$, where s is an arc-length parameterization, followed by the linearizing output. The orientation of the trailer is given by $\tau = \arctan(\dot{y}_2/\dot{y}_1)$. The angle between the robot and the trailer is given by $\phi = -l \arctan(\mathrm{d}\tau/\mathrm{d}s)$. l is the length of the trailer connection

where m is the dimension of the input with differential constraints at both ends. This problem can be easily solved using, for instance, polynomials.

For two input driftless systems like Σ, the linearizing output only depends on state $\boldsymbol{q}$. The state $\boldsymbol{q}$ depends on the linearizing output through the parameterization invariant values, namely, the linearizing output $\boldsymbol{y}$, the orientation τ of the vector tangent to the curve followed by $\boldsymbol{y}$ and the successive derivatives of τ with respect to the curvilinear abscissa s.

Thus, the configuration of a two input feedback-linearizable driftless system of dimension n can be represented by a vector $(\boldsymbol{y}, \tau, \tau^1, \ldots, \tau^{n-3})$ representing the geometric properties of the curve followed by the linearizing output along an admissible path passing through the configuration.

Therefore, designing a local steering method for such a system is equivalent to associating to any pair of vectors $(\boldsymbol{y}_1, \tau_1, \tau_1^1, \ldots, \tau_1^{n-3})$, $(\boldsymbol{y}_2, \tau_2, \tau_2^1, \ldots, \tau_2^{n-3})$ a curve in the plane starting from $\boldsymbol{y}_1$ and ending at $\boldsymbol{y}_2$ with orientation of the tangent vector and successive derivatives with respect to s and equal to $\tau_1, \tau_1^1, \ldots, \tau_1^{n-3}$ at the beginning and to $\tau_2, \tau_2^1, \ldots, \tau_2^{n-3}$ at the end. This exercise is relatively easy using polynomials and transforming the boundary conditions into linear equations over the coefficients of the polynomials. However, taking into account small-time controllability is a little bit more tricky. Reference [47.22] proposes a flatness-based steering method built on convex combinations of canonical curves.

47.4.2 Equivalence Between Chained-Formed and Feedback-Linearizable Systems

In the previous section, we proposed methods to steer feedback-linearizable control systems or systems that can be put into chained-form. We now give a necessary and sufficient condition for feedback-linearizability.

Feedback-Linearizability: A Necessary and Sufficient Condition

In [47.23], *Rouchon* gives conditions to check whether or not a system is feedback-linearizable. For two-input driftless systems a necessary and sufficient condition is the following: let us define as Δ^k, $k > 0$ the collection of distributions (i. e., set of vector fields) iteratively defined by: $\Delta_0 = \text{span}\{f_1, f_2\}$, $\Delta_1 = \text{span}\{f_1, f_2, [f_1, f_2]\}$ and $\Delta_{i+1} = \Delta_0 + [\Delta_i, \Delta_i]$ with $[\Delta_i, \Delta_i] = \text{span}\{[f, g]\,,\, f \in \Delta_i,\, g \in \Delta_i\}$. A system with two-dimensional input is feedback-linearizable iff $\text{rank}(\Delta_i) = 2 + i$.

Example: The Chained-Form System

Let us consider the chained-form system defined by (47.3)–(47.6). The control vector fields of this system are

$$f_1 = (1, 0, z_2, \ldots, z_{n-1})\,,$$
$$f_2 = (0, 1, \ldots, 0)\,,$$

$\text{rank}\Delta_0 = 2$. If we compute $f_3 = [f_1, f_2] = (0, 0, 1, 0, \ldots, 0)$, we note that $\text{rank}\Delta_1 = 3$. By computing $f_i = [f_1, f_{i-1}]$ for i up to n, we find a sequence $f_i = (0, \ldots, 0, 1, 0, \ldots, 0)$, where 1 is at position i. Therefore, $\text{rank}\Delta_i = 2 + i$ for i up to $n-2$ and the chained-form system is feedback linearizable. This conclusion could have been drawn in a more straightforward way by noticing that the state can be reconstructed from (z_1, z_2) and its derivatives. (z_1, z_2) is thus the linearizing output of the chained-form system.

47.5 Robots and Trailers

Robotics systems studied in motion planning are mainly those composed of a mobile robot alone or towing one or several trailers. The input of these systems is two dimensional.

47.5.1 Differentially-Driven Mobile Robots

The simplest mobile robot, namely the differentially-driven mobile robot displayed in Fig. 47.1, is obviously feedback-linearizable. the trajectory of the center of the wheel axis (the linearizing output) of the robot gives the orientation of the robot. It can thus be steered using a flatness-based local steering method.

47.5.2 Differentially-Driven Mobile Robots Towing One Trailer

The differentially-driven mobile robot towing a trailer hitched on top of the wheel axis of the robot displayed in Fig. 47.3 is feedback linearizable and the linearizing output is the center of the wheel axis of the trailer. The differentially-driven mobile robot towing a trailer with hitched kingpin (Fig. 47.4) is also feedback lineariz-

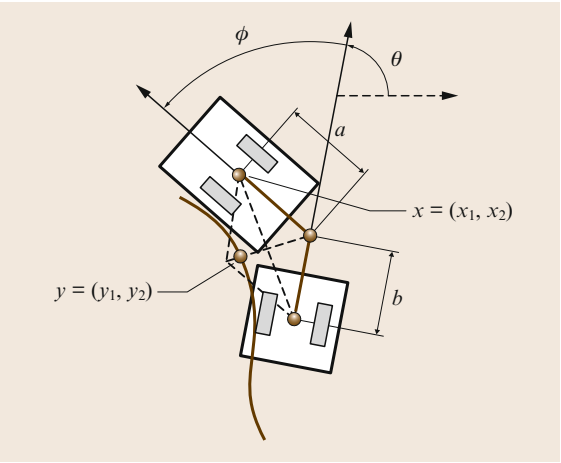


Fig. 47.4 A differentially-driven mobile robot towing a trailer with hitched kingpin is feedback linearizable

able [47.24] but the relation between linearizing output and configuration variables is more intricate

$$\begin{cases} y_1 = x_1 - b\cos\theta + L(\phi)\dfrac{b\sin\theta + a\sin(\theta+\phi)}{\sqrt{a^2+b^2+2ab\cos\phi}}\,, \\ y_2 = x_2 - b\sin\theta - L(\phi)\dfrac{a\cos(\theta+\phi) + b\cos\theta}{\sqrt{a^2+b^2+2ab\cos\phi}}\,, \end{cases}$$

where L is defined by the following elliptic integral

$$L(\phi) = ab\int_0^{\phi} \frac{\cos\sigma}{\sqrt{a^2+b^2+2ab\cos(\sigma)}}\mathrm{d}\sigma\,,$$

which is the linearizing output of the system. These relations together with the following ones

$$\tan\tau = \frac{a\sin(\theta+\phi) + b\sin\theta}{b\cos\theta + a\cos(\theta+\phi)}\,,$$

$$\kappa = \frac{\sin(\phi)}{\cos\phi\sqrt{a^2+b^2+2ab\cos\phi} + L(\phi)\sin(\phi)}\,,$$

make it possible to reconstruct the configuration variables from the linearizing output and its two first derivatives.

47.5.3 Car-Like Mobile Robots

A car-like mobile robot is constituted of a fixed rear wheel axis and of two steerable front wheels, the axes of which intersect at the center of curvature (Fig. 47.5). A car-like mobile robot is kinematically equivalent to a differentially-driven mobile robot towing a trailer hitched on top of the wheel axis of the robot (Fig. 47.3): the virtual front wheel corresponds to the differentially-driven robot, while the body of the car corresponds to the trailer.

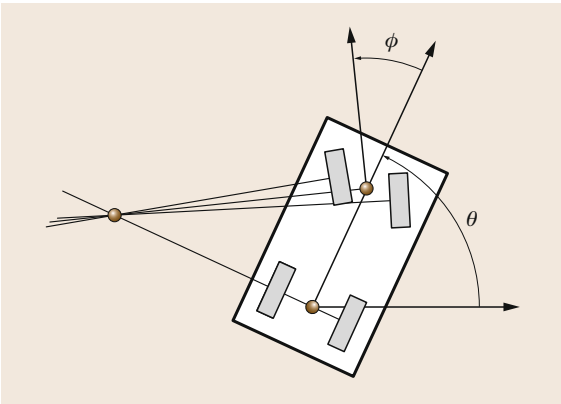


Fig. 47.5 Car-like mobile robot. The axes of the front wheels intersect at the center of curvature. The steering angle ϕ is the angle between the longitudinal axis of the car and a virtual front wheel in the middle of the two front wheels

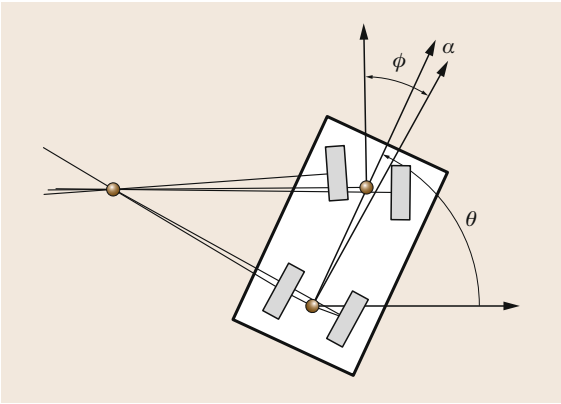


Fig. 47.6 Bi-steerable robot. The front and rear wheels are steerable and there is a relation between the rear steering angle and the front steering angle: $\alpha = f(\phi)$

47.5.4 Bi-Steerable Mobile Robots

The bi-steerable mobile robot (Fig. 47.6) is a car with front and rear steerable wheels and with a relation between the front and rear steering angles. This system has been proved to be feedback linearizable in [47.25]. As for the mobile robot towing a trailer with hitched kingpin, the linearizing output is a moving point in the robot reference frame.

47.5.5 Differentially-Driven Mobile Robots Towing Trailers

Let us consider the differentially-driven mobile robot towing a trailer connected on top of the wheel axis of the robot, and let us add an arbitrary number of trailers,

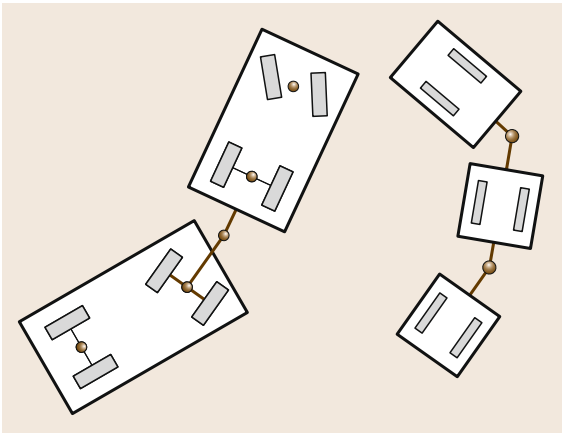


Fig. 47.7 A truck towing a towbar trailer with hitched kingpin and a differentially-driven mobile robot towing two trailers with hitched kingpin: two open problems for exact path planning for nonholonomic systems

each one connected on top of the wheel axis of the previous one. By differentiating once the curve followed by the center of the last trailer, we get the orientation of the last trailer (this orientation coincides with the orientation of the tangent to the curve). If we know the orientation and the position of the center of the last trailer along the path, we can reconstruct the curve followed by the center of the previous trailer. Repeating this reasoning, we can reconstruct the trajectory of the whole system by differentiating a sufficient number of times. The system is therefore feedback linearizable and the linearizing output is the center of the last trailer.

Combining the above reasoning with the systems reviewed in this section, we can build hybrid feedback-linearizable trailer systems. For instance, a differentially-driven mobile robot towing an arbitrary number n of trailers each one connected on top of the wheel axis of the previous trailer, except the last trailer with hitched kingpin is feedback linearizable. Simply consider the two last trailers as a feedback-linearizable system composed of a mobile robot with one trailer as in Fig. 47.4. The linearizing output of this system enables us to reconstruct the trajectory of the two last trailers. The center of trailer $n-1$ is a linearizing output for the mobile robot towing the $n-1$ last trailers.

47.5.6 Open Problems

Finally, all the systems for which we are able to plan exact motions between arbitrary pairs of configurations are included in the large class of feedback-linearizable systems. We have indeed seen that chained-form systems are also part of this class. For other systems, no exact solution has been proposed up to now. For instance, neither of the systems displayed in Fig. 47.7 is feedback linearizable. They do not satisfy the necessary condition of Sect. 47.4.2.

47.6 Approximate Methods

To deal with nonholonomic systems not belonging to any class of systems for which exact solutions exist, numerical approximate solutions have been developed. We review some of these methods in this section.

47.6.1 Forward Dynamic Programming

In [47.26] *Barraquand* and *Latombe* propose a dynamic programming approach to nonholonomic path planning. Admissible paths are generated by a sequence of constant input values, applied over a fixed interval of time δt. Starting from the initial configuration the search generates a tree: the children of a given configuration $\boldsymbol{q}$ are obtained by setting the input to a constant value and integrating the differential system over δt. The configuration space is discretized into an array of cells of equal size (i. e., hyper-parallelepipeds). A child $\boldsymbol{q}'$ of a configuration $\boldsymbol{q}$ is inserted in the search tree if and only if the computed path from $\boldsymbol{q}$ to $\boldsymbol{q}'$ is collision free and $\boldsymbol{q}'$ does not belong to a cell containing an already generated configuration. The algorithm stops when it generates a configuration belonging to the same cell as the goal (i. e., it does not necessarily reach the goal exactly).

The algorithm has been proved to be asymptotically complete with respect to both δt and the size of the cells. As a brute force method, it remains quite time consuming in practice. Its main interest is that the search is based on Dijkstra's algorithm, which allows taking into account optimality criteria such as the path length or the number of reversals. Asymptotical optimality to generate the minimum of reversals is proved for the car-like robot alone.

47.6.2 Discretization of the Input Space

In [47.27] *Divelbiss* and *Wen* propose a method to produce an admissible collision-free path for a nonholonomic mobile robot in the presence of obstacles. They restrict the set of input functions over the subspace spanned by Fourier basis over interval $[0, 1]$. An in-

put function is thus represented by a finite-dimensional vector λ. Reaching a goal configuration thus becomes a nonlinear system of equations, the unknowns of which are the coordinates λ_i's of λ. The authors use the Newton–Raphson method to find a solution. Obstacles are defined by inequality constraints over the configuration space. The path is discretized into N samples. The noncollision constraint expressed at these sample points yields the inequality constraint over vector λ. These inequality constraints are turned into equality constraints through function g defined as

$$g(c) = \begin{cases} (1-\mathrm{e}^c)^2 & \text{if } c > 0\,, \\ 0 & \text{if } c \le 0\,. \end{cases}$$

Reaching a goal configuration while avoiding obstacles thus becomes a nonlinear system of equations over vector λ again solved using the Newton–Raphson method. The method is rather efficient for short maneuvers. The main difficulty is to tune the order of the Fourier expansion. Long motions in cluttered environments require a higher order, while motion in empty space can be solved with a low order. The authors do not mention the problem of numerical instability in the integration of the dynamic system.

47.6.3 Input-Based Rapidly Exploring Random Trees

The RRT algorithms described in Chap. 7 can be used without a local steering method to plan paths for nonholonomic systems. New nodes can be generated from existing nodes by applying random input functions over an interval of time [47.28]. The main difficulty consists in finding a distance function that really accounts for the distance the system needs to travel to go from one configuration to another. Moreover, the goal is never exactly reached. This latter drawback can be overcome by post-processing the path returned by RRT using a path deformation method, as described in [47.29].

47.7 From Motion Planning to Obstacle Avoidance

Up to now we have described motion planning techniques. Their objective is to compute a collision-free trajectory to the target configuration that complies with the vehicle constraints. They assume a perfect model of the robot and scenario. The advantage of these techniques is that they provide complete and global solutions of the problem. Nevertheless, when the surroundings are unknown and unpredictable, these techniques fail.

A complementary way to face the motion problem is obstacle avoidance. The objective is to move a vehicle towards a target location free of collisions with the obstacles collected by the sensors during motion execution. The advantage of reactive obstacle avoidance is to compute motion by introducing the sensor information within the control loop, used to adapt the motion to any contingency incompatible with initial plans.

The main cost of considering the reality of the world during execution is locality. In this instance, if global reasoning is required, a trap situation could occur. Despite this limitation, obstacle avoidance techniques are mandatory to deal with mobility problems in unknown and evolving surroundings.

Notice that methods have been developed to combine both the global point of view of motion planning and the local point of view of obstacle avoidance. How can we consider robot perception at the planning level? This is so-called sensor-based motion planning. Several variants exists, such as the Bug algorithms initially introduced in [47.30]. However, non of them consider the practical context of nonholonomic mobile robots.

47.8 Definition of Obstacle Avoidance

Let $\mathcal{A}$ be the robot (a rigid object) moving in the workspace $\mathcal{W}$, whose configuration space is CS. Let $\boldsymbol{q}$ be a configuration, $\boldsymbol{q}_t$ this configuration in time t, $\mathcal{A}(\boldsymbol{q}_t) \in \mathcal{W}$ the space occupied by the robot in this configuration. In the vehicle there is a sensor, which in $\boldsymbol{q}_t$ measures a portion of the space $S(\boldsymbol{q}_t) \subset \mathcal{W}$ identifying a set of obstacles $\mathcal{O}(\boldsymbol{q}_t) \subset \mathcal{W}$.

Let $\boldsymbol{u}$ be a constant control vector and $\boldsymbol{u}(\boldsymbol{q}_t)$ this control vector applied in $\boldsymbol{q}_t$ during time δt. Given $\boldsymbol{u}(\boldsymbol{q}_t)$, the vehicle describes a trajectory $\boldsymbol{q}_{t+\delta t} = f(\boldsymbol{u}, \boldsymbol{q}_t, \delta t)$, with $\delta t \ge 0$. Let $\mathcal{Q}_{t,T}$ be the set of configurations of the trajectory followed from $\boldsymbol{q}_t$ with $\delta t \in [0, T]$, a given time interval. $T > 0$ is called the *sampling period*. Let $F : CS \times CS \to \mathbb{R}^+$ be a function that evaluates the progress of one configuration to another.

47.8.1 Obstacle Avoidance Problem

Let $\boldsymbol{q}_{\text{target}}$ be a target configuration. Then, in time t_i the robot $\mathcal{A}$ is in $\boldsymbol{q}_{t_i}$, where a sensor measurement

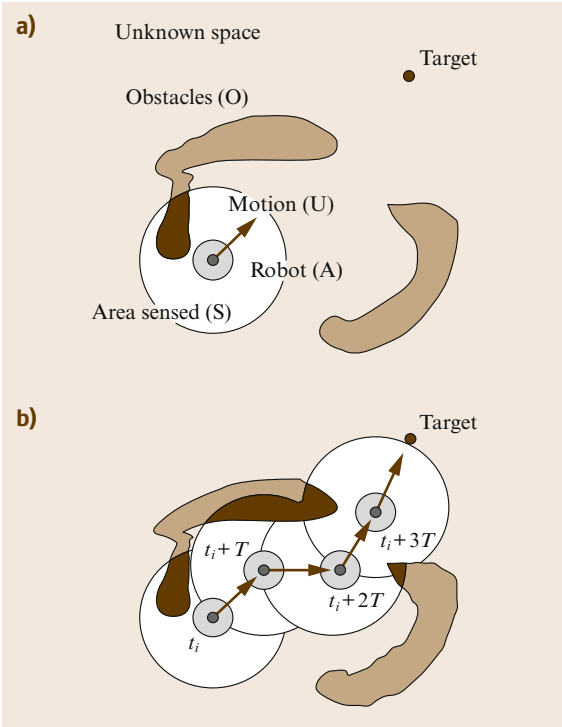


Fig. 47.8 **(a)** Obstacle avoidance problem consists of computing motion control that avoids collisions with the obstacles gathered by the sensors whilst driving the robot towards the target location. **(b)** The result of applying this technique at each time is a sequence of motions that drive the vehicle free of collisions to the target

is obtained $S(\boldsymbol{q}_{t_i})$*, and thus an obstacle description* $\mathcal{O}(\boldsymbol{q}_{t_i})$.

The objective is to compute a motion control $\boldsymbol{u}_i$*, such that (i) the trajectory generated is free of collisions with the obstacles* $\mathcal{A}(\mathcal{Q}_{t_i,T}) \bigcap \mathcal{O}(\boldsymbol{q}_{t_i}) = \emptyset$*; and (ii) it makes the vehicle progress to the target location* $F(\boldsymbol{q}_{t_i}, \boldsymbol{q}_{\text{target}}) < F(\boldsymbol{q}_{t_i+T}, \boldsymbol{q}_{\text{target}})$.

The result of solving this problem at each sample time (Fig. 47.8a) is a sequence of motion controls $\{\boldsymbol{u}_1 \ldots \boldsymbol{u}_n\}$ computed in execution time that avoids the obstacles gathered by the sensors, while making the vehicle progress towards the target location in each configuration $\{\boldsymbol{q}_{t_1} \ldots \boldsymbol{q}_{\text{target}}\}$ (Fig. 47.8b). Notice that the motion problem is a global problem. The obstacle avoidance methods are local and iterative techniques to address this problem. The disadvantage of locality (trap situations) is counteracted with the advantage of introducing the sensory information within the control cycle (to take into account the reality of the world in execution).

There are at least three aspects that affect the development of an obstacle avoidance method: the avoidance technique, the type of robot-sensor and the type of scenario. These subjects correspond to the next three sections. First, we describe the obstacle avoidance techniques (Sect. 47.9). Second, we discuss the techniques to adapt a given obstacle avoidance method to work on a vehicle taking into account the shape, kinematics, and dynamics (Sect. 47.10). The sensory processing is detailed in Chaps. 5 and 25 of this book. Finally, the usage of an obstacle avoidance technique on a vehicle in a given scenario is highly dependent on the scenario nature (static or dynamic, unknown or known, structured or not, or its size, for example). Usually, this problem is associated with the integration of planning and obstacle avoidance (Sect. 47.11).

47.9 Obstacle Avoidance Techniques

We describe here a taxonomy of obstacle avoidance techniques and some representative methods. First there are two groups, methods that compute the motion in one step and methods that do it in more than one. Methods of one step directly reduce the sensor information to motion control. There are two types:

- The *heuristic* methods were the first techniques used to generate motion based on sensors. The majority of this work was derived in classic planning methods and will not be described here. See [47.1, 30–33] for some representative work.
- The methods of *physical analogies* assimilate the obstacle avoidance to a known physical problem. We discuss here the potential field methods [47.34, 35]. Other works are variants adapted to uncertain models [47.36] or that use other analogies [47.37–39].

Methods with more than one step compute some intermediate information, which is processed next to obtain the motion:

- The methods of *subset of controls* compute an intermediate set of motion controls, and next they choose one of them as a solution. There are two types: (i) methods that compute a subset of motion directions. We describe here the vector field histogram (VFH) [47.40] and the obstacle restriction method [47.41] (see [47.42] for three-dimensional (3-D) workspaces). Another method is the steer angle field approach [47.43]. (ii) Methods that com-

pute a subset of velocity controls. We describe here the dynamic window approach (DWA) [47.44] and velocity obstacles [47.45] (see [47.46] for moving obstacles). Another method based on similar principles but developed independently is the curvature velocity method [47.47].

- Finally, there are methods that compute some *high-level information* such as intermediate information, which is translated next in motion. We describe the basic versions of the nearness diagram navigation [47.48, 49]. The reader is directed to [47.50–52] for further improvements.

All the methods outlined here have advantages and disadvantages, depending on the navigation context, like uncertain worlds, motion at high speeds, motion in confined or troublesome spaces, etc. Unfortunately there is no metric available to quantitatively measure the performance of the methods. However, for an experimental comparison in terms of their intrinsic problems see [47.48].

47.9.1 Potential Field Methods

The potential field method (PFM) uses an analogy in which the robot is a particle that moves in the configuration space under the influence of a force field. While the target location exerts a force that attracts the particle $\boldsymbol{F}_{\text{att}}$, the obstacles exert repulsive forces $\boldsymbol{F}_{\text{rep}}$. At each time t_i, the motion is computed to follow the direction of the artificial force induced by the sum of both potentials $\boldsymbol{F}_{\text{tot}}(\boldsymbol{q}_{t_i}) = \boldsymbol{F}_{\text{att}}(\boldsymbol{q}_{t_i}) + \boldsymbol{F}_{\text{rep}}(\boldsymbol{q}_{t_i})$ (the most promising motion direction), Fig. 47.9.

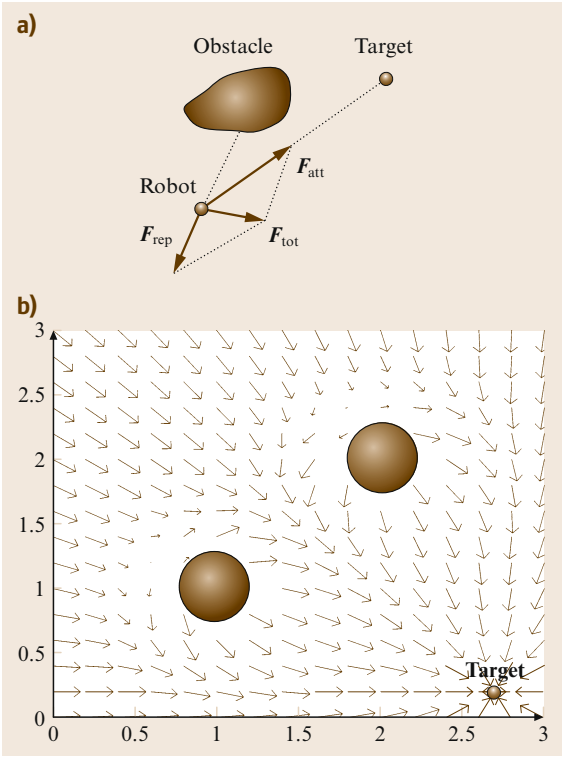


Fig. 47.9 **(a)** Computation of the motion direction with a potential field method. The target attracts the particle $\boldsymbol{F}_{\text{att}}$ while the obstacle exerts a repulsive force $\boldsymbol{F}_{\text{rep}}$. The resulting force $\boldsymbol{F}_{\text{tot}}$ is the most promising motion direction. **(b)** Motion directions computed in each point of the space with the classic method

Example

$$\boldsymbol{F}_{\text{att}}(\boldsymbol{q}_{t_i}) = K_{\text{att}}\boldsymbol{n}_{q_{\text{target}}} \tag{47.8}$$

$$\boldsymbol{F}_{\text{rep}}(\boldsymbol{q}_{t_i}) = \begin{cases} K_{\text{rep}} \sum_j \left(\dfrac{1}{d(\boldsymbol{q}_{t_i}, \boldsymbol{p}_j)} - \dfrac{1}{d_0} \right) \boldsymbol{n}_{p_j} & \text{if } d(\boldsymbol{q}_{t_i}, \boldsymbol{p}_j) < d_0 \,, \\ 0 & \text{otherwise} \end{cases} \tag{47.9}$$

where K_{att} and K_{rep} are the constants of the forces, d_0 is the influence distance of the obstacles $\boldsymbol{p}_j$, $\boldsymbol{q}_{t_i}$ is the current vehicle configuration and $\boldsymbol{n}_{q_{\text{target}}}$ and $\boldsymbol{n}_{p_j}$ are the unitary vectors that point from $\boldsymbol{q}_{t_i}$ to the target and each obstacle $\boldsymbol{p}_j$. From $\boldsymbol{F}_{\text{tot}}(\boldsymbol{q}_{t_i})$, the control $\boldsymbol{u}_i$ can be obtained with a position or force control [47.53].

This is the classic version, where the potentials only depend on the current vehicle configuration. Complementarily the generalized potentials depend also on the instantaneous robot velocity and vehicle accelerations.

Example

$$\boldsymbol{F}_{\text{rep}}(\boldsymbol{q}_{t_i}) = \begin{cases} K_{\text{rep}} \sum_j \left(\dfrac{a\dot{\boldsymbol{q}}_{t_i}}{[2ad(\boldsymbol{q}_{t_i}, \boldsymbol{p}_j) - \dot{\boldsymbol{q}}_{t_i}^2]} \right) \boldsymbol{n}_{p_j} \cdot \boldsymbol{n}_{\dot{\boldsymbol{q}}_{t_i}} & \text{if } \dot{\boldsymbol{q}}_{t_i} > 0 \,, \\ 0 \quad \text{otherwise} \end{cases} \tag{47.10}$$

where $\dot{\boldsymbol{q}}_{t_i}$ is the current robot velocity, $\boldsymbol{n}_{\dot{\boldsymbol{q}}_{t_i}}$ the unitary vector pointing in the direction of the robot velocity, and a is the maximum vehicle acceleration. This expression arises when the repulsive potential is defined as the inverse of the difference between the estimated time until a collision takes place and the time needed to stop the robot until full backward acceleration. Notice how the repulsive only affect in the direction of motion of the vehicle, as opposed to the classic potential. For a comparison of the classic and generalized versions,

and the relation with the way to compute the motion controls next, see [47.53]. This method is widely used because it is easy to understand and due to its clear mathematic formalism.

47.9.2 Vector Field Histogram

The vector field histogram (VFH) solves the problem in two steps by computing a set of candidate motion directions and then selecting one of them.

Candidate Set of Directions

Firstly the space is divided into sectors from the robot location. The method uses a polar histogram H constructed around the robot, where each component represents the obstacle polar density in the corresponding sector. The function that maps the obstacle distribution in sector k on the corresponding component of the histogram $h^k(\boldsymbol{q}_{t_i})$ is

$$h^k(\boldsymbol{q}_{t_i}) = \int_{\Omega_k} P(\boldsymbol{p})^n . \left(1 - \frac{d(\boldsymbol{q}_{t_i}, \boldsymbol{p})}{d_{\max}}\right)^m \mathrm{d}\boldsymbol{p} \,. \qquad (47.11)$$

The dominion of integration is $\Omega_k = \{\boldsymbol{p} \in \mathcal{W} \backslash \boldsymbol{p} \in k \wedge d(\boldsymbol{q}_{t_i}, \boldsymbol{p}) < d_0\}$. The density $h^k(\boldsymbol{q}_{t_i})$ is proportional to the probability $P(\boldsymbol{r})$ that a point was occupied by an obstacle, and to a factor that increases when the distance to the point decreases.

Typically the resulting histogram has peaks (directions with high density of obstacles) and valleys (directions with low density). The set of candidate directions is the set of adjacent components with density lower than a given threshold, and closest to the component that contains the target direction. This set of components (sectors) is called the selected valley, and represents a set of candidate directions (Fig. 47.10).

Motion Computation

The objective of the next step is to choose a direction of this set. The strategy is to apply three heuristics that depend on the component that contains the target or on the size of the selected valley. The cases are checked in sequence:

1. Case 1: goal sector in the selected valley. Solution: $k_{\mathrm{sol}} = k_{\mathrm{target}}$, where k_{target} is the sector that contains the goal location.
2. Case 2: goal sector not in the selected valley and number of sectors of the valley greater than m. Solution: $k_{\mathrm{sol}} = k_i \pm m/2$, where m is a fixed number of sectors and k_i the sector of the valley closer to the k_{target}.

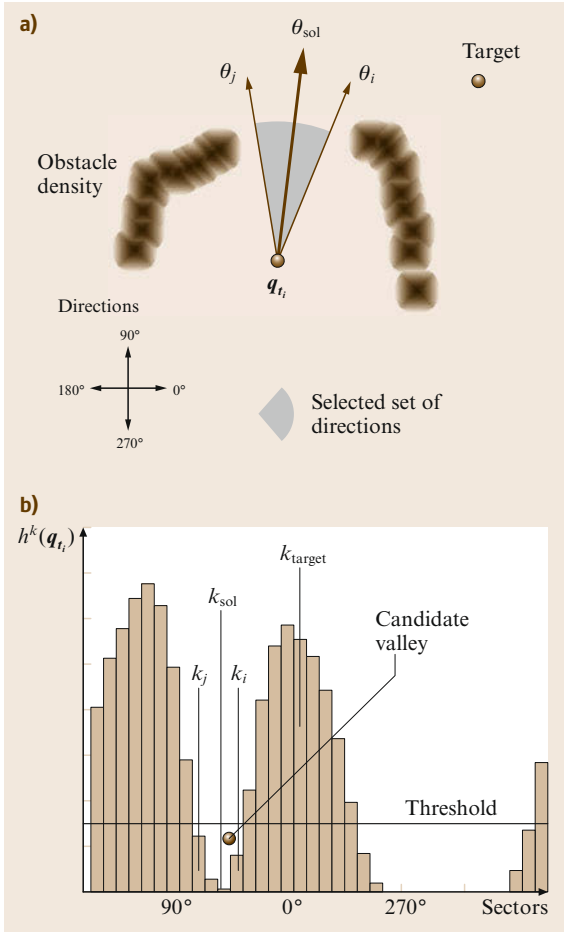


Fig.47.10a,b Computation of the motion direction θ_{sol} with VFH. **(a)** Robot and obstacle occupancy distribution. **(b)** The candidate valley is the set of adjacent components with values lower than the threshold. The navigation case is Case 3, since the sector of the target k_{target} is not in the valley and the number of sectors is lower than a fixed quantity m ($m = 8$, i. e., 45°). Thus the solution is $k_{\mathrm{sol}} = (k_i + k_j)/2$, whose bisector is θ_{sol} in **(a)**. The bisectors of k_i and k_j are θ_i and θ_j, respectively

3. Case 3: goal sector not in the selected valley and number of sectors of the valley lower or equal to m. Solution: $k_{\mathrm{sol}} = (k_i + k_j)/2$, where k_i and k_j are the extreme sectors of the valley.

The result is a component or sector k_{sol}, whose bisector is the direction solution θ_{sol}. The velocity v_{sol} is inversely proportional to the distance to the closest obstacle. The control is $\boldsymbol{u}_i = (v_{\mathrm{sol}}, \theta_{\mathrm{sol}})$.

VFH is a method formulated to work with probability obstacle distributions, and thus, it is well adapted to work with uncertain sensors as ultrasonic sonars.

47.9.3 The Obstacle Restriction Method

The obstacle restriction method (ORM) solves the problem in three steps, where the result of the two first steps is a set of candidate motion directions. The first step is to compute an instantaneous subgoal if necessary. The second one associates a motion constraint to each obstacle and joins them next to compute the set of desirable directions. The last step is an strategy to compute the motion given this set.

Instantaneous Target Selection

This step computes a subgoal when is better to direct the motion towards a given zone of the space (which ameliorates the situation to reach the goal latter), rather than directly towards the goal itself. The subgoals are located in between obstacles or in the edge of an obstacle (Fig. 47.11a). Next, the process is to check with a local algorithm whether the goal can be reached from the robot location. If it is not, then the closest reachable subgoal to the goal is selected. To check if a point can be reached, there is a local algorithm that computes the existence of a local path that joins two locations [47.48].

Let $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ be two locations of the space, R the robot radius, and L a list of obstacle points, where $\boldsymbol{p}_i$ is an obstacle of the list. Let $\mathcal{A}$ and $\mathcal{B}$ be the two semiplanes divided by the line that joins $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$. Then, if for all the points of L, $d(\boldsymbol{p}_j, \boldsymbol{p}_k) > 2R$ (with $\boldsymbol{p}_j \in \mathcal{A}$ and $\boldsymbol{p}_k \in \mathcal{B}$), there is a collision free path that joins both locations. If this condition is not satisfied, then there is no local path (although a global one could exist). The interesting result is when the result is positive, since the guarantee that one point can be reached from the other exists.

The result of the process is the goal or an instantaneous subgoal (from now this location is called the target location). Note that this process is general and can be used as a preprocess step by other methods to instantaneously validate the goal location or to compute an instantaneous subgoal to drive the vehicle.

Candidate Set of Directions

For each obstacle i a set of not desirable directions for motion $S^i_{n\mathrm{D}}$ is computed (motion constraint). This set is the union of two subsets S^i_1 and S^i_2. S^i_1 represents the side of the obstacle not suitable to do the avoidance and S^i_2 is an exclusion area around the obstacle (Fig. 47.11b). The motion constraint for the obstacle is the union of both sets $S^i_{n\mathrm{D}} = S^i_1 \cup S^i_2$. The set of desired directions of motion is the complementary $S_\mathrm{D} = \{[-\pi, \pi] \setminus S_{n\mathrm{D}}\}$, where $S_{n\mathrm{D}} = \cup_i S^i_{n\mathrm{D}}$.

Motion Computation

The final step is to select a direction of motion. There are three cases that depend on the set of desirable directions S_D and on the target direction θ_{target}. The cases are checked in sequence:

1. Case 1: $S_\mathrm{D} \neq \emptyset$ and $\theta_{\mathrm{target}} \in S_\mathrm{D}$. Solution: $\theta_{\mathrm{sol}} = \theta_{\mathrm{target}}$.
2. Case 2: $S_\mathrm{D} \neq \emptyset$ and $\theta_{\mathrm{target}} \notin S_\mathrm{D}$. Solution: $\theta_{\mathrm{sol}} = \theta_{\mathrm{lim}}$, where θ_{lim} is the direction of S_D closest to θ_{target}.
3. Case 3: $S_\mathrm{D} = \emptyset$. Solution: $\theta_{\mathrm{sol}} = (\phi^l_{\mathrm{lim}} + \phi^r_{\mathrm{lim}})/2$, where ϕ^l_{lim} and ϕ^r_{lim} are the directions of $S_{\mathrm{D_r}}$ and $S_{\mathrm{D_l}}$ closer to θ_{target} ($S_{\mathrm{D_r}}$ and $S_{\mathrm{D_l}}$ are the set of desirable

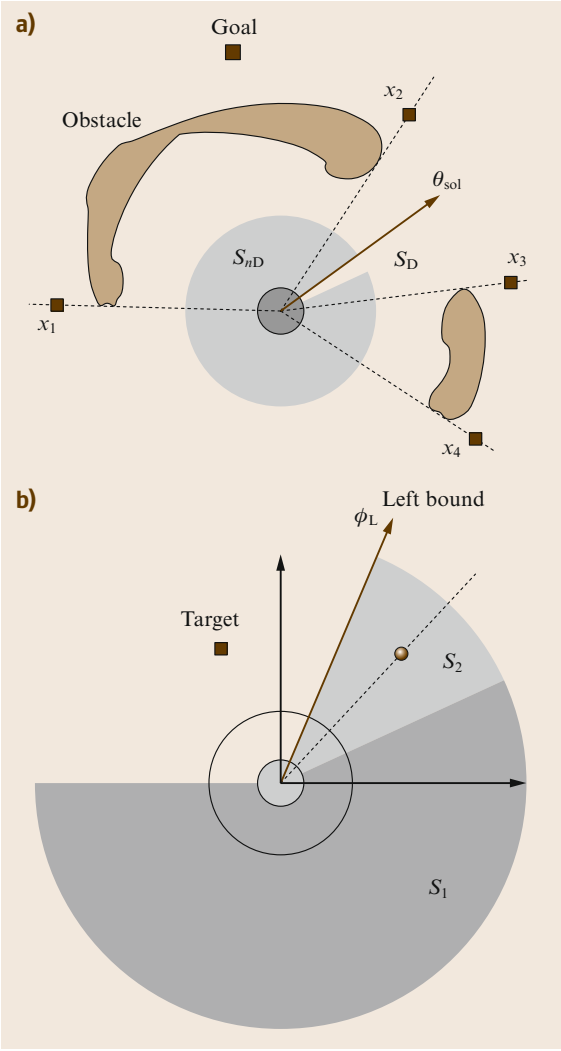


Fig. 47.11 **(a)** Distribution of the subgoals x_i. The selected instantaneous target location is x_2. The set of candidate directions is $S_{n\mathrm{D}}$ and the solution is θ_{sol} (the second case). **(b)** The two sets of not desirable directions S_1 and S_2 for a given obstacle

directions of obstacles on the left and right-hand sides of the target, respectively).

The result is a direction of motion solution θ_{sol}. The velocity v_{sol} is inversely proportional to the distance to the closest obstacle. The control is $\boldsymbol{u}_i = (v_{sol}, \theta_{sol})$. This is a geometric-based method based on cases. The advantage is that it has been demonstrated to address effective motion in confined spaces.

47.9.4 Dynamic Window Approach

The dynamic window approach (DWA) (VIDEO 712) is a method that solves the problem in two steps, by computing as intermediate information a subset of the control space $\mathcal{U}$. For simplicity, we consider a motion control such as translational and rotational velocity (v, w); $\mathcal{U}$ is defined by

$$\begin{aligned}\mathcal{U} = \{\, &(v, w) \in \mathbb{R}^2 \setminus \\ &v \in [-v_{max}, v_{max}] \wedge \\ &w \in [-w_{max}, w_{max}] \,\}\,. \end{aligned} \tag{47.12}$$

Set of Candidate Controls

The candidate set of controls $\mathcal{U}_R$ contains the controls: (i) within the maximum velocities of the vehicle $\mathcal{U}$, (ii) that generate safe trajectories $\mathcal{U}_A$, and (iii) that can be reached within a short period of time given the vehicle accelerations $\mathcal{U}_D$. The set $\mathcal{U}_A$ contains the admissible controls. These controls can be canceled before collision by applying the maximum deceleration (a_v, a_w)

$$\mathcal{U}_A = \left\{(v, w) \in \mathcal{U} | v \le \sqrt{2 d_{obs} a_v} \wedge w \le \sqrt{2 \theta_{obs} a_w}\right\}, \tag{47.13}$$

where d_{obst} and θ_{obst} are the distance to the obstacle and the orientation of the tangent to the trajectory in the obstacle. The set $\mathcal{U}_D$ contains the controls reachable in a short period

$$\begin{aligned}\mathcal{U}_D = \{\, &(v, w) \in \mathcal{U} \setminus \\ &v \in [v_o - a_v T, v_o + a_v T] \wedge \\ &w \in [w_o - a_w T, w_o + a_w T]\}\,, \end{aligned} \tag{47.14}$$

where $\dot{\boldsymbol{q}}_{t_i} = (v_o, w_o)$ is the current velocity.

The resulting subset of controls is (Fig. 47.12)

$$\mathcal{U}_R = \mathcal{U} \cap \mathcal{U}_A \cap \mathcal{U}_D\,. \tag{47.15}$$

Motion Computation

The next step is to select one control $\boldsymbol{u}_i \in \mathcal{U}_R$. The problem is set out as the maximization of an objective

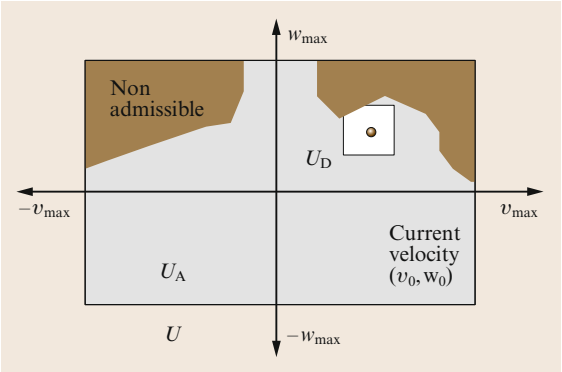


Fig. 47.12 Subset of controls $\mathcal{U}_R = \mathcal{U} \cap \mathcal{U}_A \cap \mathcal{U}_D$, where $\mathcal{U}$ contains the controls within the maximum velocities, $\mathcal{U}_A$ the admissible controls, and $\mathcal{U}_D$ the controls reachable in a short period of time

function

$$\begin{aligned}G(\boldsymbol{u}) = &\alpha_1 \cdot \text{Goal}(\boldsymbol{u}) + \alpha_2 \cdot \text{Clearance}(\boldsymbol{u}) \\ &+ \alpha_3 \cdot \text{Velocity}(\boldsymbol{u})\,. \end{aligned} \tag{47.16}$$

This function is a compromise among Goal($\boldsymbol{u}$) that favors velocities that offer progress to the goal, Clerance($\boldsymbol{u}$) that favors velocities far from the obstacles, and Velocity($\boldsymbol{u}$) that favors high speeds. The solution is the control $\boldsymbol{u}_i$ that maximizes this function.

DWA solves the problem in the control space using information of the vehicle dynamics, thus the method is well adapted to work on vehicles with slow dynamic capabilities or that work at high speeds.

47.9.5 Velocity Obstacles

The method of velocity obstacles (VO) solves the problem in two steps, by computing as intermediate information a subset of the $\mathcal{U}$. The framework is equal to that of DWA. The difference is that the computation of the set of safe trajectories $\mathcal{U}_A$ takes into account the velocity of the obstacles, which is described next.

Let $\boldsymbol{v}_i$ be the velocity of obstacle i (which became enlarged with the vehicle radius occupying an area B_i) and $\boldsymbol{u}$ a given vehicle control. The set of colliding relative velocities is called *collision cone*

$$CC_i = \left\{\boldsymbol{u}_i | \lambda_i \bigcap B_i \neq \emptyset\right\}\,, \tag{47.17}$$

where λ_i is the direction of the unitary vector $\boldsymbol{u}_i = \boldsymbol{u}_i - \boldsymbol{v}_i$. The *velocity obstacle* is this set in a common absolute system of reference

$$VO_i = CC_i \oplus \boldsymbol{v}_i\,, \tag{47.18}$$

where $\oplus$ is the Minkowski vector sum. The set of unsafe trajectories is the union of the velocity obstacles for

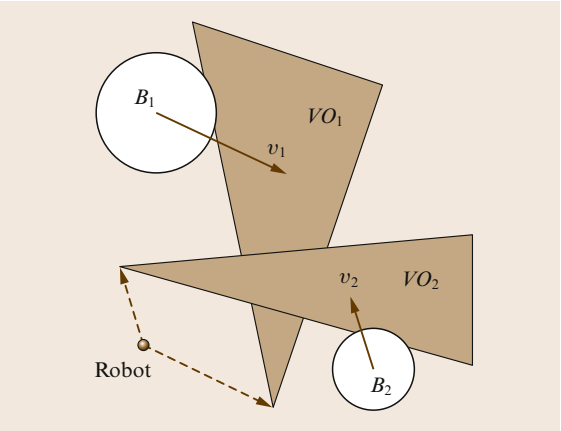


Fig. 47.13 Subset of not safe controls $\bar{\mathcal{U}}_A = VO_1 \cup VO_2$. A control vector out of this velocity generates a collision-free motion with the moving obstacles

each moving obstacle $\bar{\mathcal{U}}_A = \cup_i VO_i$ (Fig. 47.13). The advantage of this method is that takes into account the velocity of the obstacles; thus it is well suited in dynamic scenarios.

47.9.6 Nearness Diagram Navigation

This method is more a methodology to design obstacle avoidance methods rather than a method in itself. Nearness diagram navigation (ND) is an obstacle avoidance method obtained with a geometric implementation following this methodology. The idea behind is to employ a divide and conquer strategy based on situations to simplify the obstacle avoidance problem following the situated-activity paradigm [47.54]. First, there is a set of situations that represent all the cases among robot locations, obstacles, and target locations. Also, for each case there is a motion law associated with it. During the execution phase, at time t_i one situation is identified and the corresponding law is used to compute the motion.

Situations

The situations are represented in a binary decision tree. The selection of a situation depends on the obstacles $\mathcal{O}(\boldsymbol{q}_{t_i})$, on the robot location $\boldsymbol{q}_{t_i}$ and target $\boldsymbol{q}_{\text{target}}$. The criteria are based on high-level entities, like a security distance around the robot bounds and a motion area (that identifies suitable areas of motion). For example, one criterion is if there are obstacles within the security zone. Another is if the motion area is large or narrow. The result is only one situation, since by definition and representation (binary decision tree) the set of situations is complete and exclusive (Fig. 47.14).

Actions

Associated to any situation there is an action that computes the motion to adapt the behavior to the case

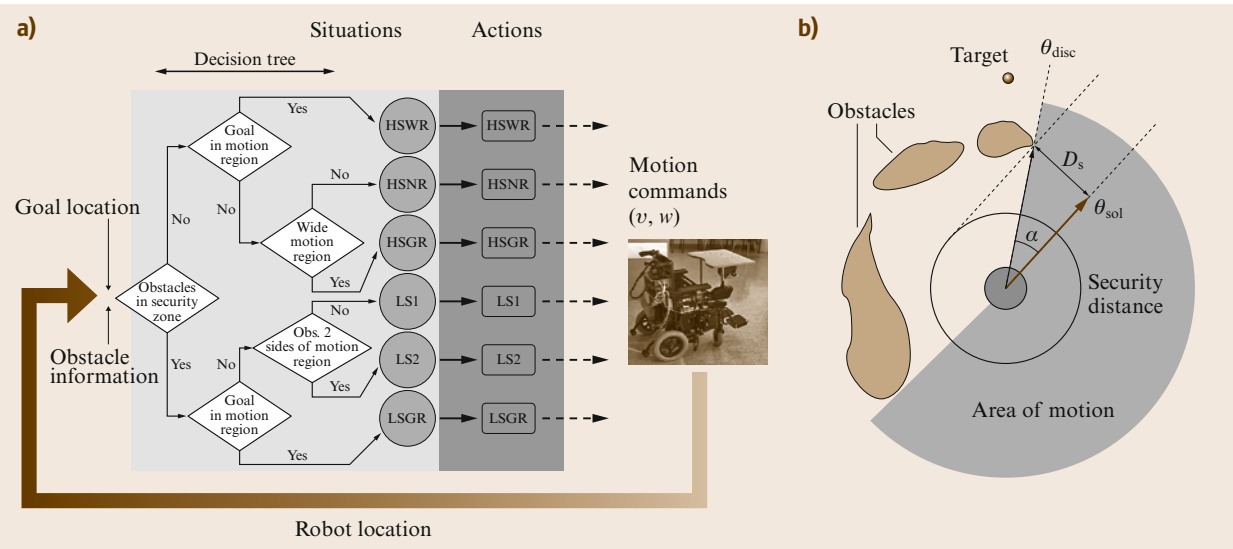


Fig. 47.14 **(a)** Design diagram of the method. Given the obstacle information and the target location, one situation is identified given some criteria. Next the associated action is executed computing the motion. **(b)** Example of computation of the solution of the ND (geometric implementation). The first step is to identify the situation. There are no obstacles closer than the security distance D_s. Next $\boldsymbol{q}_{\text{target}}$ is not within the motion area. Third, the motion area is wide. With these three criteria we identify the current situation, HSWR. In this situation, the associated action computes the control as $\boldsymbol{u}_i = (v_{sol}, \theta_{sol})$, where v_{sol} is the maximum velocity and θ_{sol} is computed as a deviation α from the limit direction θ_{disc} of the motion area closer to the target direction

represented by each situation. At a high level the actions describe the behavior desired in each situation. For example, one situation is when there are no obstacles within a security area and the goal is within the motion area (HSGR). The solution is to *move towards the goal*. Another situation is when there are no obstacles within a security area, and the goal is not within the motion area but is wide (HSWR). The solution is to *move towards the limit of the motion area but clearing the security area of obstacles*.

The interesting aspect of this method is that it employs a *divide and conquer* strategy to solve the navigation. As a consequence, it simplifies the difficulty of the problem. The first advantage is that the methodology is described at a symbolic level and, thus, there are many ways to implement the method. Second, a geometric implementation of this methodology (ND method) has been demonstrated to address difficult navigation cases, which translate in to achieve safe navigation in dense, complex, and difficult scenarios.

47.10 Robot Shape, Kinematics, and Dynamics in Obstacle Avoidance

There are three aspects of the vehicle that have to be taken into account during the obstacle avoidance process: shape, kinematics, and dynamics. The shape with the kinematics is a geometric problem that involves the representation of the vehicle configurations in collision, given the admissible trajectories $\mathcal{Q}_{t,\infty}$. The dynamics involves the accelerations and temporal considerations, and is derived in two aspects: (i) choosing a control reachable in a short period of time T given the current velocity $\dot{\boldsymbol{q}}_{t_i}$ and the maximum accelerations. (ii) Taking the braking distance into account, so that after a control execution, the vehicle can always stop before collision by applying the maximum deceleration (improving safety).

The problem of shape, kinematics, and dynamics in obstacle avoidance has been taken into account from three different points of view:

1. Designing a way to incorporate the constraints within the method (Sect. 47.9.4)
2. Developing techniques that abstract the vehicle aspects from the application of the method [47.55–58]
3. By techniques that break down the problem into subproblems and incorporate the aspects in sequence [47.59–61] after method usage.

47.10.1 Techniques that Abstract Vehicle Aspects

These techniques are based on constructing an abstraction layer between the aspects of the vehicle and the obstacle avoidance method, in such a way that when the method is applied its solutions already take into account these aspects [47.55–58]. We consider here vehicles whose elementary paths obtained under execution of constant controls can be approximated by circular arcs (for example, a differential-drive robot, syncro-drive, or a tri-cycle). To simplify, a control is translational and rotational velocity $\boldsymbol{u} = (v, w)$. The set of reachable controls $\mathcal{U}_{\mathrm{A}}$ given the current velocity $\dot{\boldsymbol{q}}_{t_i} = (v_0, w_0)$ and the maximum accelerations (a_v, a_w) is obtained by (47.14).

Abstraction Construction

For these vehicles, the configuration space CS is three-dimensional. The idea is to construct, centered in the robot at each time t_i, the manifold of the configuration space $ARM(\boldsymbol{q}_{t_i}) \equiv ARM$ defined by elementary circular paths. The function that defines the manifold is

$$\theta = f(x,y) = \begin{cases} \arctan 2\left(x, \dfrac{x^2-y^2}{2y}\right) & \text{if } y \geq 0\,, \\ -\arctan 2\left(x, -\dfrac{x^2-y^2}{2y}\right) & \text{otherwise}\,. \end{cases} \tag{47.19}$$

It is easy to see that function f is differentiable in $\mathbb{R}^2\backslash(0,0)$. Thus $(x, y, f(x,y))$ defines a two-dimensional manifold in $\mathbb{R}^2 \times S^1$ when $(x,y) \in \mathbb{R}^2\backslash(0,0)$. This manifold ARM contains all the configurations that can be reached at each step of the obstacle avoidance.

Next, in the ARM one computes the exact region of the configurations in collision CO_{ARM} given any shape of the robot (i. e., obstacle representation in the manifold). Given an obstacle point (x_p, y_p) and a point of the robot bounds (x_r, y_r), a point (x_s, y_s) of the CO_{ARM} boundary is obtained by

$$\begin{aligned} x_s &= (x_f + x_i)\cdot a\,, \\ y_s &= (y_f - y_i)\cdot a\,, \end{aligned} \tag{47.20}$$

with

$$\begin{aligned} a = \Big\{ &\left[\left(y_f^2 - y_i^2\right) + \left(x_f^2 - x_i^2\right)\right]\left[(y_f - y_i)^2 + (x_f - x_i)^2\right]\Big\} \\ &\times \Big[(y_f - y_i)^4 + 2\left(x_f^2 + x_i^2\right)(y_f - y_i)^2 \\ &+ \left(x_f^2 - x_i^2\right)^2\Big]^{-1}. \end{aligned}$$

This result is used to map the robot bounds for all obstacles in the manifold, computing the exact shape of the CO_{ARM}. Next, one compute the nonadmissible configurations CNA_{ARM} in the manifold ARM, which correspond to configurations that once reached at a given

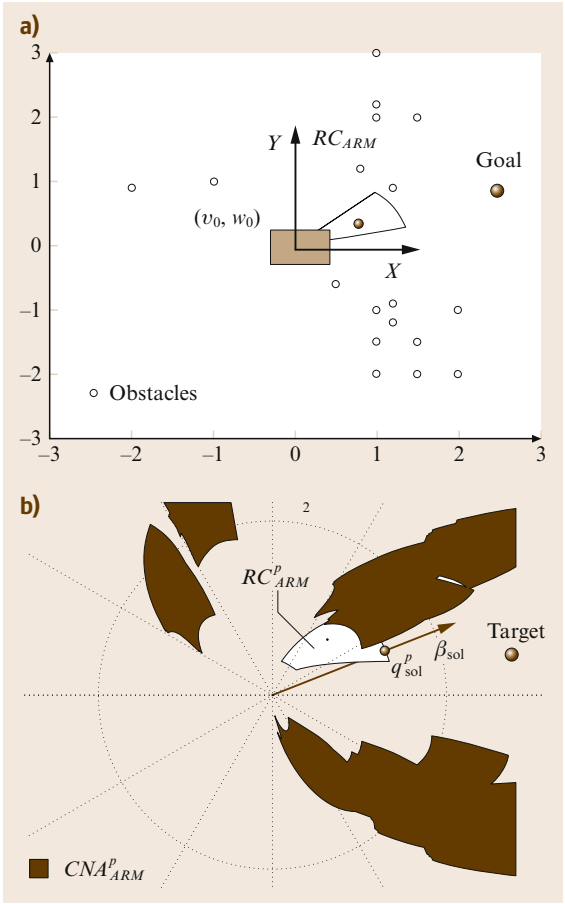


Fig. 47.15 **(a)** Robot location, obstacle information and goal location. **(b)** The abstraction layer in ARM^p. In this representation the obstacles are the nonadmissible region CNA^p_{ARM} in ARM and that the motion is omnidirectional (applicability conditions of many obstacle avoidance methods). The method is applied to obtain the most promising direction β_{sol}, used to obtain the configuration solution $\boldsymbol{q}^p_{sol}$ in the set of reachable configurations RC^p_{ARM}. Finally, the solution is the control $\boldsymbol{u}_{sol}$ that reaches this configuration at time T. This control complies with the kinematics and dynamics and takes into account the exact vehicle shape

velocity in time T, the vehicle cannot be stopped by applying the maximum deceleration without collision (i. e., there is not enough braking distance). These regions CNA_{ARM} are the CO_{ARM} *enlarged* by a magnitude that depends on the maximum vehicle accelerations. The set of configurations RC_{ARM} reachable by controls in a short period of time $\mathcal{U}_A$ is also computed in the ARM. Finally, a change of coordinates is applied to ARM leading to ARM^p. Its effect is that the elementary circular paths become straight segments in the manifold. As a consequence, the problem now is to move an omnidirectional point in a bidimensional space free of any constraint.

Method Application

The final step is to apply the avoidance method on the ARM^p to avoid the CNA^p_{ARM} regions. The method solution β_{sol} is the most promising direction in ARM^p. This direction is used to select and admissible location $\boldsymbol{q}^p_{sol} \notin CNA^p_{ARM}$ in the set of dynamic reachable configurations RC^p_{ARM}. Finally, the control $\boldsymbol{u}_{sol}$ that leads to this location at time T is selected. By construction, this control is kinematically and dynamically admissible, avoids collisions with the exact vehicle shape, and takes into account the braking distance (Fig. 47.15).

Note how with these techniques, the three-tridimensional obstacle avoidance problem is converted into the simple problem of moving a point in a two-dimensional space without kinematic and dynamic restrictions (applicability conditions of many existing methods). Thus, many existing methods can be applied in this abstraction, and as a consequence, the solutions take the vehicle constraints into account.

47.10.2 Techniques of Decomposition in Subproblems

These techniques address the obstacle avoidance problem taking into account the vehicle constraints by breaking down this problem into subproblems:

1. Obstacle avoidance
2. Kinematics and dynamics
3. The shape.

Each subproblem is dealt with in sequence (Fig. 47.16).

Obstacle Avoidance

First the obstacle avoidance method is used by assuming a circular and omnidirectional vehicle. The solution is the most promising motion direction and the velocity $\boldsymbol{u}_1 = (v_1, \theta_1)$ to direct the vehicle towards the target.

Kinematics and Dynamics

Second, this control is converted into a control that complies with the kinematics and dynamics, which tends to align the vehicle with the instantaneous motion direction θ_1 of $\boldsymbol{u}_1$. For example, [47.60, 61] modify the output of the obstacle avoidance method by a feedback action that aligns the vehicle with the direction solution in a least squares fashion. Moreover, [47.59] use a dynamic controller of the vehicle. This controller models the behavior of the vehicle as it would be pulled by a virtual force, and computes the mo-

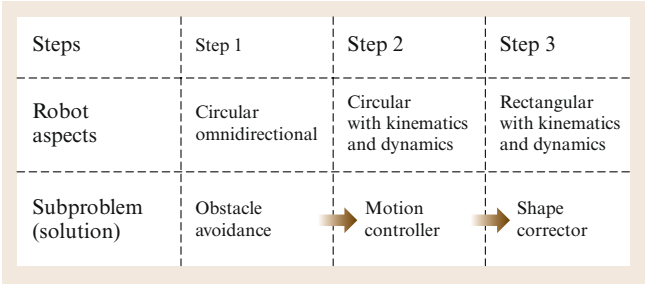


Fig. 47.16 Decomposition of the obstacle avoidance problem into subproblems that are addressed in steps, successively incorporating the vehicle aspects

tion that would result after applying this force during a short period of time. The motion generated is the new control. In order to use the controller, the previous control $\boldsymbol{u}_1$ is converted into an instantaneous force $\boldsymbol{F} = (\theta_1, F_{\max}(v_1/v_{\max}))$ input of the controller, which computes a control $\boldsymbol{u}_2$ that complies with the kinematic and dynamics.

Shape

The final step is to assure that the control $\boldsymbol{u}_2$ obtained in the previous step avoids collisions with the exact shape of the vehicle. To do this, a shape corrector is used to check collisions by dynamic simulation of the control. If there is collision, there is a search in the set of reachable controls (47.14) until one of them is collision free. The result of this process is a motion control $\boldsymbol{u}_i$ that guarantees obstacle avoidance and complies with the kinematics and dynamics of the vehicle.

47.11 Integration Planning – Reaction

In this section we show how the obstacle avoidance methods are integrated in real systems. On the one hand, the obstacle avoidance methods are local techniques to address the motion problem. Thus, they are doomed to fall in local minima that translate in trap situations or cyclic motions. This reveals the necessity of a more global reasoning. On the other hand, motion planning techniques compute a geometric path free of collisions, which guarantees global convergence. However, when the scenarios are unknown and evolve, these techniques fail, since the precomputed paths will almost surely collide with obstacles. It seems clear that one key aspect to build a motion system is to combine the best of both worlds: the global knowledge given by motion planning and the reactivity of the obstacle avoidance methods.

The most extended ways to specify the interaction between the deliberation and reaction are: (i) to precompute a path to the target deformed in execution as a function of the changes in the scenario obtained from the sensor information (systems of path deformation), for example, [47.62–67]. (ii) To use a planner at a high frequency with a tactical role, leaving the degree of execution to the reactor [47.68–73].

47.11.1 Systems of Path Deformation

Elastic bands is a method that initially assumes the existence of a geometric path to the target location (computed by a planner). The path is assimilated with a band, subjected to two types of forces: an inner contraction force and an external force. The inner force simulates the tension of a strip and maintains the stress. The external force is exerted by the obstacles and moves the band far from them. During the execution, the new obstacles produce forces that remove the band far from them, guaranteeing their avoidance. These methods are described in Chap. 37.

An extension of the method of path deformation was proposed in [47.64] for nonholonomic systems (VIDEO 80). Even though the objective is the same as for mobile robots without kinematic constraints, avoiding obstacle while following a trajectory, the concepts are completely different. The trajectory Γ of a nonholonomic system is completely defined by an initial configuration $\Gamma(0)$ and the value of the input function $\boldsymbol{u} \in C^1(I, \mathbb{R}^m)$, where I is an interval. The trajectory deformation method for nonholonomic systems is thus based on the perturbation of the input function of the current trajectory in order to achieve three objectives:

1. Keeping the nonholonomic constraints satisfied
2. Getting away from obstacles detected online by onboard sensors
3. Keeping unchanged the initial and last configurations of the trajectory after deformation.

Perturbing the input function of Γ, by a vector-valued input perturbation $\boldsymbol{v} \in C^1(I, \mathbb{R}^m)$, yields a trajectory deformation $\eta \in C^1(I, \mathbb{R}^n)$

$$\boldsymbol{u} \leftarrow \boldsymbol{u} + \tau \boldsymbol{v} \Rightarrow \Gamma \leftarrow \Gamma + \tau \eta \,,$$

where τ is a positive, asymptotically small real number. As an approximation of order 1, the relation between $\boldsymbol{u}$ and η is given by the linearized system, which we do not express here.

Potential Field of Obstacles

Obstacles are taken into account by defining a potential field over the configuration space, which increases when

the robot gets closer to obstacles. This potential field is lifted into the space of trajectories by integration along the trajectory of the configuration potential value.

Discretization of the Input Space

The space $C^1(I, \mathbb{R}^m)$ of input perturbations is an infinite-dimensional vector space. The choice of an input perturbation is restricted to a finite-dimensional subspace spanned by p arbitrary test functions, $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p$ where p is a positive integer. An input perturbation: $\boldsymbol{u} = \sum_{i=1}^{p} \lambda_i \boldsymbol{e}_i$ is thus defined by a vector $\lambda \in \mathbb{R}^p$. The variation of the trajectory potential is linear with respect to λ.

Boundary Conditions

The boundary conditions consisting in applying a trajectory perturbation equal to zero at both ends of interval I is linear with respect to λ. It is, therefore, easy to find a vector λ making the potential decrease and satisfying the boundary conditions.

Nonholonomic Constraint Deviation

The approximation of order 1 in the relation between the input perturbation and trajectory deformation induces a side effect: after a few iterations, the nonholonomic constraints are no longer satisfied. This side effect is corrected by considering an augmented system with n control vector fields $f_1, \ldots, f_n$ spanning $\mathbb{R}^n$ for each configuration and by keeping the input components u_{m+1} to u_n along the additional vector fields as close as possible to 0. Figure 47.17 shows an example of the trajectory deformation algorithm applied to a nonholonomic system.

47.11.2 Systems of Tactical Planning

The systems of tactical planning recompute at a high frequency a path to the target location, and use the main course to advise the obstacle avoidance module. The design of these motion systems involves at least the synthesis of three functionalities. The construction of a model, the deliberative planning, and the obstacle avoidance. The modeler constructs a representation base for deliberation and memory for the reactive behavior. The planner generates global plans used tactically to guide the obstacle avoidance module, which generates the local motion. Next we give a perspective of the three functionalities and three possible tools to implement them [47.73].

Model Builder Module

Construction of a model of the environment (to increase the spatial domain of the planning and used as local memory for obstacle avoidance). One possibility is to use a binary occupancy grid that is updated whenever a new sensory measurement is available, and to employ a scan matching technique [47.74, 75] to improve the vehicle odometry before integrating any new measure in the grid.

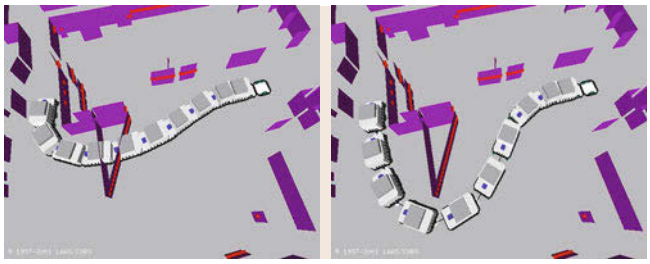


Fig. 47.17 A differentially-driven mobile robot towing a trailer applying the trajectory deformation algorithm to avoid obstacles detected online

Planner Module

Extraction of the connectivity of the free space (used to avoid the cyclical motions and trap situations). A good choice is a dynamic navigation function such as the D* [47.76, 77]. The idea behind the planner is to focus the search locally in the areas where the changes in the scenario structure have occurred and affect the computation of the path. The planner avoids the local minima and is computationally very efficient for real-time implementations.

Obstacle Avoidance Module

Computation of the collision-free motion. Any of the methods described in this chapter could be used. One possibility is the ND method (Sect. 47.9.6), since it has been demonstrated to be very effi-

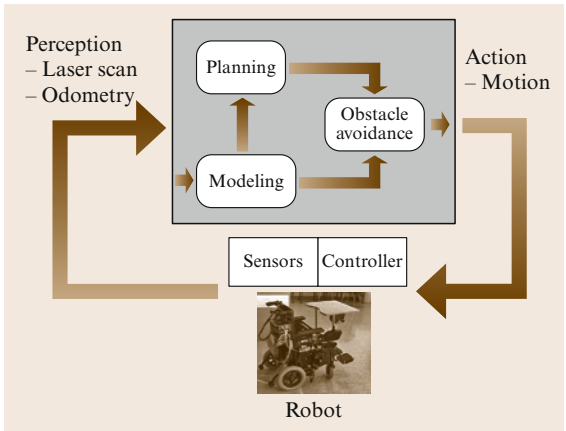


Fig. 47.18 Overview of the system of motion

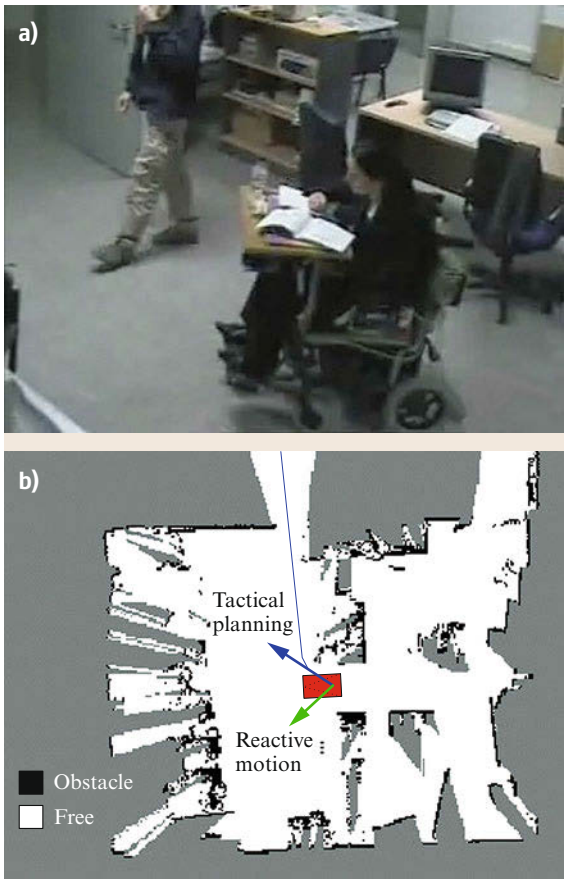


Fig. 47.19 **(a)** Snapshot of an experiment carried out with the motion system described working on a wheelchair robot equipped with a planar range laser sensor. **(b)** Information of the motion system at a given time: current accumulated map of the scenario, path computed with the planner and tactical direction of motion, and solution of the reactive obstacle avoidance method ◄

cient and robust in environments with little space to manoeuvre.

Globally the system works as follows (Fig. 47.18): given a laser scan and the odometry of the vehicle, the model builder incorporates this information into the existing model. Next, the information of the changes in obstacle and free space in the model is used by the planner module to compute the course to follow to reach the goal. Finally, the avoidance module uses the information of the obstacles contained in the grid and information of this tactical planner to generate the motion (to drive the vehicle free of collisions towards the goal). The motion is executed by the vehicle controller and the process restarts with a new sensorial measurement. It is important to stress that the three modules should work synchronously within the perception – action cycle for consistency reasons. The advantage of these systems is that the synergy of the modules allows to avoid trap situations and the cyclic motions (limitations associated to the local nature of the avoidance methods). Figure 47.19 shows a snapshot of an experiment carried out with this motion system.

47.12 Conclusions, Future Directions, and Further Reading

The algorithmic tools presented in this chapter show that motion planning and obstacle avoidance research techniques have reached a level of maturity that allow their transfer onto real platforms (VIDEO 710, VIDEO 714). Today, several indoor mobile robots use obstacle avoidance techniques on a daily basis to guide visitors in museums. Outdoor applications still require some developments in perception and modeling. For these applications, 3-D sensing capabilities are necessary to model the environment but also to detect obstacles. As an example, several car manufacturer are working on parallel parking assistance. The difficult point of this application is to build a model of a parking spot from 3-D data in a great variety of environments.

The main challenge of autonomous motion in mobile robots currently consists in integrating techniques from different robotics research domains in order to plan and execute motions for very complex systems, like humanoid robots, for instance. Integration in the sense that different software components need to work together on a machine but also in a scientific meaning: the classical motion planning formulation with configuration variables that locate the robot with respect to a global reference frame does not fit partially known environments with imprecise maps. Robotic tasks like motion need to be specified with respect to landmarks of the environment. For instance, grasping an object is by definition a motion specified with respect to the position of the object. Very little has been done to design a general framework in this direction.

For useful complementary reading on motion planning and obstacle avoidance for mobile robots see [47.5, 78–80].

Video-References

VIDEO 80 Sensor-based trajectory deformation and docking for nonholonomic mobile robots
available from http://handbookofrobotics.org/view-chapter/47/videodetails/80

VIDEO 707 Autonomous robotic smart wheelchair navigation in an urban environment
available from http://handbookofrobotics.org/view-chapter/47/videodetails/707

VIDEO 708 Sena wheelchair: Autonomous navigation at University of Malaga (2007)
available from http://handbookofrobotics.org/view-chapter/47/videodetails/708

VIDEO 709 Robotic wheelchair: Autonomous navigation with Google Glass
available from http://handbookofrobotics.org/view-chapter/47/videodetails/709

VIDEO 710 A ride in the google self driving car
available from http://handbookofrobotics.org/view-chapter/47/videodetails/710

VIDEO 711 Mobile robot navigation system in outdoor pedestrian environment
available from http://handbookofrobotics.org/view-chapter/47/videodetails/711

VIDEO 712 Mobile robot autonomous navigation in Gracia district, Barcelona
available from http://handbookofrobotics.org/view-chapter/47/videodetails/712

VIDEO 713 Autonomous navigation of a mobile vehicle
available from http://handbookofrobotics.org/view-chapter/47/videodetails/713

VIDEO 714 Autonomous robot cars drive DARPA Urban challenge
available from http://handbookofrobotics.org/view-chapter/47/videodetails/714

References

47.1 N.J. Nilson: A mobile automaton: An application of artificial intelligence techniques, 1st Int. Jt. Conf. Artif. Intell. (1969) pp. 509–520

47.2 A. Thompson: The navigation system of the JPL robot, 5th Int. Jt. Conf. Artif. Intell., Cambridge (1977) pp. 749–757

47.3 G. Giralt, R. Sobek, R. Chatila: A multi-level planning and navigation system for a mobile robot: A 1st approach to Hilare, 6th Int. Jt. Conf. Artif. Intell. (1979) pp. 335–337

47.4 T. Lozano-Pérez: Spatial planning: A configuration space approach, IEEE Trans. Comput. **32**(2), 108–120 (1983)

47.5 J.C. Latombe: *Robot Motion Planning* (Kluwer, Dordrecht 1991)

47.6 J.-P. Laumond: Feasible trajectories for mobile robots with kinematic and environment constraints. In: *Intelligent Autonomous Systems*, ed. by F.C.A. Groen (Hertzberger, Amsterdam 1987) pp. 346–354

47.7 Z. Li, J.F. Canny: *Nonholonomic Motion Planning* (Kluwer, Dordrecht 1992)

47.8 M. Likhachev, D. Ferguson: Planning long dynamically-feasible maneuvers for autonomous vehicles, Int. J. Robotics Res. **28**(8), 933–945 (2009)

47.9 H. Sussmann: Lie brackets, real analyticity and geometric control. In: *Differential Geometric Control Theory*, Progress in Mathematics, Vol. 27, ed. by R. Brockett, R. Millman, H. Sussmann (Birkhauser, New York 1982) pp. 1–116

47.10 H.J. Sussmann, V. Jurdjevic: Controllability of nonlinear systems, J. Differ. Equ. **12**, 95–116 (1972)

47.11 J.-P. Laumond: Singularities and topological aspects in nonholonomic motion planning. In: *Nonholonomic motion Planning*, ed. by Z. Li, J.F. Canny (Kluwer, Boston 1992) pp. 149–199

47.12 J.-P. Laumond, J.J. Risler: Nonholonomic systems: Controllability and complexity, Theor. Comput. Sci. **157**, 101–114 (1996)

47.13 J.-P. Laumond, P. Jacobs, M. Taix, R. Murray: A motion planner for nonholonomic mobile robot, IEEE Trans. Robotics Autom. **10**(5), 577–593 (1994)

47.14 P. Cheng, S.M. LaValle: Reducing metric sensitivity in randomized trajectory design, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 43–48

47.15 J.A. Reeds, R.A. Shepp: Optimal paths for a car that goes both forward and backwards, Pac. J. Math. **145**(2), 367–393 (1990)

47.16 P. Souères, J.-P. Laumond: Shortest path synthesis for a car-like robot, IEEE Trans. Autom. Control **41**(5), 672–688 (1996)

47.17 D. Balkcom, M. Mason: Time optimal trajectories for bounded velocity differential drive vehicles, Int. J. Robotics Res. **21**(3), 199–218 (2002)

47.18 D. Tilbury, R. Murray, S. Sastry: Trajectory generation for the *n*-trailer problem using Goursat normal form, IEEE Trans. Autom. Control **40**(5), 802–819 (1995)

47.19 S. Sekhavat, J.-P. Laumond: Topological property for collision-free nonholonomic motion planning: The case of sinusoidal inputs for chained form systems, IEEE Trans. Robotics Autom. **14**(5), 671–680 (1998)

47.20 M. Fliess, J. Lévine, P. Martin, P. Rouchon: Flatness and defect of non-linear systems: Introductory theory and examples, Int. J. Control **61**(6), 1327–1361 (1995)

47.21 P. Rouchon, M. Fliess, J. Lévine, P. Martin: Flatness and motion planning: The car with *n* trailers, Eur. Control Conf. (1993) pp. 1518–1522

47.22 F. Lamiraux, J.-P. Laumond: Flatness and small-time controllability of multibody mobile robots:

Application to motion planning, IEEE Trans. Autom. Control **45**(10), 1878–1881 (2000)
47.23 P. Rouchon: Necessary condition and genericity of dynamic feedback linearization, J. Math. Syst. Estim. Control **4**(2), 1–14 (1994)
47.24 P. Rouchon, M. Fliess, J. Lévine, P. Martin: Flatness, motion planning and trailer systems, IEEE Int. Conf. Decis. Control (1993) pp. 2700–2705
47.25 S. Sekhavat, J. Hermosillo: Cycab bi-steerable cars: A new family of differentially flat systems, Adv. Robotics **16**(5), 445–462 (2002)
47.26 J. Barraquand, J.C. Latombe: Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles, Algorithmica **10**, 121–155 (1993)
47.27 A. Divelbiss, T. Wen: A path space approach to nonholonomic motion planning in the presence of obstacles, IEEE Trans. Robotics Autom. **13**(3), 443–451 (1997)
47.28 S. LaValle, J. Kuffner: Randomized kinodynamic planning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 473–479
47.29 F. Lamiraux, E. Ferré, E. Vallée: Kinodynamic motion planning: Connecting exploration trees using trajectory optimization methods, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 3987–3992
47.30 V. Lumelsky, A. Stepanov: Path planning strategies for a point mobile automation moving admist unknown obstacles of arbitrary shape, Algorithmica **2**, 403–430 (1987)
47.31 R. Chatila: Path planning and environmental learning in a mobile robot system, Eur. Conf. Artif. Intell. (1982)
47.32 L. Gouzenes: Strategies for solving collision-free trajectories problems for mobile robots and manipulator robots, Int. J. Robotics Res. **3**(4), 51–65 (1984)
47.33 R. Chattergy: Some heuristics for the navigation of a robot, Int. J. Robotics Res. **4**(1), 59–66 (1985)
47.34 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robotics Res. **5**, 90–98 (1986)
47.35 B.H. Krogh, C.E. Thorpe: Integrated path planning and dynamic steering control for autonomous vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1986) pp. 1664–1669
47.36 J. Borenstein, Y. Koren: Real-time obstacle avoidance for fast mobile robots, IEEE Trans. Syst. Man Cybern. **19**(5), 1179–1187 (1989)
47.37 K. Azarm, G. Schmidt: Integrated mobile robot motion planning and execution in changing indoor environments, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 298–305
47.38 A. Masoud, S. Masoud, M. Bayoumi: Robot navigation using a pressure generated mechanical stress field, the biharmonical potential approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 124–129
47.39 L. Singh, H. Stephanou, J. Wen: Real-time robot motion control with circulatory fields, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 2737–2742
47.40 J. Borenstein, Y. Koren: The vector field histogram-fast obstacle avoidance for mbile robots, IEEE Trans. Robotics Autom. **7**, 278–288 (1991)
47.41 J. Minguez: The obstacle restriction method (ORM): Obstacle avoidance in difficult scenarios, IEEE/RSJ Int. Conf. Intell. Robot Syst. (IROS) (2005)
47.42 D. Vikerimark, J. Minguez: Reactive obstacle avoidance for mobile robots that operate in confined 3-D workspaces, IEEE Mediterr. Electrotech. Conf. (2006)
47.43 W. Feiten, R. Bauer, G. Lawitzky: Robust obstacle avoidance in unknown and cramped environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 2412–2417
47.44 D. Fox, W. Burgard, S. Thrun: The dynamic window approach to collision avoidance, IEEE Robotics Autom. Mag. **4**(1), 23–33 (1997)
47.45 P. Fiorini, Z. Shiller: Motion planning in dynamic environments using velocity obstacles, Int. J. Robotics Res. **17**(7), 760–772 (1998)
47.46 F. Large, C. Laugier, Z. Shiller: Navigation among moving obstacles using the NLVO: Principles and applications to intelligent vehicles, Auton. Robots **19**(2), 159–171 (2005)
47.47 R. Simmons: The curvature-velocity method for local obstacle avoidance, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 3375–3382
47.48 J. Minguez, L. Montano: Nearness diagram (ND) navigation: Collision avoidance in troublesome scenarios, IEEE Trans. Robotics Autom. **20**(1), 45–59 (2004)
47.49 J. Minguez, J. Osuna, L. Montano: A divide and conquer strategy to achieve reactive collision avoidance in troublesome scenarios, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)
47.50 J.W. Durham, F. Bullo: Smooth nearness-diagram navigation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008)
47.51 C.-C. Yu, W.-C. Chen, C.-C. Wang: Self-tuning nearness diagram navigation, Int. Conf. Serv. Interact. Robotics (SIRCon) (2009)
47.52 M. Mujahad, D. Fischer, B. Mertsching, H. Jaddu: Closest gap based (CG) reactive obstacle avoidance Navigation for highly cluttered environments, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 1805–1812
47.53 R.B. Tilove: Local obstacle avoidance for mobile robots based on the method of artificial potentials, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1990) pp. 566–571
47.54 R.C. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1999)
47.55 J. Minguez, L. Montano: Abstracting any vehicle shape and the kinematics and dynamic constraints from reactive collision avoidance methods, Eur. Conf. Mob. Robots (2007)
47.56 J. Minguez, L. Montano, J. Santos-Victor: Abstracting the vehicle shape and kinematic constraints from the obstacle avoidance methods, Auton. Robots **20**(1), 43–59 (2006)
47.57 L. Armesto, V. Girbés, M. Vincze, S. Olufs, P. Muñoz-Benavent: Mobile robot obstacle avoidance based

on quasi-holonomic smooth paths, Lect. Notes Comput. Sci. **7429**, 244–255 (2012)
47.58 J.L. Blanco, J. González, J.A. Fernández-Madrigal: Foundations of parameterized trajectories-based space transformations for obstacle avoidance. In: *Motion Planning*, ed. by X.-Y. Jing (InTech, Rijeka 2008)
47.59 J. Minguez, L. Montano: Robot navigation in very complex dense and cluttered indoor/outdoor environments, 15th IFAC World Congr. (2002)
47.60 A. De Luca, G. Oriolo: Local incremental planning for nonholonomic mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 104–110
47.61 A. Bemporad, A. De Luca, G. Oriolo: Local incremental planning for car-like robot navigating among obstacles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 1205–1211
47.62 S. Quinlan, O. Khatib: Elastic bands: Connecting path planning and control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 802–807
47.63 O. Brock, O. Khatib: Real-time replanning in high-dimensional configuration spaces using sets of homotopic paths, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 550–555
47.64 F. Lamiraux, D. Bonnafous, O. Lefebvre: Reactive path deformation for nonholonomic mobile robots, IEEE Trans. Robotics **20**(6), 967–977 (2004)
47.65 Y. Yang, O. Brock: Elastic roadmaps – Motion generation for autonomous mobile manipulation, Auton. Robots **28**(1), 113–130 (2010)
47.66 E. Yoshida, C. Esteves, I. Belousov, J.-P. Laumond, T. Sakaguchi, K. Yokoi: Planning 3-d collision-free dynamic robotic motion through iterative reshaping, IEEE Trans. Robotics **24**, 1186–1198 (2008)
47.67 H. Kurniawati, T. Fraichard: From path to trajectory deformation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007)
47.68 O. Brock, O. Khatib: High-speed navigation using the global dynamic window approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 341–346
47.69 I. Ulrich, J. Borenstein: VFH*: Local obstacle avoidance with look-ahead verification, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 2505–2511
47.70 J. Minguez, L. Montano: Sensor-based motion robot motion generation in unknown, dynamic and troublesome scenarios, Robotics Auton. Syst. **52**(4), 290–311 (2005)
47.71 C. Stachniss, W. Burgard: An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002) pp. 508–513
47.72 R. Philipsen, R. Siegwart: Smooth and efficient obstacle avoidance for a tour guide robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)
47.73 L. Montesano, J. Minguez, L. Montano: Lessons learned in integration for sensor-based robot navigation systems, Int. J. Adv. Robotics Syst. **3**(1), 85–91 (2006)
47.74 F. Lu, E. Milios: Robot pose estimation in unknown environments by matching 2-D range scans, Intell. Robotic Syst. **18**, 249–275 (1997)
47.75 J. Minguez, L. Montesano, F. Lamiraux: Metric-based iterative closest point scan matching for sensor displacement estimation, IEEE Trans. Robotics **22**(5), 1047–1054 (2006)
47.76 A. Stenz: The focussed *d** Algorithm for real-time replanning, Int. Jt. Conf. Artif. Intell. (IJCAI) (1995) pp. 1652–1659
47.77 S. Koenig, M. Likhachev: Improved fast replanning for robot navigation in unknown terrain, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)
47.78 J.-P. Laumond: Robot motion planning and control. In: *Lecture Notes in Control and Information Science*, ed. by J.P. Laumond (Springer, New York 1998)
47.79 H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, S. Thrun: *Principles of Robot Motion* (MIT Press, Cambridge 2005)
47.80 S.M. LaValle: *Planning Algorithms* (Cambridge Univ. Press, Cambridge 2006)

Multimedia Contents

48. Modeling and Control of Legged Robots

Pierre-Brice Wieber, Russ Tedrake, Scott Kuindersma

The promise of legged robots over wheeled robots is to provide improved mobility over rough terrain. Unfortunately, this promise comes at the cost of a significant increase in complexity. We now have a good understanding of how to make legged robots walk and run dynamically, but further research is still necessary to make them walk and run efficiently in terms of energy, speed, reactivity, versatility, and robustness. In this chapter, we will discuss how legged robots are usually modeled, how their stability analysis is approached, how dynamic motions are generated and controlled, and finally summarize the current trends in trying to improve their performance. The main problem is avoiding to fall. This can prove difficult since legged robots have to rely entirely on available contact forces to do so. The temporality of leg motions appears to be a key aspect in this respect, as current control solutions include continuous anticipation of future motion (using some form of model predictive control), or focusing more specifically on limit cycles and orbital stability.

48.1 **A Brief History of Legged Robots** 1204
48.2 **The Dynamics of Legged Locomotion** 1204
48.2.1 Lagrangian Dynamics 1205
48.2.2 Newton and Euler Equations of Motion 1205
48.2.3 Contact Models 1207
48.3 **Stability Analysis – Not Falling Down** 1209
48.3.1 Fixed Points 1210
48.3.2 Limit Cycles 1210
48.3.3 Viability 1211
48.3.4 Controllability 1212
48.3.5 Robust or Stochastic Stability 1213
48.3.6 Input-Output Stability 1214
48.3.7 Stability Margins 1214
48.4 **Generation of Dynamic Walking and Running Motions** 1214
48.4.1 Early Offline Motion Generation Schemes 1215
48.4.2 Online Motion Generation: A Model Predictive Control Point of View 1215
48.4.3 Motion in Constrained Environments 1219
48.4.4 Motion Generation with Limited Computing Resources 1220
48.5 **Motion and Force Control** 1222
48.6 **Towards More Efficient Walking** 1225
48.6.1 Gait Generation for Dynamic Walking 1225
48.6.2 Orbital Trajectory Stabilization and Control 1226
48.7 **Different Contact Behaviors** 1227
48.7.1 Wall Climbing 1227
48.7.2 Tethered Walking 1227
48.7.3 Legs with Wheels 1228
48.7.4 Wheels with Legs 1228
48.8 **Conclusion** 1228
References 1228

The promise of legged robots over standard wheeled robots is to provide improved mobility over rough terrain. This promise builds on the decoupling between the environment and the main body of the robot that the presence of articulated legs allows, with two consequences. First, the motion of the main body of the robot can be made largely independent from the roughness of the terrain, within the kinematic limits of the legs: legs provide an active suspension system. Indeed, one of the most advanced hexapod robots of the 1980s was aptly called the Adaptive Suspension Vehicle [48.1]. Second, this decoupling allows legs to temporarily leave their

contact with the ground: isolated footholds on a discontinuous terrain can be overcome, allowing to visit places absolutely out of reach otherwise. Note that having feet firmly planted on the ground is not mandatory here: skating is an equally interesting option, although rarely approached so far in robotics.

Unfortunately, this promise comes at the cost of a hindering increase in complexity. It is only with the unveiling of the Honda P2 humanoid robot in 1996 [48.2], and later of the Boston Dynamics BigDog quadruped robot in 2005 that legged robots finally began to deliver real-life capabilities that are just beginning to match the long sought animal-like mobility over rough terrain. Not matching yet the capabilities of humans and animals, legged robots do contribute however already to understanding their locomotion, as evidenced by the many fruitful collaborations between robotics and biomechanics researchers.

48.1 A Brief History of Legged Robots

Before the advent of digital computers, legged machines could be approached only by electromechanical means, lacking in advanced feedback control. This *prerobotics* period culminated with the General Electric Walking Truck developed by Ralph Mosher, which inspired awe in the mid 1960s. The limb motions of this elephant-size quadruped machine were directly reflecting the limb motions of the onboard operator, who was responsible for all motion control and synchronization. Unfortunately, the strenuous concentration that this required limited operation to less than 15 minutes.

Digitally controlled legged robots started to appear in the late 1960s. Among early pioneers, Robert McGhee initiated a series of quadruped and hexapod robots first at University of South California, then at Ohio State University, culminating in the mid 1980s with the Adaptive Suspension Vehicle, a human carrying hexapod vehicle walking on natural and irregular outdoor terrain [48.1], while Ichiro Kato initiated a long series of biped and humanoid robots in the Waseda University, a series still continuing nearly half a century later [48.3]. But by the end of the 1970s, all legged robots were still limited to quasi-static gaits, i. e., slow walking motions with the center of mass of the robot always kept above its feet.

The transition to dynamic legged locomotion occurred in the beginning of the 1980s, with the first dynamically walking bipedal robot demonstrated at the Tokyo University [48.4], and the famous series of hopping and running monopedal, bipedal and quadrupedal robots developed at the MIT (Massachusetts Institute of Technology) LegLab under the direction of *Marc Raibert* [48.5]. Key theoretical breakthroughs came in the end of the 1980s, when *Tad McGeer* demonstrated that stable dynamic walking motions could be obtained by pure mechanical means, giving rise to a whole new field of research, *passive dynamic walking*, introducing new, key concepts such as orbital stability using Poincaré maps [48.6], with one simple conclusion: you need not have complete (or any) control to be able to walk dynamically and efficiently.

Legged robots were still mostly research laboratory curiosities working in limited situations when Honda unveiled the P2 humanoid robot in 1996 [48.2], a decade long secret project demonstrating unprecedented versatility and robustness, followed in 2000 by the Asimo humanoid robot. The world of humanoid and legged robots was ripe for companies to begin investing. A handful of other Japanese companies such as Toyota or Kawada were quick to follow with their own humanoid robots, while Sony began selling more than 150 000 of its Aibo home companion robot dogs. Boston Dynamics, a company *Marc Raibert* founded after leaving the MIT LegLab, finally unveiled its BigDog quadruped robot in 2005 [48.7], which was the first to demonstrate true animal-like locomotion capabilities on rough terrain.

The progress over the last decades has been remarkable. Profound questions have finally been answered: we now understand how to make legged robots walk and run dynamically. But other profound questions still have to be answered, such as how best to make them walk and run *efficiently*. The performance of legged robots needs to be improved in many ways: energy, speed, reactivity, versatility, robustness, etc. We will therefore discuss in this chapter how legged robots are usually modeled in Sect. 48.2 and how dynamic motions are currently generated and controlled in Sections 48.4 and 48.5, before discussing in Sect. 48.6 the current trends in improving their efficiency.

48.2 The Dynamics of Legged Locomotion

One of the major difficulties in making a legged robot walk or run is keeping its balance: where should the robot place its feet, how should it move its body in order to move safely in a given direction, even in case

of strong perturbations? This difficulty comes from the fact that contact forces with the environment are an absolute necessity to generate and control locomotion, but they are limited by the mechanical laws of unilateral contact.

This essential role of the contact forces is particularly clear in the derivatives of the total linear and angular momenta of the robot, the former involving the motion of its center of mass. Because of the importance of contact forces for legged locomotion, we briefly discuss here their different models.

48.2.1 Lagrangian Dynamics

Structure of the Configuration Space

As for every robot moving in their 3-D environment (in space or underwater for example), the configuration space of legged robots combines the configuration $\hat{\boldsymbol{q}} \in \mathbb{R}^N$ of their N joints with a global position $\boldsymbol{x}_0 \in \mathbb{R}^3$ and orientation $\boldsymbol{\theta}_0 \in \mathbb{R}^3$ (representing an element of $SO(3)$)

$$\boldsymbol{q} = \begin{pmatrix} \hat{\boldsymbol{q}} \\ \boldsymbol{x}_0 \\ \boldsymbol{\theta}_0 \end{pmatrix} . \tag{48.1}$$

The position $\boldsymbol{x}_0$ and orientation $\boldsymbol{\theta}_0$ are typically those of a central body (pelvis or trunk) or of an extremity (foot or hand).

Structure of the Lagrangian Dynamics

The specific structure of the configuration space outlined above is naturally reflected in the Lagrangian dynamics

$$\mathbf{M}(\boldsymbol{q}) \left[\begin{pmatrix} \ddot{\hat{\boldsymbol{q}}} \\ \ddot{\boldsymbol{x}}_0 \\ \ddot{\boldsymbol{\theta}}_0 \end{pmatrix} + \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{g} \\ \boldsymbol{0} \end{pmatrix} \right] + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix} + \sum_i \boldsymbol{C}_i(\boldsymbol{q})^{\mathrm{T}} \boldsymbol{f}_i \tag{48.2}$$

of the system, where $\mathbf{M}(\boldsymbol{q}) \in \mathbb{R}^{(N+6)\times(N+6)}$ is the generalized inertia matrix of the robot, $-\boldsymbol{g} \in \mathbb{R}^3$ is the constant gravity acceleration vector, $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{N+6}$ is the vector of Coriolis and centrifugal effects, $\boldsymbol{u} \in \mathbb{R}^N$ is the vector of joint torques, and for all i, $\boldsymbol{f}_i \in \mathbb{R}^3$ is a force exerted by the environment on the robot and $\mathbf{C}_i(\boldsymbol{q}) \in \mathbb{R}^{(N+6)\times 3}$ is the associated Jacobian matrix [48.8].

Since the vector $\boldsymbol{u}$ of joint torques has the same size as the vector $\hat{\boldsymbol{q}}$ of joint positions, the whole dynamics including the global position $\boldsymbol{x}_0$ and orientation $\boldsymbol{\theta}_0$ appears to be underactuated if no external forces $\boldsymbol{f}_i$ are exerted.

48.2.2 Newton and Euler Equations of Motion

Center of Mass and Angular Momentum

A consequence of the structure (48.2) of the Lagrangian dynamics is that the part of this dynamics which is not directly actuated involves the Newton and Euler equations of motion of the robot taken as a whole ([48.8] for detailed derivations). The Newton equation can be written in the following way

$$m\,(\ddot{\boldsymbol{c}} + \boldsymbol{g}) = \sum_i \boldsymbol{f}_i \,, \tag{48.3}$$

with m the total mass of the robot and $\boldsymbol{c}$ the position of its center of mass (COM). The Euler equation can be expressed with respect to the COM in the following way

$$\dot{\mathbf{L}} = \sum_i (\boldsymbol{p}_i - \boldsymbol{c}) \times \boldsymbol{f}_i \,, \tag{48.4}$$

with $\boldsymbol{p}_i$ the points of applications of the forces $\boldsymbol{f}_i$ and

$$\mathbf{L} = \sum_k (\boldsymbol{x}_k - \boldsymbol{c}) \times m_k \dot{\boldsymbol{x}}_k + \mathbf{I}_k \boldsymbol{\omega}_k \tag{48.5}$$

the angular momentum of the whole robot with respect to its COM, with $\dot{\boldsymbol{x}}_k$ and $\boldsymbol{\omega}_k$ the translation and rotation velocities of the different parts k of the robot, m_k and $\mathbf{I}_k$ their masses and inertia tensor matrices (expressed in global coordinates).

The Newton equation makes it obvious that the robot needs external forces $\boldsymbol{f}_i$ in order to move its COM in a direction other than that of gravity. The Euler equation is more subtle, as we will see during flight phases.

Flight Phases

During flight phases, when a legged robot is not in contact with its environment, not experiencing any contact forces $\boldsymbol{f}_i$, the Newton equation (48.3) simplifies to

$$\ddot{\boldsymbol{c}} = -\boldsymbol{g} \,. \tag{48.6}$$

In this case, the COM invariably accelerates along the gravity vector $-\boldsymbol{g}$ with constant horizontal speed, following a standard falling motion: there is absolutely no possibility to control the COM to move in any different way. The Euler equation (48.4) simplifies in the same way to

$$\dot{\mathbf{L}} = 0 \,, \tag{48.7}$$

imposing a conservation of the angular momentum $\mathbf{L}$. In this case, however, the robot is still able to generate and control both joint motions and global rotations, this is how cats are able to fall back on their feet when

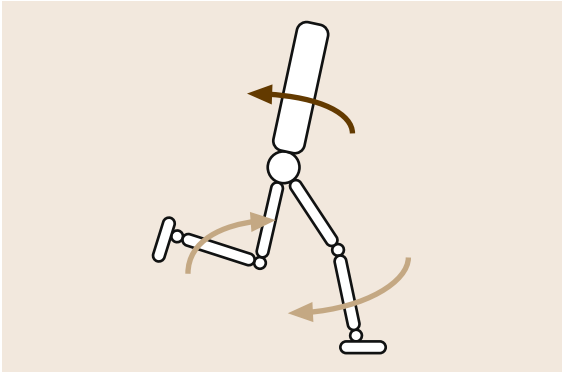


Fig. 48.1 Even though the angular momentum is constant during flight phases, the robot is still able to generate and control rotations of the whole body with the help of leg or arm motions (*light brown*), as a result of the *nonholonomy* of the angular momentum. This is how cats fall back on their feet when dropped from any initial orientation

dropped from any initial orientation (Fig. 48.1). This is a result of the *nonholonomy* of the angular momentum (48.5) which is not the derivative of any function of the configuration of the robot [48.9]. As a result, even though the angular momentum $\mathbf{L}$ is kept constant during the whole flight phase, the joint configuration $\hat{\boldsymbol{q}}$, and the global orientation $\boldsymbol{\theta}_0$ of the robot can be driven to any desired value at the end of the flight phase. We will see in Sect. 48.5 how this impacts the control of legged robots. Note that the dynamics of legged robots during flight phases is similar to the dynamics of free-floating space robots discussed in Chap. 55. Further discussion and developments can be found there.

In Contact with a Flat Ground: The Center of Pressure

In case the forces applied by the environment on the robot are due to contacts with a flat ground (while standing still, walking or running), let us consider a reference frame oriented along the ground, with the z axis orthogonal to it (therefore tilted if the ground is tilted, see Fig. 48.5). Without loss of generality, let us suppose that the points of contact, $\boldsymbol{p}_i$, with the ground are all such that $p_i^z = 0$.

Let us consider then the sum of the Euler (48.4) and the cross product of the COM $\boldsymbol{c}$ with the Newton (48.3)

$$m\boldsymbol{c} \times (\ddot{\boldsymbol{c}} + \boldsymbol{g}) + \dot{\mathbf{L}} = \sum_i \boldsymbol{p}_i \times \boldsymbol{f}_i \,, \tag{48.8}$$

and let us divide the result by the z coordinate of the Newton equation to obtain

$$\frac{m\boldsymbol{c} \times (\ddot{\boldsymbol{c}} + \boldsymbol{g}) + \dot{\mathbf{L}}}{m(\ddot{c}^z + g^z)} = \frac{\sum_i \boldsymbol{p}_i \times \boldsymbol{f}_i}{\sum_i f_i^z} \,. \tag{48.9}$$

Since $p_o^z = 0$, the x and y coordinates of this equation can be simplified in the following way

$$\boldsymbol{c}^{x,y} - \frac{c^z}{\ddot{c}^z + g^z}(\ddot{\boldsymbol{c}}^{x,y} + \boldsymbol{g}^{x,y}) + \frac{1}{m(\ddot{c}^z + g^z)}\mathbf{S}\dot{\mathbf{L}}^{x,y} = \frac{\sum_i f_i^z \boldsymbol{p}_i^{x,y}}{\sum_i f_i^z} \,, \tag{48.10}$$

with a simple rotation matrix

$$\mathbf{S} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} .$$

On the right hand side of (48.10) appears the definition of the center of pressure (COP) $\boldsymbol{z}$ of the contact forces, $\boldsymbol{f}_i$. These contact forces are usually unilateral (the robot can push on the ground, not pull)

$$f_i^z \geq 0 \,, \tag{48.11}$$

which implies that the CoP is bound to lie in the convex hull of the contact points (Fig. 48.2)

$$\boldsymbol{z}^{x,y} = \frac{\sum_i f_i^z \boldsymbol{p}_i^{x,y}}{\sum_i f_i^z} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} \,. \tag{48.12}$$

Combining this inclusion with the dynamic (48.10) reveals an ordinary differential inclusion (ODI)

$$\boldsymbol{c}^{x,y} - \frac{c^z}{\ddot{c}^z + g^z}(\ddot{\boldsymbol{c}}^{x,y} + \boldsymbol{g}^{x,y}) + \frac{1}{m(\ddot{c}^z + g^z)}\mathbf{S}\dot{\mathbf{L}}^{x,y} = \boldsymbol{z}^{x,y} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} \,, \tag{48.13}$$

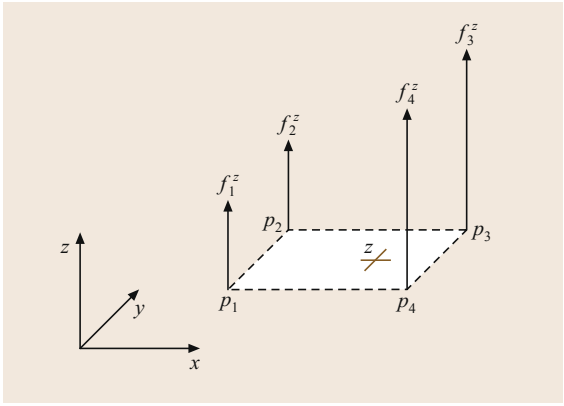


Fig. 48.2 The CoP $\boldsymbol{z}$ is bound to lie in the convex hull of contact points $\boldsymbol{p}_i$

which bounds the motion of the COM $\boldsymbol{c}$, of the robot and the variations of its angular momentum $\mathbf{L}$ with respect to the position $\boldsymbol{p}_i^{x,y}$ of the contact points.

This ODI can be reorganized in the following way

$$\frac{c^z}{\ddot{c}^z+g^z}(\ddot{\boldsymbol{c}}^{x,y}+\boldsymbol{g}^{x,y})=(\boldsymbol{c}^{x,y}-\boldsymbol{z}^{x,y})+\frac{1}{m(\ddot{c}^z+g^z)}\mathbf{S}\dot{\mathbf{L}}^{x,y}\,, \tag{48.14}$$

in order to expose its simple geometric meaning in Fig. 48.3. We can see especially that aside from the effects of gravity $\boldsymbol{g}^{x,y}$ and variations $\dot{\boldsymbol{L}}^{x,y}$ of the angular momentum, the horizontal acceleration $\ddot{\boldsymbol{c}}^{x,y}$ of the COM is the result of a force pushing the COM $\boldsymbol{c}^{x,y}$ away from the CoP $\boldsymbol{z}^{x,y}$, which is bound to lie in the convex hull of the contact points. We have here an intrinsically unstable dynamics.

Note finally that the definition (48.12) of the CoP can be reorganized to show that the horizontal momenta of the contact forces $\boldsymbol{f}_i$ with respect to the CoP $\boldsymbol{z}$ are equal to zero

$$\left[\sum_i(\boldsymbol{p}_i-\boldsymbol{z})\times\boldsymbol{f}_i\right]^{x,y}=\sum_i(\boldsymbol{p}_i^{x,y}-\boldsymbol{z}^{x,y})f_i^z=0\,. \tag{48.15}$$

Hence the CoP is also referred to as the zero moment point (ZMP) [48.10, 11].

In Contact with Multiple Surfaces

If the contact points $\boldsymbol{p}_i$ are not all on the same plane, we can introduce a CoP for each contact surface, but we cannot introduce a unique CoP for all contact forces as we did previously. Approximations, and generalizations

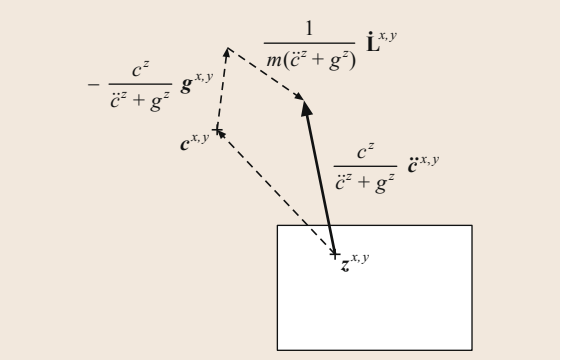


Fig. 48.3 The horizontal acceleration $\ddot{\boldsymbol{c}}^{x,y}$ of the COM of the robot is the sum of a force pushing the COM $\boldsymbol{c}^{x,y}$ away from the CoP $\boldsymbol{z}^{x,y}$ the effect of gravity $-\boldsymbol{g}^{x,y}$ and variations $\dot{\mathbf{L}}^{x,y}$ of the angular momentum

to limited multiple contact situations have been proposed but haven't been widely adopted [48.11–14]. In the general case, the Newton and Euler equations (48.3) and (48.4) have to be considered explicitly together with the unilaterality condition (48.11) in order to check which motion is feasible or not [48.15–17]. The problem we have to solve then relates very closely to the *force closure* problem discussed in Chap. 38 on grasping. Different ways to solve it as quickly as possible have been proposed [48.15, 18], and used mostly so far to measure offline the stability robustness of a given motion [48.19, 20], and only recently for motion planning, once again offline [48.21]. Refinements to curved contact surfaces have also been proposed [48.22].

48.2.3 Contact Models

The structure of the Lagrangian dynamics makes it clear that contact forces are central to the modeling and control of legged robots. But note that the only characteristics of these forces that we have introduced so far is their unilaterality (48.11). As a result, the previous analysis applies to various contact situations: walking and running motions, but also sliding situations such as when skiing or skating, or even rolling situations [48.23, 24].

Let us briefly discuss now standard contact models (more details can be found in Chap. 37 on contact modeling and manipulation). Concerning motion and forces tangential to the contact surfaces, a simple Coulomb friction model is usually considered. Concerning motion and forces orthogonal to the contact surfaces, two options are generally considered: a compliant or a rigid model, introducing impacts and other nonsmooth behaviors.

Coulomb Friction

When a contact point $\boldsymbol{p}_i$ is sliding on its contact surface, the corresponding tangential contact force $\boldsymbol{f}_i^{x,y}$ is proportional to the normal force f_i^z in a direction opposite to the sliding motion

$$\boldsymbol{f}_i^{x,y}=-\mu_0 f_i^z\frac{\dot{\boldsymbol{p}}_i^{x,y}}{\|\dot{\boldsymbol{p}}_i^{x,y}\|}\ \text{ if }\ \dot{\boldsymbol{p}}_i^{x,y}\neq 0\,, \tag{48.16}$$

with $\mu_0>0$ the friction coefficient. When the contact point is sticking and not sliding, the norm of the tangential force is simply bounded, with the same friction coefficient

$$\|\boldsymbol{f}_i^{x,y}\|\leq\mu_0 f_i^z\ \text{ if }\ \dot{\boldsymbol{p}}_i^{x,y}=0\,. \tag{48.17}$$

This is typically referred to as the *friction cone*. Note that this friction model directly implies the unilaterality condition (48.11).

Compliant Contact Models

Compliant contact models take into account the viscoelastic properties of the materials in contact in the direction orthogonal to the contact surfaces

$$f_i^z = -K_i p_i^z - \Lambda_i \dot{p}_i^z \text{ if } p_i^z \leq 0\,, \tag{48.18}$$

where K_i and Λ_i are stiffness and damping coefficients, respectively. In this case, the normal force f_i^z appears to be the result of a penetration of the contact point below the contact surface, i. e., when $p_i^z < 0$. Note that, as it is here, this model doesn't satisfy the unilaterality condition (48.11) in all situations and needs therefore to be saturated to enforce this property. Of course, when there is no contact, there is no contact force

$$f_i^z = 0 \text{ if } p_i^z > 0\,. \tag{48.19}$$

A similar spring-damper model can also be used to determine the frictional contact force $f_i^{x,y}$, subject to the bounds of the friction cone (48.17).

Rigid Contact Models

Rigid contact models are somewhat simpler to introduce: either there is contact and the normal force can take any nonnegative value,

$$f_i^z \geq 0 \text{ if } p_i^z = 0\,, \tag{48.20}$$

or there is no contact and no contact force

$$f_i^z = 0 \text{ if } p_i^z > 0\,. \tag{48.21}$$

In this case, no penetration of the contact points below the contact surfaces are considered. Note also that the unilaterality condition (48.11) is satisfied here by definition. As a matter of fact, this rigid model can be summarized in the following way

$$f_i^z \geq 0,\ p_i^z \geq 0,\ f_i^z p_i^z = 0\,, \tag{48.22}$$

where it appears that both the normal force f_i^z and the position of the contact point with respect to the contact surface p_i^z must be nonnegative, but at least one of them must be equal to 0. This is called a *complementarity condition*. Analogous to (48.20) and (48.21), it defines the normal force to be either zero, or the force necessary to maintain $p_i^z = 0$. Note that the sticking-or-sliding behavior of Coulomb friction can also be represented with such a complementarity condition. Note also that this set of implicit equations may not have a solution or may have multiple ones in certain pathologic situations [48.25].

Rigid and compliant contact models are compared in Fig. 48.4 in a static situation (when $\dot{p}_i^z = 0$), showing clearly how the rigid model corresponds to an infinitely stiff compliant model. Compliant contact models can model more accurately the deformation of the contacting bodies, but accurate models often require very stiff springs, causing the resulting differential equations to be numerically stiff, what can slow down or complicate numerical analysis. Rigid models are comparatively more straightforward to integrate and analyze, with purely rigid bodies and contact surfaces. They are therefore preferred when theoretically or numerically studying the stability of legged robots, when optimizing walking or running trajectories, etc.

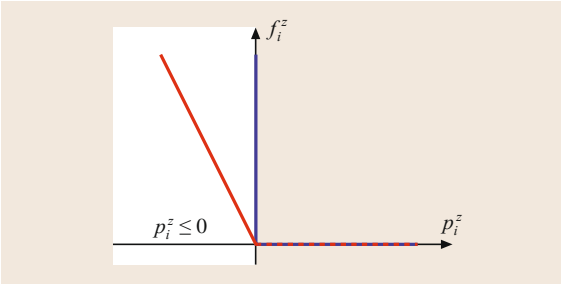


Fig. 48.4 Comparison of a rigid (*blue*) and compliant (*red*) contact model in a static situation. Unlike the rigid model, the compliant model considers a penetration of the contact point below the contact surface, when $p_i^z < 0$. The graph of the rigid model, with the specific shape of a right angle, is called a Signorini graph

Impacts

Since rigid contact models do not allow penetration of the contact points below the contact surfaces, when a point $\boldsymbol{p}_i$ reaches a contact surface $p_i^z = 0$ with a velocity $\dot{p}_i^z < 0$, this velocity has to change instantaneously in order to satisfy the no-penetration assumption, $p_i^z \geq 0$ [48.26]. What happens in this situation is an *impact*, a discontinuity of the velocity, where we need to distinguish the velocity of the robot before impact $\dot{\boldsymbol{q}}^-$, and its velocity after impact, $\dot{\boldsymbol{q}}^+$. A straightforward integration of the Lagrangian dynamics (48.2) over a time singleton [48.27] gives us a relation between pre- and post-impact velocities and impulsive forces $\boldsymbol{F}_i$

$$\mathbf{M}(\boldsymbol{q})\left(\dot{\boldsymbol{q}}^+ - \dot{\boldsymbol{q}}^-\right) = \sum_i \mathbf{C}_i(\boldsymbol{q})^{\mathrm{T}} \boldsymbol{F}_i\,. \tag{48.23}$$

But in order to completely define the impact law and compute the post-impact velocity, we also need a model of the impulsive forces. Unfortunately, this is a complex problem still open to research [48.28, 29], especially in situations of multiple contacts which are the standard situation for legged robots. The usual approach for

legged robots is therefore to assume the post-impact behavior, usually considering that the contact points eventually stick to the contact surfaces

$$\dot{\boldsymbol{p}}_i^+ = \mathbf{C}_i \dot{\boldsymbol{q}}^+ = 0\,. \tag{48.24}$$

This is, however, a strong assumption, and reality can be much more complex, as demonstrated in [48.30]. Moreover, a recent study shows that post-impact sticking may not always be desirable [48.31].

Impacts are central in most stability analyses of passive dynamic walking machines (Sect. 48.6), but they can be potentially destructive for the mechanical parts of a robot. As a result, they are often carefully avoided and therefore generally disregarded outside of passive dynamic walkers. A recent study shows however that this is probably a mistake [48.32].

Hybrid and Nonsmooth Dynamics

With rigid contacts, the dynamics of legged robots appears to switch, depending on the contact situations (48.20) or (48.21). Discontinuities of the state of the robot also occur at impacts. A classical way to combine these different aspects is with a hybrid dynamical system. This approach has however some limitations [48.33, 34], the most obvious one being its incapacity to handle properly *Zeno behaviors*, infinite accumulations of impacts in finite time [48.35]. An example of such a situation is when a legged robot is rigidly standing on one foot, and gently rocking from one contact edge to the opposite contact edge, with a continuously decreasing period (at least according to the perfectly rigid model).

Impacts with multiple contacts and Zeno behaviors are not the only difficulties with rigid contact models: there is also the Painlevé paradox, tangential impacts, impacts without collisions, etc. The nonsmooth dynamics approach [48.25, 27, 33, 34], where the Lagrangian dynamics is turned into a measure differential equation, accelerations being abstract measures, velocities being functions with locally bounded variations, appears to be a much more adequate way to deal with all these intricacies. But it has not been widely adopted for legged robots because of a significant increase in mathematical complexity.

Many of the complexities of rigid body contact dynamics can be avoided by discretizing the system dynamics in time and reasoning about the integral of contact forces acting over a time step. In particular, no distinction is made between impulses and finite contact forces over a time step. By conservatively approximating the friction cone as a polyhedron, forward dynamics can be cast as a linear complementarity problem (LCP) [48.36], for which efficient solvers exist. This has become a popular formulation for simulation of rigid bodies in frictional contact and it has recently seen applications to control design for legged systems [48.37].

48.3 Stability Analysis – Not Falling Down

The robot model described above represents a complex, constrained, nonlinear dynamical system. Our goal for this section is to understand the long-term behavior of this dynamical system, to answer questions like, *Will the robot fall down?* As we will see in the following sections, this long-term behavior can be changed by open-loop or closed-loop feedback control, with the explicit goal of minimizing the likelihood that the robot will fall down.

For controlled nonlinear dynamical systems, there are a number of useful concepts relating to their safeness and stability, including:

- *Fixed points*: Stable fixed points represent the static postures in which the robot can safely stand still.
- *Limit cycles*: Limit cycles provide a natural extension of fixed-point analysis to *periodic* walking or running motions.
- *Viability*: Viability is a concept of controlled invariance, which analyzes the set of states from which the robot is able to avoid to fall. Unfortunately, this property can be intractable to compute.
- *Controllability*: Controllability provides a slightly restricted notion of viability, analyzing the set of states from which the robot is capable of returning to a particular fixed point (or limit cycle). This can be more tractable to compute than viability, especially for simple models.

In addition, if there are unknown errors in the robot model, if there is uncertainty about the environment (e.g., the location of the ground), or if there are unmodeled disturbances, then additional tools are available from robustness analysis and stochastic stability:

- *Robust stability*: Robust stability (or viability) examines the properties of the system considering worst-case (bounded) disturbances. For instance, a robust controller may be able to guarantee that a fixed point is stable even if the estimate of the mass of the trunk is wrong by $\pm 10\%$.

- *Stochastic stability*: Stochastic analysis provides tools to investigate the *probability* of falling down. For many robot disturbance models, the system will always fall eventually (with probability one), but analysis can reveal long-living *metastable* distributions.
- *Input-output stability*: This analysis treats a particular disturbance as an input, a performance criteria as output, and attempts to compute a relative *gain* or *sensitivity* of the robot performance due to this input.
- *Stability margins*: Robustness analysis can be difficult. In practice control designers often settle for the system staying comfortably away from the boundaries of deterministic stability.

We give a very brief overview of these tools in the remainder of this section, with an attempt to emphasize features of the analysis that are particular to legged robots.

48.3.1 Fixed Points

Given a first-order ordinary differential equation governed by $\dot{\boldsymbol{x}} = \boldsymbol{a}(\boldsymbol{x})$, a *fixed point* of the system is a state $\boldsymbol{x}^*$, for which $\boldsymbol{a}(\boldsymbol{x}^*) = 0$. The governing equations for a legged robot, given in Sect. 48.2, are second-order, take a control input, and potentially require a differential inclusion to describe the contact forces, but the notion of a fixed point is still meaningful. Here a fixed point is a configuration $\boldsymbol{q}^*$, for which there exists feasible control inputs $\boldsymbol{u}^*$, and feasible contact forces $\boldsymbol{f}^*$, so that $\dot{\boldsymbol{q}}^* = \ddot{\boldsymbol{q}}^* = \boldsymbol{0}$ is a solution to (48.2). A fixed point of this system is a posture in which the legged robot is able to stand still.

In static situations, when $\ddot{\boldsymbol{c}} = \dot{\mathbf{L}} = \boldsymbol{0}$, the ODI (48.13) gives the following necessary condition

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\boldsymbol{g}^{x,y} = \boldsymbol{z}^{x,y} \in \text{conv}\left\{p_i^{x,y}\right\} \tag{48.25}$$

for a static equilibrium on a flat ground. This necessary condition states that the COM $\boldsymbol{c}$ must project on the ground along the gravity vector $-\boldsymbol{g}$, inside the convex hull of the contact points p_i, also known as the *support polygon* (Fig. 48.5). This is a very well known and widely used necessary condition. Note, however, that it is valid only when considering contacts with a flat ground [48.15].

Once a fixed point has been found, one would often like to examine its stability. For smooth nonlinear differential equations, local stability can often be established via linearizing the dynamics at the fixed point and then applying well-known tools from linear systems theory. Indeed, these tools can also be applied

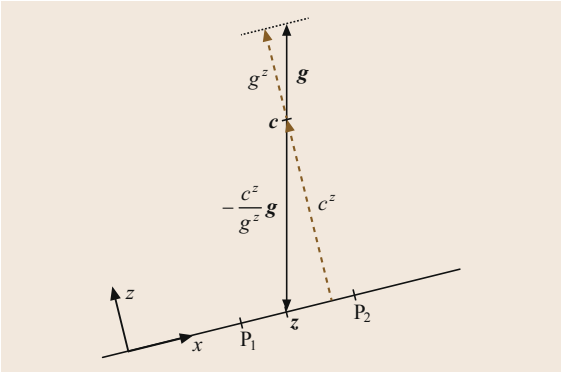


Fig. 48.5 A necessary static equilibrium condition on a flat ground is that the COM $\boldsymbol{c}$ must project on the ground along the gravity vector $-\boldsymbol{g}$ inside the convex hull of the contact points P_i also known as the *support polygon* (*red*)

here, *under the strong assumption that the contact conditions do not change* (e.g., either $f_i^z = 0$ and $p_i^z \geq 0$, or $f_i^z \geq 0$ and $p_i^z = 0$). Each active contact adds an equality constraint to the linearized system and the eigenvalues of this constrained linear system are only meaningful in the minimal coordinates; e.g., for a robot standing on horizontal ground the assumption precludes the ability to examine a perturbation of the entire robot in the vertical direction. Note that in a hybrid model of the legged robot, this assumption is equivalent to linearizing within a single hybrid mode, and it is common practice to describe each hybrid mode in its minimal coordinates [48.38]. Once local stability is established, it may also be possible to understand the *region of attraction* of the fixed point [48.39, 40]. Evaluating the stability of these models through changes in contact configurations is a new and important area of research. For recent work, see for instance [48.41].

48.3.2 Limit Cycles

A natural extension of fixed-point analysis to walking and running motions is to examine the existence and stability of periodic orbits, or limit cycles, of the dynamical system. Here a periodic orbit is a solution $\{\langle \boldsymbol{q}(t), \boldsymbol{u}(t), \boldsymbol{f}(t)\rangle \mid t \in [0, \infty)\}$ of the dynamical system with a finite period $T > 0$ such that $\boldsymbol{q}(t+T) = \boldsymbol{q}(t), \boldsymbol{u}(t+T) = \boldsymbol{u}(t)$, and $\boldsymbol{f}(t+T) = \boldsymbol{f}(t)$. Almost always, periodic orbits are discovered numerically using techniques from motion planning like those described in Sect. 48.4. For passive, or very minimally actuated robots, the very existence of a periodic solution can be exciting [48.42]; robots with more actuators typically have an abundance of periodic solutions and planning techniques can be used to find solutions that, for instance, minimize the a quantity of interest such as the

cost of transport [48.43] or a measure of open-loop stability [48.44].

A periodic solution $\langle \boldsymbol{q}_0(\cdot), \boldsymbol{u}_0(\cdot), \boldsymbol{f}_0(\cdot) \rangle$ with period T, is (asymptotically) *orbitally stable* [48.45] or *limit-cycle stable* if, given initial conditions $\boldsymbol{q}(0)$ we have

$$\lim_{t \to \infty} \left[\min_{0 \le t' < T} \| \boldsymbol{q}(t) - \boldsymbol{q}_0(t') \| \right] = 0 \,. \tag{48.26}$$

The key feature here that there is no requirement that system trajectories converge in time; only the distance to the closest point on the orbit must go to zero. This stability can be defined for initial conditions restricted to a local neighborhood of $\boldsymbol{q}_0(\cdot)$, for a region containing $\boldsymbol{q}_0(\cdot)$, or globally (though no interesting legged robot is globally stable) and for inputs determined by a variety of open- or closed-loop control policies, $\boldsymbol{u}(t) = \boldsymbol{\pi}(t, \boldsymbol{q}(t), \dot{\boldsymbol{q}}(t), \boldsymbol{f}(t))$. Under mild conditions, a sufficient criteria for establishing orbital stability is establishing fixed-point stability of a Poincaré map [48.46], as illustrated in Fig. 48.6. Due to the limitation that finding periodic solutions is typically done numerically, Poincaré map analysis is also typically done numerically, e.g., using finite differences to evaluate a local linearization of the map. This does not necessarily preclude rigorous analysis, and techniques exist to compute regions of attraction to hybrid limit cycles despite potential numerical inaccuracies [48.47]. Finally we note that limit cycle analysis can be applied (carefully) to solutions that are not periodic in all states – for instance, the x, y location of the floating base should be left out if the robot needs to make forward progress!

The primary advantage of limit cycle analysis is that, using the Poincaré map and related methods, many of the tools from fixed point analysis and linear systems theory can be easily extended to evaluate (local) stability and even to design a (locally) stabilizing controller. However, this is a very limited definition of locomotion; useful locomotion through nontrivial environments will likely require aperiodic motions.

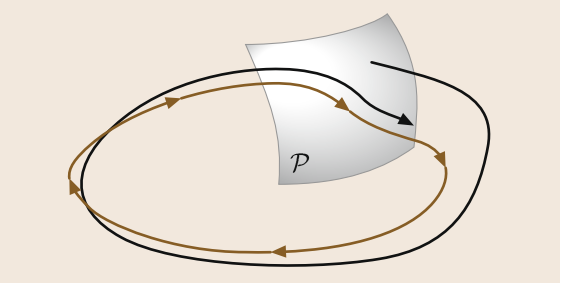


Fig. 48.6 Illustration of a Poincaré surface $\mathcal{P}$ defined on a periodic orbit (*red*) and a trajectory converging to the orbit (*black*)

Extensions to Aperiodic Stability

The notion of stability requires convergence to some nominal solution as time goes to infinity, thus the nominal solution must be defined over an infinite interval. A common approach involves stitching together periodic solutions with provably bounded finite-time transition maneuvers which can switch between them [48.48]. Orbital stability of an (infinite) aperiodic trajectory can be established using a transverse linearization or *moving Poincaré section* [48.49]. For real-world situations where the entire desired trajectory cannot be known apriori, these guarantees can also be provided using online motion planning and receding-horizon control [48.50].

48.3.3 Viability

If the objective of a legged robot is simply to avoid falling down, it can be possible then to directly define (at least theoretically) the set of postures $\mathcal{F} \subseteq \mathbb{R}^{N+6}$, where the robot has fallen down and that the robot should simply avoid. This leads to introducing the set of states from which the robot is able to avoid entering $\mathcal{F}$ – the set of states from which the robot is able to avoid falling down. These states will be called *viable*, following the *viability* theory developed in [48.51] for general ODIs and introduced in the analysis of walking robots in [48.52]. In principle, the set of all viable states represents a safe operating envelope for the robot. It is typically intractable to compute explicitly but it can be approached indirectly in the case of walking robots in the following way.

Center of Mass Dynamics During Horizontal Walking

In case the robot is walking on a horizontal ground, the z axis is aligned with gravity, so $\boldsymbol{g}^{x,y} = 0$ and the ODI (48.13) becomes

$$\boldsymbol{c}^{x,y} - \frac{c^z}{\ddot{c}^z + g^z} \ddot{\boldsymbol{c}}^{x,y} + \frac{1}{m(\ddot{c}^z + g^z)} \mathbf{S}\dot{\mathbf{L}}^{x,y} \in \operatorname{conv}\left\{ \boldsymbol{p}_i^{x,y} \right\} . \tag{48.27}$$

We can observe that it is linear with respect to the horizontal motion $\boldsymbol{c}^{x,y}$, $\ddot{\boldsymbol{c}}^{x,y}$ of the COM and variations of the angular momentum $\mathbf{L}^{x,y}$. The vertical motion c^z, $\ddot{c}^z$ of the COM can usually be bounded, but in order to simplify the following derivations, let us suppose that the COM moves strictly horizontally above the ground, i. e., c^z is constant and $\ddot{c}^z = 0$, so the ODI becomes

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z} \ddot{\boldsymbol{c}}^{x,y} + \frac{\mathbf{S}\dot{\mathbf{L}}^{x,y}}{mg^z} \in \operatorname{conv}\left\{ \boldsymbol{p}_i^{x,y} \right\} . \tag{48.28}$$

Variations of the angular momentum $\mathbf{L}$ can also be bounded in the x and y directions, but to make things

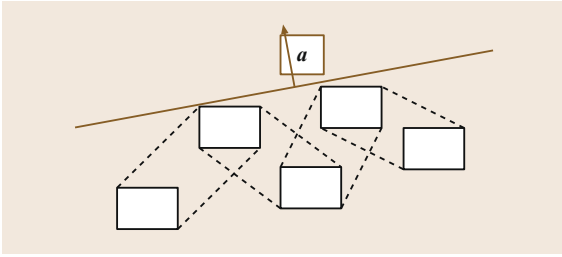


Fig. 48.7 A situation where, for some reason, no steps can be undertaken beyond a certain line (*red*), with $\boldsymbol{a}$ a vector orthogonal to this line and pointing beyond it

as simple as possible, let us consider these variations to be equal to 0, so the ODI takes a very simple second order linear form

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} = \boldsymbol{z}^{x,y} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} . \tag{48.29}$$

An Example of Inevitable Fall

Let us consider furthermore a situation where, for some reason (e.g., a cliff), no contact with the ground can be realized beyond a certain line (Fig. 48.7), with $\boldsymbol{a}$ a vector orthogonal to this line and pointing beyond it. If the COM of the robot reaches this line at a time t_0 with a speed pointing beyond it ($\boldsymbol{a}^{\mathrm{T}}\dot{\boldsymbol{c}}^{x,y}(t_0) > 0$), this linear ODI can be integrated analytically very simply and lead to the following inequality [48.53], valid at all time $t \geq t_0$

$$\boldsymbol{a}^{\mathrm{T}}\left[\boldsymbol{c}^{x,y}(t) - \boldsymbol{c}^{x,y}(t_0)\right] \geq \frac{\boldsymbol{a}^{\mathrm{T}}\dot{\boldsymbol{c}}^{x,y}(t_0)}{\omega}\sinh\left[\omega(t-t_0)\right] , \tag{48.30}$$

with

$$\omega = \sqrt{\frac{g^z}{c^z}} . \tag{48.31}$$

The right hand side is increasing exponentially with time. As a result, the position $\boldsymbol{c}^{x,y}$ of the COM is diverging exponentially in the direction of the vector $\boldsymbol{a}$, leading inexorably to a fall.

A Sufficient Condition for Viability

In the simplified linear case developed above, the inequality (48.30) establishes that in the case of a fall, the motion of the COM diverges exponentially, so the integral of the norm of any n^{th} derivative of its position,

$$\int_{t_0}^{\infty} \left\| \boldsymbol{c}^{(n)}(t) \right\| \mathrm{d}t , \tag{48.32}$$

would be infinite. Therefore, if we can find a finite value for such an integral from a given initial state of the robot, we can conclude that this state is viable.

48.3.4 Controllability

For any dynamical system, an initial condition $\boldsymbol{x}_0$ is *controllable* to a final state $\boldsymbol{x}_f$, if there exists a feasible input trajectory $\boldsymbol{u}(\cdot)$, which takes the system from $\boldsymbol{x}_0$ to $\boldsymbol{x}_f$. An initial state $\boldsymbol{x}_0$ is denoted *finite-time* controllable to $\boldsymbol{x}_f$ if the minimum time required is finite. Controllability analysis is a well understood topic for linear dynamical systems, where a simple rank condition on the *controllability matrix* provides necessary and sufficient conditions to establish whether or not any initial condition can be driven to any final state [48.54]. For nonlinear systems, the property is dependent on both the initial and final states.

In the case of legged robots, instead of considering the set of all viable states introduced earlier, we could consider instead the more limited set of states that are controllable to stable fixed-points – states that can come to a stable stop after a given number of steps. This is a sufficient condition for viability. Such states have been called *capturable* [48.55], and arguably encompass most of the states of interest for legged robots. A variant would be to consider states from which the robot is able to reach a stable limit cycle, or any other state which is known to be viable in the first hand.

The Capture Point

In the simple linear case developed above, the capturable states can be identified analytically with the help of the compound variable

$$\boldsymbol{\xi} = \boldsymbol{c} + \frac{1}{\omega}\dot{\boldsymbol{c}} , \tag{48.33}$$

introduced independently as the *extrapolated center of mass* (XCOM) [48.56], the *capture point* [48.57] or the *divergent component* of the dynamics [48.58]. These three denominations correspond to three key properties of this variable. First of all, a trivial reformulation gives

$$\dot{\boldsymbol{c}} = \omega(\boldsymbol{\xi} - \boldsymbol{c}) , \tag{48.34}$$

revealing that this point $\boldsymbol{\xi}$ is the point where the COM is converging to, hence the extrapolated COM. Following the linear dynamics (48.29), the horizontal motion of this point satisfies

$$\dot{\boldsymbol{\xi}}^{x,y} = \omega(\boldsymbol{\xi}^{x,y} - \boldsymbol{z}^{x,y}) , \tag{48.35}$$

where we can see that this point $\boldsymbol{\xi}$ diverges away from the CoP $\boldsymbol{z}$. But if this point is above the support polygon,

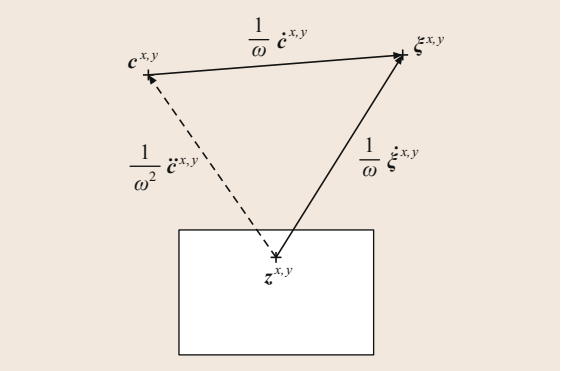


Fig. 48.8 The speed of the COM $\boldsymbol{c}$ points towards the capture point $\boldsymbol{\xi}$, while the speed of this capture point points away from the CoP, $\boldsymbol{z}$. The combination of these two first-order dynamics gives the acceleration $\ddot{\boldsymbol{c}}$, pushing the COM away from the CoP

if we have

$$\boldsymbol{\xi}^{x,y} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} , \tag{48.36}$$

we can have $\boldsymbol{z}^{x,y} = \boldsymbol{\xi}^{x,y}$ so the point $\boldsymbol{\xi}$ can stay motionless, and the COM will converge to it and come to a stop, hence it can be seen as a capture point (CP). Finally, we can observe that the second-order linear dynamics (48.29) has been decomposed here as two first-order linear dynamics (48.34) and (48.35), the first one being stable, with the COM converging to the XCOM, the second one being unstable, with the XCOM diverging away from the CoP, hence the divergent part of the dynamics (48.29). Here appears an interesting structure relating the motion of the points $\boldsymbol{z}$, $\dot{\boldsymbol{c}}$ and $\boldsymbol{\xi}$, shown in Fig. 48.8 [48.59, 60].

An important observation here is that if the state of the robot satisfies the condition (48.36), then the robot can stop without making any step and it is capturable and viable. Further analysis, considering any given number of steps and a nonzero angular momentum L can be found in [48.60].

48.3.5 Robust or Stochastic Stability

For the deterministic robot dynamics models described in Sect. 48.2, the concepts of fixed-points, limit cycles, viability, and controllability form the core issues of analysis. However, in the presence of disturbances, such as model inaccuracies or unpredicted variation in terrain, tools from robust verification must be called upon to characterize the stability of a system. Broadly speaking, robust stability analysis exists in two forms. In *worst-case* analysis, an upper bound is given for a disturbance and the goal is to demonstrate that for any disturbance within this bound, the system will not reach an unstable state. Alternatively, *stochastic* analysis can be performed where a disturbance probability distribution is specified with the goal being to demonstrate that the system avoids unstable states with high probability. Worst-case analysis has received surprisingly little attention to date in the legged robotics community, however analysis approaches based on Lyapunov functions [48.38, 41, 47] can be easily extended using the notion of a *common Lyapunov function* [48.61]. One challenge here is that the nominal solutions, e.g., the exact shape of the nominal limit cycle, are often parameter dependent: the system might still be stable with different parameters, but to a slightly different limit cycle.

Stochastic stability has received relatively more attention in the legged robotics community, perhaps because many interesting models of possible disturbances do result in robots that walk for a long time, but do eventually fall over. Such systems cannot be classified as stable, but it seems incomplete to simply call them unstable. Such systems are said to be *metastable*, a concept first established in the physics community [48.62] and later used to analyze walking systems [48.63]. From this point of view, the stability of a system can be characterized by its mean time to failure.

Recall the idea of Poincaré maps described above, but now consider the discrete time stochastic return map dynamics for a periodic walking system,

$$\boldsymbol{x}_{k+1} = r(\boldsymbol{x}_k, h_k) , \tag{48.37}$$

where $h_k \sim P(h)$ is a random variable that, e.g., captures the terrain height at time step k. Discretizing the state space $\mathcal{X} \subseteq \mathbb{R}^{(N+6)} \times \mathbb{R}^{(N+6)}$ into d states allows us to define the Markov process based on the above return map

$$\boldsymbol{p}_{k+1} = \mathbf{T}\boldsymbol{p}_k , \tag{48.38}$$

where $p_k \in \mathbb{R}^d$ is the state probably distribution vector and $\mathbf{T}$ is a stochastic matrix with entries $T_{ij} = \Pr(\boldsymbol{x}_{k+1}^i | \boldsymbol{x}_k^j)$. If we assume that all failure states are absorbing and can be grouped together into a single state, this corresponds to a column in $\mathbf{T}$ containing a single 1 and $\mathrm{d}-1$ zeros. The largest eigenvalue of $\mathbf{T}$ must then be $\lambda_1 = 1$ with a corresponding eigenvector $\boldsymbol{v}_1$, describing the stationary distribution of the absorbing state. Interestingly, the second largest eigenvalue has been found to be close to unity $\lambda_2 \approx 1$ and much larger than the remaining $\mathrm{d}-2$ eigenvalues, for several simple walking systems [48.63]. This suggests the existence of a metastable neighborhood in state space captured by the distribution $\boldsymbol{v}_2$ with an associated time constant $\tau = -1/\log(\lambda_2)$. Other approaches to computing these stochastic stability bounds are also possible [48.64].

48.3.6 Input–Output Stability

Input-output methods define sensitivity measures to summarize the system's response to disturbances. For example, for a walking system, one could define a vector $\boldsymbol{l} \in \mathbb{R}^m$ composed of m gait variables that directly relate to failure modes (such as foot-ground clearance) and measure the effect that a set of input disturbances $\boldsymbol{w}$, have on their values. The *gait sensitivity norm* does exactly this [48.65]; here the dynamic system response $\|\partial \boldsymbol{l}/\partial \boldsymbol{w}\|_2$ is used as a measure of disturbance rejection. How is this response actually computed? In practice, computing the system response can be done experimentally by generating a variety of perturbations and measuring the effect in the gait variables. Although laborious, this approach is probably the most appropriate for physical systems. Alternatively, for simulated limit cycle walkers, linearization of the Poincaré return map and perturbation analysis yields a complete characterization of the dynamic system response.

More general tools from nonlinear control theory can also be applied. The L_2 gain is a sensitivity measure that has seen widespread application in nonlinear control theory. For complex dynamical systems such as walking robots, an upper bound on the L_2 gain can be approximated using storage functions, which can often be computed using convex optimization [48.66, 67].

48.3.7 Stability Margins

In practice, computing performance measures like the ones mentioned above for complex walking systems can be very difficult. As a result, researchers often employ heuristic stability margins to keep the robot away from the boundary of stability. Here the definition of stability can vary widely. The simplest stability margins rely only on the static configuration of the robot, where if the COM ground projection leaves the support polygon, this corresponds to an uncompensated moment on the foot resulting in a rotation along its edge. Popular static metrics include the distance between the ground projection of the COM and the edges of the support polygon or the minimum potential energy required to tip the robot over sideways [48.68]. Because these stability criteria are based only on statics, they are only practically useful for slow (or *quasi-static*) motions.

The most commonly used dynamic stability margin requires that the CoP (or ZMP) remains within the boundary of the support polygon. When this condition is satisfied, the foot cannot rotate around the boundary of the support polygon. Note that this is certainly very conservative – many bipedal robots walk dynamically even with point feet (resulting in a CoP that is *always* on the boundary of the support polygon) – but this metric has proven extremely useful for flat-foot dynamic walking. Goswami introduced the related concept of the foot rotation indicator (FRI) [48.69] that is defined as the point on the ground where the net ground reaction force would have to act to keep the foot from rotating; this point need not stay inside the support polygon and can therefore be applied for a wider range of gaits. However, the stability claims based on these criteria are valid only on flat terrain and they do not constitute necessary or sufficient conditions for stable walking in general [48.15, 55].

When the COM projection exits the support polygon, there are two ways to make it return: by accelerating the COM in the direction of the support polygon through changes in angular momentum or by changing the support polygon by taking a step. This basic insight led to stability margins based on *capture regions* [48.55, 60]. A capture region is simply the set of all capture points (defined in Sect. 48.3.4). The *capture margin* is a metric that determines the *capturability* of a walking system. If the capture region and support polygon overlap, it is defined as the maximum distance between the points in the capture region and the closest edge of the support polygon, otherwise it is the negative distance between the capture region and support polygon. Note that large values under this metric are related to the total capture region area, which is in turn related to the ability of the walking system to stabilize itself, e.g., with upper body motions that affect angular momentum [48.60]. Extensions to n-step capture margins have also been developed and described in detail [48.55].

48.4 Generation of Dynamic Walking and Running Motions

As mentioned in the brief historical introduction to this chapter, early legged robots were maintaining quasi-static equilibrium postures at all time. The transition to more dynamic motions took place at the beginning of the 1980s, associated with biped [48.4] and monopod [48.5] robots. As a result, this Section on dynamic legged locomotion will focus on results obtained mostly with biped robots, although most of the methods discussed here could be applied as well to other legged robots.

48.4.1 Early Offline Motion Generation Schemes

The very limited computing power of early robots allowed very limited online decisions. As a result, trajectories were usually computed offline, especially in complex situations such as for legged robots. Online motion generation has finally become possible only by the year 2000, thanks to the exponential increase in computing power and the continuous progress of numerical methods during the last decades.

Trajectory Optimization

One of the earliest approaches to computing walking and running motions was through numerical optimization [48.70]. The idea is to leverage the capacity of trajectory optimization methods to take into account nonlinear dynamics and objectives such as minimizing energy consumption, and compute the corresponding optimal motion. Unfortunately, the dynamics of legged robots are so complex that this approach was still stuck with simple planar models by the end of the 1990s [48.71–75]. It is only by the year 2000 that the first optimal walking motions for complete 3-D models could be computed [48.76, 77].

Fortunately, numerical methods have continued to improve (as discussed in Sect. 48.6.1), allowing researchers to solve complex problems such as optimizing trajectories and the mass distribution of the robot simultaneously and maximizing the open-loop stability of trajectories [48.78].

Unfortunately, this approach still requires time consuming computations that often cannot be realized online. Not being able to compute walking or running motions online limits the robots to a predefined set of precomputed actions, potentially ruining their versatility and reactivity. One way to alleviate this serious limitation is to generate a database of trajectories [48.79] that can be queried online, possibly conditioned on orders given to the robot [48.80] or the current state of the robot in order to improve its stability [48.40, 81, 82].

Artificial Synergy Synthesis and the ZMP Approach

Partitioning the problem is another early approach, that aimed at palliating the complexity of the dynamics of legged robots that hindered trajectory optimization. The idea is to assign some degrees of freedom of the robot to take care of dynamic constraints such as the ODI (48.13), allowing the rest of the robot to be operated more or less independently. This general approach has been called *artificial synergy synthesis* [48.10].

The original proposition in [48.10] was to use trunk rotations to ensure dynamic feasibility while the legs of the robot executed a given prerecorded motion. More precisely, leg motions and contact points $\boldsymbol{p}_i$ being predefined, the trajectory of the CoP z could be predefined accordingly, so that it was just a matter of solving the ordinary differential equation (ODE) (48.13) for this predefined z to obtain the required rotations of the trunk. This original proposition was demonstrated experimentally 20 years later on the Waseda University WL-12RV biped walking robot [48.83]. This method of predefining the trajectory of the CoP has eventually been called the *ZMP approach* to walking motion generation (remember the ZMP is just another name given to the CoP). This ZMP approach has been associated later with having feet always flat on the ground, excluding heel and toe rotation phases. Note, however, that such phases were considered in the original proposition [48.10].

Some aspects of this proposition can be questioned. It has been argued first of all that predefining the evolution of the CoP is not necessary nor even desirable [48.84, 85]. Then, the ODI (48.13) clearly shows that dynamic feasibility depends on both variations of the angular momentum $\mathbf{L}$, and motion of the COM $\boldsymbol{c}$ with respect to contact points, $\boldsymbol{p}_i$. While trunk rotations mostly involve variations of the angular momentum, a recent analysis showed that this has only a weak influence on the balance of legged robots [48.86]. Dynamic feasibility can be handled much more efficiently by adapting the motion of the COM, which has been the prevailing approach.

The core idea of partitioning the motion of the robot builds on a profound and far-reaching observation: if the robot has a sufficient number of degrees of freedom and sufficient control authority, then every part of the motion that is not involved in the dynamic constraints – everything but angular momentum and motion of the COM with respect to contact points – can be operated more or less independently, and appears therefore to be relatively peripheral to the problem of legged locomotion. It is the same key observation that implicitly drives the templates and anchors approach and the long history of simple biomechanical models of legged locomotion that focus on a few meaningful degrees of freedom, mostly the motion of the COM with respect to contact points, and abstract all the rest [48.87–89]. This idea has been tremendously successful in the legged locomotion research community.

48.4.2 Online Motion Generation: A Model Predictive Control Point of View

If the algorithms for motion generation could work sufficiently fast to be applied online, then the robot

could achieve reactivity and robustness by continuously adapting the motions to the current state of the robot and its environment. But therein lies a problem: how can we make sure that the continuous re-evaluation of online decisions maintains long term viability? One solution comes from the model predictive control (MPC) theory.

The online motion generation schemes that allow most of the great humanoid robots of today and yesterday to walk and run can all be related to MPC theory and can all be seen as variants of the same MPC scheme, although they have rarely been introduced in that way. Most adopt the artificial synergy synthesis approach described in the previous Section, focusing almost exclusively on the motion of the COM with respect to contact points and assuming that the rest of the robot can be operated more or less independently.

Viability, Optimal Control and MPC

When trying to identify viable states in Sect. 48.3.3, the first option was to check if the integral (48.32) could have a finite value. A classical result from optimal control theory is that an optimal control law minimizing this integral would make it decrease with time. As a result, if this integral was finite in the beginning, it would stay finite: if the state of the robot was viable in the beginning, it would stay viable. The problem is, the dynamics of legged robots is too complex to be able to compute such an optimal control law in the general case.

MPC is one way to obtain a computable approximation to this optimal control law. The fundamental idea is to introduce a *terminal constraint* [48.90], imposing that the integral (48.32) is equal to zero after a finite length of time. Introducing such an artificial constraint naturally yields a suboptimal control law, but this allows considering the integral (48.32) over only a finite length of time while preserving viability. Note that enforcing viability in finite time means in this case enforcing capturability: here, the terminal constraint is a capturability constraint.

Many MPC variants have been proposed [48.91], but two extremes are going to be significant in the following. One extreme is to observe that minimizing an integral of the form (48.32) is not necessary to obtain a viable behavior: the terminal, capturability constraint can be sufficient in this respect [48.92]. In this case, however, it is not possible to continuously re-evaluate and adapt the motion to the state of the system: this must be done at carefully chosen moments, what limited the reactivity of the robots. The other extreme is to observe that the terminal constraint is not absolutely necessary [48.93]: simply minimizing a truncated version of the integral (48.32) over a sufficiently long but finite length of time can still lead to an integral decreasing with time, ensuring viability. In this case, it is possible to continuously re-evaluate and adapt the motion to the state of the system without any problem.

Let us see now how all this theory is applied to the online generation of walking and running motions. In all the following examples, the robot is supposed to walk on a flat horizontal ground.

Predefined Footsteps and Capturability Constraint

Let us have a look first at motion generation schemes implementing the original ZMP approach, with predefined footsteps and a predefined CoP, and considering a capturability constraint to ensure viability. Let us begin with the walking motion generation scheme implemented in the long series of Waseda University humanoid robots [48.94], which considers a four point mass model with predefined motion except for the horizontal motion of the waist and trunk masses which is assigned to follow a reference trajectory for the CoP. An iterative procedure based on fast fourier transforms is used then to solve the dynamics

$$\frac{\sum m_i(\ddot{c}_i^z + g^z)\boldsymbol{c}_i^{x,y} - m_i\, c_i^z\, \ddot{\boldsymbol{c}}_i^{x,y}}{\sum m_i(\ddot{c}_i^z + g^z)} \longrightarrow \boldsymbol{z}_{\mathrm{ref}}^{x,y} \,. \tag{48.39}$$

In order to execute this scheme online, a capturability constraint is introduced, imposing that the robot is always able to stop within two steps [48.95]. Unfortunately, details about the choice of the reference CoP and the exact terminal constraint are not disclosed.

The walking motion generation scheme implemented in the Munich University Johnny robot [48.96] considers only a three point mass model with predefined motion, except for the horizontal motion of the main mass in the trunk which is assigned to follow a piecewise linear reference trajectory for the CoP. One degree of freedom is left in this reference trajectory in order to impose a terminal constraint on the position of the COM at the end of the next two steps

$$\boldsymbol{c}^{x,y} = \boldsymbol{c}_{\mathrm{ref}}^{x,y} \,. \tag{48.40}$$

This constraint appears to be incomplete in imposing capturability, what would require to consider also the velocity of the COM (Sect. 48.3.3).

The walking motion generation scheme implemented in the Honda Asimo robot [48.58] is very similar, three point masses and a piecewise linear reference for the CoP with one degree of freedom left to satisfy the terminal constraint. The difference lies in the terminal constraint which is a true capturability constraint, imposing cyclicity of the motion through the capture point/XCOM/divergent component of the dy-

namics at the end of the next step

$$\boldsymbol{\xi}^{x,y} = \boldsymbol{\xi}^{x,y}_{\text{ref}} \,. \tag{48.41}$$

The walking motion generation scheme implemented in the Tokyo University H7 robot [48.97] considers the whole dynamics (48.13) of the robot. The whole motion of the robot is predefined, except the horizontal motion of the COM which is assigned to follow a reference trajectory for the CoP, set in the middle of the contact points. An iterative procedure is used then to solve the dynamics

$$\boldsymbol{c}^{x,y} - \frac{m\,c^z\,\ddot{\boldsymbol{c}}^{x,y} - \dot{\mathbf{SL}}^{x,y}}{m(\ddot{c}^z + g^z)} \longrightarrow \overline{\boldsymbol{p}_i^{x,y}} \,. \tag{48.42}$$

The terminal constraint (48.40) on the position of the COM (incomplete with respect to capturability) is also considered at the end of the next two steps.

The walking and running motion generation scheme implemented in the Toyota Partner robot [48.98] is exactly the same, except for the terminal constraint which is a true capturability constraint imposing cyclicity of the motion through both the position and velocity of the COM.

The walking and running motion generation scheme implemented in the Sony QRIO robot [48.99] follows a similar design, but considers only a single point mass at a constant height, with its horizontal motion assigned to minimizing the deviation of the CoP from the middle of the contact points

$$\min \int \left\| \boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} - \overline{\boldsymbol{p}_i^{x,y}} \right\|^2 \mathrm{d}t \,, \tag{48.43}$$

while imposing a capturability constraint on both the position and velocity of the COM.

Another variant tested on the Kawada HRP-2 robot [48.100] also considers a single point mass at a constant height, but the CoP follows a piecewise polynomial trajectory

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} = \boldsymbol{z}^{x,y}_{\text{ref}} \,, \tag{48.44}$$

with some degrees of freedom left to satisfy the same capturability constraint as before, through both the position and velocity of the COM. An important characteristic of this scheme is that the piecewise polynomial trajectory of the CoP may fluctuate strongly, threatening to violate the ODI (48.13). An automatic adjustment of the step timings is proposed therefore in order to minimize this risk.

All of these walking and running motion generation schemes try to impose capturability through terminal constraints, but some of them appear to fail properly doing so by only constraining the position of the COM. None of them consider an integral of the form (48.32): the integral (48.43) does not match. We have seen earlier that in this case, it is possible to re-evaluate and adapt the motion at specific instants, what has been done when having to realize new steps, or when changing the walking speed or direction. But to adapt the motion to a perturbation, a specific observer would be required to trigger the adaptation at the correct instant [48.92], and it appears that this option hasn't been investigated: no state feedback has been experimented with these motion generation schemes.

Predefined Footsteps, Without Capturability Constraint

In the standard walking motion generation scheme implemented in the Kawada HRP-2 humanoid robot [48.101], the whole motion of the robot is predefined as before, except the horizontal motion of the COM which is assigned this time to minimize the weighted integral

$$\min \int \|\dddot{\boldsymbol{c}}^{x,y}\|^2 + \beta \left\| \boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} + \frac{\dot{\mathbf{SL}}^{x,y}}{mg^z} - \overline{\boldsymbol{p}_i^{x,y}} \right\|^2 \mathrm{d}t \tag{48.45}$$

of the norm of the third derivative of the motion of the COM and the deviation of the CoP from a reference in the middle of the contact points. This is clearly an integral of the form (48.32), and we have seen that viability can be ensured in this case if we consider a truncated version of this integral over a sufficiently long but finite length of time, typically the next two steps, without the need to impose any terminal constraint. We have also seen that it is possible in this case to continuously re-evaluate and adapt the motion to the state of the system, a clear improvement over the previous approaches based solely on terminal constraints. This has been validated experimentally in various situations, effectively adapting the walking motion to perturbations [48.102].

An interesting variant [48.103] introduces variations of the angular momentum L as an additional variable to minimize the combined integral

$$\min \int \|\dddot{\boldsymbol{c}}^{x,y}\|^2 + \beta \left\| \boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} + \frac{\dot{\mathbf{SL}}^{x,y}}{mg^z} - \overline{\boldsymbol{p}_i^{x,y}} \right\|^2 + \gamma \|\boldsymbol{L}^{x,y}\|^2 \mathrm{d}t \,, \tag{48.46}$$

showing an improvement in the tracking of the generated motion (and the same closed loop robustness to small perturbations).

A problem however with these approaches is that simply minimizing the deviation of the CoP from the middle of the contact points doesn't preclude it from fluctuating, so the ODI (48.13) could be violated, especially in case of perturbations. This is monitored in [48.102] to trigger a change of footstep if necessary, but without any clear guarantee that this change is appropriate. For this reason, it has been proposed in [48.85] to impose the ODI (48.13) as a strict constraint, considering a single point mass model with a constant height

$$\boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} \in \text{conv}\left\{\boldsymbol{p}_i^{x,y}\right\} , \tag{48.47}$$

and simply minimize the integral

$$\min \int \|\ddot{\boldsymbol{c}}^{x,y}\|^2 \mathrm{d}t \tag{48.48}$$

over a sufficiently long but finite length of time to ensure viability. This is the walking motion generation scheme implemented in the Aldebaran Nao robot [48.104].

A variant tested successfully on the DLR (Deutsches Zentrum für Luft- und Raumfahrt) biped robot [48.105] considers minimizing the deviation from a reference trajectory for the XCOM together with the derivative of the CoP

$$\min \int \left\|\boldsymbol{\xi}^{x,y} - \boldsymbol{\xi}_{\text{ref}}^{x,y}\right\|^2 + \beta\|\dot{\boldsymbol{z}}^{x,y}\|^2 \mathrm{d}t \, . \tag{48.49}$$

An additional terminal constraint was considered but not tested.

A last proposition, tested only in simple simulations, not with a real robot [48.106], considers a single point mass which is assigned to minimize the deviation of the leg length l from a given reference, and the deviation of the CoP from the middle of the contact points over the next two steps

$$\min \int (l - l_{\text{ref}})^2 + \beta \left\| \boldsymbol{c}^{x,y} - \frac{c^z\, \ddot{\boldsymbol{c}}^{x,y}}{\ddot{c}^z + g^z} - \overline{\boldsymbol{p}_i^{x,y}} \right\|^2 \mathrm{d}t \, . \tag{48.50}$$

What makes this proposition particularly stimulating is that this is not an integral of the form (48.32), that can ensure viability, and no terminal constraint is there to impose capturability, so there is no direct relation with the viability/capturability analysis developed in Sect. 48.3.3, on which all the previous schemes were based. What this minimization only imposes is the ability to realize two more steps in the future, with legs approximately at a nominal length: the situation in the end of these two steps is not controlled (no terminal constraint), so the robot might very well be falling afterwards. However, in the presented simulations, it appears that maintaining this ability to make two more steps in the future is a sufficient condition for generating stable walking and running motions. The conclusion is that terminal constraints or integrals of the form (48.32) are in fact not mandatory and could be relaxed altogether.

In all of these approaches, the footsteps were predefined and kept fixed, which is obviously a strong limitation on the robot's capacity to adapt to a changing environment or to strong perturbations.

Adaptive Footsteps

In fact, adapting foot placement is straightforward, and has already been validated on a Kawada HRP-2 robot [48.107]: the only required change is to consider foot placement as a decision variable, used in addition to the horizontal motion of the COM, in order to both satisfy the ODI (48.47) and minimize the integral

$$\min \int \left\|\dot{\boldsymbol{c}}^{x,y} - \dot{\boldsymbol{c}}_{\text{ref}}^{x,y}\right\|^2 \mathrm{d}t \tag{48.51}$$

over a sufficiently long length of time to ensure viability (since this is clearly an integral of the form (48.32)), and have the COM follow on top of that the reference velocity $\dot{\boldsymbol{c}}_{\text{ref}}^{x,y}$. Now, since the footstep placement is decided online, geometric feasibility needs to be checked online as well. A simple but effective option is to consider a polygonal approximation of the reachable volume of the COM with respect to each foot on the ground [48.108].

Yet, the timing of the steps was still predefined, what has a huge influence on the reactivity of legged robots [48.60]. In order to adapt it as well, it has been proposed in [48.109] to minimize the combined integral

$$\min \int \left\|\dot{\boldsymbol{c}}^{x,y} - \dot{\boldsymbol{c}}_{\text{ref}}^{x,y}\right\|^2 + \|\ddot{\boldsymbol{f}}^{x,y}\|^2 \mathrm{d}t \, , \tag{48.52}$$

with $\ddot{\boldsymbol{f}}^{x,y}$ the horizontal acceleration of the feet. But this approach has been tested only in simple simulations so far. Moreover, the underlying optimization problem becomes nonlinear, what is significantly more involved to solve.

Another approach, quite unique, starts with a singular linear quadratic regulator (LQR) design [48.110], considering the classical single point mass at a constant height, assigned to minimize the deviation of the CoP from a combination of a reference CoP and the XCOM

$$\min \int \left\| \boldsymbol{c}^{x,y} - \frac{c^z}{g^z}\ddot{\boldsymbol{c}}^{x,y} + \alpha \boldsymbol{z}_{\text{ref}}^{x,y} - (1+\alpha)\boldsymbol{\xi}^{x,y} \right\|^2 \mathrm{d}t \tag{48.53}$$

for some $\alpha > 0$. An interesting feature of this singular LQR design is that it can be solved analytically. And an

interesting property of this singular objective function is that it can be reduced to zero, in which case we have

$$c^{x,y} - \frac{c^z}{g^z}\ddot{c}^{x,y} = (1+\alpha)\xi^{x,y} - \alpha z_{\text{ref}}^{x,y} , \tag{48.54}$$

what can be combined with the dynamics (48.35) of the XCOM to obtain a stable first order dynamics

$$\dot{\xi}^{x,y} = \alpha\omega(z_{\text{ref}}^{x,y} - \xi^{x,y}) , \tag{48.55}$$

according to which the XCOM ξ is going to converge to $z_{\text{ref}}^{x,y}$, and the CoP z as well according to (48.54). But in a very unusual twist, this LQR design is not used as is, and is inverted analytically to find under which condition does a piecewise constant trajectory of the CoP generate a nondiverging motion of the COM. Unsurprisingly, this nondiverging condition ends up being a terminal constraint on the capture point/XCOM/diverging part of the dynamics, with trajectories corresponding in the end exactly to those found in [48.59]. But here, this terminal constraint is used in the end to decide online the footstep placement that will ensure viability, what is eventually validated experimentally with very strong perturbations.

All Variants of a Common Design

The viability analysis of Sect. 48.3.3 attested the crucial importance of anticipation to avoid falling. We observe now that the walking and running motion generation schemes that power most of the greatest humanoid robots of today and yesterday, the Honda Asimo, the Toyota Partner, the Sony Qrio, the Aldebaran Nao, the Kawada HRP-2, the Tokyo University H7, the Munich University Johnny, the DLR biped, the long series of Waseda University humanoids and many others, are all structured around this necessity to anticipate, at least a couple steps ahead of time. Viability is secured then in different ways, always related to MPC theory, through capturability or through minimizing an integral of the form (48.32). In all cases, adopting the artificial synergy synthesis approach, focusing on the motion of the COM of the robot with respect to contact points, considering that the rest of the motion can be handled more or less independently, is the key to enable efficient computations. All these motion generation schemes appear then to share the same general design: MPC of the COM of the robot with respect to contact points. The rest is details.

The compelling consequence of identifying this common design, is revealing the possibility to cross-breed all these approaches, retaining the best features of each: more precise multiple mass models, for generating both walking and running motions, with adaptive footstep placement and adaptive timing, sensor feedback, all in order to obtain in the end the ultimate robust and versatile online motion generation scheme. These are the next obvious steps.

Missing in this gallery of great legged robots, the Boston Dynamics biped and quadruped robots. They are without a doubt the legged robots exhibiting the most impressive dynamic motions today, robust and versatile. Exact details about their control algorithms are scarce, but various clues suggest that they might share the same design.

48.4.3 Motion in Constrained Environments

We have seen so far how to generate dynamic walking and running motions by considering the dynamic feasibility constraint (48.13) independently from any other concern. But on truly rough terrain or in cluttered spaces, kinematic constraints on the motion of the robot cannot be disregarded and can further complicate the problem. More precisely, kinematic constraints generally make the reachable space of the robot nonconvex, so that deciding actions and motions with only a local view on the situation can quickly get the robot stuck in local loops or dead-ends. One way to decide actions and motions with a global view on the situation is with planning techniques, as described in Chap. 7 on motion planning and Chap. 47 on motion planning and obstacle avoidance. In the case of legged robots, three classes of problems of increasing complexity have been considered: dynamic manipulation without locomotion, locomotion on a flat ground with obstacles, and locomotion on rough terrain with complex contact transitions.

Dynamic Manipulation Without Locomotion

In the case of manipulation tasks without locomotion, the only difference of legged robots with respect to more traditional manipulator robots is the dynamic feasibility constraint (48.13). Therefore, a mere application of the standard planning techniques described in Chaps. 7 and 47 including this feasibility constraint has proved to be fully sufficient. But these techniques are strongly biased towards quasi-static motions. As a result, a sequence of statically stable postures is computed first, and the motion is smoothed and accelerated afterwards, finally taking into account dynamic feasibility [48.111]. Much more involved are the problems involving locomotion, which attracted therefore much more attention.

Legged Locomotion on a Flat Ground with Obstacles

If we consider that the problem of generating dynamic legged locomotion is properly solved thanks to the dif-

ferent methods described earlier in this section, we can try to abstract its details and plan the navigation of the upper body of a legged robot between obstacles as for any other mobile robot, with the same planning techniques, regardless of the exact locomotion method. It has been shown indeed that legged robots present a form of *small-space controllability*, which means that their upper-body can follow any given path with any given accuracy [48.112], although this can require impractically quick steps. A different option to follow exactly any given path with the upper-body is to cross legs, as if walking on a balance beam, and some robots have been specifically designed to be able to realize such feats [48.113], but this can require impractically slow steps. Ruling out impractically slow or impractically quick steps, the legged locomotion may finally differ from the planned path and fail to avoid obstacles. In this case, iterative replanning techniques have been proposed [48.114].

In this approach, however, it is assumed that finally finding proper footholds to follow the path planned for the upper body is not a problem. Yet, the appeal of legged robots is to provide improved mobility on difficult terrain, situations where finding footholds is actually a problem. A solution then is to check the availability of proper footholds when planning the upper body motion [48.115], an approach recently refined to show how to obtain in the general case an exact equivalence between planning a continuous path and planning a sequence of footholds on a flat ground [48.116, 117].

A different approach is to plan directly the footholds with respect to obstacles on the ground, and generate afterwards the corresponding dynamic legged locomotion [48.118]. This has been applied successfully to long distance navigation with a tiered strategy [48.119], with computations fast enough to run online in changing environments [48.120], taking into account deterministically moving obstacles [48.121], on mildly rough terrain [48.122–124], using on-board sensing [48.125]. Obstacles above the ground can be considered then with a hybrid bounding box or more refined swept volumes [48.126, 127], or one can approximate the free-space with convex segmentation [48.128].

Legged Locomotion with Complex Contact Transitions

On truly rough terrain or in cluttered environments, legged locomotion may require much more complex contact situations and contact transitions than in standard walking or running, involving not only contacts between the feet and the ground, but all potential contact surfaces on the robot and on the environment. The variability of these situations is beyond comparison with the previous case on flat ground, and chances of precomputations or simplifications are strongly lacking in respect. Notably, the kinematic and dynamic constraints cannot be tackled independently, and having to consider them simultaneously results in an overwhelming computational problem. As a result, a major effort has been to find a proper ordering of the different subproblems in order to obtain a tractable problem in the end, finally opting for planning contacts before motion [48.129–131]. Complex real-life use-cases have been solved this way [48.132–134], but the computational requirements still limit our ability to achieve the versatility of locomotion possible on flat ground. Online adaptation to changing environments has yet to be convincingly demonstrated and remains an active area of research.

As noted before, the underlying planning techniques manifest a strong bias towards quasi-static motions: some post-processing is necessary to obtain a truly dynamic motion in the end. The same two options as in Sect. 48.4.1 have been proposed, standard trajectory optimization [48.135], or following the artificial synergy synthesis approach and computing first the motion of the COM of the robot with respect to the different contact points, independently from the rest of the motion of the robot, which is computed only afterwards [48.21]. But the strong interdependency between kinematic and dynamic constraints defeats the core assumption of the artificial synergy synthesis approach in this case: dynamic feasibility can't always be handled independently from the rest of the motion of the robot, so there is a significant risk that this method fails finding in the end a feasible motion even if one exists.

Further discussion of all these problems in the specific case of humanoid robots can be found in Chap. 67.5 on whole-body activities of humanoid robots, or in [48.136]. There is also a rich literature on gait selection for many-legged robots, starting as early as [48.137] but this is still an active area of research [48.138].

48.4.4 Motion Generation with Limited Computing Resources

The walking and running motion generation schemes presented so far in this chapter all require significant computation, even when they consider only simplified dynamical models. Early legged robots, which had very limited computing resources, had to rely on much less demanding approaches. One option is to combine simple explicit rules, each one focusing on a different part of the overall motion. Another option is to try to mimic biological hypotheses on how animals generate and control their motion, what ended up in fact with somehow similar propositions.

Combining Simple Rules

Thanks to simplifying symmetries in biped or quadruped running gaits such as trot, pace and bound, the whole family of robots hopping in the MIT LegLab throughout the 1980s on one, two or four legs, in 2-D (two-dimensional) or 3-D (three-dimensional), could rely on the same simple control design [48.5]. The general idea is to apply simple control laws independently to the different parts of the overall locomotion task. The vertical oscillations were controlled by open-loop vertical thrusts, simply stabilized by mechanical energy losses. Since the angular momentum is conserved during flight phases, the body attitude was controlled only during stance phases, with standard proportional–derivative (PD) control. The key to maintaining long-term balance, and controlling the global motion was foot placement. An elementary analysis of the motion of the COM during the stance phase revealed the role of the so-called *neutral* foot position, which would make the stance phase perfectly symmetric. It is then just a matter of linear corrections of the foot position with respect to this neutral position to make the stance phase asymmetric in order to control the robot's speed and balance. The direct combination of these utterly simple control laws resulted in impressively robust and versatile locomotion.

The first biped robot that walked dynamically relied on an equally simple approach [48.4]. This robot emulated walking on stilts, with contact surfaces between the feet and the ground reduced to single points, so $z = p$. With a single, fixed contact point during single support phases (the double support phases were supposed negligible), the ODI (48.29) can be solved analytically: for $t \geq t_0$,

$$\begin{pmatrix} c^{x,y}(t) - p^{x,y}(t_0) \\ \dot{c}^{x,y}(t) \end{pmatrix} = A(t) \begin{pmatrix} c^{x,y}(t_0) - p^{x,y}(t_0) \\ \dot{c}^{x,y}(t_0) \end{pmatrix} \tag{48.56}$$

with

$$A(t) = \begin{pmatrix} \cosh \omega(t-t_0) & \omega^{-1} \sinh \omega(t-t_0) \\ \omega \sinh \omega(t-t_0) & \cosh \omega(t-t_0) \end{pmatrix},$$
$$\omega = \sqrt{\frac{g^z}{c^z}}\,. \tag{48.57}$$

If the next single support phase starts at time t_1 with a foot position $p^{x,y}(t_1)$, we end up having

$$\begin{pmatrix} c^{x,y}(t_1) - p^{x,y}(t_1) \\ \dot{c}^{x,y}(t_1) \end{pmatrix} = A(t_1) \begin{pmatrix} c^{x,y}(t_0) - p^{x,y}(t_0) \\ \dot{c}^{x,y}(t_0) \end{pmatrix} + \begin{pmatrix} p^{x,y}(t_0) - p^{x,y}(t_1) \\ 0 \end{pmatrix}, \tag{48.58}$$

which describes the motion of the COM $c^{x,y}$ with respect to the contact points $p^{x,y}$, from one step, starting at time t_k, to the next, starting at time t_{k+1}. This discrete time dynamical system can be controlled then with the foot placement $p^{x,y}(t_k)$. Since it is linear, standard pole placement can be applied, yielding a standard PD control of foot placement. This simple approach has been successfully applied on more complex biped robots [48.139, 140].

The drawback of combining independent control laws is to lack coordination yielding potentially harsh, unrefined motion, but this approach demonstrates that stable locomotion is possible even with very limited control resources. Of primary importance here are foot placement and upper body attitude; the rest of the motion appears to be secondary, at least with respect to balance control. Pushing this observation further, it has been proposed, and validated in simulation, that a large variety of walking patterns could be used, and stabilized only with upper body attitude and foot placement control, both with simple PD control laws [48.141]. It is not surprising then that hand-designed motion patterns can be used successfully on robots as complex as the Korea Advanced Institute of Science and Technology (KAIST) Hubo humanoid robot [48.142], although this may require delicate fine tuning.

Reducing even further the control requirements, the mechanics of the robots could be tuned so that passive motions would automatically land the feet on the appropriate locations with respect to stability, ending up with perfectly passive stable dynamic locomotion. This idea will be discussed more thoroughly in Sect. 48.6.

Biomimetic Motion Generation

Current theories on locomotion control in animals include central pattern generators (CPGs) and cascades of reflex motions, which interact and combine to generate the final motion. CPGs are tunable oscillators, generating synchronized quasi-cyclic motion patterns in response to simple control signals such as locomotion speed or turning angle [48.143]. In order to introduce such CPGs in legged robots, standard oscillators such as the Van der Pol equation or the Hopf oscillator have been proposed [48.144, 145], or more biologically inspired neural oscillators [48.146] (ironically, *central* pattern generators often happen to be *decentralized*, composed of a collection of synchronized oscillators [48.143, 145, 147]).

But CPGs are intrinsically open-loop motion generators, so to stabilize and adapt the motion of the robot to their environment, limited feedback loops have been introduced, similar to the simple control laws discussed earlier in this Section, focusing on upper body attitude and foot placement [48.145, 147, 148]. Contin-

uing with the biological analogy, these simple feedback loops have been called reflexes. Following ideas discussed more thoroughly in Chap. 13 on behavior-based systems, the *subsumption architecture* has been proposed to build much more complex networks of reactive behaviors or reflexes [48.149], and has allowed a hexapod robot to walk successfully on mildly rough terrain without any careful planning of the motion [48.150].

Fundamentally the Same Decoupling Approach

Being inspired by simple mechanical analyses or by animals, these approaches based on combinations of simple rules demonstrate that stable walking can be realized without much computation, at least in simple situations. More complex situations may require more refined motions and therefore more refined motion generation schemes, such as the ones discussed earlier in this chapter. But it is striking to observe again the same focus here on the motion of the COM of the robot with respect to the contact points, when controlling either the upper body attitude or the foot placement (or both), regardless of the rest of the motion, which can be generated by CPGs, inverse kinematics or any other approach, simple or complex. This is fundamentally the same decoupling as proposed earlier with the artificial synergy synthesis approach.

48.5 Motion and Force Control

Properly executing the walking and running motions computed in the previous section requires precise control of the motion of the robot and of the contact forces with the environment, applying techniques from Chapters 8 and 9 on motion and force control. But since legged robots are often complex mechanical systems with many degrees of freedom and some form of redundancy, techniques from Chap. 10 on redundant manipulators also need to be applied frequently. And in case of contact with multiple surfaces, problems of contact force distribution appear, requiring techniques similar to those found in Chap. 38 on grasping. And in the opposite case, during flight phases, when the robot is not at all in contact with its environment, its dynamics appear to share important properties with free-floating space robots, requiring control techniques similar to those found in Chap. 55 on space robotics. All these techniques (and many more) are necessary to make legged robots walk and run efficiently. The purpose of this Section is to see how they connect and are implemented in the case of legged robots.

Whole-Body Motion

Humanoid and other legged robots are often complex mechanical systems with a large number of degrees of freedom in order to accomplish the various kinematic and dynamic tasks necessary for simultaneously achieving locomotion, perception and manipulation of the environment, interaction with humans, etc. Making use of all available degrees of freedom, and not only those in the legs, to achieve multiple simultaneous objectives, and not only locomotion, is usually called *whole-body motion control* (Chap. 67.5). We have seen previously that locomotion mostly involves the motion of the COM $\boldsymbol{c}$ with respect to points $\boldsymbol{p}_i$ of contact with the environment. Other objectives involve end-effector motion for manipulation, gaze control for perception, motion of specific parts of the robot for obstacle avoidance, etc. In all of these cases, what needs to be controlled is Cartesian motion of different parts of the robot, rather than joint motion. Several control schemes have been proposed to do this, standard inverse kinematics followed by joint space control [48.139], virtual model control [48.151], the task function approach [48.152, 153], operational space control [48.154].

Interestingly, the last two options allow specifying different priorities to the different objectives assigned to the robot, so in case these objectives are not feasible jointly, the robot tries first to achieve the objectives with higher priority. In the case of legged robots, maintaining balance and avoiding falls appear to be of primary importance for the safety of both the robot and its environment, so being able to consider this objective with a higher priority can be crucial in this regard. These control schemes allow moreover some form of decoupling between the different objectives, further contributing to the artificial synergy synthesis approach introduced earlier in the previous Section. We are going therefore to continue focusing here exclusively on the locomotion objective, since the other ones can be considered more or less independently and are not specific to legged robots.

Flight Phases

We have seen earlier in this chapter that during flight phases, when a legged robot is not in contact with its environment, the motion of the COM invariably follows a standard falling motion: there is no possibility to make it move in any other way. But we have seen that even though the angular momentum $\mathbf{L}$ is constant during flight phases, the joint configuration $\hat{\boldsymbol{q}}$ and the global orientation $\boldsymbol{\theta}_0$ of the robot can still be driven

concurrently together to any desired value thanks to the nonholonomy of the angular momentum (48.5). However, a famous theorem due to *Brockett* [48.155] implies that continuous time-invariant feedback control laws such as those proposed in [48.156, 157] are not able to do so. The solution is to use discontinuous or time-varying control laws [48.158], an approach well established in space robotics [48.159, 160].

In the case of running motions, it has been proposed to specifically adapt leg trajectories during flight phases in order to control both the joint configuration and the global orientation at the end of the flight phases [48.161]. It has been measured indeed that a simple difference in the height of a single step can result in 1 or 2 degrees of difference in the global orientation of the robot [48.8]. Albeit small, this effect is far from negligible. More thorough discussions of nonholonomic constraints and the impact of Brockett's theorem can be found in Chap. 49 on the modeling and control of wheeled mobile robots or in [48.158].

Angular Momentum

Either considering the ODI (48.13) when walking on a flat ground, or the Newton and Euler equations (48.3) and (48.4) in the general case, the dynamic feasibility of legged motions appears to be directly and entirely related to the motion of the COM $\boldsymbol{c}$, and the angular momentum, $\mathbf{L}$. It has been naturally proposed therefore to control specifically those two quantities concurrently [48.162–164]. Of course, since COM motion and angular momentum are directly related to contact forces $\boldsymbol{f}_i$ (or equivalently to the CoP, z), controlling the former is absolutely equivalent to controlling the latter. The question however is whether the angular momentum should be specifically controlled to some given value and what that value should be [48.163, 164].

Of course, since the angular momentum is directly related to contact forces, it should comply with the unilaterality and Coulomb friction limitations on contact forces (48.11) and (48.17). As a matter of fact, the regulation of angular momentum has been discussed only in quasi-static situations [48.162–164], where the reference value is clearly zero, which generally complies well with these limitations. And in the only case where walking was considered, the horizontal angular momentum $\mathbf{L}^{x,y}$, that appears in the ODI (48.13) when walking on a flat horizontal ground, was explicitly not controlled [48.162]. It appears, in fact, that during walking motions, the question of which value the angular momentum should have is very intricate, because of its nonholonomy, but it is definitely not zero [48.8, 103].

Even in quasi-static situations, both joint configuration and global orientation can be controlled independently from the angular momentum; the connection between these different values is very subtle. As a result, controlling the angular momentum does not induce a control of either joint configuration or global orientation (due to its nonholonomy). The opposite is true however: controlling joint configuration and global orientation does induce a control of the angular momentum as a derived quantity. It has been argued furthermore that common whole body motion objectives such as manipulation or interaction with humans are more directly related to joint configuration and global orientation than to angular momentum [48.165]. As a matter of fact, disregarding the exact variations of the angular momentum has been the most frequent option so far [48.139, 151–154, 165]. In the end, it is not clear if specifically controlling the angular momentum can help controlling legged robots: this is still an open question.

COM Motion Control

Putting the angular momentum aside, and considering only the simple linear dynamics (48.29) when walking on a flat ground, and more precisely its decomposition (48.34) and (48.35), we have seen that every state satisfying the condition (48.36), such that the capture point $\boldsymbol{\xi}$ is in the support polygon, is capturable and can therefore be stabilized. If the capture point is strictly inside the support polygon, the COM can even be stabilized to any reference position $\boldsymbol{c}_{\text{ref}}$ above the support polygon, with a simple linear feedback of the capture point

$$z^{x,y} = c^{x,y}_{\text{ref}} + k(\xi^{x,y} - c^{x,y}_{\text{ref}}) \,, \tag{48.59}$$

where the feedback gain $k > 1$ must be tuned in order to keep the CoP z within the support polygon, ensuring dynamic feasibility of the motion. Indeed, using this feedback with the dynamics (48.35) yields a stable closed loop dynamics

$$\dot{\xi}^{x,y} = \omega(k-1)(c^{x,y}_{\text{ref}} - \xi^{x,y}) \,, \tag{48.60}$$

according to which the capture point $\boldsymbol{\xi}$ will converge to the reference position $\boldsymbol{c}_{\text{ref}}$, and the COM $\boldsymbol{c}$ will converge to it as a result of the stable dynamics (48.34). It has been shown that among all proportional-derivative (PD) feedback of the motion of the COM

$$z^{x,y} = c^{x,y}_{\text{ref}} + k(c^{x,y} + \lambda \dot{c}^{x,y} - c^{x,y}_{\text{ref}}) \,, \tag{48.61}$$

the feedback (48.59) of the capture point, with a choice of $\lambda = \omega^{-1}$, is the only one that will successfully stabilize all capturable states [48.166]. It has been successfully applied to walking motions on the DLR biped robot [48.59].

Now, if there is a time-varying perturbation or error $\boldsymbol{\varepsilon}$ in our dynamics, such that

$$c^{x,y} - \frac{c^z}{g^z}\ddot{c}^{x,y} = z^{x,y} + \varepsilon \,, \tag{48.62}$$

the PD feedback (48.61) can be shown to be robustly stable (input-to-state stable, to be precise) with respect to $\boldsymbol{\varepsilon}$, but with $\lambda < \omega^{-1}$ and $k > \frac{1}{\lambda^2\omega^2}$ [48.167]. As discussed above, with this choice of λ, not all capturable states will be stabilized successfully, even if there is no perturbation. This is unfortunately a usual trade-off in robust control.

Force Feedback

Motion control of legged robots without contact force sensing and feedback has been shown to be possible for simple quasi-static balancing tasks, using passivity based control [48.168]. However, force sensing and feedback is commonly used when realizing more dynamic motions such as walking or running. One of the first problems to overcome in this case is the compliance of both the robot and its environment, which easily generates destabilizing oscillations [48.169–172]. Contact force control schemes of various complexity have been proposed, but always with the goal of damping these undesired oscillations. The exact implementation of these force control schemes generally relies on a stiff lower-level joint motion control in order to enhance disturbance rejection, leading to admittance control implementations [48.139, 169–173]. More details on such control schemes can be found in Chap. 9 on force control.

Now, these damping control schemes inevitably introduce some delay in the realization of the desired contact forces. One option is to show that the COM motion control scheme described earlier, which entirely relies on contact forces, is robust to such delay [48.59]. Another option is to take this delay into account with a simple first-order dynamics of the CoP

$$\dot{z} = \omega_z(z_d - z) \,, \tag{48.63}$$

and adapt the previous COM motion control scheme accordingly [48.174, 175]

$$z_d^{x,y} = c_{\text{ref}}^{x,y} + k(\xi^{x,y} - c_{\text{ref}}^{x,y}) + k'(z^{x,y} - c_{\text{ref}}^{x,y}) \,. \tag{48.64}$$

This introduces some force feedback directly in the COM motion control through the measurement of the CoP z, what can help improve the reactivity of the controller to perturbations.

Contact Forces Distribution

In case of contact with multiple surfaces, computing contact forces $\boldsymbol{f}_i$, that allow realizing the desired motion control scheme through the Newton and Euler equations (48.3) and (48.4), and yet comply with both the unilaterality condition (48.11) and the Coulomb friction model (48.17), can be slightly more involved. The standard approach is through optimization, and more precisely quadratic programming (QP) with linear constraints, using a faceted, linear approximation of the Coulomb friction model (48.17)

$$\boldsymbol{A}\boldsymbol{f}_i \leq \boldsymbol{b} \,. \tag{48.65}$$

It has been proposed then to minimize when possible the norm of the joint torques $\boldsymbol{u}$ in order to reduce energy consumption [48.152],

$$\min_{\boldsymbol{f}_i} \ \|\boldsymbol{u}\|^2 \,, \tag{48.66}$$

or to balance the contact forces between the different contact points, either by minimizing the sum of their norms [48.165],

$$\min_{\boldsymbol{f}_i} \ \sum_i \|\boldsymbol{f}_i\|^2 \,, \tag{48.67}$$

or the norm of their difference [48.176],

$$\min_{\boldsymbol{f}_i} \ \|\boldsymbol{f}_{\text{left}} - \boldsymbol{f}_{\text{right}}\|^2 \,, \tag{48.68}$$

or minimizing the tangential forces in order to reduce the risk of slipping [48.177],

$$\min_{\boldsymbol{f}_i} \ \sum_i \|\boldsymbol{f}_i^{x,y}\|^2 \,. \tag{48.69}$$

More general QP formulations have been proposed [48.178, 179], including hierarchies of kinematic and dynamic tasks and constraints [48.153].

But contact forces obtained in these ways are not explicitly kept away from the edges of the friction cone and unilaterality condition, so that the slightest perturbation may quickly lead to sliding or tipping, what can lead to serious issues. The simplest way to steadily improve the robustness of the robot to such perturbations is to introduce a security margin w

$$\boldsymbol{A}\boldsymbol{f}_i \leq \boldsymbol{b} - w\boldsymbol{1} \,, \tag{48.70}$$

where $\boldsymbol{1}$ is a vector of ones, but there is no obvious way to decide which value would be appropriate. One option then is to try to maximize it with a linear program (LP)

$$\max_{w,\boldsymbol{f}_i} \ w \,, \tag{48.71}$$

improving as much as possible the robustness of the robot to perturbations by selecting contact forces $\boldsymbol{f}_i$ the farthest possible from the edges of the (faceted) friction cone and unilaterality condition. Note that the maximal value w^* obtained in this way is the *residual radius* discussed in [48.19].

48.6 Towards More Efficient Walking

Examples of remarkably efficient locomotion are ubiquitous in nature. From sprinting cheetahs to leaping sifakas, the variety and beauty of animal motion has inspired roboticists for decades. Yet some of the most impressive examples of humanoid walking, e.g., Honda's Asimo or Boston Dynamics' Atlas, have employed high-impedance joint control to achieve stable locomotion. As a result, the fluidity of their motions is far from that of their biological counterpart. Indeed, these robots expend an order of magnitude more energy than a typical human while walking [48.180].

In contrast, the passive dynamic walking machines of McGeer and others [48.6, 181] have been able to achieve surprisingly human-like gaits without any actuation, relying solely upon their carefully designed body dynamics. These machines moved down sloped terrain, relying upon the conversion of potential to kinetic energy to balance the losses introduced by impacts and friction. Subsequent examples of minimally-actuated bipeds capable of walking under their own power on flat terrain made the first steps toward realizing efficient walking machines [48.180, 182].

The existence of these robots leads to several interesting questions. For example, how can we find limit cycles of underactuated walkers and reason about their stability? Simple models of walking systems have been used to try to answer these questions for over two decades. Such analyses typically rely upon linearization of the model dynamics to integrate the equations of motion or numerical simulation of the nonlinear dynamics and analysis via Poincaré maps (Sect. 48.3.2). For example, the compass gait model is one of the simplest walking models, consisting of three point masses – one at the hip and one on each leg – and the legs connected to the hip with a pin joint (Fig. 48.10). Although even this very simple model cannot be analyzed analytically, a variety of stable gaits have been identified using numerical techniques [48.186].

Today there exists a community of researchers working to bridge the gap between stable and versatile machines like Asimo and efficient dynamic walkers. Topics of particular interest include navigating robustly over rough terrain, generating and stabilizing nonperiodic gaits, and scaling up to more complex robot models. New computational tools for generating and stabilizing trajectories for underactuated dynamical systems have been an important result of this work. In this section we highlight the fundamental insights provided by work on passive dynamic walking machines and describe ways in which these ideas have been extended to more general systems.

48.6.1 Gait Generation for Dynamic Walking

The most direct approach to producing efficient walking gaits in underactuated robots is to design controllers that produce actuated trajectories that mimic passive ones. Fundamental to these methods is the idea of restoration of mechanical energy. Since mechanical energy is

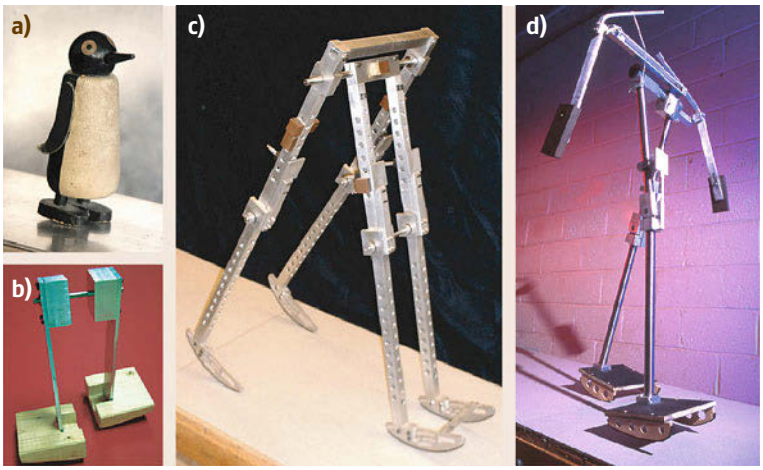


Fig.48.9a–f Passive-dynamic walking machines. **(a)** The Wilson *Walkie* (after [48.183]). **(b)** A slightly improved version (after [48.184]). Both **(c)** and **(d)** walk down a slight ramp with the *comical, awkward, waddling gait of the penguin* (after [48.183]). **(e)** Cornell copy (after [48.185]) of McGeer's capstone design. This four-legged *biped* has two pairs of legs, an inner and outer pair, to prevent falling sideways. **(f)** The Cornell passive biped with arms (photo by Hank Morgan) (after [48.181]). This walker has knees and arms and is perhaps the most human-like passive-dynamic walker to date (photo reproduced with permissions from [48.180])

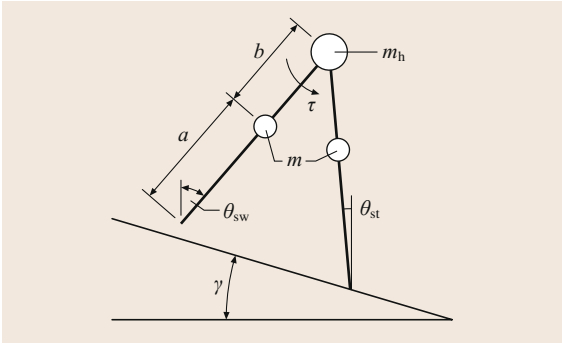


Fig. 48.10 The compass gait

lost at each heel strike, passive dynamic walkers require gravitational force to restore energy to the system before the next step. By choosing control inputs that regulate mechanical energy, for example by simulating gravity, walking on level terrain can be achieved [48.187, 188]. However, such approaches are still largely limited to modest terrains and periodic motions.

Trajectory optimization methods offer the potential to move beyond limit cycle gaits and to more complicated systems without passive stability properties. However, the fact that walking systems must make and break contact with the world complicates the trajectory optimization problem. Simulating rigid-body contact forces continuously using spring and damper models often results in stiff differential equations that can lead to numerical and computational issues. An alternative approach is to capture discontinuities in velocity by modeling collisions as impulsive events using a hybrid system representation (Sect. 48.2.3). For a walking hybrid system, a set of autonomous *guard functions* are defined such that $\phi_i(\boldsymbol{q}) = 0$ implies that body i is in contact and $\phi_i(\boldsymbol{q}) > 0$ implies the opposite. When a body comes into contact, the generalized velocity undergoes an instantaneous change, $\dot{\boldsymbol{q}}^+ = f_\Delta(\boldsymbol{q}, \dot{\boldsymbol{q}}^-)$, where $\dot{\boldsymbol{q}}^-$ and $\dot{\boldsymbol{q}}^+$ are the velocities pre and post-impact, respectively. The *mode* of the system is defined by the set of active contacts, $C = \{i | \phi_i(\boldsymbol{q}) = 0\}$, which in turn defines the set of Jacobian matrices in (48.2).

The hybrid system representation creates several challenges for trajectory optimization methods. The discontinuities due to mode transitions are typically accounted for explicitly using either *direct* or *multiple-shooting* methods [48.189]. Since each mode implies a different dynamics and mode changes are necessary for walking, this leads to the question of how to formulate the dynamics constraints. One solution is to predefine the sequence of mode transitions. For simple models with point feet, this approach can work quite well. However, for more complex models with many contacts, the number of modes grows exponentially, making mode sequence prespecification impractical.

The complementarity-based methods for simulating rigid body systems in contact described in Sect. 48.2.3 suggest another approach. By discretizing the dynamics in time and considering only the integral of contact forces over a time interval, the forward dynamics of a walking system can be greatly simplified. Since many trajectory optimization methods already discretize by evaluating a finite set of points along a trajectory, incorporation of contact forces as optimization variables leads to trajectory optimization as a nonlinear optimization problem with complementarity constraints [48.37].

Many impressive examples of trajectory optimization for walking and running exist in the literature. For example, *Mombaur* [48.78] simultaneously optimized motion and physical parameters for simulated bipeds to achieve open-loop stable running and *Remy* [48.190] applied trajectory optimization to generate efficient running for a quadruped. There are many more examples and it remains an active area of research. The problem of stabilizing trajectories optimized offline remains a challenge, particularly when the execution deviates from the planned mode sequence. Computational requirements still prevent nonlinear trajectory optimization methods from being applied online as an MPC algorithm, but improvements in numerical solvers and computing hardware will likely continue to reduce this gap.

48.6.2 Orbital Trajectory Stabilization and Control

Once an open-loop gait is generated, it must typically be stabilized in order to be executed on a robot. Classical techniques for trajectory stabilization from linear control can work, but a strong theme in stabilization for legged robots has been the idea that one should not enforce the precise timing of the trajectory. Section 48.3.2 discussed the notion of orbital stability and introduced Poincaré maps as a tool for stability analysis. Orbital stabilization, rather than traditional trajectory stabilization, sacrifices little and appears to be more compatible with underactuated robots. These ideas have also played an important role in developing stable dynamic walking and running controllers using *virtual constraints* [48.38]. Virtual constraints are holonomic constraints on monotonic functions of the robot's configuration variables. For example, for a forward periodic gait, it might be reasonable to assume that the angle of the stance leg with respect to the ground θ_{st} is monotonically increasing in time throughout the stance phase. The remaining configuration variables are written as functions of θ_{st}, effectively reparametrizing time, and constraints are imposed that, e.g., enforce sym-

metry in stance and swing leg angles ($\theta_{sw} = -\theta_{st}$) or coordinate arm and leg motion. This leads to outputs

$$\boldsymbol{y} = \begin{bmatrix} h_1(\boldsymbol{q}) \\ \cdots \\ h_d(\boldsymbol{q}) \end{bmatrix}, \tag{48.72}$$

where each $h_i(\boldsymbol{q})$ encodes a virtual constraint. For example, $h(\boldsymbol{q}) = \theta_{sw} + \theta_{st}$ encodes the virtual constraint for swing-stance leg symmetry. By driving the output dynamics asymptotically to 0 using high-gain control, the system can be seen to evolve according to the virtual constraints, yielding a lower dimensional problem for control design and stability analysis. This approach has been used to produce remarkable dynamic walking and running examples in real robots [48.191].

Poincaré maps are defined with respect to a transverse surface at a single point on a periodic orbit, which limits their usefulness for designing stabilizing controllers. The *transverse dynamics*, on the other hand, is defined with respect to a *continuous* family of surfaces $S(t)$, transverse to the orbit [48.45]. In the neighborhood of these coordinates, the dynamics can be written as

$$\dot{\tau} = 1 + f_1(\tau, \boldsymbol{x}_\perp) \tag{48.73}$$

$$\dot{\boldsymbol{x}}_\perp = \boldsymbol{A}(\tau)\boldsymbol{x}_\perp + f_2(\tau, \boldsymbol{x}_\perp)\,, \tag{48.74}$$

where $\tau \in [0, t_f]$ represents the orbit phase, $\boldsymbol{x}_\perp$ are the $2N-1$ coordinates orthogonal to the flow of the system, and $f_1(\cdot), f_2(\cdot)$ are functions containing nonlinear terms. Note that unlike the discrete Poincaré map, these dynamics do not require the trajectory of interest to be a periodic orbit.

By extracting the linear part of the transversal dynamics and incorporating control inputs, we obtain the transverse linearization

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{A}(t)\boldsymbol{z}(t) + \boldsymbol{B}(t)\boldsymbol{u}(t)\,, \tag{48.75}$$

for $t \in [0, t_f]$. Stabilization of this system, and therefore the underlying trajectory, can be approached using standard techniques from linear control theory. For walking systems with impulses, the dynamics can be extended to include a linearized impact map, $\boldsymbol{z}(t_i^+) = \mathbf{C}_i\boldsymbol{z}(t_i^-)$, where $t_i, i \in C$ are times at which impacts occur. Figure 48.11 illustrates the idea of transversal surfaces about a trajectory for a system with impacts. Manchester et al. [48.50] showed that under some mild assumptions, local exponential orbital stability of planned motions on a hybrid nonlinear system can be achieved using a receding horizon control with the transverse linear dynamics.

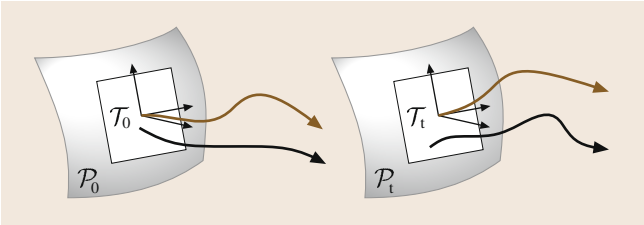


Fig. 48.11 A simple illustration of transverse linearization along a nominal trajectory (*red*) for systems with impacts

Minor variations in impact timing can be handled in linearization-based approaches to stabilizing walking trajectories, but perturbations that lead to changes in mode ordering often cannot. To address this problem, we need algorithms for adjusting walking trajectories online. As described in the previous section, the hybrid nature of walking systems makes this challenging. One way to avoid this issue is to employ smooth approximations of the dynamics that permit the use of efficient local trajectory optimization techniques at the expense of violating complementarity condition (48.22) by, e.g., allowing interpenetration of rigid bodies. This approach has led to impressive examples of online trajectory optimization for simulated systems [48.192], but it is yet unknown how well these approximations will transfer to physical systems.

48.7 Different Contact Behaviors

Contact forces have appeared throughout this chapter to be central for the modeling and control of legged robots. As a result, a different contact behavior can result in a significantly different dynamical behavior of the whole robot, and this can be extremely useful in situations where standard walking or running is inefficient or downright impossible.

48.7.1 Wall Climbing

Wall climbing legged robots rely on various devices such as suction cups, magnets, adhesive materials, miniature spine arrays, that can generate adhesive forces in order to stick to various vertical surfaces such as glass, steel, concrete, brick, stone [48.193–195]. In this case, the unilaterality condition (48.11), which was central throughout this chapter, is no more relevant. As a result, the whole locomotion modeling and control is deeply transformed.

48.7.2 Tethered Walking

On steep slopes, rappelling is an interesting option, tethering the robot to anchors in order to avoid tumbling

down. This has been experienced successfully on construction sites [48.196], or in a volcano in the case of the impressive Carnegie Mellon University (CMU) Dante II octopod robot [48.197]. In this case, the locomotion problem is similar to the general case described throughout this chapter, but the tether introduces an additional contact between the robot and the environment, on which the robot can pull but not push, so a unilateral contact exactly opposite to the standard unilateral contact between the feet and the ground, what can be treated however in exactly the same way.

48.7.3 Legs with Wheels

Adapting wheeled vehicles to rough terrain has led in some cases to implant wheels on legs, combining the flexibility of articulated legs on rough terrain with the efficiency of wheels on flat terrain. Different options are possible, passive or active wheels, passive or active legs [48.198–202]. In this case, the unilaterality condition (48.11) is fundamental like in the rest of this chapter, but the contact points can move freely on the contact surfaces along the direction of the wheels, what can significantly increase the array of possible motions [48.203]. Problems similar to those of standard legged robots include finding stable postures and allocating contact forces for maximizing stability and efficiency [48.200–202]. Interestingly, the case of passive wheels results in motions similar to skating [48.198].

48.7.4 Wheels with Legs

Evolving the mechanical design of legs for greater simplicity and improved efficiency has led in some cases to directly implant the legs on wheels [48.204, 205]. This results in very impressive outdoor performance, but not much related to standard legged robots in the end.

48.8 Conclusion

Legged systems are at the heart of some of the most exciting work in modern robotics. They offer the opportunity to travel to places beyond the reach of wheeled systems and gain fundamental insights into the conditions under which stable and efficient locomotion is possible. At the same time, their complex dynamics pose significant challenges for our computational approaches to control and stability analysis. Indeed, the simple fact that every walking system must engage in intermittent contact with its environment has significant mathematical implications.

A large number of success stories for legged systems moving with remarkable precision and reliability have been witnessed over the past decade, and the development of exciting new tools for optimization-based planning, control, and analysis offers the promise to develop machines that achieve even higher levels of robustness and efficiency.

References

48.1 K.J. Waldron, R.B. McGhee: The adaptive suspension vehicle, IEEE Control Syst. Mag. **6**, 7–12 (1986)

48.2 K. Hirai, M. Hirose, Y. Haikawa, T. Takenaka: The development of honda humanoid robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 1321–1326

48.3 H. Lim, A. Takanishi: Biped walking robots created at waseda university: WL and WABIAN family, Philos. Trans. Royal Soc. A **365**(1850), 49–64 (2007)

48.4 H. Miura, I. Shimoyama: Dynamic walk of a biped, Int. J. Robotics Res. **3**(2), 60–74 (1984)

48.5 M. Raibert: *Legged Robots that Balance* (MIT Press, Cambridge 1986)

48.6 T. McGeer: *Passive Dynamic Walking*, Simon Fraser University Tech. Rep., (Simon Fraser Univ., Burnaby 1988)

48.7 M. Raibert, K. Blankespoor, G. Nelson, R. Playter, the BigDog Team: BigDog, the rough-terrain quadruped robot, Proc. 17th World Cong. Int. Fed. Autom. Control. (2008)

48.8 P.-B. Wieber: Holonomy and nonholonomy in the dynamics of articulated motion, Proc. Ruperto Carola Symp. Fast Motion Biomech. Robotics (2005)

48.9 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)

48.10 M.K. Vukobratović: Contribution to the study of anthropomorphic systems, Kybernetika **8**(5), 404–418 (1972)

48.11 P. Sardain, G. Bessonnet: Forces acting on a biped robot. center of pressure–zero moment point, IEEE Trans. Syst. Man. Cybern. A **34**(5), 630–637 (2004)

48.12 K. Harada, S. Kajita, K. Kaneko, H. Hirukawa: ZMP analysis for arm/leg coordination, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 75–81

48.13 K. Harada, H. Hirukawa, F. Kanehiro, K. Fujiwara, K. Kaneko, S. Kajita, M. Nakamura: Dynamical

balance of a humanoid robot grasping an environment, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 1167–1173
48.14 Y. Or, E. Rimon: Analytic characterization of a class of 3-contact frictional equilibrium postures in 3D gravitational environments, Int. J. Robotics Res. **29**(1), 3–22 (2010)
48.15 P.-B. Wieber: On the stability of walking systems, Proc. Int. Workshop Humanoids Hum. Friendly Robots (2002)
48.16 T. Saida, Y. Yokokoji, T. Yoshikawa: FSW (feasible solution of wrench) for multi-legged robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)
48.17 H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, M. Morisawa: A universal stability criterion of the foot contact of legged robots – adios ZMP, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1976–1983
48.18 T. Bretl, S. Lall: Testing static equilibrium for legged robots, IEEE Trans. Robotics **24**(4), 794–807 (2008)
48.19 S. Barthélemy, P. Bidaud: Stability measure of postural dynamic equilibrium based on residual radius, Proc. Int. Symp. Adv. Robot Kinemat. (2008)
48.20 Z. Qiu, A. Escande, A. Micaelli, T. Robert: Human motions analysis and simulation based on a general criterion of stability, Proc. Int. Symp. Digit. Hum. Model. (2011)
48.21 Z. Qiu, A. Escande, A. Micaelli, T. Robert: A hierarchical framework for realizing dynamically-stable motions of humanoid robot in obstacle-cluttered environments, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)
48.22 E. Rimon, R. Mason, J.W. Burdick, Y. Or: A general stance stability test based on stratified morse theory with application to quasi-static locomotion planning, IEEE Trans. Robotics **24**(3), 626–641 (2008)
48.23 Q. Huang, S. Sugano, K. Tanie: Stability compensation of a mobile manipulator by manipulator motion: Feasibility and planning, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 1285–1292
48.24 J. Kim, W.K. Chung, Y. Youm, B.H. Lee: Real-time ZMP compensation method using null motion for mobile manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 1967–1972
48.25 B. Brogliato: *Nonsmooth Mechanics*, Communications and Control Engineering (Springer, London 1999)
48.26 F. Pfeiffer, C. Glocker: *Multibody Dynamics with Unilateral Contacts* (Wiley, New York 1996)
48.27 S. Chareyron, P.-B. Wieber: *Stability and Regulation of Nonsmooth Dynamical Systems* INRIA Res. Rep. RR-5408 (INRIA, Montbonnot Saint-Ismier 2004)
48.28 C. Liu, Z. Zhao, B. Brogliato: Frictionless multiple impacts in multibody systems. I. Theoretical framework, Proc. R. Soc. A **464**, 3193–3211 (2008)
48.29 Y.-B. Jia, M. Mason, M. Erdmann: Multiple impacts: A state transition diagram approach, Int. J. Robotics Res. **32**(1), 84–114 (2013)
48.30 J.-M. Bourgeot, C. Canudas de Wit, B. Brogliato: Impact shaping for double support walk: From the rocking block to the biped robot, Proc. Int. Conf. Climb. Walk. Robots (2005)
48.31 B. Gamus, Y. Or: Analysis of dynamic bipedal robot locomotion with stick-slip transitions, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 3348–3355
48.32 S. Kajita, K. Miura, M. Morisawa, K. Kaneko, F. Kanehiro, K. Yokoi: Evaluation of a stabilizer for biped walk with toe support phase, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)
48.33 V. Acary, B. Brogliato: *Numerical Methods for Nonsmooth Dynamical Systems*, Lect. Notes Appl. Comput. Mech., Vol. 35 (Springer, Berlin, Heidelberg 2008)
48.34 R.I. Leine, N. van de Wouw: *Stability and Convergence of Mechanical Systems with Unilateral Constraints*, Lect. Notes Appl. Comput. Mech., Vol. 36 (Springer, Berlin, Heidelberg 2008)
48.35 Y. Or, A.D. Ames: Stability and completion of zeno equilibria in lagrangian hybrid systems, IEEE Trans. Autom. Control **56**(6), 1322–1336 (2011)
48.36 D.E. Stewart, J.C. Trinkle: An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction, Int. J. Numer. Methods Eng. **39**(15), 2673–2691 (1996)
48.37 M. Posa, C. Cantu, R. Tedrake: A direct method for trajectory optimization of rigid bodies through contact, Int. J. Robotics Res. **33**(1), 69–81 (2014)
48.38 E.R. Westervelt, J.W. Grizzle, C. Chevallereau, J.H. Choi, B. Morris: *Feedback Control of Dynamic Bipedal Robot Locomotion* (CRC, Boca Raton 2007)
48.39 M. Wisse: Essentials of Dynamic Walking: Analysis and Design of Two-Legged Robots, Dissertation (Technische Universiteit, Delft 2004)
48.40 R. Tedrake, I.R. Manchester, M.M. Tobenkin, J.W. Roberts: LQR-Trees: Feedback motion planning via sums of squares verification, Int. J. Robotics Res. **29**, 1038–1052 (2010)
48.41 M. Posa, M. Tobenkin, R. Tedrake: Lyapunov analysis of rigid body systems with impacts and friction via sums-of-squares, Proc. Int. Conf. Hybrid Syst. Comput. Control (2013) pp. 63–72
48.42 S. Cotton, I. Olaru, M. Bellman, T. van der Ven, J. Godowski, J. Pratt: Fastrunner: A fast, efficient and robust bipedal robot. concept and planar simulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)
48.43 M. Srinivasan, A. Ruina: Computer optimization of a minimal biped model discovers walking and running, Nature **439**, 72–75 (2006)
48.44 K. Mombaur, H.G. Bock, J.P. Schloder, R.W. Longman: Open-loop stable solutions of periodic optimal control problems in robotics, Z. Angew. Math. Mech. **85**(7), 499–515 (2005)
48.45 J. Hauser, C.C. Chung: Converse Lyapunov functions for exponentially stable periodic orbits, Syst. Control Lett. **23**(1), 27–34 (1994)

48.46 J. Guckenheimer, P. Holmes: *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields* (Springer, Berlin, Heidelberg 1983)

48.47 I.R. Manchester, M.M. Tobenkin, M. Levashov, R. Tedrake: Regions of attraction for hybrid limit cycles of walking robots, Proc. 21st World Cong. Int. Fed. Autom. Control (2011)

48.48 E.R. Westervelt, G. Buche, J.W. Grizzle: Experimental validation of a framework for the design of controllers that induce stable walking in planar bipeds, Int. J. Robotics Res. **23**(6), 559–582 (2004)

48.49 A.S. Shiriaev, L.B. Freidovich, I.R. Manchester: Can we make a robot ballerina perform a pirouette? Orbital stabilization of periodic motions of underactuated mechanical systems, Annu. Rev. Control **32**(2), 200–211 (2008)

48.50 I.R. Manchester, U. Mettin, F. Iida, R. Tedrake: Stable dynamic walking over uneven terrain, Int. J. Robotics Res. **30**(3), 265–279 (2011)

48.51 J.-P. Aubin: *Viability Theory* (Birkhäuser, Basel 1991)

48.52 P.-B. Wieber: Constrained dynamics and parametrized control in biped walking, Proc. Int. Symp. Math. Theory Networks Syst. (2000)

48.53 P.-B. Wieber: Viability and predictive control for safe locomotion, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008)

48.54 K. Ogata: *Modern Control Engineering*, 3rd edn. (Prentice Hall, Upper Saddle River 1996)

48.55 J. Pratt, R. Tedrake: Velocity based stability margins for fast bipedal walking, Proc. Ruperto Carola Symp. Fast Motion Biomech. Robotics (2005)

48.56 A.L. Hof, M.G.J. Gazendam, W.E. Sinke: The condition for dynamic stability, J. Biomech. **38**, 1–8 (2005)

48.57 J. Pratt, J. Carff, S. Drakunov, A. Goswami: Capture point: A step toward humanoid push recovery, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2006)

48.58 T. Takenaka, T. Matsumoto, T. Yoshiike: Real time motion generation and control for biped robot – 1st report: Walking gait pattern generation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009)

48.59 J. Englsberger, C. Ott, M.A. Roa, A. Albu-Schäffer, G. Hirzinger: Bipedal walking control based on capture point dynamics, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011)

48.60 T. Koolen, T. de Boer, J. Rebula, A. Goswami, J. Pratt: Capturability-based analysis and control of legged locomotion, Part 1: Theory and application to three simple gait models, Int. J. Robotics Res. **31**(9), 1094–1113 (2012)

48.61 A. Papachristodoulou, S. Prajna: Robust stability analysis of nonlinear hybrid systems, IEEE Trans. Autom. Control **54**(5), 1035–1041 (2009)

48.62 P. Hanggi, P. Talkner, M. Borkovec: Reaction-rate theory: Fifty years after Kramers, Rev. Mod. Phys. **62**(2), 251–342 (1990)

48.63 K. Byl: Metastable Legged-Robot Locomotion, Dissertation (MIT, Cambridge 2008)

48.64 J. Steinhardt, R. Tedrake: Finite-time regional verification of stochastic nonlinear systems, Int. J. Robotics Res. **31**(7), 901–923 (2012)

48.65 D.G.E. Hobbelen, M. Wisse: A disturbance rejection measure for limit cycle walkers: The gait sensitivity norm, IEEE Trans. Robotics **23**(6), 1213–1224 (2007)

48.66 C. Ebenbauer: Polynomial Control Systems: Analysis and Design via Dissipation Inequalities and Sum of Squares, Dissertation (Univ. Stuttgart, Stuttgart 2005)

48.67 H. Dai, R. Tedrake: L2-gain optimization for robust bipedal walking on unknown terrain, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)

48.68 E. Garcia, J. Estremera, P. Gonzalez de Santos: A classification of stability margins for walking robots, Proc. Int. Conf. Climb. Walk. Robots (2002)

48.69 A. Goswami: Postural stability of biped robots and the foot rotation indicator (FRI) point, Int. J. Robotics Res. **18**(6), 523–533 (1999)

48.70 C.K. Chow, D.H. Jacobson: *Studies of human locomotion via optimal programming*, Tech. Rep. No. 617 (Harvard Univ., Cambridge 1970)

48.71 P.H. Channon, S.H. Hopkins, D.T. Phan: Derivation of optimal walking motions for a biped walking robot, Robotica **10**(2), 165–172 (1992)

48.72 G. Cabodevilla, N. Chaillet, G. Abba: Energy-minimized gait for a biped robot, Proc. Auton. Mob. Syst. (1995)

48.73 C. Chevallereau, A. Formal'sky, B. Perrin: Low energy cost reference trajectories for a biped robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998)

48.74 M. Rostami, G. Bessonnet: Impactless sagittal gait of a biped robot during the single support phase, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998)

48.75 L. Roussel, C. Canudas de Wit, A. Goswami: Generation of energy optimal complete gait cycles for biped robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998)

48.76 J. Denk, G. Schmidt: Synthesis of a walking primitive database for a humanoid robot using optimal control techniques, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2001)

48.77 T. Buschmann, S. Lohmeier, H. Ulbrich, F. Pfeiffer: Optimization based gait pattern generation for a biped robot, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2005)

48.78 K. Mombaur: Using optimization to create self-stable human-like running, Robotica **27**(3), 321–330 (2009)

48.79 J. Denk, G. Schmidt: Synthesis of walking primitive databases for biped robots in 3D-environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

48.80 S.A. Setiawan, S.H. Hyon, J. Yamaguchi, A. Takanishi: Quasi real-time walking control of a bipedal humanoid robot based on walking pattern synthesis, Proc. Int. Symp. Exp. Robotics (1999)

48.81 P.-B. Wieber, C. Chevallereau: Online adaptation of reference trajectories for the control of walking systems, Robotics Auton. Syst. **54**(7), 559–566 (2006)

48.82 C. Liu, C.G. Atkeson: Standing balance control using a trajectory library, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 3031–3036

48.83 J. Yamaguchi, A. Takanishi, I. Kato: Development of biped walking robot compensating for three-axis moment by trunk motion, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1993)

48.84 Q. Huang, K. Yokoi, S. Kajita, K. Kaneko, H. Arai, N. Koyachi, K. Tanie: Planning walking patterns for a biped robot, IEEE Trans. Robotics Autom. **17**(3), 280–289 (2001)

48.85 P.-B. Wieber: Trajectory free linear model predictive control for stable walking in the presence of strong perturbations, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2006)

48.86 T. de Boer: Foot placement in robotic bipedal locomotion, Dissertation (Technische Univ. Delft, Dleft 2012)

48.87 R.J. Full, D.E. Koditschek: Templates and anchors: Neuromechanical hypotheses of legged locomotion on land, J. Exp. Biol. **202**, 3325–3332 (1999)

48.88 R.M. Alexander: Mechanics of bipedal locomotion, Persp. Exp. Biol. **1**, 493–504 (1976)

48.89 R.M. Alexander: Simple models of human movement, ASME Appl. Mech. Rev. **48**(8), 461–470 (1995)

48.90 S.S. Keerthi, E.G. Gilbert: Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations, J. Optim. Theory Appl. **57**(2), 265–293 (1988)

48.91 D.Q. Mayne, J.B. Rawlings, C.V. Rao, P.O.M. Scokaert: Constrained model predictive control: Stability and optimality, Automatica **26**(6), 789–814 (2000)

48.92 M. Alamir, N. Marchand: Numerical stabilisation of non-linear systems: Exact theory and approximate numerical implementation, Eur. J. Control **5**(1), 87–97 (1999)

48.93 M. Alamir, G. Bornard: Stability of a truncated infinite constrained receding horizon scheme: The general discrete nonlinear case, Automatica **31**(9), 1353–1356 (1995)

48.94 A. Takanishi, M. Tochizawa, H. Karaki, I. Kato: Dynamic biped walking stabilized with optimal trunk and waist motion, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1989)

48.95 H. Lim, Y. Kaneshima, A. Takanishi: Online walking pattern generation for biped humanoid robot with trunk, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

48.96 T. Buschmann, S. Lohmeier, M. Bachmayer, H. Ulbrich, F. Pfeiffer: A collocation method for real-time walking pattern generation, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2007)

48.97 K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, H. Inoue: Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired ZMP, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002)

48.98 R. Tajima, D. Honda, K. Suga: Fast running experiments involving a humanoid robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009)

48.99 K. Nagasaka, Y. Kuroki, S. Suzuki, Y. Itoh, J. Yamaguchi: Integrated motion control for walking, jumping and running on a small bipedal entertainment robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

48.100 M. Morisawa, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, S. Nakaoka, H. Hirukawa: A biped pattern generation allowing immediate modification of foot placement in real-time, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2006)

48.101 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Biped walking pattern generation by using preview control of zero moment point, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 1620–1626

48.102 K. Nishiwaki, S. Kagami: Online walking control systems for humanoids with short cycle pattern generation, Int. J. Robotics Res. **28**(6), 729–742 (2009)

48.103 J. Park, Y. Youm: General ZMP preview control for bipedal walking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007)

48.104 D. Gouaillier, C. Collette, C. Kilner: Omni-directional closed-loop walk for NAO, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2010)

48.105 M. Krause, J. Englsberger, P.-B. Wieber, C. Ott: Stabilization of the capture point dynamics for bipedal walking based on model predictive control, Proc. IFAC Symp. Robot Control (2012)

48.106 M. van de Panne: From footprints to animation, Comput. Graph. **16**(4), 211–223 (1997)

48.107 A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, M. Diehl: Online walking motion generation with automatic foot step placement, Adv. Robotics **24**(5-6), 719–737 (2010)

48.108 A. Herdt, N. Perrin, P.-B. Wieber: LMPC based online generation of more efficient walking motions, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)

48.109 Z. Aftab, T. Robert, P.-B. Wieber: Ankle, hip and stepping strategies for humanoid balance recovery with a single model predictive control scheme, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)

48.110 J. Urata, K. Nishiwaki, Y. Nakanishi, K. Okada, S. Kagami, M. Inaba: Online decision of foot placement using singular LQ preview regulation, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2011)

48.111 J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, H. Inoue: Dynamically-stable motion planning for humanoid robots, Auton. Robots **12**, 105–118 (2002)

48.112 S. Dalibard, A. El Khoury, F. Lamiraux, M. Taix, J.-P. Laumond: Small-space controllability of a walking humanoid robot, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2011)

48.113 K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, T. Isozumi: Humanoid robot HRP-2, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 1083–1090

48.114 E. Yoshida, C. Esteves, I. Belousov, J.-P. Laumond, T. Sakaguchi, K. Yokoi: Planning 3-D collision-free dynamic robotic motion through iterative reshaping, IEEE Trans. Robotics **24**(5), 1186–1198 (2008)

48.115 J.-D. Boissonnat, O. Devillers, S. Lazard: Motion planning of legged robots, SIAM J. Comput. **30**(1), 218–246 (2000)

48.116 N. Perrin, O. Stasse, F. Lamiraux, E. Yoshida: Weakly collision-free paths for continuous humanoid footstep planning, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011)

48.117 N. Perrin: From discrete to continuous motion planning, Proc. Int. Workshop Algorithm. Found. Robotics (2012)

48.118 J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, H. Inoue: Online footstep planning for humanoid robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

48.119 J. Chestnutt, J. Kuffner: A tiered planning strategy for biped navigation, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2004)

48.120 P. Michel, J. Chestnutt, J. Kuffner, T. Kanade: Vision-Guided Humanoid Footstep Planning for Dynamic Environments. In: *Proc. IEEE/RAS Int. Conf. Humanoid Robots* 2005)

48.121 J. Chestnutt, P. Michel, J. Kuffner, T. Kanade: Locomotion among dynamic obstacles for the honda asimo, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007)

48.122 J.-M. Bourgeot, N. Cislo, B. Espiau: Path-planning and tracking in a 3D complex environment for an anthropomorphic biped robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002)

48.123 J. Chestnutt, J. Kuffner, K. Nishiwaki, S. Kagami: Planning biped navigation strategies in complex environments, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2003)

48.124 M. Zucker, J.A. Bagnell, C. Atkeson, J. Kuffner: An optimization approach to rough terrain locomotion, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

48.125 J. Chestnutt, Y. Takaoka, K. Suga, K. Nishiwaki, J. Kuffner, S. Kagami: Biped navigation in rough environments using on-board sensing, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009)

48.126 N. Perrin, O. Stasse, F. Lamiraux, Y. Kim, D. Manocha: Real-time footstep planning for humanoid robots among 3D obstacles using a hybrid bounding box, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

48.127 N. Perrin, O. Stasse, L. Baudoin, F. Lamiraux, E. Yoshida: Fast humanoid robot collision-free footstep planning using swept volume approximations, IEEE Trans. Robotics **28**(2), 427–439 (2012)

48.128 R.L.H. Deits, R. Tedrake: Computing large convex regions of obstacle-free space through semidefinite programming, Proc. Int. Workshop Algorithmic Found. Robotics (2014)

48.129 T. Bretl, S. Lall, J.-C. Latombe, S. Rock: Multi-step motion planning for free-climbing robots, Proc. Int. Workshop Algorithmic Found. Robotics (2004)

48.130 K. Hauser, T. Bretl, J.-C. Latombe: Non-gaited humanoid locomotion planning, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2005)

48.131 K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, B. Wilcox: Motion planning for legged robots on varied terrain, Int. J. Robotics Res. **27**(11/12), 1325–1349 (2008)

48.132 A. Escande, A. Kheddar, S. Miossec: Planning support contact-points for humanoid robots and experiments on HRP-2, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006)

48.133 A. Escande, A. Kheddar, S. Miossec, S. Garsault: Planning support contact-points for acyclic motions and experiments on HRP-2, Proc. Int. Symp. Exp. Robotics (2008)

48.134 K. Bouyarmane, J. Vaillant, F. Keith, A. Kheddar: Exploring humanoid robots locomotion capabilities in virtual disaster response scenarios, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)

48.135 S. Lengagne, J. Vaillant, E. Yoshida, A. Kheddar: Generation of whole-body optimal dynamic multi-contact motions, Int. J. Robotics Res. **32**(9/10), 1104–1119 (2013)

48.136 K. Harada, E. Yoshida, K. Yokoi: *Motion Planning for Humanoid Robots* (Springer, Berlin, Heidelberg 2010)

48.137 R.B. McGhee, A.A. Frank: On the stability properties of quadruped creeping gaits, Math. Biosci. **3**, 331–351 (1968)

48.138 G.C. Haynes, A.A. Rizzi: Gaits and gait transitions for legged robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1117–1122

48.139 Y. Fujimoto, A. Kawamura: Proposal of biped walking control based on robust hybrid position/force control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Minneap. (1996) pp. 2724–2730

48.140 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Yokoi, H. Hirukawa: A realtime pattern generator for biped walking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 31–37

48.141 K. Yin, K. Loken, M. van de Panne: SIMBICON: Simple Biped Locomotion Control, Proc. ACM SIGGRAPH (2007)

48.142 I.-W. Park, J.-Y. Kim, J.-H. Oh: Online biped walking pattern generation for humanoid robot KHR-3(KAIST humanoid robot – 3: HUBO), Proc. IEEE-RAS Int. Conf. Humanoid Robots (2006)

48.143 A. Ijspeert, A. Crespi, D. Ryczko, J.-M. Cabelguen: From swimming to walking with a salamander robot driven by a spinal cord model, Science **315**(5817), 1416–1420 (2007)

48.144 R. Katoh, M. Mori: Control method of biped locomotion giving asymptotic stability of trajectory, Automatica **20**(4), 405–414 (1984)

48.145 L. Righetti, A. Ijspeert: Programmable central pattern generators: An application to biped locomotion control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

48.146 K. Matsuoka: Sustained oscillations generated by mutually inhibiting neurons with adaptation, Biol. Cybern. **52**, 367–376 (1985)

48.147 G. Endo, J. Morimoto, J. Nakanishi, G. Cheng: An empirical exploration of a neural oscillator for

biped locomotion control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 3036–3042
48.148 Y. Fukuoka, H. Kimura, A. Cohen: Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts, Int. J. Robotics Res. **22**(3–4), 187–202 (2003)
48.149 R. Brooks: Elephants don't play chess, Robotics Auton. Syst. **6**, 3–15 (1990)
48.150 R. Brooks: A robot that walks; emergent behaviors from a carefully evolved network, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1989) pp. 292–296
48.151 J. Pratt, C.-M. Chew, A. Torres, P. Dilworth, G. Pratt: Virtual model control: An intuitive approach for bipedal locomotion, Int. J. Robotics Res. **20**, 129–143 (2001)
48.152 F. Génot, B. Espiau: On the control of the mass center of legged robots under unilateral constraints, Proc. Int. Conf. Climb. Walk. Robots (1998)
48.153 L. Saab, O.E. Ramos, F. Keith, N. Mansard, P. Souères, J.-Y. Fourquet: Dynamic whole-body motion generation under rigid contacts and other unilateral constraints, IEEE Trans. Robotics **29**(2), 346–362 (2013)
48.154 L. Sentis, J. Park, O. Khatib: Compliant control of multicontact and center-of-mass behaviors in humanoid robots, IEEE Trans. Robotics **26**(3), 483–501 (2010)
48.155 R.W. Brockett: Asymptotic stability and feedback stabilization. In: *Differential Geometric Control Theory*, (Birkhäuser, Boston 1983)
48.156 S. Kajita, T. Nagasaki, K. Kaneko, K. Yokoi, K. Tanie: A running controller of humanoid biped HRP-2LR, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 618–624
48.157 L. Sentis, O. Khatib: Control of free-floating humanoid robots through task prioritization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1730–1735
48.158 A. De Luca, G. Oriolo: Modelling and control of nonholonomic mechanical systems, CISM Int. Centre Mech. Sci. **360**, 277–342 (1995)
48.159 Y. Nakamura, R. Mukherjee: Exploiting nonholonomic redundancy of free-flying space robots, IEEE Trans. Robotics Autom. **9**(4), 499–506 (1993)
48.160 E. Papadopoulos: Nonholonomic behaviour in free-floating space manipulators and its utilization. In: *Nonholonomic Motion Planning*, ed. by Y. Xu, T. Kanade (Kluwer Academic, Dordrecht 1993)
48.161 C. Chevallereau, E.R. Westervelt, J.W. Grizzle: Asymptotically stable running for a five-link, four-actuator, planar bipedal robot, Int. J. Robotics Res. **24**(6), 431–464 (2005)
48.162 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Resolved momentum control: Humanoid motion planning based on the linear and angular momentum, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1644–1650
48.163 M. Popovic, A. Hofmann, H. Herr: Zero spin angular momentum control: definition and applicability, IEEE/RAS Int. Conf. Humanoid Robots (2004)
48.164 S.-H. Lee, A. Goswami: Ground reaction force control at each foot: A momentum-based humanoid balance controller for non-level and non-stationary ground, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 3157–3162
48.165 C. Ott, M.A. Roa, G. Hirzinger: Posture and balance control for biped robots based on contact force optimization, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2011)
48.166 T. Sugihara: Standing stabilizability and stepping maneuver in planar bipedalism based on the best COM-ZMP regulator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009)
48.167 Y. Choi, D. Kim, B.-J. You: On the walking control for humanoid robot based on the kinematic resolution of CoM jacobian with embedded motion, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2655–2660
48.168 S.-H. Hyon: Compliant terrain adaptation for biped humanoids without measuring ground surface and contact forces, IEEE Trans. Robotics **25**(1), 171–178 (2009)
48.169 S. Kajita, K. Yokoi, M. Saigo, K. Tanie: Balancing a humanoid robot using backdrive concerned torque control and direct angular momentum feedback, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 3376–3382
48.170 S. Lohmeier, K. Löffler, M. Gienger, H. Ulbrich, F. Pfeiffer: Computer system and control of biped *Johnnie*, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4222–4227
48.171 J.-H. Kim, J.-H. Oh: Walking control of the humanoid platform KHR-1 based on torque feedback control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 623–628
48.172 M.-S. Kim, J.-H. Oh: Posture control of a humanoid robot with a compliant ankle joint, Int. J. Humanoid Robotics **7**(1), 5–29 (2010)
48.173 T. Buschmann, S. Lohmeier, H. Ulbrich: Biped walking control based on hybrid position/force control, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 3019–3024
48.174 S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, K. Yokoi: Biped walking stabilization based on linear inverted pendulum tracking, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 4489–4496
48.175 M. Morisawa, S. Kajita, F. Kanehiro, K. Kaneko, K. Miura, K. Yokoi: Balance control based on capture point error compensation for biped walking on uneven terrain, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2012)
48.176 Y. Fujimoto, A. Kawamura: Simulation of an autonomous biped walking robot including environmental force interaction, IEEE Robotics Autom. Mag. **5**(2), 33–41 (1998)
48.177 L. Righetti, J. Buchli, M. Mistry, M. Kalakrishnan, S. Schaal: Optimal distribution of contact forces

with inverse dynamics control, Int. J. Robotics Res. **32**(3), 280–298 (2013)
48.178 A.D. Ames: Human-inspired control of bipedal robotics via control lyapunov functions and quadratic programs, Proc. Int. Conf. Hybrid Syst. Comput. Control (2013)
48.179 S. Kuindersma, F. Permenter, R. Tedrake: An efficiently solvable quadratic program for stabilizing dynamic locomotion, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2014)
48.180 S.H. Collins, A. Ruina, R. Tedrake, M. Wisse: Efficient bipedal robots based on passive-dynamic walkers, Science **307**, 1082–1085 (2005)
48.181 S.H. Collins, M. Wisse, A. Ruina: A three-dimensional passive-dynamic walking robot with two legs and knees, Int. J. Robotics Res. **20**(7), 607–615 (2001)
48.182 R. Tedrake, T.W. Zhang, H.S. Seung: Stochastic policy gradient reinforcement learning on a simple 3D biped, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 2849–2854
48.183 J.E. Wilson: Walking Toy, Patent 0 (1936)
48.184 R. Tedrake, T.W. Zhang, M.F. Fong, H.S. Seung: Actuating a simple 3D passive dynamic walker, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 4656–4661
48.185 M. Garcia, A. Chatterjee, A. Ruina: Efficiency, speed, and scaling of two-dimensional passive-dynamic walking, Dyn. Stab. Sytems **15**(2), 75–99 (2000)
48.186 A. Goswami, B. Thuilot, B. Espiau: *Compass-Like Biped Robot Part I : Stability and Bifurcation of Passive Gaits*, INRIA Res. Rep. No. 2996 (INRIA, Les Chesnay Cedex 1996)
48.187 M.W. Spong, G. Bhatia: Further results on control of the compass gait biped, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1933–1938
48.188 F. Asano, Z.-W. Luo, M. Yamakita: Biped gait generation and control based on a unified property of passive dynamic walking, IEEE Trans. Robotics **21**(4), 754–762 (2005)
48.189 J.T. Betts: Survey of numerical methods for trajectory optimization, J. Guid. Control, Dyn. **21**(2), 193–207 (1998)
48.190 C.D. Remy: Optimal Exploitation of Natural Dynamics in Legged Locomotion, Dissertation (ETH, Zurich 2011)
48.191 K. Sreenath, H.W. Park, I. Poulakakis, J.W. Grizzle: A compliant hybrid zero dynamics controller for stable, efficient and fast bipedal walking on MABEL, Int. J. Robotics Res. **30**(9), 1170–1193 (2011)
48.192 Y. Tassa, T. Erez, E. Todorov: Synthesis and stabilization of complex behaviors through online trajectory optimization, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 4906–4913
48.193 S. Hirose, A. Nagakubo, R. Toyama: Machine that can walk and climb on floors, walls and ceilings, Proc. Int. Conf. Adv. Robotics (1991) pp. 753–758
48.194 T. Yano, S. Numao, Y. Kitamura: Development of a self-contained wall climbing robot with scanning type suction cups, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998) pp. 249–254
48.195 S. Kim, A. Asbeck, W. Provancher, M.R. Cutkosky: SpinybotII: Climbing hard walls with compliant microspines, Proc. Int. Conf. Adv. Robotics (2005) pp. 18–20
48.196 S. Hirose, K. Yoneda, H. Tsukagoshi: TITAN VII: quadruped walking and manipulating robot on a steep slope, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 494–500
48.197 J. Bares, D. Wettergreen: Dante II: technical description, results and lessons learned, Int. J. Robotics Res. **18**(7), 621–649 (1999)
48.198 G. Endo, S. Hirose: Study on roller-walker (system integration and basic experiments), IEEE Int. Conf. Robotics Autom. (1999) pp. 2032–2037
48.199 R. Siegwart, P. Lamon, T. Estier, M. Lauria, R. Piguet: Innovative design for wheeled locomotion in rough terrain, Robotics Auton. Syst. **40**, 151–162 (2002)
48.200 K. Iagnemma, A. Rzepniewski, S. Dubowsky: Control of robotic vehicles with actively articulated suspensions in rough terrain, Auton. Robots **14**, 5–16 (2003)
48.201 K. Iagnemma, S. Dubowsky: Traction control of wheeled robotic vehicles in rough terrain with application to planetary rovers, Int. J. Robotics Res. **23**(10–11), 1029–1040 (2004)
48.202 C. Grand, F. Ben Amar, F. Plumet, P. Bidaud: Stability and traction optimization of a wheel-legged robot, Int. J. Robotics Res. **23**(10–11), 1041–1058 (2004)
48.203 G. Besseron, C. Grand, F. Ben Amar, F. Plumet, P. Bidaud: Locomotion modes of an hybrid wheel-legged robot, Proc. Int. Conf. Climb. Walk. Robots (2004)
48.204 U. Saranli, M. Buehler, D.E. Koditschek: Rhex - a simple and highly mobile hexapod robot, Int. J. Robotics Res. **20**(7), 616–631 (2001)
48.205 T. Allen, R. Quinn, R. Bachmann, R. Ritzmann: Abstracted biological principles applied with reduced actuation improve mobility of legged vehicles, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1370–1375

Multimedia Contents

49. Modeling and Control of Wheeled Mobile Robots

Claude Samson, Pascal Morin, Roland Lenain

This chapter may be seen as a follow up to Chap. 24, devoted to the classification and modeling of basic wheeled mobile robot (WMR) structures, and a natural complement to Chap. 47, which surveys motion planning methods for WMRs. A typical output of these methods is a feasible (or admissible) reference state trajectory for a given mobile robot, and a question which then arises is how to make the physical mobile robot track this reference trajectory via the control of the actuators with which the vehicle is equipped. The object of the present chapter is to bring elements of the answer to this question based on simple and effective control strategies.

The chapter is organized as follows. Section 49.2 is devoted to the choice of control models and the determination of modeling equations associated with the path-following control problem. In Sect. 49.3, the path following and trajectory stabilization problems are addressed in the simplest case when no requirement is made on the robot orientation (i. e., position control). In Sect. 49.4 the same problems are revisited for the control of both position and orientation. The previously mentionned sections consider an ideal robot satisfying the rolling-without-sliding assumption. In Sect. 49.5, we relax this assumption in order to take into account nonideal wheel-ground contact. This is especially important for field-robotics applications and the proposed results are validated through full scale experiments on natural terrain. Finally, a few complementary issues on the feedback control of mobile robots are briefly discussed in the concluding Sect. 49.6, with a list of commented references for further reading on WMRs motion control.

49.1 **Background** 1236
49.1.1 Path Following 1237
49.1.2 Stabilization of Trajectories 1237
49.1.3 Stabilization of Fixed Postures 1237

49.2 **Control Models** 1238
49.2.1 Kinematics Versus Dynamics 1238
49.2.2 Modeling in an Absolute Frame 1238
49.2.3 Modeling in a Frénet Frame 1239

49.3 **Adaptation of Control Methods for Holonomic Systems** 1240
49.3.1 Stabilization of Trajectories for a Nonconstrained Point 1240
49.3.2 Path Following with no Orientation Control 1240

49.4 **Methods Specific to Nonholonomic Systems** 1241
49.4.1 Transformation of Kinematic Models into the Chained Form 1242
49.4.2 Tracking of a Reference Vehicle with the Same Kinematics 1242
49.4.3 Path Following with Orientation Control 1244
49.4.4 Asymptotic Stabilization of Fixed Postures 1246
49.4.5 Limitations Inherent to the Control of Nonholonomic Systems 1250
49.4.6 Practical Stabilization of Arbitrary Trajectories 1251

49.5 **Path Following in the Case of Nonideal Wheel-Ground Contact** 1255
49.5.1 Extended Control Models in the Presence of Sliding 1255
49.5.2 On-Line Estimation of Sliding Angles 1257
49.5.3 Feedback Laws for Path Following 1259

49.5.4 Path Following Illustrations in Low Grip Conditions 1259
49.6 **Complementary Issues and Bibliographical Guide** 1261
49.6.1 General Trailer Systems 1261
49.6.2 Sensor-Based Motion Control 1262
49.6.3 Sliding Effects and Other Dynamical Issues 1262
49.6.4 Bibliographical Guide 1262
Video-References 1263
References 1263

49.1 Background

Wheeled mobile robotics remains a very active research topic, as witnessed by the Darpa Grand Challenges [49.1]. The first Grand Challenge, in 2005, took place in a clear environment and was won by Stanley (Fig. 49.1a). The second Grand Challenge, in 2007, took place in urban environment and demonstrated autonomous driving capabilities, especially in complex environments with strong safety and reliability issues. This made possible the development of autonomous cars, such as google car (Fig. 49.1b). Whatever the sensors used for localization and detection, the use in every day life of cybercars, such as VipaLab (Fig. 49.1c), is becoming a reality. Numerous tests are under progress all around the world showing the benefits of autonomous transportation systems. On-road transportation probably constitutes the main and most visible application domain of emerging autonomous wheeled robots. Driving assistance for human activities in off-road conditions constitutes another application domain. Indeed, WMRs can also be used in hazardous environments, or in environments that are not reachable by humans. Planetary exploration (Chap. 55), with the example of Curiosity (Fig. 49.1d), is a popular example of such applications. Other developments, concerning people well-being and safety, are also investigated in the field of environment monitoring, surveillance, agriculture, civil protection or defence. Figure 49.1e,f illustrate these developments with the first one representing the Claas etrion robot (for autonomous farming, see also Chap. 56) and the second one representing the SUGV robot manufactured by iRobot. This robot, dedicated to civil protection, was the first one to enter the damaged Fukushima power plant. All these examples highlight the possible benefits and different applications of WMRs. The variety of robot designs (in term of scale, locomotion mode, or related actuation strategy see Chap. 24), is in line with the diversity of application needs. Whatever the system and application, however, autonomy relies on the design of efficient feedback laws that can ensure precise autonomous motion of the vehicle despite all possible modeling errors and perturbations. This chapter is dedicated to this issue and aims at providing the basics of feedback control design for nonholonomic WMRs. Implementation of these control laws supposes that one is able to measure the variables involved in the control loop (typically the position and orientation of the mobile robot with respect to either a fixed frame or a path that the vehicle should follow). Throughout this chapter we will assume that these measurements are available continuously in time and that they are not corrupted by noise. In a general manner, robustness considerations will not be discussed in detail, one reason being that, beyond imposed space limitations, a large part of the presented approaches are based on linear control theory. The feedback control laws then inherit the strong robustness properties associated with stable linear systems. Results can also be subsequently refined by using

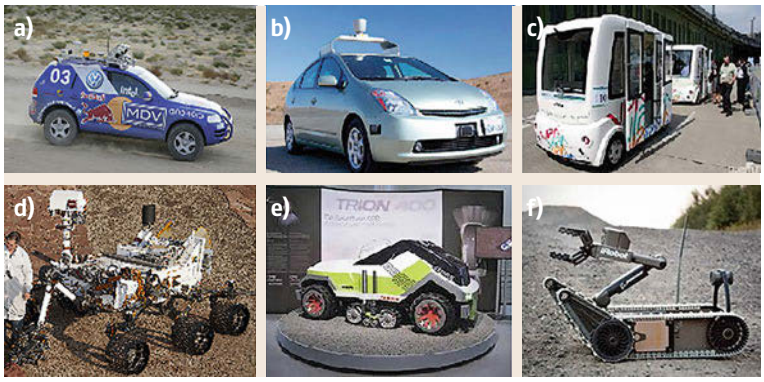


Fig.49.1a–f Some example of mobile robots for different applications: **(a)** Stanley (Stanford Univ.), off-road motion; **(b)** Google car, autonomous car; **(c)** Vipa (Institut Pascal/Ligier), urban transportation; **(d)** Curiosity (NASA) planetary exploration; **(e)** Etrion (Claas), agriculture robot; **(f)** PackBot (iRobot), public safety

complementary, eventually more elaborate, automatic control techniques. For the sake of simplicity, the control methods are developed mainly for unicycle-type and car-like mobile robots, which correspond respectively to the types $(2,0)$ and $(1,1)$ in the classification proposed in Chap. 24. Most of the results can in fact be easily extended/adapted to other mobile robots, in particular to systems with trailers. We will mention the cases where such extensions are straightforward. All reported simulation results, illustrating various control problems and solutions, are carried out for a car-like vehicle, whose kinematics is slightly more complex than that of unicycle-type vehicles.

Recall (Fig. 49.2) that:

1. A *unicycle-type mobile robot* is schematically composed of two independent actuated wheels on a common axle whose direction is rigidly linked to the robot chassis, and one or several passively orientable – or caster – wheels, which are not controlled and serve for sustentation purposes.
2. A (rear-drive) *car-like mobile robot* is composed of a motorized wheeled axle at the rear of the chassis, and one (or a pair of) orientable front steering wheel(s).

Note also, as illustrated by the diagram below (Fig. 49.3), that a car-like mobile robot can be viewed (at least kinematically) as a unicycle-type mobile robot to which a trailer is attached.

Despite the variety of application, three generic control problems can mainly be considered in mobile robotics and are detailed in this chapter.

49.1.1 Path Following

Given a curve C on the plane, a (nonzero) longitudinal velocity v_0 for the robot chassis, and a point P attached to the chassis, the goal is to have the point P follow the curve C when the robot moves with the velocity v_0. The variable that one has to stabilize at zero is thus the distance between the point P and the curve (i. e., the distance between P and the closest point M on C). This type of problem typically corresponds to driving on a road while trying to maintain the distance between the vehicle chassis and the side of the road constant. Automatic wall following is another possible application.

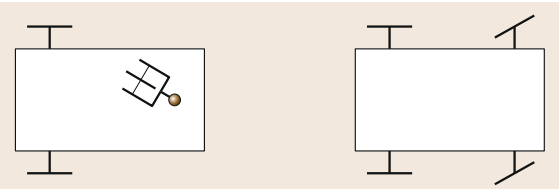


Fig.49.2a,b Unicycle-type (a) and car-like (b) mobile robots

49.1.2 Stabilization of Trajectories

This problem differs from the previous one in that the vehicle's longitudinal velocity is no longer predetermined because one also aims to monitor the distance gone along the curve C. This objective supposes that the geometric curve C is complemented with a time schedule, i. e., that it is parameterized with the time variable t. This boils down to defining a trajectory $t \longmapsto (x_r(t), y_r(t))$ with respect to a reference frame $\mathcal{F}_0$. Then the goal is to stabilize the position error vector $(x(t) - x_r(t), y(t) - y_r(t))$ at zero, with $(x(t), y(t))$ denoting the coordinates of point P in $\mathcal{F}_0$ at time t. The problem may also be formulated as one of controlling the vehicle in order to track a reference vehicle whose trajectory is given by $t \longmapsto (x_r(t), y_r(t))$. Note that perfect tracking is achievable only if the reference trajectory is feasible for the physical vehicle, and that a trajectory which is feasible for a unicycle-type vehicle is not necessarily feasible for a car-like vehicle. Also, in addition to monitoring the position $(x(t), y(t))$ of the robot, one may be willing to control the chassis orientation $\theta(t)$ at a desired reference value $\theta_r(t)$ associated with the orientation of the reference vehicle. For a nonholonomic unicycle-type robot, a reference trajectory $(x_r(t), y_r(t), \theta_r(t))$ is feasible if it is produced by a reference vehicle which has the same kinematic limitations as the physical robot. For instance, most trajectories produced by an omnidirectional vehicle (omnibile vehicle in the terminology of Chap. 24) are not feasible for a nonholonomic mobile robot. However, nonfeasibility does not imply that the reference trajectory cannot be tracked in an approximate manner, i. e., with small (although nonzero) tracking errors.

49.1.3 Stabilization of Fixed Postures

Let $\mathcal{F}_1$ denote a frame attached to the robot chassis. In this chapter, we call a robot posture (or situation) the association of the position of a point P located on the robot chassis with the orientation $\theta(t)$ of $\mathcal{F}_1$ with respect to a fixed frame $\mathcal{F}_0$ in the plane of motion. For

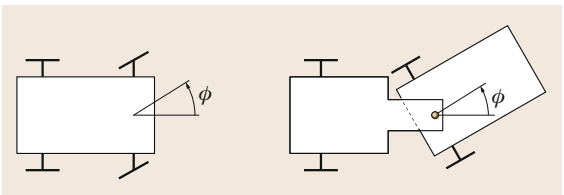


Fig. 49.3 Analogy car/unicycle with trailer

this last problem, the objective is to stabilize at zero the posture vector $\xi(t) = (x(t), y(t), \theta(t))$, with $(x(t), y(t))$ denoting the position of P expressed in $\mathcal{F}_0$. Although a fixed desired (or reference) posture is obviously a particular case of a feasible trajectory, this problem cannot be solved by classical control methods.

49.2 Control Models

49.2.1 Kinematics Versus Dynamics

In order to proceed with the control of mobile robots, a model describing its motion has to be derived. Several levels of motion equations may be derived, pending on the targeted application (expected velocity, terrain configuration, etc.). Chapter 24 provides a general configuration dynamic model for WMRs. Its particularization to the case of unicycle-type and car-like mobile robots gives

$$\mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{u}} + \boldsymbol{F}(\boldsymbol{q}, \boldsymbol{u})\boldsymbol{u} = \boldsymbol{\Gamma}(\phi)\boldsymbol{\tau} \,, \tag{49.1}$$

with $\boldsymbol{q}$ denoting a robot's configuration vector, $\boldsymbol{u}$ a vector of independent velocity variables associated with the robot's degrees of freedom, $\mathbf{H}(\boldsymbol{q})$ a reduced inertia matrix (which is invertible for any $\boldsymbol{q}$), $\boldsymbol{F}(\boldsymbol{q}, \boldsymbol{u})\boldsymbol{u}$ a vector of forces combining the contribution of Coriolis and wheel–ground contact forces, ϕ the orientation angle of the car's steering wheel, $\boldsymbol{\Gamma}$ an invertible control matrix (which is constant in the case of a unicycle-type vehicle), and $\boldsymbol{\tau}$ a vector of independent motor torques (whose dimension is equal to the number of degrees of freedom in the case of full actuation, i. e., equal to two for the vehicles considered herein). In the case of a unicycle-type vehicle, a configuration vector is composed of the components of the chassis posture vector $\boldsymbol{\xi}$ and the orientation angles of the castor wheels (with respect to the chassis). In the case of a car-like vehicle, a configuration vector is composed of the components of $\boldsymbol{\xi}$ and the steering wheel angle ϕ. To be complete, this dynamic model must be complemented with kinematic equations in the form

$$\dot{\boldsymbol{q}} = \mathbf{S}(\boldsymbol{q})\boldsymbol{u} \,, \tag{49.2}$$

from which one can extract a reduced kinematic model

$$\dot{\boldsymbol{z}} = \mathbf{B}(\boldsymbol{z})\boldsymbol{u} \,, \tag{49.3}$$

with $\boldsymbol{z} = \boldsymbol{\xi}$, in the case of a unicycle-type vehicle, and $\boldsymbol{z} = (\boldsymbol{\xi}, \phi)$ in the case of a car-like vehicle. In the automatic control terminology, the complete dynamic model (49.1) and (49.2) forms a *control system* which can be written as

$$\dot{\boldsymbol{X}} = f(\boldsymbol{X}, \boldsymbol{\tau}) \quad \text{with} \quad \boldsymbol{X} = (\boldsymbol{q}, \boldsymbol{u})$$

denoting the state vector of this system, and $\boldsymbol{\tau}$ the vector of control inputs. The kinematic models (49.2) and (49.3) are also control systems with respective state vectors $\boldsymbol{q}$ and $\boldsymbol{z}$ and control vector $\boldsymbol{u}$. Any of these models can be used for control design and analysis purposes. In the remainder of this chapter, we have chosen to work with the kinematic model (49.3). By analogy with the motion control of manipulator arms, this boils down to using a model with velocity control inputs, rather than a model with torque control inputs. This is justified when dynamic effects are not preponderant and low-level velocity control loops on the wheels motor are powerful enough to ensure good velocity tracking. In Sect. 49.5, however, when nonideal wheel-ground contact is addressed, dynamics is considered.

49.2.2 Modeling in an Absolute Frame

For the unicycle-type mobile robot, the kinematic model (49.3) used from now on is

$$\begin{cases} \dot{x} = u_1 \cos\theta \,, \\ \dot{y} = u_1 \sin\theta \,, \\ \dot{\theta} = u_2 \,, \end{cases} \tag{49.4}$$

where (x, y) represents the coordinates of the point $\mathrm{P_m}$ located at mid-distance of the actuated wheels, and the angle θ characterizes the robot's chassis orientation (Fig. 49.4). In this equation, u_1 represents the intensity of the vehicle's longitudinal velocity, and u_2 is the chassis instantaneous velocity of rotation. The variables u_1 and u_2 are themselves related to the angular velocity of the actuated wheels via the one-to-one relations

$$u_1 = \frac{r}{2}(\dot{\psi}_\mathrm{r} + \dot{\psi}_\ell) \,,$$
$$u_2 = \frac{r}{2R}(\dot{\psi}_\mathrm{r} - \dot{\psi}_\ell) \,,$$

with r the wheels' radius, R the distance between the two actuated wheels, and $\dot{\psi}_\mathrm{r}$ (resp. $\dot{\psi}_\ell$) the angular velocity of the right (resp. left) rear wheel.

For the car-like mobile robot, the kinematic model (49.3) used from now on is

$$\begin{cases} \dot{x} = u_1 \cos\theta \,, \\ \dot{y} = u_1 \sin\theta \,, \\ \dot{\theta} = \frac{u_1}{L} \tan\phi \,, \\ \dot{\phi} = u_2 \,, \end{cases} \tag{49.5}$$

where ϕ represents the vehicle's steering wheel angle, and L is the distance between the rear and front wheels' axles. In all forthcoming simulations, L is set equal to 1.2 m.

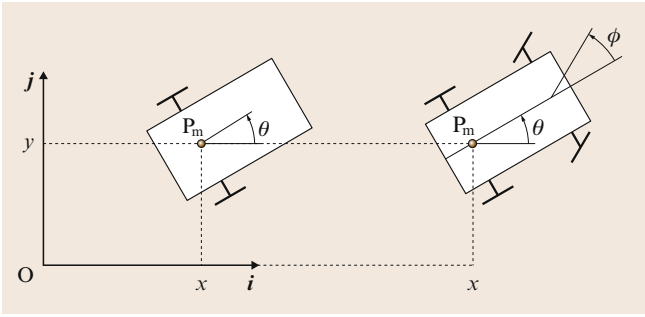


Fig. 49.4 Configuration variables

49.2.3 Modeling in a Frénet Frame

The object of this subsection is to generalize the previous kinematic equations when the reference frame is a Frénet frame. This generalization will be used later on when addressing the path following problem.

Let us consider a curve C in the plane of motion, as illustrated on Fig. 49.5, and let us define three frames $\mathcal{F}_0$, $\mathcal{F}_m$, and $\mathcal{F}_s$, as follows. $\mathcal{F}_0 = \{0, \boldsymbol{i}, \boldsymbol{j}\}$ is a fixed frame, $\mathcal{F}_m = \{P_m, \boldsymbol{i}_m, \boldsymbol{j}_m\}$ is a frame attached to the mobile robot with its origin – the point P_m – located on the rear wheels axle, at the mid-distance of the wheels, and $\mathcal{F}_s = \{P_s, \boldsymbol{i}_s, \boldsymbol{j}_s\}$, which is indexed by the curve's curvilinear abscissa s, is such that the unit vector $\boldsymbol{i}_s$ tangents C. Consider now a point P attached to the robot chassis, and let (l_1, l_2) denote the coordinates of P expressed in the basis of $\mathcal{F}_m$. To determine the equations of motion of P with respect to the curve C let us introduce three variables s, d, and θ_e, defined as follows:

- s is the curvilinear abscissa at the point P_s obtained by projecting P orthogonally on C. This point exists and is unique if the point P is close enough to the curve. More precisely, it suffices that the distance between P and the curve be smaller than the lower bound of the curve radii. We will assume that this condition is satisfied.
- d is the ordinate of P in the frame $\mathcal{F}_s$; its absolute value is also the distance between P and the curve.
- $\theta_e = \theta - \theta_s$ is the angle characterizing the orientation of the robot chassis with respect to the frame $\mathcal{F}_s$.

We also define the curvature $c(s)$ of C at P_s, i. e., $c(s) = \partial\theta_s / \partial s$.

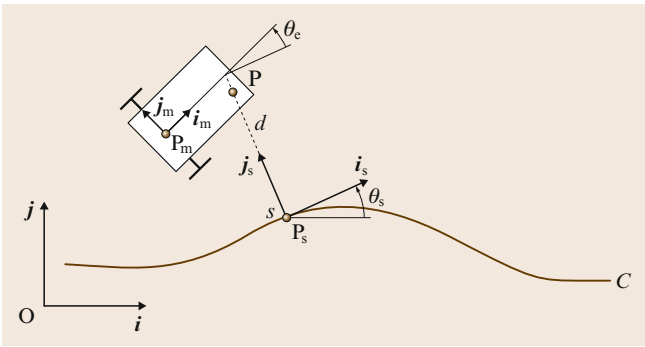


Fig. 49.5 Representation in a Frénet frame

Using these notations one easily deduces from (49.4) the following system of equations ([49.2] for details), which generalizes (49.4)

$$\begin{cases} \dot{s} = \frac{1}{1 - dc(s)} \left[(u_1 - l_2 u_2) \cos\theta_e - l_1 u_2 \sin\theta_e \right] , \\ \dot{d} = (u_1 - l_2 u_2) \sin\theta_e + l_1 u_2 \cos\theta_e \,, \\ \dot{\theta}_e = u_2 - \dot{s} c(s) \,. \end{cases} \tag{49.6}$$

For car-like vehicles, one easily verifies, by using (49.5), that the system (49.6) becomes

$$\begin{cases} \dot{s} = \frac{u_1}{1 - dc(s)} \left[\cos\theta_e - \frac{\tan\phi}{L} (l_2 \cos\theta_e + l_1 \sin\theta_e) \right] , \\ \dot{d} = u_1 \left[\sin\theta_e + \frac{\tan\phi}{L} (l_1 \cos\theta_e - l_2 \sin\theta_e) \right] , \\ \dot{\theta}_e = \frac{u_1}{L} \tan\phi - \dot{s} c(s) \,, \\ \dot{\phi} = u_2 \,. \end{cases} \tag{49.7}$$

49.3 Adaptation of Control Methods for Holonomic Systems

We address in this section the problems of trajectory stabilization and path following. When we defined these problems in the introduction, we considered a reference point P attached to the robot chassis. It turns out that the choice of this point is important. Indeed, consider for instance the equations (49.6) for a unicycle point P when C is the axis $(O, \boldsymbol{i})$. Then, $s = x_P$, $d = y_P$, and $\theta_e = \theta$ represent the robot's posture with respect to the fixed reference frame $\mathcal{F}_0$. There are two possible cases depending on whether P is, or is not, located on the actuated wheels axle. Let us consider the first case, for which $l_1 = 0$. From the first two equations of (49.6), one has

$$\dot{x}_P = (u_1 - l_2 u_2)\cos\theta\,, \quad \dot{y}_P = (u_1 - l_2 u_2)\sin\theta\,.$$

These relations indicate that P can move only in the direction of the vector $(\cos\theta, \sin\theta)$. This is a direct consequence of the nonholonomy constraint to which the vehicle is subjected. Now, if P is not located on the wheels axle, then

$$\begin{pmatrix}\dot{x}_P\\ \dot{y}_P\end{pmatrix} = \begin{pmatrix}\cos\theta & -l_1\sin\theta\\ \sin\theta & l_1\cos\theta\end{pmatrix}\begin{pmatrix}1 & -l_2\\ 0 & 1\end{pmatrix}\begin{pmatrix}u_1\\ u_2\end{pmatrix}. \tag{49.8}$$

The fact that the two square matrices in the right-hand side of this equality are invertible indicates that $\dot{x}_P$ and $\dot{y}_P$ can take any values, and thus that the motion of P is not constrained. By analogy with holonomic manipulator arms, this means that P may be seen as the extremity of a two-degree-of-freedom (2-DOF) manipulator, and thus that it can be controlled by applying the same control laws as those used for manipulators. In this section, we assume that the point P, used to characterize the robot's position, is chosen away from the rear wheels axle. In this case we will see that the problems of trajectory stabilization and path following can be solved very simply. However, as shown in the subsequent section, choosing P on the wheels axle may also be of interest in order to better control the vehicle's orientation.

49.3.1 Stabilization of Trajectories for a Nonconstrained Point

Unicycle

Consider a differentiable reference trajectory $t \longmapsto (x_r(t), y_r(t))$ in the plane. Let $e = (x_P - x_r, y_P - y_r)$ denote the tracking error in position. The control objective is to asymptotically stabilize this error at zero. In view of (49.8), the error equations are

$$\dot{e} = \begin{pmatrix}\cos\theta & -l_1\sin\theta\\ \sin\theta & l_1\cos\theta\end{pmatrix}\begin{pmatrix}u_1 - l_2 u_2\\ u_2\end{pmatrix} - \begin{pmatrix}\dot{x}_r\\ \dot{y}_r\end{pmatrix}. \tag{49.9}$$

Introducing new control variables (v_1, v_2) defined by

$$\begin{pmatrix}v_1\\ v_2\end{pmatrix} = \begin{pmatrix}\cos\theta & -l_1\sin\theta\\ \sin\theta & l_1\cos\theta\end{pmatrix}\begin{pmatrix}u_1 - l_2 u_2\\ u_2\end{pmatrix}, \tag{49.10}$$

the equations (49.9) become simply

$$\dot{e} = \begin{pmatrix}v_1\\ v_2\end{pmatrix} - \begin{pmatrix}\dot{x}_r\\ \dot{y}_r\end{pmatrix}.$$

The classical techniques of stabilization for linear systems can then be used. For instance, one may consider a proportional feedback control with precompensation such as

$$v_1 = \dot{x}_r - k_1 e_1 = \dot{x}_r - k_1(x_P - x_r)\,, \quad (k_1 > 0)\,,$$
$$v_2 = \dot{y}_r - k_2 e_2 = \dot{y}_r - k_2(y_P - y_r)\,, \quad (k_2 > 0)\,,$$

which yields the closed-loop equation $\dot{e} = -Ke$. Of course, this control can be rewritten for the initial control variables $\boldsymbol{u}$, since the mapping $(u_1, u_2) \longmapsto (v_1, v_2)$ is bijective.

Car

This technique extends to car-like vehicles (and also to trailer systems) by choosing a point P attached to the steering wheel frame and not located on the steering wheel axle.

49.3.2 Path Following with no Orientation Control

Unicycle

Let us adopt the notation of Fig. 49.5 to address the problem of following a path associated with a curve C in the plane. The control objective is to stabilize the distance d at zero. From (49.6), one has

$$\dot{d} = u_1\sin\theta_e + u_2(-l_2\sin\theta_e + l_1\cos\theta_e)\,. \tag{49.11}$$

Recall that in this case the vehicle's longitudinal velocity u_1 is either imposed or prespecified. We will assume that the product $l_1 u_1$ is positive, i. e., the position of the point P with respect to the actuated wheels axle is chosen in relation to the sign of u_1. This assumption will

be removed in Sect. 49.4. To simplify, we will also assume that $l_2 = 0$, i. e., the point P is located on the axis $(P_m, \boldsymbol{i}_m)$. Let us then consider the following feedback control law

$$u_2 = -\frac{u_1 \tan\theta_e}{l_1} - \frac{u_1}{\cos\theta_e} k(d, \theta_e) d \,, \qquad (49.12)$$

with k a continuous, strictly positive, function on $\mathbb{R} \times (-\pi/2, \pi/2)$ such that $k(d, \pm\pi/2) = 0$. Since $l_2 = 0$, applying the control (49.12) to (49.11) gives

$$\dot{d} = -l_1 u_1 k(d, \theta_e) d \,,$$

which suggests that the control law (49.12) ensures convergence of d to zero under adequate conditions on l_1, u_1, and k. This is made precise in the following result ([49.2] for details).

Proposition 49.1

Consider the path following problem for a unicycle-type mobile robot with

A. A strictly positive, or strictly negative, longitudinal velocity u_1.

B. A reference point P of coordinates $(l_1, 0)$ in the vehicle's chassis frame, with $l_1 u_1 > 0$.

Let k denote a continuous function, strictly positive on $\mathbb{R} \times (-\pi/2, \pi/2)$, and such that $k(d, \pm\pi/2) = 0$ for every d (for instance, $k(d, \theta_e) = k_0 \cos\theta_e$). Then, for any initial conditions $(s(0), d(0), \theta_e(0))$ such that

$$\theta_e(0) \in (-\pi/2, \pi/2) \,, \quad \frac{l_1 c_{\max}}{1 - |d(0)| c_{\max}} < 1 \,,$$

with $c_{\max} = \max_s |c(s)|$, the feedback control (49.12) makes the distance $|d|$ between P and the curve non-increasing, and makes it converge to zero if

$$\int_0^t |u_1(s)| \mathrm{d}s \longrightarrow +\infty \quad \text{when } t \longrightarrow +\infty \,.$$

Car

This control technique also applies to car (or trailer systems) by considering a point P attached to the steering wheel frame, with u_1 positive.

49.4 Methods Specific to Nonholonomic Systems

The control methods recalled in the previous section are simple and sufficient for many applications. Their main limitation comes from the fact that the robot's orientation is not actively controlled. This becomes a problem when the application requires manœuvers (i. e., when the vehicle's velocity u_1 has to become negative). This issue is especially critical for trailer systems (including car-like vehicles). Indeed, when the longitudinal velocity is positive, the leader vehicle has a *pulling* action which tends to align the followers along the curve. In the other case, the leader has a pushing action which tends to misalign them, thus leading to collisions between the vehicles' components (the jackknife effect, see Fig. 49.6 for illustration). In order to remove this constraint on the sign of the longitudinal velocity, the control has to be designed so that all orientation angles are actively stabilized. An indirect way to do this consists in choosing the point P on the actuated wheels axle, at the mid-distance of the wheels, for instance. In this case, the nonholonomy constraints intervene much more explicitly, and the control can no longer be obtained by applying the techniques used for holonomic manipulators.

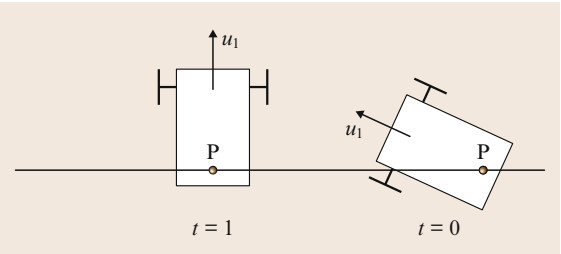


Fig. 49.6 Path-following instability with reverse longitudinal velocity

This section is organized as follows. First, the modeling equations with respect to a Frénet frame are recast into a canonical form called the chained form. From there, a solution to the path-following problem with active stabilization of the vehicle's orientation is worked out. The problem of (feasible) trajectory stabilization is also revisited with the complementary objective of controlling the vehicle's orientation. The asymptotic stabilization of fixed postures is then addressed. Finally, some comments on the limitations of the proposed control strategies, in relation to the objective of asymptotic stabilization, serve to motivate and introduce a new control approach developed in the subsequent section.

49.4.1 Transformation of Kinematic Models into the Chained Form

The chapter dedicated to path planning shows how the kinematic equations of the mobile robots here considered (unicycle-type, car-like, with trailers) can be transformed into the chain form via a change of state and control variables. In particular, the equations of a unicycle (49.4), and those of a car (49.5), can be transformed into a three-dimensional and a four-dimensional chained system, respectively. Those of a unicycle-type vehicle with N trailers yield a chained system of dimension $N+3$ when the trailers are hooked to each other in a specific way. As shown below, this transformation can be generalized to the kinematic models derived with respect to a Frénet frame. The result will be given only for the unicycle and car cases (49.6) and (49.7), but it also holds for trailer systems. The reference point P is now chosen at the mid-distance of the vehicle's rear wheels (or at the mid-distance of the wheels of the last trailer, when trailers are involved).

Let us start with the unicycle case. Under the assumption that P corresponds to the origin of $\mathcal{F}_m$, one has $l_1 = l_2 = 0$ so that the system (49.6) simplifies to

$$\begin{cases} \dot{s} = \dfrac{u_1}{1-dc(s)} \cos\theta_e \,, \\ \dot{d} = u_1 \sin\theta_e \,, \\ \dot{\theta}_e = u_2 - \dot{s}c(s) \,. \end{cases} \tag{49.13}$$

Let us determine a change of coordinates and control variables $(s, d, \theta_e, u_1, u_2) \longmapsto (z_1, z_2, z_3, v_1, v_2)$ allowing to (locally) transform (49.13) into the three-dimensional chained system

$$\begin{cases} \dot{z}_1 = v_1 \,, \\ \dot{z}_2 = v_1 z_3 \,, \\ \dot{z}_3 = v_2 \,. \end{cases} \tag{49.14}$$

By first setting

$$z_1 = s \,, \quad v_1 = \dot{s} = \frac{u_1}{1-dc(s)} \cos\theta_e \,,$$

we already obtain $\dot{z}_1 = v_1$. This implies that

$$\begin{aligned} \dot{d} = u_1 \sin\theta_e &= \frac{u_1}{1-dc(s)} \cos\theta_e [1-dc(s)] \tan\theta_e \\ &= v_1 [1-dc(s)] \tan\theta_e \,. \end{aligned}$$

We then set $z_2 = d$ and $z_3 = [1-dc(s)]\tan\theta_e$, so that the above equation becomes $\dot{z}_2 = v_1 z_3$. Finally, we define

$$\begin{aligned} v_2 &= \dot{z}_3 \\ &= \left[-\dot{d}c(s) - d\frac{\partial c}{\partial s}\dot{s} \right] \tan\theta_e \\ &\quad + [1-dc(s)](1+\tan^2\theta_e)\dot{\theta}_e \,. \end{aligned}$$

The equations (49.14) are satisfied with the variables z_i and v_i so defined.

From this construction it is simple to verify that the mapping $(s, d, \theta_e) \longmapsto z$ is a *local* change of coordinates defined on $\mathbb{R}^2 \times (-\pi/2, \pi/2)$ (to be more rigorous, one should also take the constraint $|d| < 1/c(s)$ into account). Let us finally remark that the change of control variables involves the derivative $(\partial c/\partial s)$ of the path's curvature (whose knowledge is thus needed for the calculations). In summary, we have shown that the change of coordinates and of control variables $(s, d, \theta_e, u_1, u_2) \longmapsto (z_1, z_2, z_3, v_1, v_2)$ defined by

$$\begin{aligned} (z_1, z_2, z_3) &= (s, d, [1-dc(s)]\tan\theta_e) \\ (v_1, v_2) &= (\dot{z}_1, \dot{z}_3) \end{aligned}$$

transforms the model (49.13) of a unicycle-type vehicle into a three-dimensional (3-D) chained system. One can similarly transform the car's equations into a four-dimensional (4-D) chained system, although the calculations are slightly more cumbersome. More precisely, the change of coordinates and control variables $(s, d, \theta_e, \phi, u_1, u_2) \longmapsto (z_1, z_2, z_3, z_4, v_1, v_2)$ defined by

$$\begin{aligned} (z_1, z_2, z_3, z_4) = \Big\{ & s, d, [1-dc(s)]\tan\theta_e, \\ & -c(s)[1-dc(s)]\left(1+2\tan^2\theta_e\right) \\ & -d\frac{\partial c}{\partial s}\tan\theta_e \\ & +[1-dc(s)]^2 \frac{\tan\phi}{L}\frac{1+\tan^2\theta_e}{\cos\theta_e} \Big\} \,, \\ (v_1, v_2) = (\dot{z}_1, \dot{z}_4) & \end{aligned}$$

transforms the model (49.7) of a car-like vehicle (with $l_1 = l_2 = 0$) into a four-dimensional chained system.

49.4.2 Tracking of a Reference Vehicle with the Same Kinematics

Let us first consider the problem of tracking, in both position and orientation, a reference vehicle (Fig. 49.7). For simplicity, we choose P as the origin P_m of the robot's chassis frame $\mathcal{F}_m$.

Although the terminology is rather loose, the *tracking problem* is usually associated, in the control literature, with the problem of *asymptotically stabilizing* the reference trajectory. In this case, a necessary condition for the existence of a control solution is that the reference is feasible. Feasible trajectories $t \longmapsto (x_r(t), y_r(t), \theta_r(t))$ are smooth time functions which are solution to the robot's kinematic model for some specific control input $t \longmapsto \boldsymbol{u}_r(t) = (u_{1,r}(t), u_{2,r}(t))^T$, called the *reference control*. For a unicycle-type robot for example, this means in view of (49.4) that

$$\begin{cases} \dot{x}_r = u_{1,r} \cos\theta_r \,, \\ \dot{y}_r = u_{1,r} \sin\theta_r \,, \\ \dot{\theta}_r = u_{2,r} \,. \end{cases} \tag{49.15}$$

In other words, feasible reference trajectories correspond to the motion of a reference frame $\mathcal{F}_r = \{P_r, \boldsymbol{i}_r, \boldsymbol{j}_r\}$ rigidly attached to a reference unicycle-type robot, with P_r (alike $P = P_m$) located at the mid-distance of the actuated wheels (Fig. 49.7). From there, the problem is to determine a feedback control which asymptotically stabilizes the tracking error $(x - x_r, y - y_r, \theta - \theta_r)$ at zero, with (x_r, y_r) being the coordinates of P_r in $\mathcal{F}_0$, and θ_r the oriented angles between $\boldsymbol{i}$ and $\boldsymbol{i}_r$. One can proceed as in the path-following case, first by establishing the error equations with respect to the frame $\mathcal{F}_r$, then by transforming these equations into the chain form via a change of variables like the one used to transform the kinematic equations of a mobile robot into a chained system, and finally by designing stabilizing control laws for the transformed system.

Expressing the tracking error in position $(x - x_r, y - y_r)$ with respect to the frame $\mathcal{F}_r$ gives the vector (Fig. 49.7)

$$\begin{pmatrix} x_e \\ y_e \end{pmatrix} = \begin{pmatrix} \cos\theta_r & \sin\theta_r \\ -\sin\theta_r & \cos\theta_r \end{pmatrix} \begin{pmatrix} x - x_r \\ y - y_r \end{pmatrix} . \tag{49.16}$$

Calculating the time derivative of this vector yields

$$\begin{pmatrix} \dot{x}_e \\ \dot{y}_e \end{pmatrix} = \begin{pmatrix} u_{2,r} y_e + u_1 \cos(\theta - \theta_r) - u_{1,r} \\ -u_{2,r} x_e + u_1 \sin(\theta - \theta_r) \end{pmatrix} .$$

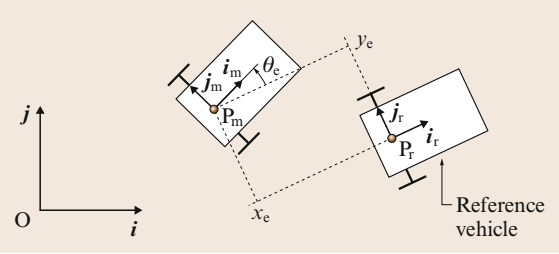


Fig. 49.7 Reference vehicle and error coordinates

By denoting $\theta_e = \theta - \theta_r$, the orientation error between the frames $\mathcal{F}_m$ and $\mathcal{F}_r$, we obtain

$$\begin{cases} \dot{x}_e = u_{2,r} y_e + u_1 \cos\theta_e - u_{1,r} \,, \\ \dot{y}_e = -u_{2,r} x_e + u_1 \sin\theta_e \,, \\ \dot{\theta}_e = u_2 - u_{2,r} \,. \end{cases} \tag{49.17}$$

To determine a control (u_1, u_2) which asymptotically stabilizes the error (x_e, y_e, θ_e) at zero, let us consider the following change of coordinates and control variables

$$(x_e, y_e, \theta_e, u_1, u_2) \longmapsto (z_1, z_2, z_3, w_1, w_2) \,,$$

defined by

$$\begin{aligned} z_1 &= x_e \,, \\ z_2 &= y_e \,, \\ z_3 &= \tan\theta_e \,, \\ w_1 &= u_1 \cos\theta_e - u_{1,r} \,, \\ w_2 &= \frac{u_2 - u_{2,r}}{\cos^2\theta_e} \,. \end{aligned} \tag{49.18}$$

Note that, around zero, this mapping is only defined when $\theta_e \in (-\pi/2, \pi/2)$. In other words, the orientation error between the physical robot and the reference robot has to be smaller than $\pi/2$.

It is immediate to verify that, in the new variables, the system (49.17) can be written as

$$\begin{cases} \dot{z}_1 = u_{2,r} z_2 + w_1 \,, \\ \dot{z}_2 = -u_{2,r} z_1 + u_{1,r} z_3 + w_1 z_3 \,, \\ \dot{z}_3 = w_2 \,. \end{cases} \tag{49.19}$$

We remark that the last term in each of the above three equations corresponds to the one of a chained system. From here, it can be shown that the feedback control law

$$\begin{cases} w_1 = -k_1 |u_{1,r}| (z_1 + z_2 z_3) \quad (k_1 > 0) \,, \\ w_2 = -k_2 u_{1,r} z_2 - k_3 |u_{1,r}| z_3 \,, \quad (k_2, k_3 > 0) \,, \end{cases} \tag{49.20}$$

makes the origin of system (49.19) globally asymptotically stable provided essentially that $u_{1,r}$ does not tend to zero as t tends to infinity. A precise stability statement and associated proof are provided in [49.2], together with complementary details about the gain tuning. Finally, the velocity control inputs u_1 and u_2 can be deduced from w_1 and w_2 by using the two last lines of (49.18).

Generalization to a Car-Like Vehicle

The previous method extends to the car case. We provide below the main steps of this extension. Consider the car's kinematic model (49.5) and a reference vehicle satisfying the same kinematic equations with all variables indexed by the subscript r. We assume that there exists $\delta \in (0, \pi/2)$ such that the reference steering angle ϕ_r belongs to the interval $[-\delta, \delta]$. By defining x_e, y_e, and θ_e as in the unicycle case, and by setting $\phi_e = \phi - \phi_r$, one obtains the following system

$$\begin{cases} \dot{x}_e = \left(\dfrac{u_{1,r}}{L}\tan\phi_r\right) y_e + u_1 \cos\theta_e - u_{1,r}\,, \\ \dot{y}_e = -\left(\dfrac{u_{1,r}}{L}\tan\phi_r\right) x_e + u_1 \sin\theta_e\,, \\ \dot{\theta}_e = \dfrac{u_1}{L}\tan\phi - \dfrac{u_{1,r}}{L}\tan\phi_r\,, \\ \dot{\phi}_e = u_2 - u_{2,r}\,. \end{cases} \tag{49.21}$$

Let us introduce the following new state variables

$$\begin{cases} z_1 = x_e\,, \\ z_2 = y_e\,, \\ z_3 = \tan\theta_e\,, \\ z_4 = \dfrac{\tan\phi - \cos\theta_e \tan\phi_r}{L\cos^3\theta_e} + k_2 y_e\,, (k_2 > 0)\,. \end{cases}$$

We note that for any $\phi_r \in (-\pi/2, \pi/2)$, the mapping $(x_e, y_e, \theta_e, \phi) \longmapsto z$ defines a diffeomorphism between $\mathbb{R}^2 \times (-\pi/2, \pi/2)^2$ and $\mathbb{R}^4$. Introduce now the new control variables

$$\begin{cases} w_1 = u_1 \cos\theta_e - u_{1,r}\,, \\ w_2 = \dot{z}_4 = k_2 \dot{y}_e + \left(\dfrac{3\tan\phi}{\cos\theta_e} - 2\tan\phi_r\right)\dfrac{\sin\theta_e}{L\cos^3\theta_e}\dot{\theta}_e \\ \quad - \dfrac{u_{2,r}}{L\cos^2\phi_r\cos^2\theta_e} + \dfrac{u_2}{L\cos^2\phi\cos^3\theta_e}\,. \end{cases} \tag{49.22}$$

One shows that $(u_1, u_2) \longmapsto (w_1, w_2)$ defines a change of variables for θ_e, ϕ, and ϕ_r, inside the interval $(-\pi/2, \pi/2)$. These changes of state and control variables transform the system (49.21) into

$$\begin{cases} \dot{z}_1 = \left(\dfrac{u_{1,r}}{L}\tan\phi_r\right) z_2 + w_1\,, \\ \dot{z}_2 = -\left(\dfrac{u_{1,r}}{L}\tan\phi_r\right) z_1 + u_{1,r} z_3 + w_1 z_3\,, \\ \dot{z}_3 = -k_2 u_{1,r} z_2 + u_{1,r} z_4 \\ \quad + w_1\left[z_4 - k_2 z_2 + (1 + z_3^2)\dfrac{\tan\phi_r}{L}\right]\,, \\ \dot{z}_4 = w_2\,. \end{cases} \tag{49.23}$$

From here, one can deduce that the control law w_1 and w_2 defined as

$$\begin{cases} w_1 = -k_1 |u_{1,r}| \\ \qquad \times \left\{ z_1 + \dfrac{z_3}{k_2}\left[z_4 + (1 + z_3^2)\dfrac{\tan\phi_r}{L}\right]\right\}\,, \\ w_2 = -k_3 u_{1,r} z_3 - k_4 |u_{1,r}| z_4\,, \end{cases} \tag{49.24}$$

with $k_{1,2,3,4}$ denoting positive numbers, makes the origin of system (49.23) globally asymptotically stable provided essentially that $u_{1,r}$ does not tend to zero as t tends to infinity.

Simulation Results

The simulation shown on Fig. 49.8 and VIDEO 181 illustrates this control scheme. The gain parameters k_i have been chosen as $(k_1, k_2, k_3, k_4) = (1, 1, 3, 3)$. The initial configuration of the reference vehicle (i. e., at $t = 0$), which is represented in Fig. 49.8a by dashed lines, is $(x_r, y_r, \theta_r)(0) = (0, 0, 0)$. The reference control $\boldsymbol{u}_r$ is defined by (49.25). The initial configuration of the controlled robot, represented in the figure by plain lines, is $(x, y, \theta)(0) = (0, -1.5, 0)$. The configurations at time $t = 10$, 20, and 30, are also represented on the figure. Due to the fast convergence of the tracking error to zero (see the time evolution of the components x_e, y_e, θ_e of the tracking error in Fig. 49.8b), one can basically consider that the configurations of both vehicles coincide after time $t = 10$.

$$u_r(t) = \begin{cases} (1, 0)^T\,, & \text{if } t \in [0, 10]\,, \\ \{-1, 0.5\cos[2\pi(t-10)/5]\}^T\,, \\ \text{if } t \in [10, 20]\,, \\ (1, 0)^T\,, & \text{if } t \in [20, 30]\,. \end{cases} \tag{49.25}$$

49.4.3 Path Following with Orientation Control

We reconsider the path-following problem with the reference point P now located on the actuated wheels axle, at the mid-distance of the wheels. The objective is to synthesize a control law which allows the vehicle to follow the path in a stable manner, independently of the sign of the longitudinal velocity.

Unicycle Case

We have seen in Sect. 49.4.1 how to transform kinematic equations with respect to a Frénet frame into the

three-dimensional chained system

$$\begin{cases} \dot{z}_1 = v_1 \,, \\ \dot{z}_2 = v_1 z_3 \,, \\ \dot{z}_3 = v_2 \,. \end{cases} \tag{49.26}$$

Recall that $(z_1, z_2, z_3) = (s, d, (1 - dc(s)) \tan \theta_e)$ and that $v_1 = u_1/(1 - dc(s)) \cos \theta_e$. The objective is to determine a control law which asymptotically stabilizes $(d = 0, \theta_e = 0)$ and also ensures that the constraint on the distance d to the path (i. e., $|dc(s)| < 1$) is satisfied along the trajectories of the controlled system. For the control law, a first possibility consists in considering a proportional feedback like

$$v_2 = -v_1 k_2 z_2 - |v_1| k_3 z_3 \,, \quad (k_2, k_3 > 0) \,. \tag{49.27}$$

It is straightforward to verify that the origin of the closed-loop subsystem

$$\begin{cases} \dot{z}_2 = v_1 z_3 \,, \\ \dot{z}_3 = -v_1 k_2 z_2 - |v_1| k_3 z_3 \end{cases} \tag{49.28}$$

is asymptotically stable when v_1 is constant, either positive or negative. Since u_1 (not v_1) is the intensity of the vehicle's longitudinal velocity, one would rather establish stability conditions which depend on u_1. As detailed in [49.2], the control law (49.27) in fact makes the origin of system (49.28) asymptotically stable under the main condition that u_1 does not tend to zero a t tends to infinity. Note that the sign of u_1 now does not need to be constant. As for the constraint $|dc(s)| < 1$, it is satisfied along any solution to the controlled system provided that

$$z_2^2(0) + \frac{1}{k_2} z_3^2(0) < \frac{1}{c_{\max}^2} \,,$$

with $c_{\max} = \max_s |c(s)|$.

From a practical point of view it can be useful to complement the control action with an integral term. More precisely, let us define a variable z_0 by $\dot{z}_0 = v_1 z_2$ with $z_0(0) = 0$. The control (49.27) can be modified as follows

$$\begin{aligned} v_2 &= -|v_1| k_0 z_0 - v_1 k_2 z_2 - |v_1| k_3 z_3 \,, \\ &= -|v_1| k_0 \int_0^t v_1 z_2 - v_1 k_2 z_2 - |v_1| k_3 z_3 \,, \\ &(k_0, k_2, k_3 > 0, k_0 < k_2 k_3) \end{aligned} \tag{49.29}$$

leading to similar stability results as before. The constraint $|dc(s)| < 1$ is now satisfied along any solution to the controlled system provided that

$$z_2^2(0) + \frac{1}{k_2 - \frac{k_0}{k_3}} z_3^2(0) < \frac{1}{c_{\max}^2} \,.$$

Generalization

One of the assets of this type of approach, besides the simplicity of the control law and little demanding conditions of stability associated with it, is that it can be generalized in a straightforward manner to car-like vehicles and unicycle-type vehicles with trailers. The result is summarized hereafter by considering an n-dimensional chained system

$$\begin{cases} \dot{z}_1 = v_1 \,, \\ \dot{z}_2 = v_1 z_3 \,, \\ \vdots \\ \dot{z}_{n-1} = v_1 z_n \,, \\ \dot{z}_n = v_2 \,, \end{cases} \tag{49.30}$$

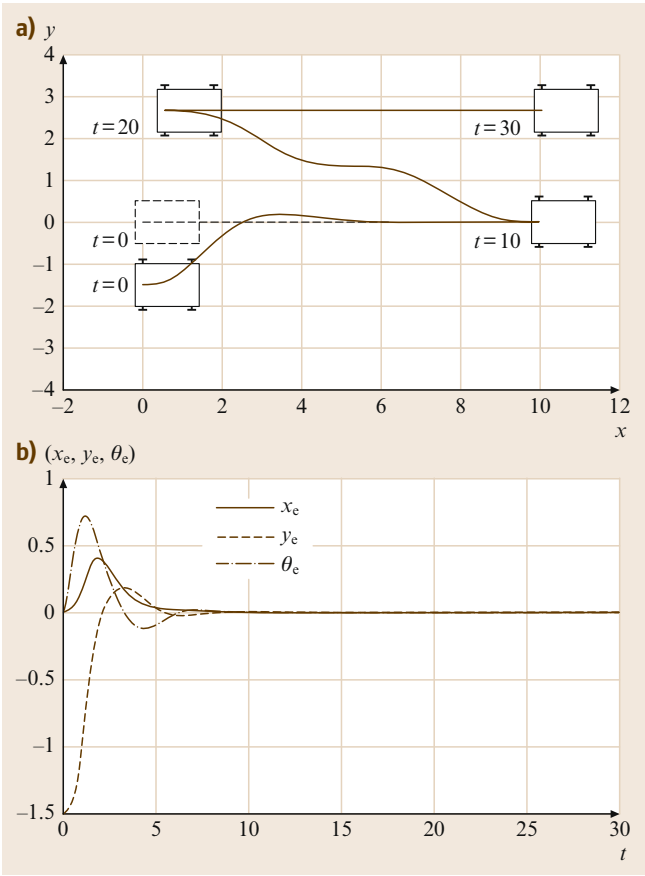


Fig.49.8a,b Tracking of a reference vehicle. **(a)** Cartesian motion; **(b)** error coordinates versus time

with $n \geq 3$. Its proof is a direct extension of the one in the three-dimensional case. The dimension $n = 4$ corresponds to the car case (Sect. 49.4.1). As for a unicycle-type vehicle with N trailers, one has $n = N + 3$. Recall also that, in all cases, z_2 represents the distance d between the path and the point P located at the mid-distance of the rear wheels of the last vehicle. The control objective is then to ensure the convergence of $z_2, \ldots, z_n$ to zero. A solution to this problem, which extends the solution (49.27) proposed for the unicycle, is given by

$$v_2 = -v_1 \sum_{i=2}^{n} \text{sign}(v_1)^{n+1-i} k_i z_i \,. \tag{49.31}$$

where the control gains $k_2, \ldots, k_n$ are chosen such that the polynomial

$$s^{n-1} + k_n s^{n-2} + k_{n-1} s^{n-3} + \ldots + k_3 s + k_2$$

is Hurwitz stable. Satisfaction of the constraint $|dc(s)| < 1$ along any solution to the controlled system can also be guaranteed, upon a condition on $(z_2(0), \cdots, z_n(0))$ specified in [49.2]. The possibility of adding an integral correction term in (49.31) is also addressed in [49.2].

Simulation Results

The simulation results reported in Fig. 49.9 and VIDEO 181 illustrate how this control scheme performs for a car-like vehicle. The reference curve is the circle of radius equal to four, centered at the origin. The robot's longitudinal velocity u_1 is defined by $u_1 = 1$ for $t \in [0, 5]$, and $u_1 = -1$ for $t > 5$. The control gains have been chosen as $(k_2, k_3, k_4) = (1, 3, 3)$. The motion of the car-like robot in the plane is represented in Fig. 49.9a, and its configuration at times $t = 0, 5$, and 25 are also depicted in the figure. The time evolution of the variables z_2, z_3, z_4 (defined at the end of Sect. 49.4.1) is represented in Fig. 49.9b. One can observe that the (discontinuous) change of the longitudinal velocity u_1 at $t = 5$ does not affect the convergence of these variables to zero.

49.4.4 Asymptotic Stabilization of Fixed Postures

We now consider the problem of asymptotic stabilization of a fixed desired (reference) posture (i. e., position and orientation) for the robot chassis. This problem may be seen as a limit case of the trajectory tracking problem. However, none of the feedback controllers proposed previously in this chapter provides a solution to this problem. From the automatic control point of view, the asymptotic stabilization of fixed postures is very different from the problems of path following and trajectory tracking with nonzero longitudinal velocity, much in the same way as a human driver knows, from experience, that parking a car at a precise location involves techniques and skills different from those exercised when cruising on a road. In particular, it cannot be solved by any classical control method for linear systems (or based on linearization). Technically, the underlying general problem is the one of asymptotic stabilization of equilibria of controllable driftless systems with less control inputs than state variables. This problem has motivated numerous studies during the last decade of the last century, from many authors and with various angles of attack, and it has remained a subject of active research at the beginning of this century.

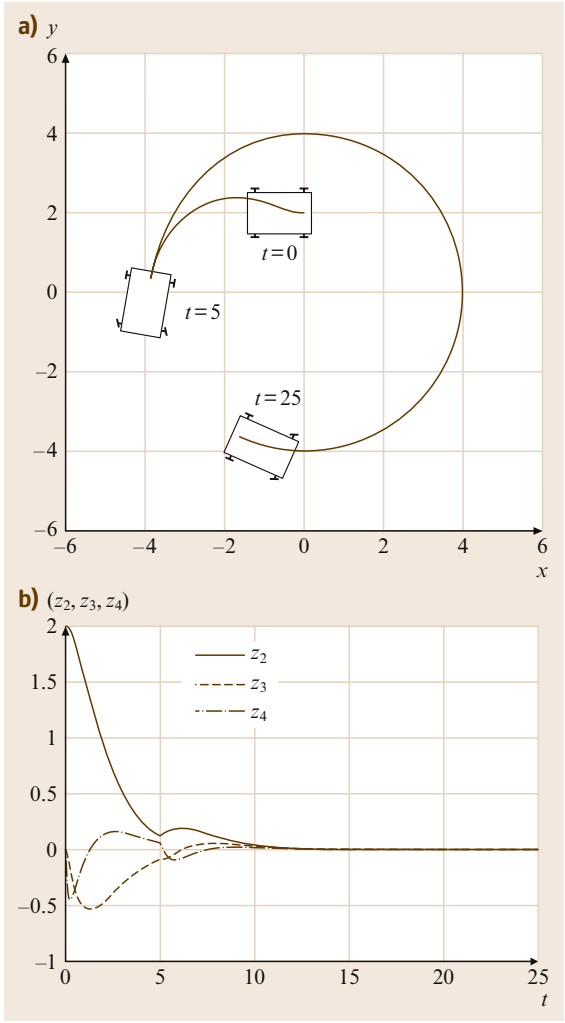


Fig.49.9a,b Path following along a circle. **(a)** Cartesian motion; **(b)** coordinates $z_{2,3,4}$ versus time

The variety of candidate solutions proposed until now, the mathematical technicalities associated with several of them, together with unsolved difficulties and limitations, particularly (but not only) in terms of robustness (an issue on which we will return), prevent us from attempting to cover the subject exhaustively. Instead, we have opted for a somewhat informal exposition of approaches which have been considered, with the illustration of a few control solutions, without going into technical and mathematical details.

A central aspect of the problem, which triggered much of the subsequent research on the control of nonholonomic systems, is that asymptotic stabilization of equilibria (or fixed points) cannot be achieved by using continuous feedbacks which depend on the state only (i. e., continuous pure-state feedbacks). This is a consequence of an important result due to Brockett in 1983. The original result by Brockett concerned differentiable feedbacks; it has later been extended to the larger set of feedbacks which are only continuous.

Theorem 49.1 (Brockett [49.3])
Consider a control system

$$\dot{x} = f(\boldsymbol{x}, \boldsymbol{u}) \quad (\boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{u} \in \mathbb{R}^m) \,,$$

with f a differentiable function and $(\boldsymbol{x}, \boldsymbol{u}) = (0, 0)$ an equilibrium of this system. A necessary condition for the existence of a continuous feedback control $\boldsymbol{u}(\boldsymbol{x})$ which renders the origin of the closed-loop system

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}(\boldsymbol{x}))$$

asymptotically stable is the local surjectivity of the application $(\boldsymbol{x}, \boldsymbol{u}) \longmapsto f(\boldsymbol{x}, \boldsymbol{u})$. More precisely, the image by f of any neighborhood Ω of $(0, 0)$ in $\mathbb{R}^{n+m}$ must be a neighborhood of 0 in $\mathbb{R}^n$.

This result implies that the equilibria of many controllable (nonlinear) systems are not asymptotically stabilizable by continuous pure-state feedbacks. All nonholonomic WMRs belong to this category of systems. This will be shown in the case of a unicycle-type vehicle; the proof for the other mobile robots is similar. Let us thus consider a unicycle-type vehicle, whose kinematic equations (49.4) can be written as $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$ with $\boldsymbol{x} = (x_1, x_2, x_3)^{\mathrm{T}}$, $\boldsymbol{u} = (u_1, u_2)^{\mathrm{T}}$, and $f(\boldsymbol{x}, \boldsymbol{u}) = (u_1 \cos x_3, u_1 \sin x_3, u_2)^{\mathrm{T}}$, and let us show that f is not locally onto in the neighborhood of $(\boldsymbol{x}, \boldsymbol{u}) = (0, 0)$. To this purpose, take a vector in $\mathbb{R}^3$ of the form $(0, \delta, 0)^{\mathrm{T}}$. It is obvious that the equation $f(\boldsymbol{x}, \boldsymbol{u}) = (0, \delta, 0)^{\mathrm{T}}$ does not have a solution in the neighborhood of $(\boldsymbol{x}, \boldsymbol{u}) = (0, 0)$ since the first equation, namely $u_1 \cos x_3 = 0$, implies that $u_1 = 0$, so that the second equation cannot have a solution if δ is different from zero.

It is also obvious that the linear approximation about the equilibrium $(\boldsymbol{x}, \boldsymbol{u}) = (0, 0)$ of the unicycle kinematic equations is not controllable. If it were, it would be possible to (locally) asymptotically stabilize this equilibrium with a linear (thus continuous) state feedback.

Therefore, by application of the above theorem, a unicycle-type mobile robot (like other nonholonomic robots) cannot be asymptotically stabilized at a desired posture (position/orientation) by using a continuous pure-state feedback. This impossibility has motivated the development of other control strategies in order to solve the problem. Three major types of controls have been considered:

1. *Continuous time-varying feedbacks*, which, besides from depending on the state x, depend also on the exogenous time variable (i. e., $\boldsymbol{u}(\boldsymbol{x}, t)$ instead of $\boldsymbol{u}(\boldsymbol{x})$ for classical feedbacks).
2. *Discontinuous feedbacks*, in the classical form $\boldsymbol{u}(\boldsymbol{x})$, except that the function $\boldsymbol{u}$ is not continuous at the equilibrium that one wishes to stabilize.
3. *Hybrid discrete/continuous feedbacks*. Although this class of feedbacks is not defined as precisely as the other two sets of controls, it is mostly composed of time-varying feedbacks, either continuous or discontinuous, such that the part of the control which depends upon the state is only updated periodically, e.g., $\boldsymbol{u}(t) = \bar{\boldsymbol{u}}[\boldsymbol{x}(kT), t]$ for any $t \in [kT, (k+1)T]$, with T denoting a constant period, and $k \in \mathbb{N}$.

We focus hereafter on continuous time-varying feedbacks. Examples of hybrid discrete/continuous feedbacks for unicycle and car-like vehicles are provided in [49.2, Sect 34.4.4], together with associated simulation results. As for discontinuous feedbacks, they involve difficult questions (existence of solutions, mathematical meaning of these solutions, etc.) which complicate their analysis and for which complete answers are not available. Moreover, for most of the discontinuous control strategies described in the literature, the property of stability in the sense of Lyapunov is either not granted or remains an open issue.

The use of time-varying feedbacks for the asymptotic stabilization of a fixed desired equilibrium, for a nonholonomic WMR, in order to circumvent the obstruction pointed out by Brockett's Theorem, was first proposed in [49.4]. Since then, very general results about the stabilization of nonlinear systems by means of time-varying feedbacks have been obtained. For instance, it has been proved that any controllable driftless system can have any of its equilibria asymptotically stabilized with a control of this type [49.5]. This includes the kinematic models of the nonholo-

nomic mobile robots here considered. We will illustrate this approach in the case of unicycle-type and car-like mobile robots modeled by three- and four-dimensional chained systems, respectively. In order to consider the three-dimensional case, let us come back on the results obtained in Sect. 49.4.3 for path following. The control law (49.27), i. e., $v_2 = -v_1 k_2 z_2 - |v_1| k_3 z_3$ applied to the system

$$\begin{cases} \dot{z}_1 = v_1 \,, \\ \dot{z}_2 = v_1 z_3 \,, \\ \dot{z}_3 = v_2 \,, \end{cases}$$

renders the function

$$V(z) := \frac{1}{2}(z_2^2 + \frac{1}{k_2} z_3^2)$$

nonincreasing along any trajectory of the controlled system, i. e.,

$$\dot{V} = -\frac{k_3}{k_2} |v_1| z_3^2 \,,$$

and ensures the convergence of z_2 and z_3 to zero if, for instance, v_1 does not tend to zero as t tends to infinity. For example, if $v_1(t) = \sin t$, z_2 and z_3 tend to zero, and

$$\begin{aligned} z_1(t) &= z_1(0) + \int_0^t v_1(s)\mathrm{d}s = z_1(0) + \int_0^t \sin s \mathrm{d}s \\ &= z_1(0) + 1 - \cos t \,, \end{aligned}$$

so that $z_1(t)$ oscillates around the mean value $z_1(0) + 1$. To reduce these oscillations, one can multiply v_1 by a factor which depends on the current state. Take, for example, $v_1(z,t) = \|(z_2, z_3)\| \sin t$, that we complement with a stabilizing term like $-k_1 z_1$ with $k_1 > 0$, i. e.,

$$v_1(z,t) = -k_1 z_1 + \|(z_2, z_3)\| \sin t \,.$$

The feedback control so obtained is time-varying and asymptotically stabilizing.

Proposition 49.2

The continuous time-varying feedback

$$\begin{cases} v_1(z,t) = -k_1 z_1 + \alpha \|(z_2, z_3)\| \sin t \,, \\ v_2(z,t) = -v_1(z,t) k_2 z_2 - |v_1(z,t)| k_3 z_3 \,, \end{cases} \tag{49.32}$$

with $\alpha, k_{1,2,3} > 0$, renders the origin of the 3-D chained system globally asymptotically stable [49.6].

The above proposition can be extended to chained systems of arbitrary dimension [49.6]. For the case $n = 4$, which corresponds to the car-like robot, one has the following result.

Proposition 49.3

The continuous time-varying feedback

$$\begin{cases} v_1(z,t) = -k_1 z_1 + \alpha \|(z_2, z_3, z_4)\| \sin t \,, \\ v_2(z,t) = -|v_1(z,t)| k_2 z_2 - v_1(z,t) k_3 z_3 \\ -|v_1(z,t)| k_4 z_4 \,, \end{cases} \tag{49.33}$$

with $\alpha, k_{1,2,3,4} > 0$ chosen such that the polynomial $s^3 + k_4 s^2 + k_3 s + k_2$ is Hurwitz stable, renders the origin of the 4-D chained system globally asymptotically stable [49.6].

Figure 49.10 below illustrates the previous result. For this simulation, the parameters $\alpha, k_{1,2,3,4}$ in the feedback law (49.33) have been chosen as $\alpha = 3$ and $k_{1,2,3,4} = (1.2, 10, 18, 17)$. Figure 49.10a shows the motion of the car-like robot in the plane. The initial configuration, at time $t = 0$, is depicted in plain lines, whereas the desired configuration is shown in dashed lines. The time evolution of the variables x, y, and θ (i. e., the position and orientation variables corresponding to the model (49.5)) is shown in Fig. 49.10b.

A shortcoming of this type of control, very clear from this simulation, is that the system's state converges to zero quite slowly. One can show that the rate of convergence is only polynomial, i. e., it is commensurable with $t^{-\alpha}$ (for some $\alpha \in (0,1)$) for most of the trajectories of the controlled system. This slow rate of convergence is related to the fact that the control function is Lipschitz-continuous with respect to x. It is a characteristics of systems the linear approximation of which is not stabilizable. Indeed, it can be shown that exponential stability (in the classical sense of linear systems) of an equilibrium point of a nonlinear system cannot be obtained with a Lipschitz-continuous feedback when the system's linear approximation at that point is not stabilizable ([49.2, Prop. 34.11] for a precise statement and details). The need of better performance and efficiency, has triggered the development of stabilizing time-varying feedbacks which are continuous, but not Lipschitz-continuous. Examples of such feedbacks, yielding uniform exponential convergence, are given in the following propositions for chained systems of dimension three and four, respectively.

Proposition 49.4

Let $\alpha, k_{1,2,3} > 0$ denote scalars such that the polynomial $p(s) = s^2 + k_3 s + k_2$ is Hurwitz stable. For any integers $p, q \in \mathbb{N}^*$, let $\rho_{p,q}$ denote the function defined on $\mathbb{R}^2$ by

$$\begin{aligned} \forall \bar{z}_2 = (z_2, z_3) \in \mathbb{R}^2, \rho_{p,q}(\bar{z}_2) \\ = \left(|z_2|^{\frac{p}{q+1}} + |z_3|^{\frac{p}{q}} \right)^{\frac{1}{p}} \,. \end{aligned}$$

Then, there exists $q_0 > 1$ such that, for any $q \geq q_0$ and $p > q + 2$, the continuous state feedback

$$\begin{cases} v_1(z,t) = -k_1(z_1 \sin t - |z_1|) \sin t \\ \quad +\alpha \rho_{p,q}(\bar{z}_2) \sin t \\ v_2(z,t) = -v_1(z,t) k_2 \dfrac{z_2}{\rho_{p,q}^2(\bar{z}_2)} - |v_1(z,t)| k_3 \dfrac{z_3}{\rho_{p,q}(\bar{z}_2)} \end{cases} \tag{49.34}$$

renders the origin of the three-dimensional chained system globally asymptotically stable, with a uniform exponential rate of convergence [49.7].

The parenthood of the controls (49.32) and (49.34) is noticeable. One can also verify that the control (49.34) is well defined (by continuity) at $\bar{z}_2 = 0$. More precisely, the ratios

$$\frac{z_2}{\rho_{p,q}^2(\bar{z}_2)} \quad \text{and} \quad \frac{z_3}{\rho_{p,q}(\bar{z}_2)} ,$$

which are obviously well defined when $\bar{z}_2 \neq 0$, tend to zero when $\bar{z}_2$ tends to zero. This guarantees the continuity of the control law.

The property of exponential convergence pointed out in the above result calls for some remarks. Indeed, this property does not exactly correspond to the classical exponential convergence property associated with stable linear systems. In this latter case, exponential convergence means that for some constant K, γ, and along any solution to the controlled system, one has

$$\|z(t)\| \leq K \|z(t_0)\| e^{-\gamma(t-t_0)}$$

This corresponds to the common notion of *exponential stability*. In the present case, this inequality becomes

$$\rho[z(t)] \leq K \rho[z(t_0)] e^{-\gamma(t-t_0)}$$

for some function ρ, defined for example by

$$\rho(z) = |z_1| + \rho_{p,q}(z_2, z_3) ,$$

with $\rho_{p,q}$ as specified in Proposition 49.4. Although the function ρ shares common features with the Euclidean norm of the state vector (it is definite positive and it tends to infinity when $\|z\|$ tends to infinity), it is not equivalent to this norm. Of course, this does not change the fact that each component z_i of z converges to zero exponentially. However, the transient behavior is different because one only has

$$|z_i(t)| \leq K \|z(t_0)\|^{\alpha} e^{-\gamma(t-t_0)} ,$$

with $\alpha < 1$, instead of

$$|z_i(t)| \leq K \|z(t_0)\| e^{-\gamma(t-t_0)} .$$

In the case of the four-dimensional chained system, one can establish the following result, which is similar to Proposition 49.4.

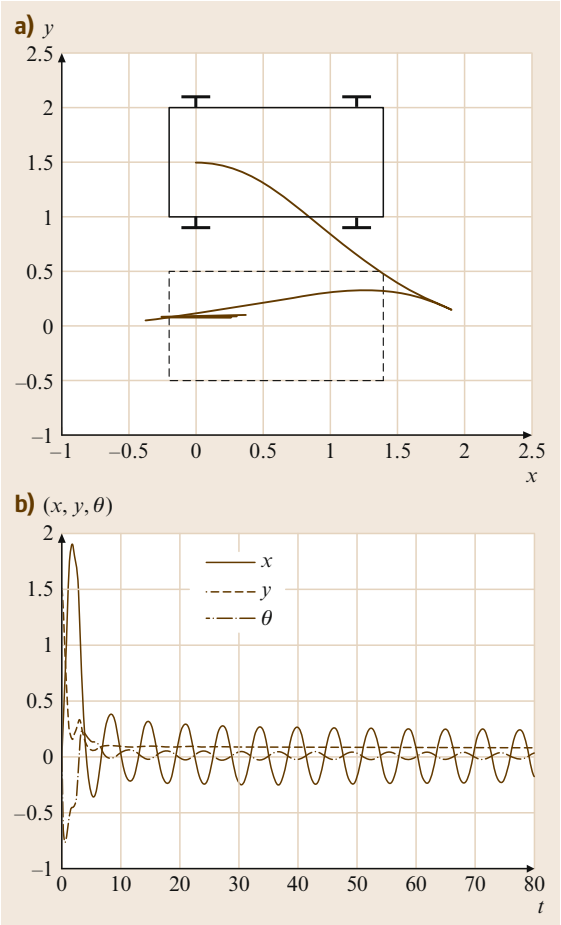


Fig.49.10a,b Asymptotic stabilization with a Lipschitz-continuous controller. **(a)** Cartesian motion; **(b)** error coordinates versus time

Proposition 49.5

Let $\alpha, k_1, k_2, k_3, k_4 > 0$ be chosen such that the polynomial $p(s) = s^3 + k_4 s^2 + k_3 s + k_2$ is Hurwitz stable. For any integers $p, q \in \mathbb{N}^*$, let $\rho_{p,q}$ denote the function defined on $\mathbb{R}^3$ by

$$\rho_{p,q}(\bar{z}_2) = \left(|z_2|^{\frac{p}{q+2}} + |z_3|^{\frac{p}{q+1}} + |z_4|^{\frac{p}{q}} \right)^{\frac{1}{p}}$$

with $\bar{z}_2 = (z_2, z_3, z_4) \in \mathbb{R}^3$. Then, there exists $q_0 > 1$ such that, for any $q \geq q_0$ and $p > q + 2$, the continuous

state feedback

$$\begin{cases} v_1(z,t) = -k_1(z_1 \sin t - |z_1|) \sin t \\ +\alpha \rho_{p,q}(\bar{z}_2) \sin t \,, \\ v_2(z,t) = -|v_1(z,t)|k_2 \dfrac{z_2}{\rho^3_{p,q}(\bar{z}_2)} - v_1(z,t)k_3 \dfrac{z_3}{\rho^2_{p,q}(\bar{z}_2)} \\ -|v_1(z,t)|k_4 \dfrac{z_4}{\rho_{p,q}(\bar{z}_2)} \,, \end{cases} \tag{49.35}$$

renders the origin of the 4-D chained system globally asymptotically stable, with a uniform exponential rate of convergence [49.7].

The performance of the control law (49.35) is illustrated by the simulation results shown in Fig. 49.11. The control parameters have been chosen as follows: $\alpha = 0.6$, $k_{1,2,3,4} = (1.6, 10, 18, 17)$, $q = 2$, $p = 5$. The comparison with the simulation results of Fig. 49.10 shows a clear gain in performance.

49.4.5 Limitations Inherent to the Control of Nonholonomic Systems

Let us first mention some problems associated with the nonlinear time-varying feedbacks just presented. An ever important issue, when studying feedback control, is robustness. Indeed, if it were not for the sake of robustness, feedback control would lose much of its value and interest with respect to open-loop control solutions. There are various robustness issues. One of them concerns the sensitivity to modeling errors. For instance, in the case of a unicycle-type robot whose kinematic equations are in the form $\dot{\boldsymbol{x}} = u_1 b_1(\boldsymbol{x}) + u_2 b_2(\boldsymbol{x})$, one would like to know whether a feedback law which stabilizes an equilibrium of this system also stabilizes this equilibrium for the *neighbor* system $\dot{\boldsymbol{x}} = u_1[b_1(\boldsymbol{x}) + \varepsilon g_1(\boldsymbol{x})] + u_2[b_2(\boldsymbol{x}) + \varepsilon g_2(\boldsymbol{x})]$, with g_1 and g_2 denoting continuous applications, and ε a parameter which quantifies the modeling error. This type of error can account, for example, for a small uncertainty concerning the orientation of the actuated wheels axle with respect to the chassis, which results in a bias in the measurement of this orientation. One can show that time-varying control laws like (49.34) are not robust with respect to this type of error in the sense that, for certain functions g_1 and g_2, and for ε arbitrarily small, the system's solutions end up oscillating in the neighborhood of the origin, instead of converging to the origin. In other words, both the properties of stability of the origin and of convergence to this point can be jeopardized by arbitrarily small modeling errors, even in the absence of measurement noise. In view of these problems, one is brought to question the existence of fast (exponential) stabilizers endowed with robustness properties similar to those of stabilizing linear feedbacks for linear systems. The answer is that, to our knowledge, no such control solution (either continuous or discontinuous) has ever been found. More than likely such a solution does not exist for nonholonomic systems. Robustness of the stability property against modeling errors, and control discretization and delays, has been proved in some cases, but this could only be achieved with Lipschitz-continuous feedbacks which, as we have seen, yield slow convergence. The classical compromise between robustness and performance thus seems much more acute than in the case of stabilizable linear systems (or nonlinear systems whose linear approximation is stabilizable).

A second issue is the proven nonexistence of a *universal* feedback controller capable of stabilizing any feasible reference state trajectory asymptotically [49.8].

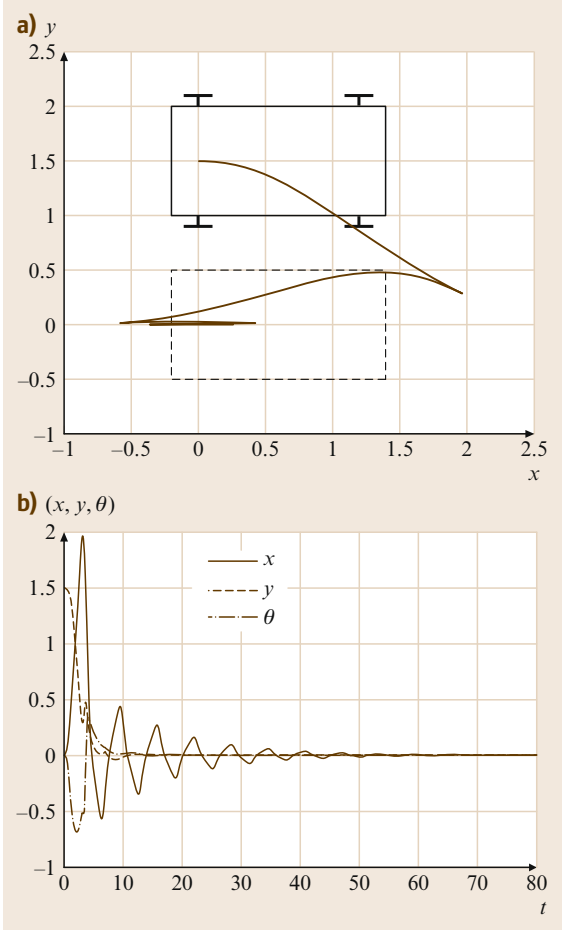


Fig.49.11a,b Asymptotic stabilization with a continuous (non-Lipschitz) time-varying feedback. **(a)** Cartesian motion; **(b)** error coordinates versus time

This is another notable difference with the linear case. Indeed, given a controllable linear system $\dot{\boldsymbol{x}} = \mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{u}$, the feedback controller $\boldsymbol{u} = \boldsymbol{u}_\mathrm{r} + \mathbf{K}(\boldsymbol{x} - x_\mathrm{r})$, with $\mathbf{K}$ a gain matrix such that $\mathbf{A} + \mathbf{BK}$ is Hurwitz stable, exponentially stabilizes any feasible reference trajectory $\boldsymbol{x}_\mathrm{r}$ (solution to the system) associated with the control input $\boldsymbol{u}_\mathrm{r}$. The nonexistence of such a controller, in the case of nonholonomic mobile robots, is related to the conditions upon the longitudinal velocity stated in previous propositions concerning trajectory stabilization. This basically indicates that such conditions cannot be removed entirely: whatever the chosen feedback controller, there always exists a feasible reference trajectory that this feedback cannot asymptotically stabilize. Note that this limitation persists when considering nonstandard feedbacks (such as, e.g., time-varying periodic feedbacks capable of asymptotically stabilizing reference trajectories which are reduced to a single point). Moreover, it has clear practical consequences because there are applications (automatic tracking of a human-driven car, for instance) for which the reference trajectory, and thus its properties, are not known in advance (is the leading car going to keep moving or stop?) so that one cannot easily decide on which controller to use. Switching between various controllers is a possible strategy, which has been studied by some authors and may give satisfactory results in many situations. However, since implementing a predefined switching strategy between two controllers sums up to designing a third controller, this does not solve the core of the problem nor grant any certitude of success.

A third issue, which is not specific to nonholonomic systems, but has seldom been addressed in the nonlinear control literature, concerns the problem of tracking *nonfeasible* trajectories (i. e., trajectories which are not solutions to the system's equations). Since exact tracking is not possible, by the definition of a nonfeasible trajectory, the control objective is then to ensure that the tracking errors shrink to, and ever after never exceed, certain nonzero thresholds. The fact that these thresholds can theoretically be arbitrarily small in the case of nonholonomic systems, if the amplitude of the velocity control inputs is not limited, makes this problem particularly relevant for these systems. This can be termed as a *practical* stabilization objective which, although slightly less ambitious than the objective of asymptotic stabilization considered in previous sections, opens up both the control design problem and the range of applications significantly. For instance, it allows one to address the problem of tracking an omnidirectional vehicle with a unicycle-type, or a car-like, vehicle. In the context of planning a trajectory with obstacle avoidance, transforming a nonfeasible trajectory into a feasible approximation for a certain mobile robot can be performed by applying a practical stabilizer to a model of this robot and by numerical integration of the system's closed-loop equations. Also, if one reformulates the former question about the existence of a *universal* stabilizer, with the objective of asymptotic stabilization now replaced by the one of practical stabilization, then the answer becomes positive: such a stabilizer exists and, moreover, the reference trajectories do not even have to be feasible.

49.4.6 Practical Stabilization of Arbitrary Trajectories

A possible approach for the design of practical stabilizers in the case of controllable driftless systems is described in [49.9]. Some of its basic principles, here adapted to the specific examples of unicycle-type and car-like mobile robots, are recalled next.

Let us introduce some matrix notation that will be used in this section.

$$\mathbf{R}(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \quad \mathbf{S} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

$$\bar{\mathbf{R}}(\theta) = \begin{pmatrix} R(\theta) & 0 \\ 0 & 1 \end{pmatrix}.$$

Unicycle Case

With the above notation, the kinematic model (49.4) can be written as

$$\dot{\boldsymbol{g}} = \bar{\mathbf{R}}(\theta)\mathbf{C}\boldsymbol{u}\,, \tag{49.36}$$

with $\boldsymbol{g} = (\boldsymbol{x}, \boldsymbol{y}, \theta)'$ and

$$\mathbf{C} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

Let us now consider a smooth function

$$f : \alpha \longmapsto f(\alpha) = \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \\ f_\theta(\alpha) \end{pmatrix},$$

with $\alpha \in S^1 = \mathbb{R}/2\pi\mathbb{Z}$ (i. e., α is an angle variable), and define

$$\bar{\boldsymbol{g}} := \begin{pmatrix} \bar{\boldsymbol{x}} \\ \bar{\boldsymbol{y}} \\ \bar{\theta} \end{pmatrix} := \boldsymbol{g} - \bar{\mathbf{R}}\,(\theta - f_\theta(\alpha))f(\alpha)$$

$$= \begin{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} - \mathbf{R}(\theta - f_\theta(\alpha)) \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix} \\ \theta - f_\theta(\alpha) \end{pmatrix}. \tag{49.37}$$

Note that $\bar{g}$ can be viewed as the situation of a frame $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ the origin of which has components

$$-\mathbf{R}(-f_\theta(\alpha))\begin{pmatrix} f_x(\alpha) \\ f_y(\alpha)\end{pmatrix}$$

in the frame $\mathcal{F}_{\mathrm{m}}$. In term of differential geometry, $\bar{g}$ is the product of g by the inverse $f(\alpha)^{-1}$ of $f(\alpha)$, in the sense of the Lie group operations in $SE(2)$. Hence, $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ is all the closer to $\mathcal{F}_{\mathrm{m}}$ as the components of $f(\alpha)$ are small. For any smooth time function $t \longmapsto \alpha(t)$, and along any solution to system (49.36), the time derivative of $\bar{g}$ is given by

$$\dot{\bar{g}} = \bar{\mathbf{R}}(\bar{\theta})\bar{\boldsymbol{u}}\,, \tag{49.38}$$

with

$$\bar{\boldsymbol{u}} = \mathbf{A}(\alpha)\left(\bar{\mathbf{R}}(f_\theta(\alpha)) - \frac{\partial f}{\partial \alpha}(\alpha)\right)\begin{pmatrix}\mathbf{C}\boldsymbol{u} \\ \dot{\alpha}\end{pmatrix}, \tag{49.39}$$

and

$$\mathbf{A}(\alpha) = \begin{pmatrix} I_2 & -S\begin{pmatrix} f_x(\alpha) \\ f_y(\alpha)\end{pmatrix} \\ 0 & 1\end{pmatrix}. \tag{49.40}$$

From (49.38) and (49.39), one can view $\dot{\alpha}$ as a complementary control input that can be used to monitor the motion of the frame $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$. More precisely, $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ can be viewed as an omnidirectional frame provided that $\bar{\boldsymbol{u}}$ can be rendered equal to any vector of $\mathbb{R}^3$, i. e., provided that the mapping $(\boldsymbol{u}, \dot{\alpha}) \longmapsto \bar{\boldsymbol{u}}$ is onto. Let us determine when this condition is satisfied. Equation (49.39) can also be written as

$$\bar{\boldsymbol{u}} = \mathbf{A}(\alpha)\mathbf{H}(\alpha)\begin{pmatrix}\boldsymbol{u} \\ \dot{\alpha}\end{pmatrix}, \tag{49.41}$$

with

$$\mathbf{H}(\alpha) = \begin{pmatrix} \cos f_\theta(\alpha) & 0 & -\frac{\partial f_x}{\partial \alpha}(\alpha) \\ \sin f_\theta(\alpha) & 0 & -\frac{\partial f_y}{\partial \alpha}(\alpha) \\ 0 & 1 & -\frac{\partial f_\theta}{\partial \alpha}(\alpha)\end{pmatrix}. \tag{49.42}$$

Since $\mathbf{A}(\alpha)$ is invertible, $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ is omnidirectional if and only if the matrix $\mathbf{H}(\alpha)$ is also invertible. A function f which satisfies this property for *any* $\alpha \in S^1$ is called a *transverse function* [49.10]. The issue of the existence of such functions has been treated in the much more general context of the transverse function approach [49.9, 10]. In the present case, a family of transverse functions is given by

$$f(\alpha) = \begin{pmatrix} \varepsilon \sin\alpha \\ \varepsilon^2\eta\dfrac{\sin 2\alpha}{4} \\ \arctan(\varepsilon\eta\cos\alpha)\end{pmatrix} \quad \text{with} \quad \varepsilon, \eta > 0\,. \tag{49.43}$$

Indeed, with this function one can verify that, for any $\alpha \in S^1$,

$$\det\mathbf{H}(\alpha) = -\frac{\varepsilon^2\eta}{2}\cos(\arctan(\varepsilon\eta\cos\alpha)) < 0\,.$$

Note that the components of f uniformly tend to zero as ε tends to zero, so that the associated omnidirectional frame $\bar{\mathcal{F}}_{\mathrm{m}}(\alpha)$ can be made arbitrarily close to $\mathcal{F}_{\mathrm{m}}$ by choosing ε small (but different from zero).

Now, let

$$t \longmapsto \boldsymbol{g}_{\mathrm{r}}(t) = [\boldsymbol{x}_{\mathrm{r}}(t), \boldsymbol{y}_{\mathrm{r}}(t), \theta_{\mathrm{r}}(t)]^{\mathrm{T}}$$

denote a smooth, but otherwise *arbitrary*, reference trajectory. It is not difficult to derive from (49.38) a feedback law $\bar{\boldsymbol{u}}$ which asymptotically stabilizes $\bar{\boldsymbol{g}}$ at $\boldsymbol{g}_{\mathrm{r}}$. A possible choice is given by

$$\bar{\boldsymbol{u}} = \bar{\mathbf{R}}(-\bar{\theta})[\dot{\boldsymbol{g}}_{\mathrm{r}} - k(\bar{\boldsymbol{g}} - \boldsymbol{g}_{\mathrm{r}})]\,, \tag{49.44}$$

which implies that

$$(\dot{\bar{\boldsymbol{g}}} - \dot{\boldsymbol{g}}_{\mathrm{r}}) = -k(\bar{\boldsymbol{g}} - \boldsymbol{g}_{\mathrm{r}})$$

and therefore that $\bar{g} - g_{\mathrm{r}} = 0$ is an exponentially stable equilibrium of the above equation for any $k > 0$. Then, it follows from (49.37) that

$$\lim_{t\to+\infty}\{g(t) - g_{\mathrm{r}}(t) - \bar{\mathbf{R}}[\theta_{\mathrm{r}}(t)]f[\alpha(t)]\} = 0\,. \tag{49.45}$$

The norm of the tracking error $\|\boldsymbol{g} - \boldsymbol{g}_{\mathrm{r}}\|$ is thus ultimately bounded by the norm of $f(\alpha)$ which, in view of (49.43), can be made arbitrarily small via the choice of ε. It is in this sense that practical stabilization is achieved. The control $\boldsymbol{u}$ for the unicycle-like robot is then calculated by inverting the relation (49.41) and using the expression (49.44) of $\bar{\boldsymbol{u}}$.

While it can be tempting to use very small values of ε for the transverse function f in order to obtain a good tracking precision, one must be aware of the limits of this strategy. Indeed, when ε tends to

zero, the matrix $\mathbf{H}(\alpha)$ defined by (49.42) becomes ill-conditioned, and its determinant tends to zero. This implies, by (49.41), that the robot's velocities u_1 and u_2 may become very large. In particular, when the reference trajectory $\boldsymbol{g}_\mathrm{r}$ is not feasible, many manoeuvres are likely to occur. Note that this difficulty is intrinsic to the robot's nonholonomy and that it cannot be circumvented (think about the problem of parking a car in a very narrow parking place). For this reason, trying to impose very accurate tracking of nonfeasible trajectories is not necessarily a good option in practice. On the other hand, when the trajectory is feasible, manoeuvres are not needed to achieve accurate tracking, so that smaller values of ε can be used in this case. This clearly leads to a dilemma when the reference trajectory is not known in advance and its properties in term of feasibility can vary with time. A control strategy which addresses this issue, based on the use of transverse functions whose magnitude can be adapted online, is proposed in [49.11]. Experimental validations of the present approach on a unicycle-like robot can also be found in [49.12].

Car Case

The control approach presented above can be extended to car-like vehicles (and also to the trailer case). Again, the idea is to associate with the robot's frame $\mathcal{F}_\mathrm{m}$ an omnidirectional *companion* frame $\bar{\mathcal{F}}_\mathrm{m}(\alpha)$ which can be maintained arbitrarily close to $\mathcal{F}_\mathrm{m}$ via the choice of some design parameters. Let us show how this can be done for a car-like vehicle. To simplify the forthcoming equations, let us rewrite system (49.5) as

$$\begin{cases} \dot{x} = u_1 \cos\theta \,, \\ \dot{y} = u_1 \sin\theta \,, \\ \dot{\theta} = u_1 \xi \,, \\ \dot{\xi} = u_\xi \,, \end{cases}$$

with $\xi = (\tan\phi)/L$ and $u_\xi = u_2(1+\tan^2\phi)/L$. This system can also be written as (compare with (49.36))

$$\begin{cases} \dot{\boldsymbol{g}} = \bar{\mathbf{R}}(\theta)\mathbf{C}(\xi)u_1 \,, \\ \dot{\xi} = \boldsymbol{u}_\xi \,, \end{cases} \tag{49.46}$$

with $\boldsymbol{g} = (\boldsymbol{x}, \boldsymbol{y}, \theta)^\mathrm{T}$ and $\mathbf{C}(\xi) = (1, 0, \xi)^\mathrm{T}$. Let us now consider a smooth function

$$f : \alpha \longmapsto f(\alpha) = \begin{pmatrix} f_g(\alpha) \\ f_\xi(\alpha) \end{pmatrix} = \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \\ f_\theta(\alpha) \\ f_\xi(\alpha) \end{pmatrix},$$

with $\alpha \in S^1 \times S^1$ (i. e., $\alpha = (\alpha_1, \alpha_2)$), and define (compare with (49.37))

$$\begin{aligned} \bar{\boldsymbol{g}} &:= \begin{pmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{pmatrix} := \boldsymbol{g} - \bar{\mathbf{R}}(\theta - f_\theta(\alpha)) f_g(\alpha) \\ &= \begin{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} - \mathbf{R}(\theta - f_\theta(\alpha)) \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix} \\ \theta - f_\theta(\alpha) \end{pmatrix}, \end{aligned} \tag{49.47}$$

which, as in the unicycle case, can be viewed as the situation of some *companion* frame $\bar{\mathcal{F}}_\mathrm{m}(\alpha)$. By differentiating $\bar{\boldsymbol{g}}$ along any smooth time function $t \longmapsto \alpha(t)$ and any solution to system (49.46), one can verify that (49.38) is still satisfied, except that $\bar{\boldsymbol{u}}$ is now given by

$$\begin{aligned} \bar{\boldsymbol{u}} = \mathbf{A}(\alpha) &\left(\bar{\mathbf{R}}[f_\theta(\alpha)] - \frac{\partial f_g}{\partial \alpha_1}(\alpha) - \frac{\partial f_g}{\partial \alpha_2}(\alpha) \right) \\ &\times \begin{pmatrix} \mathbf{C}(\xi)u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix} \end{aligned} \tag{49.48}$$

rather than by (49.39) (with $\mathbf{A}(\alpha)$ still defined by (49.40)). Using the fact that

$$\begin{aligned} \mathbf{C}(\xi)u_1 &= \mathbf{C}(f_\xi(\alpha))u_1 + \{\mathbf{C}(\xi) - \mathbf{C}[f_\xi(\alpha)]\}u_1 \\ &= \begin{pmatrix} 1 \\ 0 \\ f_\xi(\alpha) \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 0 \\ \xi - f_\xi(\alpha) \end{pmatrix} u_1 \,, \end{aligned}$$

(49.48) can also be written as

$$\bar{\boldsymbol{u}} = A(\alpha)\mathbf{H}(\alpha) \begin{pmatrix} u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix} + A(\alpha) \begin{pmatrix} 0 \\ 0 \\ u_1[\xi - f_\xi(\alpha)] \end{pmatrix}, \tag{49.49}$$

with

$$\mathbf{H}(\alpha) = \begin{pmatrix} \cos f_\theta(\alpha) & -\frac{\partial f_x}{\partial \alpha_1}(\alpha) & -\frac{\partial f_x}{\partial \alpha_2}(\alpha) \\ \sin f_\theta(\alpha) & -\frac{\partial f_y}{\partial \alpha_1}(\alpha) & -\frac{\partial f_y}{\partial \alpha_2}(\alpha) \\ f_\xi(\alpha) & -\frac{\partial f_\theta}{\partial \alpha_1}(\alpha) & -\frac{\partial f_\theta}{\partial \alpha_2}(\alpha) \end{pmatrix}. \tag{49.50}$$

By setting

$$\boldsymbol{u}_\xi = \dot{f}_\xi(\alpha) - k(\xi - f_\xi(\alpha) \,, \tag{49.51}$$

with $k > 0$, it follows from (49.46) that $\xi - f_\xi(\alpha)$ exponentially converges to zero. Hence, after some transient phase whose duration is commensurable with $1/k$, $\xi - f_\xi(\alpha) \approx 0$, and (49.49) reduces to

$$\bar{\boldsymbol{u}} = \mathbf{A}(\alpha)\mathbf{H}(\alpha)\begin{pmatrix} u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix}. \qquad (49.52)$$

Provided that the function f is such that $\mathbf{H}(\alpha)$ is always invertible, this latter relation means that the frame $\bar{\mathcal{F}}_\mathrm{m}(\alpha)$ associated with $\bar{\boldsymbol{g}}$ is omnidirectional. Any function f for which this property is satisfied is called a transverse function. Once it has been determined, one can proceed as in the unicycle case to asymptotically stabilize an arbitrary reference trajectory $\boldsymbol{g}_\mathrm{r}$ for $\bar{\boldsymbol{g}}$, for example, by defining $\bar{\boldsymbol{u}}$ as in (49.44). The control u_1 for the car is then obtained by inverting relation (49.52).

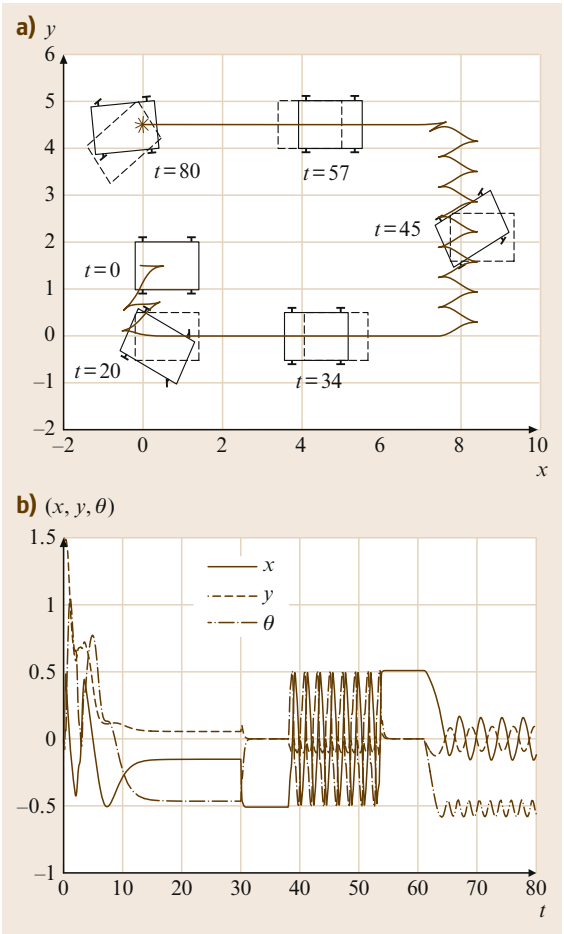


Fig.49.12a,b Practical stabilization of an arbitrary trajectory by the transverse function approach. **(a)** Cartesian motion; **(b)** error coordinates versus time

The following lemma specifies a family of transverse functions for the car case.

Lemma 49.1

For any $\varepsilon > 0$ and any η_1, η_2, η_3 such that $\eta_1, \eta_2, \eta_3 > 0$ and $6\eta_2\eta_3 > 8\eta_3 + \eta_1\eta_2$, the function f defined by

$$f(\alpha) = \begin{pmatrix} \bar{f}_1(\alpha) \\ \bar{f}_4(\alpha) \\ \arctan(\bar{f}_3(\alpha)) \\ \bar{f}_2(\alpha)\cos^3 f_3(\alpha) \end{pmatrix}$$

with

$$\bar{f} : S^1 \times S^1 \longrightarrow \mathbb{R}^4$$

given by

$$\bar{f}(\alpha) = \begin{pmatrix} \varepsilon(\sin\alpha_1 + \eta_2 \sin\alpha_2) \\ \varepsilon\eta_1 \cos\alpha_1 \\ \varepsilon^2\left(\dfrac{\eta_1 \sin 2\alpha_1}{4} - \eta_3 \cos\alpha_2\right) \\ \varepsilon^3\left(\eta_1 \dfrac{\sin^2\alpha_1 \cos\alpha_1}{6} - \dfrac{\eta_2\eta_3 \sin 2\alpha_2}{4} - \eta_3 \sin\alpha_1 \cos\alpha_2\right) \end{pmatrix},$$

satisfies the transversality condition $\det\mathbf{H}(\alpha) \neq 0\ \forall\alpha$, with $\mathbf{H}(\alpha)$ defined by (49.50) [49.13].

The simulation results reported in Fig. 49.12 illustrate the application of this control approach for a car-like robot. The reference trajectory is defined by the initial condition $g_\mathrm{r}(0) = 0$ and its time derivative

$$\dot{g}_\mathrm{r}(t) = \begin{cases} (0,0,0)^\mathrm{T} & \text{if } t \in [0,30] \\ (1,0,0)^\mathrm{T} & \text{if } t \in [30,38] \\ (0,0.3,0)^\mathrm{T} & \text{if } t \in [38,53] \\ (-1,0,0)^\mathrm{T} & \text{if } t \in [53,61] \\ (0,0,0.2)^\mathrm{T} & \text{if } t \in [61,80] \end{cases}.$$

This corresponds to a fixed situation when $t \in [0,30]$, three sequences of pure translational motion when $t \in [30,61]$, and a pure rotational motion when $t \in [61,80]$. Let us remark that this trajectory is not feasible for the car-like robot when $t \in [38,53]$, since it corresponds to a lateral translation in the direction of the unit vector $\boldsymbol{j}_\mathrm{r}$ of the frame $\mathcal{F}_\mathrm{r}$ associated with g_r, nor when $t \in [61,80]$, since a rear-drive car cannot perform pure rotational motion. The initial configuration of the car-like robot, at $t = 0$, is $\boldsymbol{g}(0) = (0, 1.5, 0)$, and the initial steering wheel angle is $\phi(0) = 0$.

In Fig. 49.12a, the robot is drawn with a solid line at several time instants, whereas the chassis of the reference vehicle is shown as a dashed line at the same time instants. The figure also shows the trajectory of the point located at the mid-distance of the robot's rear wheels. Figure 49.12b shows the time evolution of the tracking error expressed in the reference frame (i. e., (x_e, y_e) as defined by (49.16), and $\theta_e = \theta - \theta_r$). It follows from (49.45) that, after the transient phase associated with the exponential convergence of $\bar{g}$ to zero, the ultimate bound for $|x_e|$, $|y_e|$, and $|\theta_e|$ is upper-bounded by the maximum amplitude of the functions f_x, f_y, and f_θ, respectively. For this simulation, the control parameters of the transverse function f of Lemma 49.1 have been chosen as follows: $\varepsilon = 0.17$, $\eta_{1,2,3} = (12, 2, 20)$. With these values, one can verify that $|f_x|$, $|f_y|$, and $|f_\theta|$ are bounded by 0.51, 0.11, and 0.6, respectively. This is consistent with the time evolution of the tracking error observed in the figure. As pointed out for the unicycle case, tracking errors could be diminished by decreasing the value of ε, but this would involve larger values of the control inputs and also more frequent manoeuvres, especially on the time intervals [38, 53] and [61, 80] when the reference trajectory is not feasible. Additional simulation and experimental results on this approach are illustrated by VIDEO 182 and VIDEO 243.

49.5 Path Following in the Case of Nonideal Wheel–Ground Contact

The kinematic models used in the previous sections are derived under the classical rolling-without-sliding assumption for the vehicle's wheels. This assumption is satisfied with a good degree of accuracy for many applications both indoor and outdoor (e.g., on-road). In some cases, however, sliding can be significant. This happens for example when a vehicle operates on natural terrain with poor grip conditions (grass, earth) and when the vehicle's speed is significant and/or the terrain is not perfectly horizontal. In these cases, the control laws presented in the previous sections may not give full satisfaction. We show in this section that these control laws can still be used successfully, provided that sliding is taken into account at the modelling level and estimated online via a dedicated observer. For simplicity, only the path following problem is addressed, but the techniques here presented can be extended to other control problems.

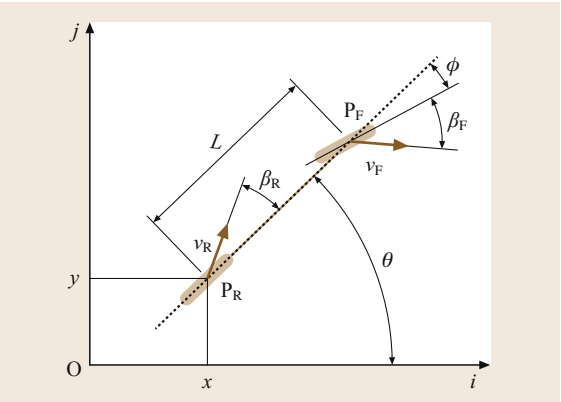


Fig. 49.13 Car-like vehicle in the presence of sliding

49.5.1 Extended Control Models in the Presence of Sliding

Extended Kinematic Model

Consider the two-wheels schematic representation of a car-like vehicle on Fig. 49.13. The angles β_R and β_F are introduced in order to represent the sliding of the rear and front wheels respectively. More precisely, denoting the centres of the rear and front wheels as P_R and P_F respectively, β_R is the angle between the vector $\boldsymbol{P_R P_F}$ and the velocity vector $\boldsymbol{v}_R$ of $\boldsymbol{P}_R$, whereas β_F represents the angle between the steering direction and the velocity vector $\boldsymbol{v}_F$ of $\boldsymbol{P}_F$. The kinematic modeling of Sect. 49.2 is easily extended to the present case [49.14], resulting in the following model

$$\begin{cases} \dot{x} = u_1 \cos(\theta + \beta_R) \\ \dot{y} = u_1 \sin(\theta + \beta_R) \\ \dot{\theta} = u_1 \cos(\beta_R) \dfrac{\tan(\phi + \beta_F) - \tan(\beta_R)}{L} \\ \dot{\phi} = u_2 \end{cases} , \tag{49.53}$$

with u_1 the (signed) intensity of the vector $\boldsymbol{v}_R$. Note that these equations reduce to (49.5) when $\beta_F = \beta_R = 0$.

Kinematic Model in a Frénet Frame

In the context of path following, a kinematic model with reference to a Frénet frame moving along the desired path is obtained via a straightforward adaptation of the pure-rolling case discussed in Sect. 49.2.3. Defining the distance between the vehicle and the desired path C as the distance d between the point P_R and this path (Fig. 49.5), the equations of this model are readily de-

duced from (49.53) (compare with (49.13))

$$\begin{cases} \dot{s} = u_1 \dfrac{\cos(\theta_e + \beta_R)}{1 - d\,c(s)} \\ \dot{d} = u_1 \sin(\theta_e + \beta_R) \\ \dot{\theta}_e = u_1 \left[\cos(\beta_R)\lambda_1 - \lambda_2\right] \\ \dot{\phi} = u_2 \end{cases}, \tag{49.54}$$

with θ_e the angle between the vehicle's body axis $P_R P_F$ and the tangent to the desired path evaluated at the projection of the point P_R on this path, $c(s)$ the path curvature at the projected point, $\lambda_1 = (\tan(\phi + \beta_F) - \tan(\beta_R))/L$ and $\lambda_2 = (c(s)\cos(\theta_e + \beta_R))/(1 - d\,c(s))$.

Dynamic Model of Sliding Angles

The kinematic model (49.53) can be used for control design once it is completed by a model of the dynamics of the sliding angles β_R and β_F. Such a model can be obtained from Newton's law and a model of tire/ground interactions. A few notation (Fig. 49.14 for details) and assumptions are introduced for this purpose:

- The vehicle's mass is denoted as m and its moment of inertia with respect to the (body-fixed) vertical axis is denoted as I_z. The vehicle's center of mass G is located on the segment joining P_R to P_F, at a distance L_R from P_R and L_F from P_F.
- The longitudinal dynamics is neglected. More precisely, it is assumed that the traction force applied to the vehicle, in relation to the monitoring of longitudinal tire/ground contact forces, is controlled independently of the vehicle's lateral dynamics and that, as a result of this control, the longitudinal velocity of the vehicle expressed in body frame, i. e., $u_1 \cos\beta_r$, varies slowly so that its time-derivative can be neglected in the calculus of the lateral dynamics.
- Lateral tire/ground contact forces, that are orthogonal to the wheels planes, are denoted as $\boldsymbol{F}_R$ and $\boldsymbol{F}_F$ respectively.
- The lateral component of the gravity force is denoted as $\boldsymbol{F}_G$. This force applies at G and its magnitude is $mg\sin\alpha$ with α denoting the terrain slope angle in the lateral direction.

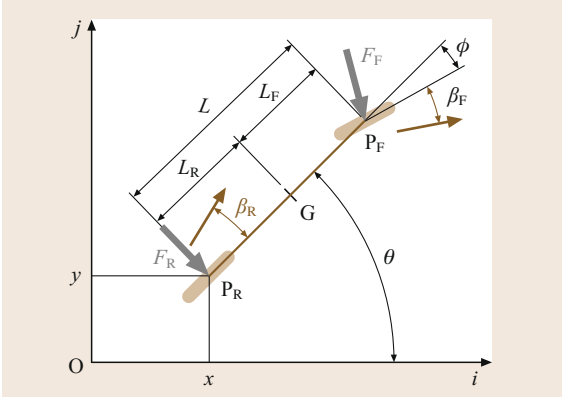


Fig. 49.14 Lateral forces applying to a car-like vehicle

Application of Newton's law yields [49.15], after projection in the direction orthogonal to $\boldsymbol{v}_R$,

$$\dot{\beta}_R = \frac{1}{mu_1}\Big[(F_R + mg\sin\alpha)\cos\beta_R + F_F\cos(\beta_R - \phi) - mL_R\ddot{\theta}\cos\beta_R\Big] - \dot{\theta}$$

with F_F and F_R the (signed) intensity of the lateral forces $\boldsymbol{F}_F$ and $\boldsymbol{F}_R$ and

$$\ddot{\theta} = \frac{1}{I_z}(L_F F_F\cos\phi - L_R F_R) \tag{49.55}$$

From (49.53)

$$\tan(\phi + \beta_F) = \tan\beta_R - \frac{L\dot{\theta}}{u_1\cos\beta_R}\,.$$

Differentiating this equation with respect to time and using the assumption that $u_1\cos\beta_R$ is constant yields

$$\dot{\beta}_F = \cos^2(\phi + \beta_F)\left(\frac{\dot{\beta}_R}{\cos^2\beta_R} - L\frac{\ddot{\theta}}{u_1\cos\beta_R}\right) - \dot{\phi}\,.$$

The dynamics of the sliding angles is thus given by

$$\begin{cases} \dot{\beta}_R = \dfrac{1}{mu_1}\Big[(F_R + mg\sin\alpha)\cos\beta_R + F_F\cos(\beta_R - \phi) - mL_R\ddot{\theta}\cos\beta_R\Big] - \dot{\theta}\,, \\ \dot{\beta}_F = \cos^2(\phi + \beta_F)\left(\dfrac{\dot{\beta}_R}{\cos^2\beta_R} - L\dfrac{\ddot{\theta}}{u_1\cos\beta_R}\right) - \dot{\phi}\,. \end{cases} \tag{49.56}$$

Replacing the vehicle's angular acceleration $\ddot{\theta}$ by its expression (49.55), and $\dot{\theta}, \dot{\phi}$ by (49.53), one obtains expressions of the sliding angles dynamics in terms of ϕ, α, u_1, u_2, F_R, and F_F.

Tire/Ground Interaction Models for Lateral Forces Computation

In view of (49.56), knowledge of the lateral ground forces $\boldsymbol{F}_R$ and $\boldsymbol{F}_F$ is needed to calculate the evolution of the sliding angles and the vehicle's motion. In

this respect, a model of tire/ground interactions is useful, in particular for simulation purposes. The Coulomb model, although very popular, does not allow one to describe the complexity of tire/ground interactions in a large operating domain. For this reason, other contact models have been proposed in the literature. For instance, the *Dahl* [49.16] and LuGre [49.17] models, eventhough they are not dedicated to tire/ground contact description, give a relationship between contact force and sliding velocity with a small number of parameters. They are used to describe a vehicle's dynamics in [49.18]. Of particular interest is the celebrated tire/ground contact model of Pacejka et al. [49.19], of which several versions depending on the application have been derived. A convenient model for mobile robot simulation and analysis is the so-called *magic formula* proposed in [49.20]. Alike other contact models, this formula expresses the lateral ground force in term of the sliding angle. It is defined as follows,

$$\begin{cases} F_* = D\sin\left[c\arctan(B[1-E])\beta_* + \dfrac{E}{B}\arctan(B\beta_*)\right] \\ D = a_1(F_z^*)^2 + a_2F_z^* \\ E = a_6(F_z^*)^2 + a_7F_z^* + a_8 \\ B = \dfrac{a_3\sin(a_4\arctan(a_5F_z^*))}{cD} \end{cases}, \tag{49.57}$$

with $* \in \{F, R\}$ used to denote either the front (F) or Rear (R) wheel, and F_z^* the corresponding tire load here defined as $\frac{L_*}{L}m$. The parameters a_i, $i \in [1, \ldots, 8]$ and c are representative of the grip conditions and tire properties (pressure, contact patches). Table 49.1 shows typical values of these parameters for a vehicle moving on wet grass at a velocity of 2 m/s (Sect. 49.5.4 further on). Fig. 49.15 depicts the corresponding relation between the front lateral force F_F and the front sliding angle β_F. This relationship is symmetric with respect to the origin, is quasi linear for small angles, has an extremum near $\pm -10°$, and involves a saturation for large sliding angles. The linear part of the function is often sufficient to account for slow to moderately fast motion in the case of good grip conditions, whereas its nonlinear part becomes important when the vehicle moves on natural ground or at high speed with possibly large sliding angles.

Table 49.1 Parameters used for dynamic modelling

Coefficients	Values
a_1	-25
a_2	500
a_3	1000
a_4	2
a_5	1
a_6	0
a_7	-0.35
a_8	5
c	1.6
$L = L_F + L_R$	$1.3 = 0.6 + 0.7$
m	380
I_z	300

Tire/ground contact models are necessary to describe the sliding angles' dynamics with some accuracy and are useful to evaluate the performance of control laws in simulation. The main drawback of these models is the large number of involved parameters and the difficulty to evaluate appropriate values in practical applications, since such values heavily depend on imprecisely known ground characteristics which, furthermore, may rapidly change along the vehicle's trajectory. For these reasons, previously evoked models may not be best suited for control design. In many cases, rather than trying to exploit a complete model involving unreliable and rapidly varying parameters it is preferable to use cruder models associated with an online estimation procedure. The control design proposed further on follows this latter avenue.

49.5.2 On-Line Estimation of Sliding Angles

As already mentionned, some knowledge of the sliding angles β_F and β_R is useful to precisely control the lateral distance d between the vehicle and the desired path. Since direct measurement of these angles via the use of dedicated sensors is quite difficult in practice, an alternative consists in designing an observer that produces on-line estimates of the sliding angles based on the measurement of the relative vehicle/path position and orientation. The underlying implicit (crude) model used

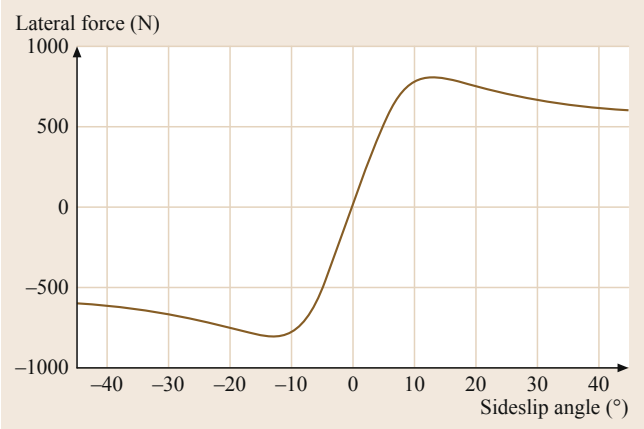


Fig. 49.15 Relationship between lateral force and sliding angle obtained using Pacejka Model

for the design of this observer is that the sliding angles do not change rapidly, as can be anticipated in common situations like driving along a road with a slowly varying curvature or following a straight line across a field with a constant slope. A solution of this type, exploiting also the vehicle's kinematic model (49.54), is proposed next.

Define $\xi = (d, \theta_e)^T$, $\beta = (\beta_F, \beta_R)^T$, and consider the function f defined by

$$f(\xi, \beta, \phi, c(s)) = \begin{pmatrix} \sin(\xi_2 + \beta_2) \\ \cos(\beta_2)\dfrac{\tan(\phi + \beta_1) - \tan(\beta_2)}{L} - \dfrac{c(s)\cos(\xi_2 + \beta_2)}{1 - \xi_1 c(s)} \end{pmatrix} .$$

From (49.54),

$$\dot{\xi} = u_1 f(\xi, \beta, \phi, c(s)) . \tag{49.58}$$

Define an observer of the form

$$\begin{cases} \dot{\hat{\xi}} = u_1 f(\xi, \hat{\beta}, \phi, c(s)) + \alpha_\xi , \\ \dot{\hat{\beta}} = \alpha_\beta . \end{cases} \tag{49.59}$$

The estimation error $(\tilde{\xi}, \tilde{\beta})^T = (\xi - \hat{\xi}, \beta - \hat{\beta})^T$ satisfies

$$\begin{cases} \dot{\tilde{\xi}} = u_1 [f(\xi, \beta, \phi, c(s)) - f(\xi, \hat{\beta}, \phi, c(s))] - \alpha_\xi , \\ \dot{\tilde{\beta}} = -\alpha_\beta + \dot{\beta} . \end{cases} \tag{49.60}$$

The objective is to define the terms α_ξ and α_β so as to ensure the asymptotic stability of the origin of the above estimation error system when β is constant, i. e., when $\dot{\beta} = 0$, and thus the asymptotic convergence of the estimate $\hat{\beta}$ to β in this case. Furthermore, if β varies slowly the estimation error should remain small.

Proposition 49.6

Let

$$\begin{cases} \alpha_\xi = |u_1| K_1 \tilde{\xi} \\ \alpha_\beta = K_2 u_1 \left[\dfrac{\partial f}{\partial \beta}(\xi, \hat{\beta}, \phi, c(s)) \right]^T \tilde{\xi} \end{cases} \tag{49.61}$$

with K_1 a 2×2 positive definite matrix and K_2 a positive scalar. Assume that β is constant and that

i) The variables $u_1, \xi, \phi, c(s), 1 - dc(s)$ are bounded, differentiable, and their time-derivatives are bounded.

ii) There exists $\tau > 0$ such that

$$|\phi + \beta_1|, \ |\beta_2|, \ |\xi_2 + \beta_2| \leq \pi/2 - \tau .$$

iii) The function u_1 is persistently exciting in the sense that there exist two constants $T, \delta > 0$ such that

$$\forall t, \quad \int_t^{t+T} |u_1(s)| \, \mathrm{d}s \geq \delta .$$

Then, the origin of system (49.60) is locally exponentially stable.

Sketch of Proof: From (49.61), the estimation error dynamics (49.60) can be written as

$$\begin{cases} \dot{\tilde{\xi}} = u_1 \left[\dfrac{\partial f}{\partial \beta}(\xi, \hat{\beta}, \phi, c(s)) \tilde{\beta} + O^2(\tilde{\beta}) \right] - |u_1| K_1 \tilde{\xi} , \\ \dot{\tilde{\beta}} = -K_2 u_1 \left[\dfrac{\partial f}{\partial \beta}(\xi, \hat{\beta}, \phi, c(s)) \right]^T \tilde{\xi} , \end{cases} \tag{49.62}$$

where $O^2(\tilde{\beta})$ denotes a second-order term in $\tilde{\beta}$ in a neighborhood of $\tilde{\beta} = 0$. Note that this term also depends on $\xi, \phi, c(s)$ and $1 - dc(s)$. The linearized system associated with system (49.62) is thus given by

$$\begin{cases} \dot{\tilde{\xi}} = u_1 \dfrac{\partial f}{\partial \beta}(\xi, \beta, \phi, c(s)) \tilde{\beta} - |u_1| K_1 \tilde{\xi} , \\ \dot{\tilde{\beta}} = -K_2 u_1 \left[\dfrac{\partial f}{\partial \beta}(\xi, \beta, \phi, c(s)) \right]^T \tilde{\xi} . \end{cases} \tag{49.63}$$

Consider the candidate Lyapunov function V defined by

$$V(\tilde{\xi}, \tilde{\beta}) = K_2 |\tilde{\xi}|^2 + |\tilde{\beta}|^2 .$$

One verifies that the time-derivative of V along the solutions of system (49.63) satisfies

$$\dot{V} = -|u_1| K_2 \tilde{\xi}^T K_1 \tilde{\xi} .$$

Since K_1 is positive definite and K_2 is positive, V is decreasing. This ensures the stability of the origin $(\tilde{\xi}, \tilde{\beta}) = 0$. Using assumptions i) and ii) and the fact that the matrix $\partial f / \partial \beta(\xi, \beta, \phi, c(s))$ is invertible one can show, that the origin of system (49.63) is asymptotically stable. Assumption iii) implies that the origin is in fact exponentially stable. Using assumptions i) and ii) again, one shows that the origin of the original (nonlinear) system (49.62) is locally exponentially stable. ■

Proposition 49.6 calls for a few remarks. Although convergence of the estimation errors is proved in the case of constant sliding angles only, the observer can still provide a good estimate of β if $\dot{\beta}$ is *small*. For fast variations of β, estimation based on a dynamic model and IMU measurements (inertial measurement

unit) will usually provide better results [49.21]. Satisfaction of assumptions i) and ii) is strongly related to the choice of the controller used to determine the steering wheel angular velocity u_2 in (49.54). In this respect, let us recall that the separation principle, which allows one to design independently a feedback controller and an observer for a linear system with a guarantee of stability of the coupled system, is not systematically satisfied for nonlinear systems.

49.5.3 Feedback Laws for Path Following

The kinematic model (49.54) can be transformed into a chained system. This is a direct extension of the results of Sect. 49.4.1. More precisely, the change of coordinates and control variables $(s, d, \theta_e, \phi, u_1, u_2) \longmapsto (z_1, z_2, z_3, z_4, v_1, v_2)$ defined by

$$
\begin{aligned}
&(z_1, z_2, z_3, z_4) = \\
&\Big\{ s,\ d,\ [1 - dc(s)] \tan(\theta_e + \beta_R), \\
&-c(s)[1 - dc(s)] \left[1 + 2\tan^2(\theta_e + \beta_R)\right] \\
&- d\frac{\partial c}{\partial s} \tan(\theta_e + \beta_R) \\
&+ [1 - dc(s)]^2 \frac{\tan(\phi + \beta_F) - \tan\beta_R}{L} \frac{1 + \tan^2(\theta_e + \beta_R)}{\cos(\theta_e + \beta_R)} \Big\}, \\
&(v_1, v_2) = (\dot{z}_1, \dot{z}_4)
\end{aligned}
$$

transforms the model (49.54) of a car-like vehicle into a 4-D chained system.

Based on this transformation, Sect. 49.4 provides a feedback control solution v_2 to the path following problem. Since the relation between v_2 and the original control variable u_2 involves the sliding angles β_F and β_R, u_2 can be computed from the outputs $\hat{\beta}_F$, $\hat{\beta}_R$ of the observer proposed in the previous section.

In practice, preexisting low-level control loops may use the steering angle ϕ itself, rather than its time-derivative, as a control input. In this case the expression for the *desired* steering angle can be obtained by identifying the expression of $\dot{\theta}_e$ in (49.54) with the one in (49.13), where u_2 is the feedback law deduced from the (chained-form) control law (49.27) of a unicycle-like robot. Further assuming that $c(s)$ is constant, one obtains the following expression

$$
\phi_{ref} = \arctan\left[\tan(\hat{\beta}_R) + \frac{L}{\cos(\hat{\beta}_R)} \left(\frac{c(s)\cos\tilde{\theta}_2}{\kappa} + \frac{A\cos^3\tilde{\theta}_2}{\kappa^2} \right) \right] - \hat{\beta}_F \tag{49.64}
$$

with

$$
\begin{cases}
\tilde{\theta}_2 = \theta_e + \hat{\beta}_R \\
\kappa = 1 - c(s)\, d \\
A = -k_2 d - k_3\, \kappa\, \mathrm{sign}(u_1) \tan\tilde{\theta}_2 + c(s)\kappa \tan^2\tilde{\theta}_2 \,.
\end{cases} \tag{49.65}
$$

49.5.4 Path Following Illustrations in Low Grip Conditions

Simulation Results

The simulation results presented next have been obtained by using the kinematic and dynamic models (49.54) and (49.56) respectively, with the set of parameters for the tire/ground interaction specified in Table 49.1. The reference path, depicted on Fig. 49.16 in black plain line, is composed of two straight lines connected by a circular path with a radius of 13 m. The simulated grip conditions generate sliding along the circular part of the path. The transitions between this circular part and the straight lines yield transient errors due to the curvature discontinuity. The vehicle's velocity is 3 m/s.

The steering velocity u_2 is computed according to the simple proportional feedback law

$$
u_2 = -k_\phi(\phi - \phi_{ref})
$$

with ϕ_{ref} defined by (49.64) and the control gain $k_\phi = 12$ (settling time of 250 ms). The other control gains in (49.64) are defined as $(k_2, k_3) = (0.09, 0.6)$, yielding a settling distance of 12m without overshoot. First,

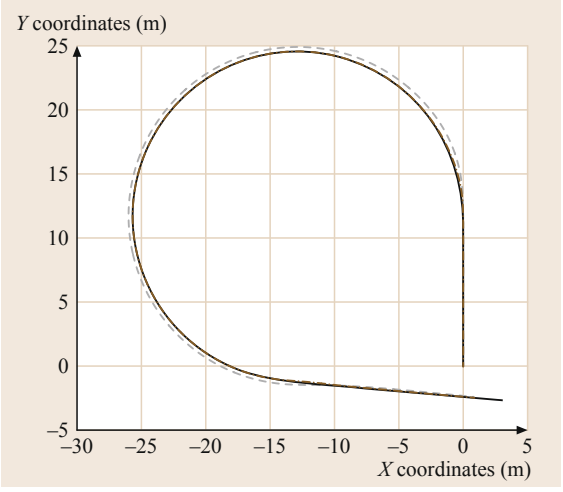


Fig. 49.16 Reference path and trajectories in Cartesian plane

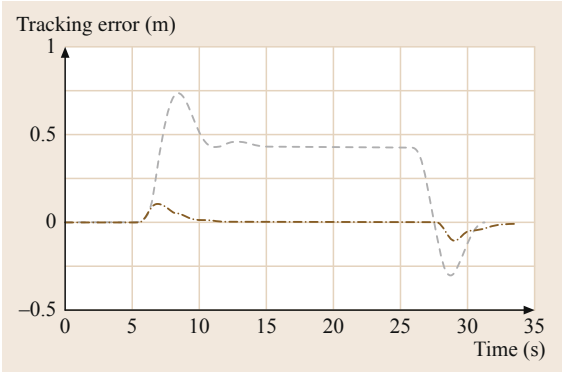


Fig. 49.17 Lateral tracking error (d)

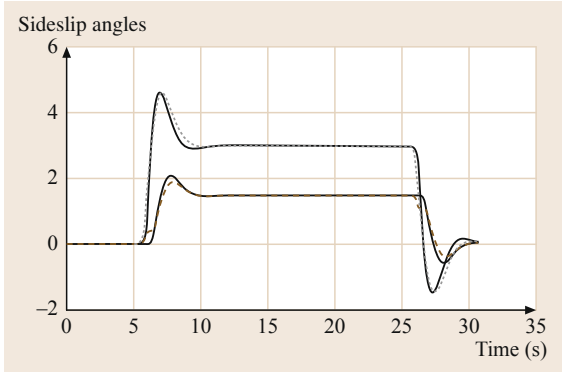


Fig. 49.18 Comparison of estimated and *actual* sliding angles

the control law is applied with $(\hat{\beta}_F, \hat{\beta}_R) = (0, 0)$ (i. e., without considering sliding). The results are shown in blue on Figs. 49.16 and 49.17. Then, the control law is applied with $\hat{\beta}_F$ and $\hat{\beta}_R$ obtained via the observer (49.59)–(49.61), using for K_1 a diagonal matrix with elements equal to 10 and 20 and the gain $K_2 = 10$. The results are shown in red. Fig. 49.17 clearly illustrates the reduction of lateral error resulting from the sliding angles estimation.

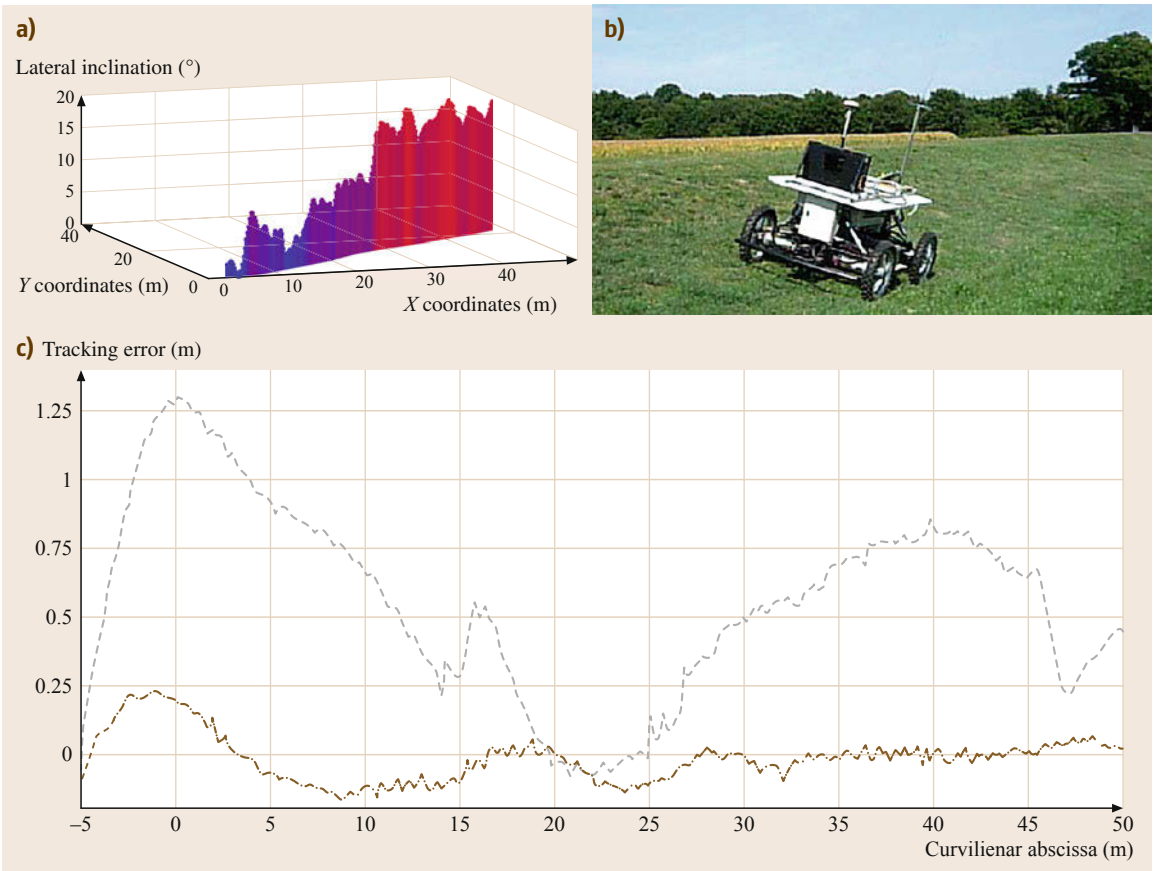


Fig.49.19a–c Path following with sliding effects for an autonomous off-road robot: **(a)** Reference trajectory in 3-D frame **(b)** Robot during autonomous tracking **(c)** Tracking errors (*dashed*: without sliding compensation, *dash-dotted*: with sliding estimation and compensation)

Performance of the proposed observer is illustrated on Fig. 49.18 with the *true* sliding angles (i. e., obtained from the simulated dynamics) in plain lines and the estimated angles in dashed lines. One can verify that the sliding angles are correctly estimated. In practice, using an observer allows one to also compensate for various unmodeled dynamics resulting, for instance, from imprecise knowledge of the vehicle's geometry (the distance L, for example) or from an unknown offset on the steering angle.

Experimental Results

Full scale experiments on natural terrain with the car-like vehicle of Fig. 49.19 are reported next and in VIDEO 435. This mobile robot is equipped with a real time kinematics global positioning system (RTK-GPS) producing a $\pm 2\,\mathrm{cm}$ accurate absolute position measurement. The GPS antenna is located straight above the control point P_R. The reference path, consisting of a straight line perpendicular to the slope direction, has been previously recorded during a manual driving. The top left of Fig. 49.19 shows the recorded coordinates in x-y plane, and the z axis represents the absolute value of the lateral slope angle α. This angle progressively reaches the value of 10° at the curvilinear abscissa 27 m, and it subsequently rapidly increases up to about 18°.

As for the simulation results reported previously, the feedback control (49.64) was first applied with

$$\hat{\beta}_F = \hat{\beta}_R = 0$$

(i. e., without taking sliding into account), then with $\hat{\beta}_F, \hat{\beta}_R$ given by the observer. In both cases, control gains equal to these used in simulation have been used, i. e., $(k_2, k_3) = (0.09, 0.6)$. The vehicle's longitudinal velocity during experiments was $3\,\mathrm{m/s}$. The bottom part of Fig. 49.19 shows the tracking error along the path with the blue dashed line corresponding to using $(\beta_F, \beta_R) = (0, 0)$ in the control law, and the red dashed-dotted line corresponding to the use of sliding angles estimated values.

These experimental results confirm the simulation ones reported before. The improvement in tracking accuracy resulting from the use of estimated sliding angles in the control law is clearly illustrated: the lateral error remains small (smaller than 0.25 m) all the time and it becomes negligible as soon as the terrain's slope is alsmost constant (beyond the curvilinear abcissa 30 m). This is consistent with the observer's stability and convergence analysis when the sliding angles are assumed constant.

49.6 Complementary Issues and Bibliographical Guide

49.6.1 General Trailer Systems

Most of the control design approaches here presented and illustrated for unicycle-like and car-like vehicles can be extended to the case of trains of vehicles composed of trailers hitched to a leading vehicle. In particular, the methods of Sect. 49.4 which are specific to nonholonomic systems can be extended to this case, provided that the kinematic equations of motion of the system can be transformed (at least semiglobally) into a chained system [49.6]. This basically requires that the hitch point of each trailer is located on the rear-wheel axle of the preceding vehicle [49.22]. For instance, the transformation to the chained form is not possible when there are two (or more) successive trailers with off-axle hitch points [49.23]. So-called *general trailer systems* (with off-axle hitch points) raise difficult control design issues, and the literature devoted to them is sparse. For this reason, and also because these systems are not met in applications as frequently as simpler vehicles, control methods specifically developed for them are not reported here. Nonetheless, a few related references are given next. The path-following problem has been considered in, e.g., [49.24] for a system with two trailers and, more generally, in [49.25, Chap. 3] and [49.26] for an arbitrary number of trailers. To our knowledge, the problem of stabilizing nonstationary reference trajectories has not been addressed for these systems (except in the single trailer case for which the system can be transformed into the chained form [49.23, 27]). In fact, the explicit calculation of feasible trajectories joining a given configuration to another is already a very difficult problem, even in the absence of obstacles. As for the asymptotic stabilization of fixed configurations, the problem can (in theory) be solved by using existing general methods developed for the larger class of controllable driftless systems. However, the calculations associated with these methods quickly become intractable when the number of trailers increases. More specific and simpler ones have been proposed in [49.28], for an arbitrary number of trailers and the asymptotic stabilization of a reduced set of configurations, and in [49.29], in the case of two trailers and arbitrary fixed configurations.

49.6.2 Sensor-Based Motion Control

The control laws described in the present chapter, and their calculation, involve the online measurement, eventually complemented by the online estimation, of variables depending on the position of the vehicles in their environment. Measures can be acquired via the use of various sensors (odometry, GPS, proximetry, vision, etc.). Usually, various treatments are applied to raw sensory data prior to computing the control variables themselves. For instance, noise filtering and state estimation are such basic operations, well documented in the automatic-control literature. Among all sensors, vision sensors play a particularly important role in robotic applications, due to the richness and versatility of the information which they provide. The combination of visual data with feedback control is often referred to as *visual servoing*. In Chap. 34, a certain number of visual servoing tasks are addressed, mostly in the context of manipulation and/or under the assumption that realizing the robot task is equivalent to controlling the pose of a camera mounted on an omnidirectional manipulator. In a certain number of cases, the concepts and methods described in this chapter can be adapted, without much effort, to the context of mobile robots. These cases basically correspond to the control methods adapted from robotic manipulation which are described in Sect. 49.3 of the present chapter. For instance, automatic driving via the control of the visually estimated lateral distance between a robotic vehicle and the side of a road, or car-platooning by controlling the frontal and lateral distances to a leading vehicle, can be addressed with the control techniques reported in Chap. 34. The reason is that it is possible to simply recast these techniques in the form of the control laws proposed in Sect. 49.3. However, there are also vision-based applications for nonholonomic mobile robots which cannot be solved by applying classical visual-servoing techniques. This is the case, for instance, of the task objectives addressed in Sect. 49.4, an example of which is the stabilization of the complete posture (i. e., position and orientation) of a nonholonomic vehicle at a desired one. Vision-based control problems of this type have been addressed in [49.11, 30].

49.6.3 Sliding Effects and Other Dynamical Issues

Results of Sect. 49.5 can be viewed as a first step to addressing dynamical issues associated with sliding effects. The kinematic control models (49.53)–(49.54) can be extended to other wheeled vehicles ([49.31] for the mobile robot classification proposed by [49.32] and [49.33] for a trailer system). On-line estimation of sliding angles via an observer can also be achieved by exploiting the dynamics of these angles and making use of complementary measurements (from an IMU for example). A result in this direction is proposed in [49.21] where a model of sliding angle dynamics is used at the observer level, thus yielding a much more reactive estimation of these angles. Note that the kinematic control law of Sect. 49.5 may still be used in this case. When prior knowledge of the reference path is available, a predictive control strategy may also be adopted to reduce transient error effects associated with slow steering angle dynamics.

Various risks related to the vehicle's integrity when operating at high speed, like e.g., swing around or roll-over, were not addressed in this chapter. In [49.34] these issues are treated at the path planning level. Knowledge of the robot's dynamics can also be used to relate control inputs to a metrics measuring the importance of such risks in order to introduce constraints at the control level [49.35]. A multimodels approach (kinematic and dynamic) can be used to translate stability aspects into control constraints, for instance by defining a maximal admissible longitudinal velocity. A predictive control strategy may also help to limit the risk of roll-over, of steering saturation, or of swing around [49.36]. Finally, it is possible to take advantage of additional actuators, as in the case of four independent actuated wheels [49.37], or by using inclination actuators [49.38].

49.6.4 Bibliographical Guide

A few former surveys on the control of WMRs have been published. Let us mention [49.39–41], which contain chapters on the modeling and control issues. A detailed classification of kinematic and dynamic models for the different types of WMR structures, on which Chap. 24 is based, is provided in [49.32]. The use of the chain form to represent WMR equations has been proposed in [49.42], then generalized in [49.22].

Path following may have been the first mobile robot control problem addressed by researchers in robotics. Among the pioneering works, let us cite [49.43, 44]. Several results presented in the present chapter are based on [49.6, 45].

The problem of tracking admissible trajectories for unicycle-type and car-like vehicles is treated in the books [49.39–41], and also in numerous conference and journal papers. Several authors have addressed this problem by applying dynamic feedback linearization techniques. In this respect, one can consult [49.46–48], and [49.39, Chap. 8], for instance.

Numerous papers on the asymptotic stabilization of fixed configurations have been published. Among

them, [49.49] provides an early overview of feedback control techniques elaborated for this purpose, and also a list of references. The first result presenting a time-varying feedback solution to this problem, in the case of a unicycle-type vehicle, is in [49.4]. The conference paper [49.50] provides a survey on time-varying feedback stabilization, in the more general context of nonlinear control systems. More specific results, like Propositions 49.2 and 49.4, are given in [49.6, 7]. Other early results on the design of smooth time-varying feedbacks can be found in [49.51, 52], for example. Concerning continuous (but not Lipschitz-continuous) time-varying feedbacks yielding exponential convergence, one can consult [49.53]. Designs of hybrid discrete/continuous fixed-point stabilizers can be found in [49.54–57], for instance. Discontinuous control design techniques are not addressed in the present chapter, but the interested reader will find examples of such feedbacks in [49.58, 59].

To our knowledge, the control approach presented in Sect. 49.4.6, which is based on the concept of transverse functions [49.9, 10], is the first attempt to address the problem of tracking *arbitrary* trajectories (i. e., not necessarily feasible for the controlled robot). Implementation issues and experimental results for this approach can be found in [49.11, 12]. An overview of trajectory tracking problems for wheeled mobile robots, with a detailed case study of car-like systems, is presented in [49.13].

Video-References

VIDEO 181 Tracking of an admissible trajectory with a car-like vehicle
available from http://handbookofrobotics.org/view-chapter/49/videodetails/181

VIDEO 182 Tracking of arbitrary trajectories with a truck-like vehicle
available from http://handbookofrobotics.org/view-chapter/49/videodetails/182

VIDEO 243 Tracking of an omnidirectional frame with a unicycle-like robot
available from http://handbookofrobotics.org/view-chapter/49/videodetails/243

VIDEO 435 Mobile robot control in off-road condition and under high dynamics
available from http://handbookofrobotics.org/view-chapter/49/videodetails/435

References

49.1 M. Buehler, K. Iagnemma, S. Sanjiv (Eds.): *The 2005 DARPA Grand Challenge: The Great Robot Race*, Springer Tracts in Advanced Robotics, Vol. 36 (Springer, Berlin, Heidelberg 2007)

49.2 P. Morin, C. Samson: Motion control of wheeled mobile robots. In: *Springer Handbook of Robotics*, ed. by B. Siciliano, O. Khatib (Springer, Berlin, Heidelberg 2008) pp. 799–826

49.3 R.W. Brockett: Asymptotic stability and feedback stabilization. In: *Differential Geometric Control Theory*, ed. by R.W. Brockett, R.S. Millman, H.J. Sussmann (Birkhäuser, Boston 1983)

49.4 C. Samson: Velocity and torque feedback control of a nonholonomic cart, Lect. Notes Control Inform. Sci. **162**, 125–151 (1991)

49.5 J.-M. Coron: Global asymptotic stabilization for controllable systems without drift, Math. Control Signals Syst. **5**, 295–312 (1992)

49.6 C. Samson: Control of chained systems. Application to path following and time-varying point-stabilization, IEEE Trans. Autom. Control **40**, 64–77 (1995)

49.7 P. Morin, C. Samson: Control of non-linear chained systems. From the Routh–Hurwitz stability criterion to time-varying exponential stabilizers, IEEE Trans. Autom. Control **45**, 141–146 (2000)

49.8 D.A. Lizárraga: Obstructions to the existence of universal stabilizers for smooth control systems, Math. Control Signals Syst. **16**, 255–277 (2004)

49.9 P. Morin, C. Samson: Practical stabilization of driftless systems on Lie groups: the transverse function approach, IEEE Trans. Autom. Control **48**, 1496–1508 (2003)

49.10 P. Morin, C. Samson: A characterization of the Lie algebra rank condition by transverse periodic functions, SIAM J. Control Optim. **40**(4), 1227–1249 (2001)

49.11 G. Artus, P. Morin, C. Samson: Control of a maneuvering mobile robot by transverse functions, Symp. Adv. Robot Kinemat. (ARK) (2004) pp. 459–468

49.12 G. Artus, P. Morin, C. Samson: Tracking of an omnidirectional target with a nonholonomic mobile robot, IEEE Conf. Adv. Robotics (ICAR) (2003) pp. 1468–1473

49.13 P. Morin, C. Samson: Trajectory tracking for nonholonomic vehicles: overview and case study, Proc. 4th Int. Workshop Robot Motion Control (RoMoCo), ed. by K. Kozlowski (2004) pp. 139–153

49.14 R. Lenain, B. Thuilot, C. Cariou, P. Martinet: High accuracy path tracking for vehicles in presence of sliding. application to farm vehicle automatic guidance for agricultural tasks, Auton. Robots **21**(1), 79–97 (2006)

49.15 T.D. Gillespie: *Fundamentals of Vehicle Dynamics* (SAE, Warrendale 1992)
49.16 P.R. Dahl: Solid friction damping of mechanical vibrations, AIAA J. **14**(12), 1675–1682 (1976)
49.17 C. Canudas de Wit, H. Olsson, K.J. Astrom, P. Lischinsky: A new model for control of systems with friction, IEEE Trans. Autom. Control **40**(3), 419–425 (1995)
49.18 C. Canudas de Wit, P. Tsiotras: Dynamic tire friction models for vehicle traction control, Proc. 38th IEEE Conf. Decis. Control, Vol. 4 (1999)
49.19 E. Bakker, L. Nyborg, H.B. Pacejka: Tyre modeling for use in vehicle dynamics studies, International Conference of the Society of Automotive Engineers (SAE) (1987) pp. 2190–2204
49.20 H.B. Pacejka: *Tyre and Vehicle Dynamics* (Butterworth-Heinemann, Oxford 2002)
49.21 R. Lenain, B. Thuilot, C. Cariou, P. Martinet: Mixed kinematic and dynamic sideslip angle observer for accurate control of fast off-road mobile robots, J. Field Robotics **27**(2), 181–196 (2010)
49.22 O.J. Sørdalen: Conversion of the kinematics of a car with *n* trailers into a chained form, IEEE Int. Conf. Robot. Autom. (ICRA) (1993) pp. 382–387
49.23 P. Rouchon, M. Fliess, J. Lévine, P. Martin: Flatness, motion planning and trailer systems, IEEE Int. Conf. Decis. Control (1993) pp. 2700–2705
49.24 P. Bolzern, R.M. DeSantis, A. Locatelli, D. Masciocchi: Path-tracking for articulated vehicles with off-axle hitching, IEEE Trans. Control Syst. Technol. **6**, 515–523 (1998)
49.25 D.A. Lizárraga: Contributions à la Stabilisation des Systèmes Non-Linéaires et à la Commande de Véhicules Sur Roues, Ph.D. Thesis (INRIA-INPG, University of Grenoble, Grenoble 2000)
49.26 C. Altafini: Path following with reduced off-tracking for multibody wheeled vehicles, IEEE Trans. Control Syst. Technol. **11**, 598–605 (2003)
49.27 F. Lamiraux, J.-P. Laumond: A practical approach to feedback control for a mobile robot with trailer, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 3291–3296
49.28 D.A. Lizárraga, P. Morin, C. Samson: Chained form approximation of a driftless system. Application to the exponential stabilization of the general N-trailer system, Int. J. Control **74**, 1612–1629 (2001)
49.29 M. Venditelli, G. Oriolo: Stabilization of the general two-trailer system, IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 1817–1823
49.30 M. Maya-Mendez, P. Morin, C. Samson: Control of a nonholonomic mobile robot via sensor-based target tracking and pose estimation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 5612–5618
49.31 D. Wang, C.B. Low: An analysis of wheeled mobile robots in the presence of skidding and slipping: Control design perspective, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2379–2384
49.32 G. Campion, G. Bastin, B. d'Andréa-Novel: Structural properties and classification of kynematic and dynamic models of wheeled mobile robots, IEEE Trans. Robotics Autom. **12**, 47–62 (1996)
49.33 C. Cariou, R. Lenain, M. Berducat, B. Thuilot: Autonomous maneuvers of a farm vehicle with a trailed implement in headland, Proc. 7th Int. Conf. Inform. Control Autom. Robotics (ICINCO), Vol. 2 (2010) pp. 109–114
49.34 K. Iagnemma, S. Shimoda, Z. Shiller: Near-optimal navigation of high speed mobile robots on uneven terrain, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 4098–4103
49.35 N. Bouton, R. Lenain, B. Thuilot, P. Martinet: A new device dedicated to autonomous mobile robot dynamic stability: application to an off-road mobile robot, Proc. IEEE International Conference on Robotics and Automation (ICRA) (2010) pp. 3813–3818
49.36 O. Hach, R. Lenain, B. Thuilot, P. Martinet: Avoiding steering actuator saturation in off-road mobile robot path tracking via predictive velocity control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5523–5528
49.37 E. Lucet, C. Grand, D. Salle, P. Bidaud: Stabilization algorithm for a high speed car-like robot achieving steering maneuver, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 2540–2545
49.38 M. Krid, F. Ben-Amar: Design and control of an active anti-roll system for a fast rover, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 274–279
49.39 C. Canudas de Wit, B. Siciliano, G. Bastin (Eds.): *Theory of Robot Control* (Springer, Berlin, Heidelberg 1996)
49.40 J.-P. Laumond (Ed.): *Robot Motion Planning and Control*, Lecture Notes in Control and Information Sciences, Vol. 229 (Springer, Berlin, Heidelberg 1998)
49.41 Y.F. Zheng (Ed.): *Recent Trends in Mobile Robots*, World Scientific Series in Robotics and Automated Systems, Vol. 11 (World Scientific, Singapore 1993)
49.42 R.M. Murray, S.S. Sastry: Steering nonholonomic systems in chained form, IEEE Int. Conf. Decis. Control (1991) pp. 1121–1126
49.43 E.D. Dickmanns, A. Zapp: Autonomous high speed road vehicle guidance by computer vision, Proc. IFAC 10th World Congr. Autom. Control. (1987)
49.44 W.L. Nelson, I.J. Cox: Local path control for an autonomous vehicle, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1998) pp. 1504–1510
49.45 C. Samson: Path following and time-varying feedback stabilization of a wheeled mobile robot, Proc. Int. Conf. Autom. Robotics Comput. Vis. (1992)
49.46 B. d'Andréa-Novel, G. Campion, G. Bastin: Control of nonholonomic wheeled mobile robots by state feedback linearization, Int. J. Robotics Res. **14**, 543–559 (1995)
49.47 A. De Luca, M.D. Di Benedetto: Control of nonholonomic systems via dynamic compensation, Kybernetica **29**, 593–608 (1993)
49.48 M. Fliess, J. Lévine, P. Martin, P. Rouchon: Flatness and defect of non-linear systems: Introductory

theory and examples, Int. J. Control **61**, 1327–1361 (1995)
49.49 I. Kolmanovsky, N.H. McClamroch: Developments in nonholonomic control problems, IEEE Control Syst. **15**, 20–36 (1995)
49.50 P. Morin, J.-B. Pomet, C. Samson: Developments in time-varying feedback stabilization of nonlinear systems, IFAC Nonlinear Control Syst. Design Symp. (1998) pp. 587–594
49.51 J.-B. Pomet: Explicit design of time-varying stabilizing control laws for a class of controllable systems without drift, Syst. Control Lett. **18**, 467–473 (1992)
49.52 A.R. Teel, R.M. Murray, G. Walsh: Nonholonomic control systems: from steering to stabilization with sinusoids, Int. J. Control **62**, 849–870 (1995)
49.53 R.T. M'Closkey, R.M. Murray: Exponential stabilization of driftless nonlinear control systems using homogeneous feedback, IEEE Trans. Autom. Control **42**, 614–6128 (1997)
49.54 M.K. Bennani, P. Rouchon: Robust stabilization of flat and chained systems, Eur. Control Conf. (1995) pp. 2642–2646
49.55 P. Lucibello, G. Oriolo: Stabilization via iterative state feedback with application to chained-form systems, IEEE Conf. Decis. Control (1996) pp. 2614–2619
49.56 O.J. Sørdalen, O. Egeland: Exponential stabilization of nonholonomic chained systems, IEEE Trans. Autom. Control **40**, 35–49 (1995)
49.57 P. Morin, C. Samson: Exponential stabilization of nonlinear driftless systems with robustness to unmodeled dynamics, ESAIM Control Optim. Calc. Var. **4**, 1–36 (1999)
49.58 A. Astolfi: Discontinuous control of nonholonomic systems, Syst. Control Lett. **27**, 37–45 (1996)
49.59 C. Canudas de Wit, O.J. Sørdalen: Exponential stabilization of mobile robots with nonholonomic constraints, IEEE Trans. Autom. Control **37**(11), 1791–1797 (1992)

Multimedia Contents

50. Modeling and Control of Robots on Rough Terrain

Keiji Nagatani, Genya Ishigami, Yoshito Okada

In this chapter, we introduce modeling and control for wheeled mobile robots and tracked vehicles. The target environment is rough terrains, which includes both deformable soil and heaps of rubble. Therefore, the topics are roughly divided into two categories, wheeled robots on deformable soil and tracked vehicles on heaps of rubble.

After providing an overview of this area in Sect. 50.1, a modeling method of wheeled robots on a deformable terrain is introduced in Sect. 50.2. It is based on terramechanics, which is the study focusing on the mechanical properties of natural rough terrain and its response to off-road vehicle, specifically the interaction between wheel/track and soil. In Sect. 50.3, the control of wheeled robots is introduced. A wheeled robot often experiences wheel slippage as well as its sideslip while traversing rough terrain. Therefore, the basic approach in this section is to compensate the slip via steering and driving maneuvers. In the case of navigation on heaps of rubble, tracked vehicles have much advantage. To improve traversability in such challenging environments, some tracked vehicles are equipped with subtracks, and one kinematical modeling method of tracked vehicle on rough terrain is introduced in Sect. 50.4. In addition, stability analysis of such vehicles is introduced in Sect. 50.5. Based on such kinematical model and stability analysis, a sensor-based control of tracked vehicle on rough terrain is introduced in Sect. 50.6. Sect. 50.7 summarizes this chapter.

50.1 **Overview** 1268
- 50.1.1 Modeling Deformable Terrain for Robots 1268
- 50.1.2 Controlling Robots in Deformable Terrain 1268
- 50.1.3 Modeling and Controlling Robots in Heaps of Rubble 1269

50.2 **Modeling of Wheeled Robot in Rough Terrain** 1270
- 50.2.1 Dynamics of Mobile Robots 1270
- 50.2.2 Wheel–Terrain Interaction Mechanics 1271
- 50.2.3 Wheel Traction Performance 1273
- 50.2.4 Model Uncertainty and Soil Parameter Identification 1273

50.3 **Control of Wheeled Robot in Rough Terrain** 1274
- 50.3.1 Slip-Compensated Path Follower 1274
- 50.3.2 Steering and Driving Maneuvers 1275

50.4 **Modeling of Tracked Vehicle on Rough Terrain** 1276
- 50.4.1 Parameterization of a Track 1277
- 50.4.2 Contact with a Single Point in the Terrain Surface 1277
- 50.4.3 Contact with the Terrain Surface Represented by a Point Cloud 1277

50.5 **Stability Analysis of Tracked Vehicles** 1278

50.6 **Control of Tracked Vehicle on Rough Terrain** 1279
- 50.6.1 Ground Detection and Trimming of the Scanned Data 1280
- 50.6.2 Determination of Desired Posture 1280
- 50.6.3 Determination of Desired Subtrack Positions 1280
- 50.6.4 Stability Evaluation of Desired Pose 1281
- 50.6.5 Stabilization of Desired Pose 1281
- 50.6.6 Position Control of Subtracks 1281

50.7 **Summary** 1281

Video-References 1281

References 1282

50.1 Overview

50.1.1 Modeling Deformable Terrain for Robots

The category of rough terrain includes a variety of ground conditions. In the case of weak and fine gravel, the motion of a wheeled mobile robot becomes relatively complicated because the interaction mechanics of the wheel in a rough terrain differs from that of an indoor robot on a flat surface; a wheel on gravel has multiple point contact, and a wheel on loose soil has a surface contact that deforms the soil around the wheel.

There has been a large amount of research on robot mobility analysis in the military community [50.1, 2]. These studies have primarily focused on empirical analyses of large vehicles (i. e., gross vehicle weights of several tons). The mobility analyses of small mobile robots that consider the interaction mechanics of a slipping wheel on rough terrain have also been investigated. Additionally, a multibody system simulation for the longitudinal slip of tires with respect to the tire–soil interaction has been demonstrated [50.3]. A dynamic interaction model, denoted as soil contact model (SCM), has been developed to provide the dynamics of a plastically deformable surface using an elevation grid, as well as to consider the multipass effect of wheels, shown in Fig. 50.1 [50.4]. Recently, multibody dynamics toolkit, such as Vortex (VIDEO 184), can be used to simulate terrain reaction forces for mobility analyses of mobile robots [50.5].

Recently, terramechanics [50.6] has been widely used in applications such as off-road UGVs (unmanned ground vehicles) [50.7, 8] and planetary rovers on Mars. A fundamental requirement for such off-road mobile robots is to maintain a traversing capability in rough terrain composed of loose sand, mud, pebbles, etc. One typical issue in such an environment is wheel slippage, which is generated because of the soil deformation at the wheel–terrain contact patch. Wheel slippage often degrades the mobility performance of the robot because some amount of tractive power from a driving actuator is consumed by the soil deformation. Furthermore, the wheel slippage induces vehicle sideslip, which is an additional disturbance for the motion control of a wheeled robot. Figure 50.2 shows one typical example of lateral- and side-slip situation of a wheeled mobile robot in rough terrain. One typical example of terramechanics-based modeling of rough terrain is described in Sect. 50.2. Recently, a study of terradynamics of legged locomotion was proposed for traversal in granular media in [50.9] and is shown in VIDEO 186.

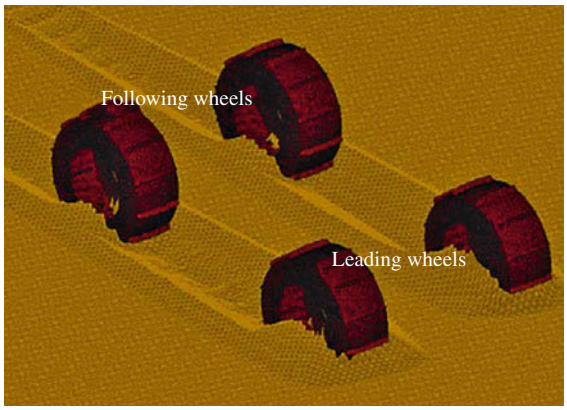


Fig. 50.1 Dynamics of a plastically deformable surface in consideration with the multipass effect of wheels (after [50.4])

50.1.2 Controlling Robots in Deformable Terrain

A wheeled robot often experiences wheel slippage as well as vehicle sideslip while traversing rough terrain. This degrades the motion control performance of the robot, i. e., a robot could deviate from the path to be followed or a wheel may get trapped in loose soil. Therefore, a control scheme that can derive the appropriate steering/driving maneuvers is necessary for a wheeled robot to achieve traversability on a rough terrain and to manage any slippage problems that may arise.

A large number of studies related to the path-following control of mobile robots have been published. General information on path-following issues can be obtained from [50.10–12]. *Rezaei* et al. investigated an online path-following strategy combined with a simultaneous localization and mapping (SLAM) algorithm for a car-like robot in outdoor environments [50.13].

Fig. 50.2 Lateral- and side-slip situation of wheeled mobile robot

Coelho and *Nunes* et al. proposed a Kalman-based active observer controller for the path-following of wheeled mobile robots [50.14]. *Helmick* et al. developed a path-following algorithm that included slip compensation by using visual odometry and a Kalman filter [50.15, 16]. The method was applied to the rocker-bogie configuration with six steerable wheels robot, shown in Fig. 50.3. In Sect. 50.3, a path-following control scheme for a four-wheeled mobile robot is introduced. The experimental validation of the control described in the section was reported in [50.17].

50.1.3 Modeling and Controlling Robots in Heaps of Rubble

In environments that involve heaps of rubble, such as an urban search and rescue situation, the approach for modeling and controlling robots is different from the afore mentioned methodology. To tackle such challenging environments, tracks are used to enhance the mobility of a robot in such environments because they have more contact with the terrain than wheels. Some tracked vehicles are equipped with active subtracks, and multitracks configuration enables the vehicle to get over relatively large steps. A typical geometrical model of tracked vehicles with subtracks is introduced in Sect. 50.4.

In case that the target rubble is stable, kinematic approaches are useful for navigation and control of the robot. In this case, stability analysis is very important. In classical stability analysis, the energy stability margin (ESM) proposed by *Messuri* and *Klein* was defined in terms of the potential energy of the robot [50.18]. Hirose assumed that increasing the robot's weight did not always contribute to its stability because it also increased the dynamic disturbance around its center of gravity. Therefore, he proposed the normalized ESM (NESM), in which the robot's weight was normalized [50.19]. For this criterion, the NESM is used to evaluate the stability of a robot on the basis of the vertical distance between the initial and largest heights of the center of gravity during a rollover. Recently, *Tubouchi*'s group proposed a method to control tracked vehicles exposed to random steps [50.20] (Fig. 50.4). An artificial uneven environment is random steps, arranged with different lengths of squared timbers. It also appears in VIDEO 189. This environment is used to evaluate the performance for urban search and rescue robots [50.21]. The basic objective is to maintain the maximal stability of the robot at every step of its path in the random step field. The earlier-given topic relating to stability analysis is introduced in Sect. 50.5. Another approach for stability analysis of tracked vehicles-based NESM was reported in [50.22].

To evaluate the stability of vehicles, terrain detection is an important technology. Therefore, many robotics researchers have focused on the terrain mapping, particularly based on the SLAM, described in Chap. 46, and machine learning. LIDAR (light detection and ranging) and stereo cameras are widely used for the terrain mapping (Sect. 22.3). *Vandapel* et al. proposed a classification method of three-dimensional (3-D) point cloud data on the basis of its geometrical features [50.23], and *Lacaze* et al. demonstrated a method of finding a path in heavy grasses [50.25]. To improve the LIDAR-based mapping, *Ohno* et al. proposed a method to obtain thin objects such as nets, poles, and wires [50.26]. Figure 50.5 shows a mapping result at an outdoor rough terrain.

If the rubble is unstable, modeling the machine-rubble interaction is very complicated. One challenging

Fig. 50.3 Rocky 8: rocker-bogie configuration with six steerable wheels robot

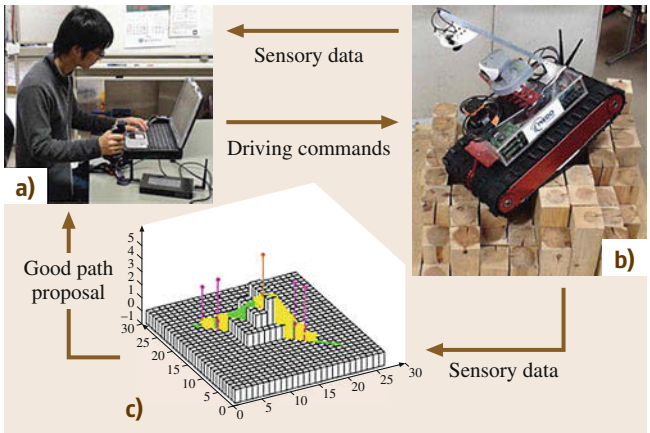


Fig. 50.4a–c Teleoperated system of tracked vehicle. Itconsists of **(a)** a remote operation station, **(b)** a mobile robot platform, and **(c)** path planner that maintain the maximal stability of the robot (after [50.20])

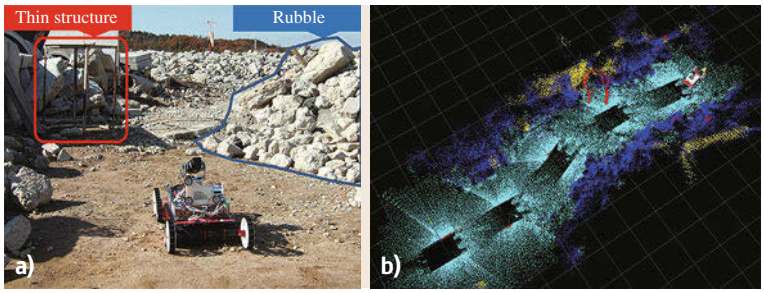


Fig.50.5a,b Classification of outdoor rubble data. **(a)** Target field: Hyogo Prefectural Emergency Management and Training Center, and **(b)** classified result for traversal (after [50.23])

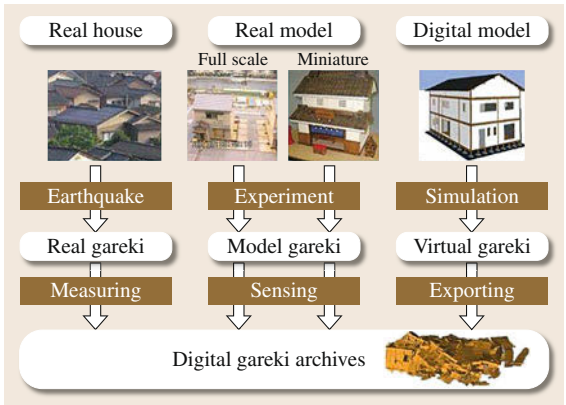


Fig. 50.6 Approaches for gareki data acquisition (after [50.24])

approach to modeling rubble is called gareki (rubble in Japanese) engineering [50.27, 28]. The model includes not only the structure data, but also the internal force of each gareki element. To obtain the data, they proposed to use information about the following, as shown in Fig. 50.6: (1) a real-collapsed building, (2) artificial full- and miniature-scale collapsed buildings, and (3) a simulated virtual digital building. Their research aims to eventually apply the gareki model for the control of mobile robots and rescue missions.

The sensor-based method is another approach for controlling robots in unstable rubbles. Such robots often have multiple tracks connected by joints to maneuver over natural/artificial steps, bumps, and stairs. Some are equipped with passive joints [50.29], but many are equipped with active joints [50.30]. Multitrack configurations enable the vehicle to maneuver over a step that is higher than the radius of the tracks. To enable such motion, a sensor-based method is proposed to control the multitracks. The robot moves and detects the surface shape of the ground continuously, and it changes the configuration of the subtracks based on the surface information. Once the surface of the ground is deformed by the traversal of the robot, the subtracks adapt quickly. A representative example for the control of robots in unstable rubble is shown in Sect. 50.6. Another sensor-based approach is to use force-feedback from contact points. *Inoue* et al., embedded force sensors in their subtracks to obtain the contact force, and changed the configuration of the subtracks [50.31].

50.2 Modeling of Wheeled Robot in Rough Terrain

This section introduces a mechanics of wheeled mobile robot in rough terrain, particularly focusing on a modeling of wheel–terrain interaction.

50.2.1 Dynamics of Mobile Robots

Figure 50.7 depicts a schematic illustration of a dynamic model for a wheeled mobile robot. The robot is modeled as an articulated multibody system. The equation of motion for the robot is numerically given by

$$\mathbf{H}\begin{pmatrix}\dot{\mathcal{V}}_\mathrm{b}\\ \ddot{\boldsymbol{q}}\end{pmatrix}+\boldsymbol{C}+\boldsymbol{G}=\begin{pmatrix}\mathcal{F}_\mathrm{b}\\ \boldsymbol{\tau}\end{pmatrix}+\mathbf{J}^\mathrm{T}\mathcal{F}_\mathrm{e}\,, \tag{50.1}$$

where the symbols have the following meanings: $\mathbf{H}$: inertia matrices composed of multibody of the robot, $\boldsymbol{C}$: velocity depending term, $\boldsymbol{G}$: gravity term, $\mathcal{V}_\mathrm{b}$: translational and angular velocities of the vehicle, $\boldsymbol{q}$: joint angles (such as wheel rotation and steering angles), $\mathcal{F}_\mathrm{b}$: forces and moments at the centroid of the vehicle body, $\boldsymbol{\tau}$: torques acting at each joint (driving/steering torques), $\mathbf{J}$: Jacobian matrix, $\mathcal{F}_\mathrm{e}$: external forces and moments acting at the centroid of each wheel, namely

$$f_{ij}(i=\{r,l\},j=\{r,m,f\})\,.$$

The dynamic motion of the robot with given traveling and steering maneuvers are numerically obtained

by successively solving (50.1). Here, a key approach for the dynamic model is to incorporate a well-defined contact model into the equation of motion in order to calculate the external forces and torques at the contact patch between the wheel and terrain. Some recent works have reported dynamic simulation tools combined with the terramechanics wheel model: for example, [50.32, 33] for the NASA's rovers, and [50.4, 34] for the ExoMars.

50.2.2 Wheel–Terrain Interaction Mechanics

Terramechanics is a study focusing on the mechanical properties of natural rough terrain and its response to off-road vehicle, specifically the interaction between wheel/track and soil. In 1960s, *Bekker* developed a classical terramechanics with many of the key concepts related to vehicle–terrain response, such as a well-known pressure-sinkage equation and shear stress model [50.35, 36]. Wong developed a comprehensive procedure for predicting the performance of both driven and towed wheels [50.37–39]. The procedure calculates the wheel mechanics with applying the stress distribution model beneath the wheel.

Terramechanics-based approach for vehicle–terrain mechanics includes mainly three methodologies [50.40, 41]: (1) Analytical approach, (2) empirical approach, and (3) numerical approach.

The analytical approach is based on a theoretical model for vehicle–terrain interactions along with experimental model validation. The empirical method is to utilize a practical measurement of soil strength

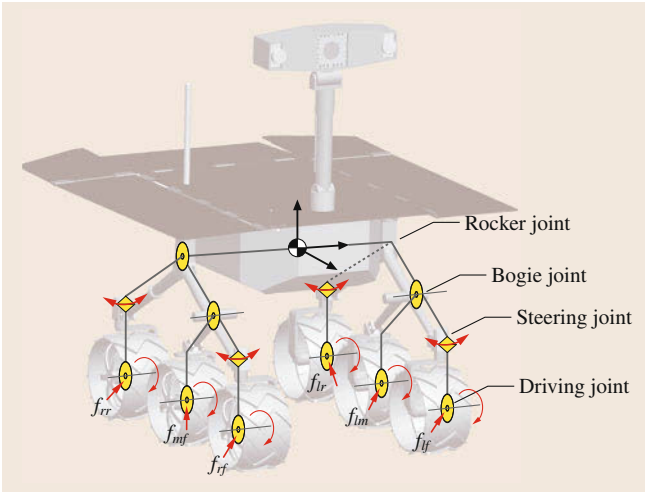


Fig. 50.7 Dynamic model of wheeled mobile robot

by a specialized apparatus in order to estimate vehicle traversability on weak/rough terrain. The numerical method includes the finite element method (FEM) and discrete element method (DEM) that models soils as vast numbers of multiple particles and simulates each particle's behavior while accounting vehicle–terrain interaction [50.42–44].

This subsection focuses on the analytical method and introduces a typical interaction model of rigid wheel on deformable terrain (Fig. 50.8). It should be noted that Chap. 55.3.13 also addresses the wheel traction mechanics in detail, and therefore, the following subsection recalls some key equations in order to dis-

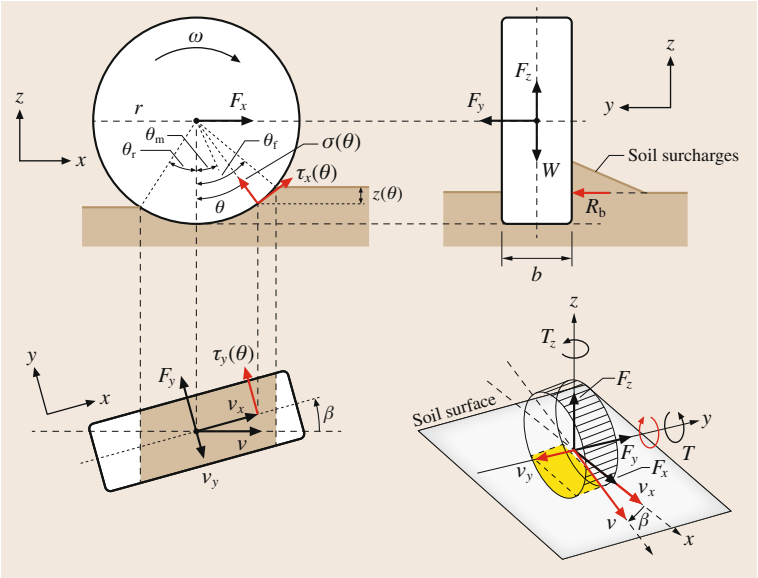


Fig. 50.8 Wheel–terrain contact model

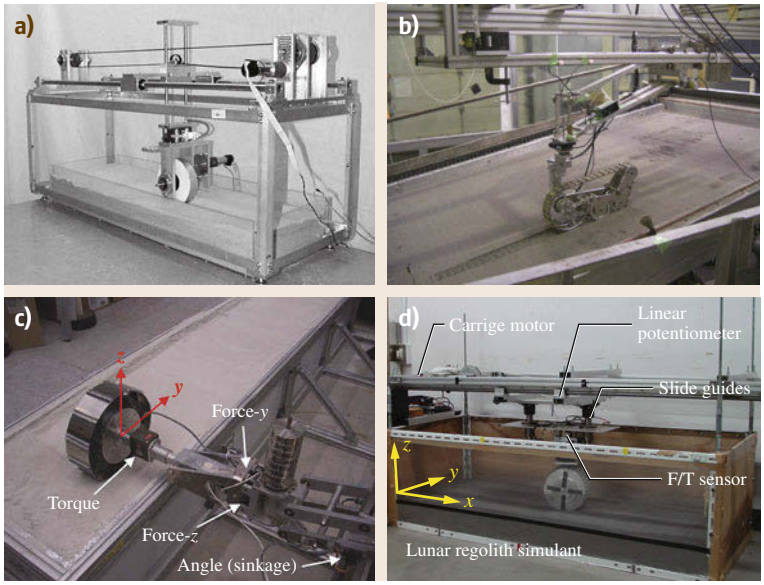


Fig.50.9a–d Single wheel test beds for experimental validation of terramechanics models. **(a)** Single wheel test bed at MIT (after [50.45]), **(b)** Single track test bed at JAXA (after [50.46]), **(c)** Single wheel test bed at DLR (after [50.47]), **(d)** Single wheel test bed at Tohoku University. (after [50.48])

cuss a wheel traction characteristics on rough terrain, soil parameter identification, and model uncertainty.

Wheel Slippage

Wheel slippage is generally observed when a robot travels on loose soil. Also, the wheel will slip in the lateral direction while the robot steering or traversing on sloped terrain. The slip ratio (i. e., slip in the longitudinal direction of wheel travel) is defined as a function of the longitudinal traveling velocity of the wheel v_x and the circumferential velocity of the wheel $r\omega$, where r is the wheel radius and ω represents the angular velocity of the wheel [50.37]

$$s = \begin{cases} (r\omega - v_x)/r\omega & (|r\omega| \geq |v_x| : \text{driving}) , \\ (r\omega - v_x)/v_x & (|r\omega| < |v_x| : \text{braking}) . \end{cases} \tag{50.2}$$

The slip ratio assumes a value in the range from -1 to 1.

The slip in the lateral direction of the wheel is expressed by the slip angle β, which is defined as a function of v_x and the lateral traveling velocity v_y, as

$$\beta = \begin{cases} \tan^{-1}(v_y/v_x) & (v_x \neq 0) , \\ \text{sgn}(v_y) \cdot \pi/2 & (v_x = 0) . \end{cases} \tag{50.3}$$

Wheel Traction Forces

The wheel–terrain contact forces, including the drawbar pull F_x, side force F_y, vertical force F_z, and resistance torque T, are calculated as the integral of the normal and shear stresses from the entry angle θ_f and the exit angle θ_r [50.38, 48]

$$F_x = rb \int_{\theta_r}^{\theta_f} \{\tau_x(\theta)\cos\theta - \sigma(\theta)\sin\theta\}\, d\theta , \tag{50.4}$$

$$F_y = \int_{\theta_r}^{\theta_f} \left[rb\tau_y(\theta) + R_b\{r - z(\theta)\cos\theta\}\right] d\theta , \tag{50.5}$$

$$F_z = rb \int_{\theta_r}^{\theta_f} \{\tau_x(\theta)\sin\theta + \sigma(\theta)\cos\theta\}\, d\theta , \tag{50.6}$$

$$T_x = r^2 b \int_{\theta_r}^{\theta_f} \tau_x(\theta)\, d\theta , \tag{50.7}$$

where $\sigma(\theta)$ is the normal stress, $\tau_x(\theta)$ and $\tau_y(\theta)$ are the shear stresses in the longitudinal or lateral direction. $z(\theta)$ is the wheel sinkage at an angle of θ, b is the wheel width, and R_b is the bulldozing resistance generated on a side wall of the wheel.

Experimental Validation

An experimental validation of the afore mentioned wheel–terrain interaction model is performed by the use of a single wheel test bed (Fig. 50.9), with varied state parameters such as soil or wheel slip conditions. A carriage velocity controlled relative to a wheel velocity realizes a wheel slip (or traction load) while measuring wheel traction forces, wheel

sinkage, and others. Experimental data obtained from the test bed are then compared with the values obtained from numerical simulation of the wheel traction model.

50.2.3 Wheel Traction Performance

The wheel–terrain model can be exploited for an evaluation of the mobility performance of a wheeled robot (i. e., traversability on sloped or deformable terrain [50.49, 50]). Such mobility evaluation technique is further valuable for the mobility system design in order to find a feasible wheel/track design that maximizes the traction performance for off-road locomotion under some specific constraints [50.46, 47, 51].

Ishigami et al. proposed a wheel traction diagram called a thrust–cornering characteristic diagram that determines the slope traversal criteria with respect to wheel slip conditions [50.17]). Figure 50.10 illustrates the relationship between the thrust and cornering forces of wheel on rough terrain, with varied slip ratios/slip angles. The thrust and cornering forces F_c and F_T are obtained by calculating the drawbar pull and side force

$$F_c = F_x \sin\beta + F_y \cos\beta \,, \tag{50.8}$$

$$F_T = F_x \cos\beta - F_y \sin\beta \,. \tag{50.9}$$

Considering the slope traversal scenario as shown in Fig. 50.11, the thrust–cornering characteristic diagram can draw the criteria with a minimum cornering force and thrust force required for a slope traversal: Case 1 as a traversal and uphill traveling, Case 2 as a straight-line traversal, and Case 3 as a traversal and downhill traveling. In addition, a robot decelerates or cannot move forward due to its negative thrust force. This diagram, for example, indicates that the slip ratio must exceed 0.6 in order to generate positive thrust force, and also, the cornering force varies primarily with the slip angle. Therefore, the diagram can determine the slope traversability on the basis of the corresponding slippage, indicated by several characteristic curves on the diagram.

50.2.4 Model Uncertainty and Soil Parameter Identification

It should be noted that the classical terramechanics model has primarily developed for an application for large-heavy vehicles (hundreds/thousands kg weight). Therefore, several assumptions that may cause an inaccurate calculation of wheel traction performance of the classical model would be omitted while exploiting the classical model to an analysis of small lightweight robot, resulting in an inaccurate calculation of wheel traction performance. The error due to the inadequate assumptions for the small-lightweight robot can be addressed by the following approaches: model improvement and uncertainty analysis of model parameters. It should be noted that the Bekker's pressure-sinkage model is assumed that the contact patch of wheel on deformable soil (circumferential section) is discretized

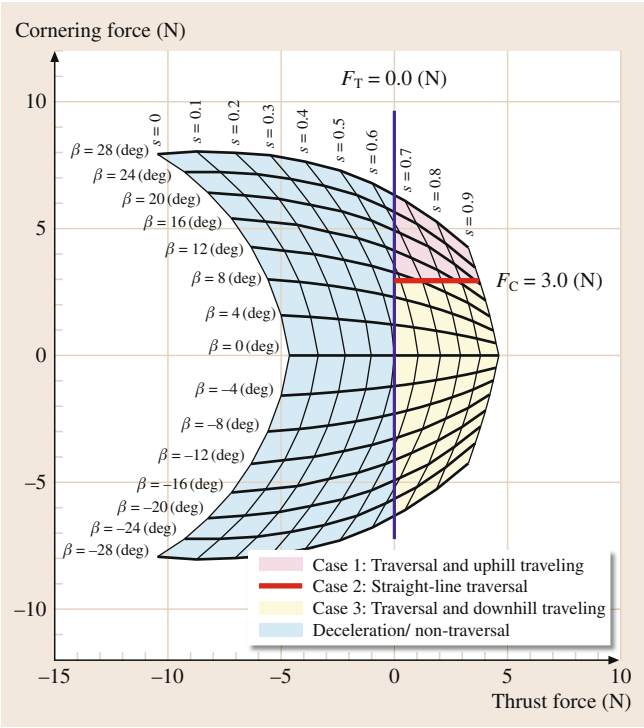


Fig. 50.10 Thrust–cornering characteristic diagram

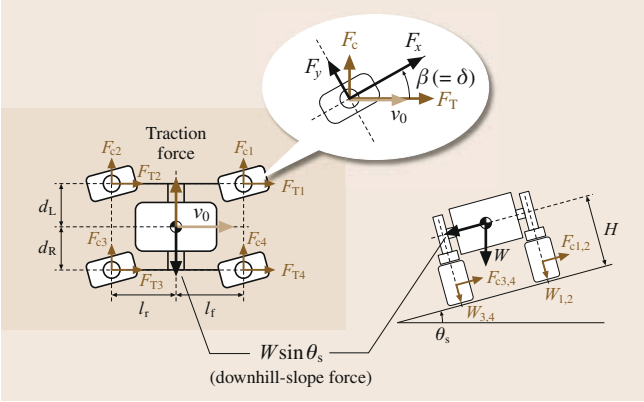


Fig. 50.11 Force balance on slope traversal

as a series of consecutive flat plate; however, *Bekker* noted [50.36]:

> *Predictions for wheels smaller than 20 inches in diameter become less accurate as wheel diameter decreases, because the sharp curvature of the loading area was neither considered in its entirety nor is it reflected in bevameter tests.*

Also another research reported that the assumptions provide inaccurate prediction for vehicles with wheels diameter less than about 50 cm (approx.) and normal loading less than 45 N (approx.) [50.36].

Some researchers have tried to update/improve the classical terramechanics model that can be successfully applied to relatively light weight vehicles: For example, *Nagatani* et al. developed a direct measurement device for the normal stress distribution [50.52]. A wheel diameter dependent pressure-sinkage model has been proposed by [50.53]. *Senatore* and *Iagnemma* proposed an improved approach for the calculation of shear deformation modulus [50.54].

Model parameters in the wheel–terrain interaction related to the soil properties are subject to uncertainties since their values would stochastically vary with location. Several researches have addressed the soil parameter identification: an on-line terrain parameter estimator with simplifying classical terramechanics equations was proposed by [50.55]; a modified nonlinear wheel–terrain interaction model for an identification of pressure-sinkage coefficient, internal friction angle, and shear deformation modulus [50.56]. Some recent works have also attempted to predict rover mobility even in uncertain conditions: a learning-based approach for slip-prediction and traversability analysis of a wheeled robot [50.57]; and a statistical method for mobility prediction of a robot with taking into account of terrain uncertainty [50.58].

50.3 Control of Wheeled Robot in Rough Terrain

This section introduces a control scheme for the wheeled robot that compensates wheel-and-vehicle slippages while following a given path is described.

A wheeled robot often experiences wheel slippages as well as vehicle sideslip while traversing in rough terrain, which degrades a performance of a motion control of the robot: i. e., a robot will deviate from a path to be followed, or a wheel might get stuck into loose soil. Therefore, a control scheme that can derive appropriate steering/driving maneuvers is necessary for a wheeled robot to achieve a rough terrain traverse and to manage any slippage problems.

A significant number of studies related to the path following control of mobile robot have been published. A general note for path following issues can be seen in [50.10–12]. *Rezaei* et al. investigated an online path following strategy combined with a simultaneous localization and mapping (SLAM) algorithm for a car-like robot in outdoor environments [50.13]. *Coelho* and *Nunes* et al. proposed a Kalman-based active observer controller for the path following of wheeled mobile robots [50.14].

As noted before, a key issue of the path following on rough terrain is to compensate wheel-and-vehicle slippage. This subsection briefly introduces a slip-compensated path follower along with a derivation for steering and driving maneuvers for path following [50.15–17] which is shown in VIDEO 188.

50.3.1 Slip-Compensated Path Follower

A two-dimensional path-following problem of a mobile robot is shown in Fig. 50.12. Here, a path following control generally performs to let the vehicle velocity v_0 coincide with the tangent of a given path such that the distance and orientation errors decrease to zero. However, once the vehicle experiences a sideslip with a certain amount of slip angle β_0, the vehicle has an additional orientation error of β_0 even

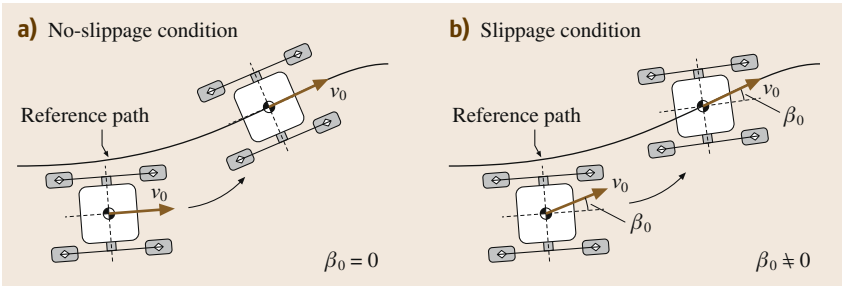


Fig.50.12a,b Path following problem without **(a)** and with **(b)** slip angle of vehicle

when the vector of vehicle velocity completely follows the path (Fig. 50.12b).

The path follower basically derives a commanded robot velocity vector $(\dot{x}_{\mathrm{cmd}}, \dot{y}_{\mathrm{cmd}}, \dot{\theta}_{\mathrm{cmd}})$ while the robot position (x_0, y_0, θ_0) and a reference path vector $(x_{\mathrm{p}}, y_{\mathrm{p}})$ are given. The slip-compensated path follower described here consists of two algorithms to achieve a path following task through high-slip environments.

First, the carrot heading algorithm determines the heading error of the robot θ_{e}, which is an angle calculated from the intersection of a circle centered on the robot coordinate with the desired path (Fig. 50.13):

$$\theta_{\mathrm{e}} = \theta_{\mathrm{d}} - \theta_0 \tag{50.10}$$

Here, θ_{d} is the desired heading angle which is a tangent angle on the intersection point and θ_0 is the robot heading angle. A large radius of the circle will generate a smooth motion of the robot while neglect a small feature of the path. A small radius will reduce the total path following error, but requires frequent heading changes for small path errors.

Once the heading error is determined, the slip-compensated path following algorithm executes the heading controller that calculates the commanded yaw rate of the robot $\dot{\theta}_{\mathrm{cmd}}$ from the heading error θ_{e} and the yaw slip rate $\dot{\beta}_0$

$$\dot{\theta}_{\mathrm{cmd}} = \frac{\left(K_1 \cdot \theta_{\mathrm{e}} + K_2 \cdot \dot{\beta}_0\right)}{T_{\mathrm{s}}}\,, \tag{50.11}$$

where K_1 and K_2 are the tuned controller gains, and T_{s} is the sampling period.

Note that the command for the linear velocity $\dot{x}_{\mathrm{cmd}}$ and $\dot{y}_{\mathrm{cmd}}$ are closely related to the maximum speed of the driving motors and the desired heading angle $\dot{\theta}_{\mathrm{cmd}}$. Practically, these commands are accomplished by actuators of the robotic mobility system, namely steering and driving mechanisms. The following subsection describes the derivation of those steering and driving maneuvers that satisfies the heading controller command $\dot{\theta}_{\mathrm{cmd}}$ as defined in (50.11).

50.3.2 Steering and Driving Maneuvers

Here, a nonholonomic kinematic model of a four-wheeled mobile robot with each wheel being independently steerable is considered. Figure 50.14 illustrates a two-dimensional kinematic model of the mobile robot having the slip angle of the vehicle β_0 and lateral wheel slippage β_i (the subscript i denotes the wheel identification number, $i = 1, \ldots, 4$ in this case). The dimension of the rover is defined by l_{f}, l_{r}, d_{R}, and d_{L}. The position and orientation of the vehicle are defined as $(x_0,\ y_0,\ \theta_0)$, and the position of each wheel is defined as $(x_i,\ y_i)$. In this model, the following assumptions are considered: the distance between the wheels is constant, and the steering axle of each wheel is perpendicular to the terrain surface.

The nonholonomic constraints of the mobile robot are defined by the following equations, with taking the lateral slips of wheels and that of vehicle into account

$$\begin{aligned} \dot{x}_0 \sin\phi_0 - \dot{y}_0 \cos\phi_0 &= 0\,, \\ \dot{x}_i \sin\phi_i - \dot{y}_i \cos\phi_i &= 0\,, \end{aligned} \tag{50.12}$$

where $\phi_0 = \theta_0 + \beta_0$ and $\phi_i = \theta_0 + \delta_i + \beta_i$. The geometric constraints between each wheel and the centroid of

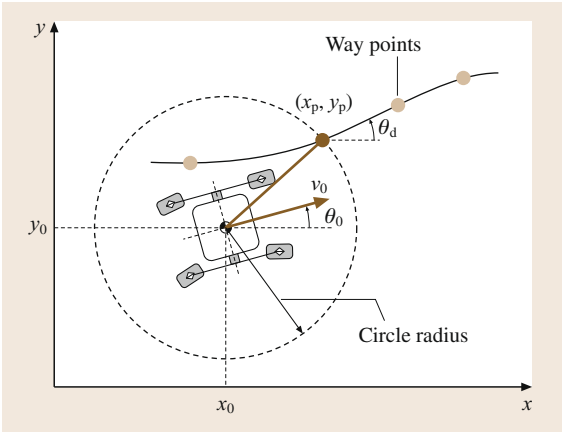


Fig. 50.13 Path following with carrot heading calculation

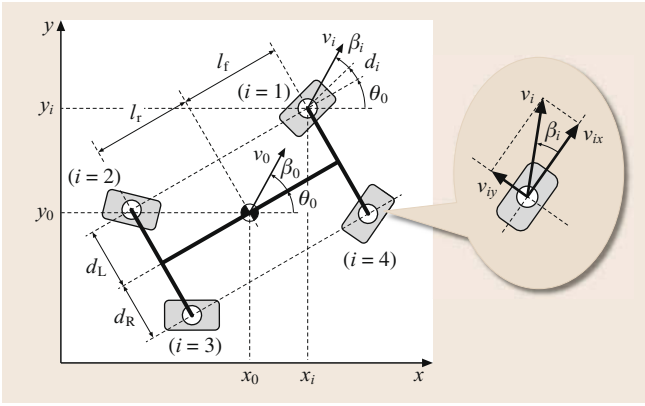


Fig. 50.14 Kinematic model of four-wheeled rover with wheel/vehicle slips

the vehicle are written as

$$\left.\begin{aligned} x_1 &= x_0 + l_\mathrm{f}\cos\theta_0 - d_\mathrm{L}\sin\theta_0 \\ x_2 &= x_0 - l_\mathrm{r}\cos\theta_0 - d_\mathrm{L}\sin\theta_0 \\ x_3 &= x_0 - l_\mathrm{r}\cos\theta_0 + d_\mathrm{R}\sin\theta_0 \\ x_4 &= x_0 + l_\mathrm{f}\cos\theta_0 + d_\mathrm{R}\sin\theta_0 \end{aligned}\right\} \rightarrow x_i = x_0 + X_i \,, \tag{50.13}$$

$$\left.\begin{aligned} y_1 &= y_0 + l_\mathrm{f}\sin\theta_0 + d_\mathrm{L}\cos\theta_0 \\ y_2 &= y_0 - l_\mathrm{r}\sin\theta_0 + d_\mathrm{L}\cos\theta_0 \\ y_3 &= y_0 - l_\mathrm{r}\sin\theta_0 - d_\mathrm{R}\cos\theta_0 \\ y_4 &= y_0 + l_\mathrm{f}\sin\theta_0 - d_\mathrm{R}\cos\theta_0 \end{aligned}\right\} \rightarrow y_i = y_0 + Y_i \,. \tag{50.14}$$

Given the desired heading angle $\theta_0 = \theta_\mathrm{d}$ and desired linear velocity v_d, the desired steering angle $\delta_{\mathrm{d}i}$ that satisfies the path following control as well as the slip compensation is obtained as follows: first, transform (50.12)

$$\delta_{\mathrm{d}i} = \tan^{-1}(\dot{y}_i/\dot{x}_i) - \theta_\mathrm{d} - \beta_i \,. \tag{50.15}$$

Subsequently, substituting (50.13) and (50.14) into (50.15), the desired steering angle is determined as follows

$$\delta_{\mathrm{d}i} = \tan^{-1}\left(\frac{v_\mathrm{d}\sin\theta_\mathrm{d} - \dot{Y}_i(\dot{\theta}_\mathrm{d})}{v_\mathrm{d}\cos\theta_\mathrm{d} - \dot{X}_i(\dot{\theta}_\mathrm{cmd})}\right) - \theta_\mathrm{d} - \beta_i \,. \tag{50.16}$$

The desired yaw rate $\dot{\theta}_\mathrm{cmd}$ is given from (50.11).

The driving maneuver is practically executed by the wheel angular velocity of the robot. From Fig. 50.14, the wheel linear velocity v_i is given as

$$v_i = r\omega_i/\cos\beta_i \,, \tag{50.17}$$

v_i is also expressed by $\dot{x}_i$ or $\dot{y}_i$

$$v_i = \dot{x}_i/\cos\phi_i = \dot{y}_i/\sin\phi_i \,. \tag{50.18}$$

Substituting (50.13) or (50.14) into the above equations, the desired wheel angular velocity $\omega_{\mathrm{d}i}$ can be obtained

$$\omega_{\mathrm{d}i} = \begin{cases} \left(v_\mathrm{d}\cos\theta_\mathrm{d} + \dot{X}_i(\dot{\theta}_\mathrm{cmd})\right)\cos\beta_i/r\cos\phi_i \\ \qquad (\theta_\mathrm{d} \le \pi/4) \,, \\ \left(v_\mathrm{d}\sin\theta_\mathrm{d} + \dot{Y}_i(\dot{\theta}_\mathrm{cmd})\right)\cos\beta_i/r\sin\phi_i \\ \qquad (\theta_\mathrm{d} \ge \pi/4) \,. \end{cases} \tag{50.19}$$

Position and orientation errors relative to a desired path should be accurately determined for deriving the control input as well as calculating the corresponding steering and driving maneuvers. A visual odometry method is of promising technique for such accurate measurement of position, orientation, velocity, and slippage, and also it is sufficiently robust for loose sandy terrain.

The visual odometry estimates the traveling velocity of the vehicle based on the optical flow vectors obtained from the time-consecutive images taken by an onboard camera(s). Integrating the velocity estimates with inertial measurement unit (IMU) readouts or stereo images for pose estimation provides an accurate estimation of the six degrees of freedom of the rover motion.

50.4 Modeling of Tracked Vehicle on Rough Terrain

Tracks are one of typical mechanism that can enhance the mobility of a robot on rough terrain because they can have contacts between the terrain more widely than wheels. Thus, many types of robots which are required to maneuver rough terrain such as military and search-and-rescue robots have been equipped with tracks.

Some tracked vehicles often have multiple tracks connected by joints to maneuver natural/artificial steps, bumps and stairs. Some are equipped with passive joints, but many are equipped with active joints. Multitracks configuration enables the vehicle to get over a step which is higher than the radius of the tracks.

It should be noted that many tracked vehicles are intended to maneuver the unknown rough terrain consisted of rubbles, rocks, concrete, and woods. Because such types of terrain are nonuniform unlike fine soil discussed in the Sect. 50.2, to dynamicaly model them is obviously difficult. In other words, we can easily sense the shape of the terrain, but dynamic parameters as well as center of gravity (COG), weight or stiffness of terrain cluster.

Thus, to geometrically consider the interaction between tracks and terrain is a reasonable approach. In this section, a geometrical model of a contact between a track and the terrain surface is introduced. The analysis includes (1) a parameterization of a generic shaped track, (2) contacts between the track and a single point in the terrain surface, and (3) contacts between the track and the terrain surface represented by a point cloud.

50.4.1 Parameterization of a Track

The geometry of a generic-shaped track is the key issue that should be discussed for modeling a tracked vehicle on rough terrain. In this section, we introduce a parameterization of the shape of a generic track.

As shown in Fig. 50.15, a generic track can be parameterized by the radius of the first and the second pulleys, R and r respectively, and the distance between the centers of the pulleys L.

In addition, let us embed the coordinate frame; its origin is at the center of the adjacent track and the x axis is along the center line of the adjacent track. The relative pose of the track respect to the adjacent track can be parameterized by the distance between the origin and the center of the first pulley, and the angle of the joint that connects the tracks θ.

50.4.2 Contact with a Single Point in the Terrain Surface

To discuss contacts between a track and a single point in the terrain surface is a good entrance to understand it between a track and the whole terrain surface. In the following sections, we geometrically describe the former contacts with the parameters of the track defined in the previous section.

As shown in Fig. 50.16, a track generally has four sections that can contact with the terrain surface; since it is tracked, it comprises a round and straight sections. Moreover, it can be in two states, namely, folded and spread states. When the track is in the spaded state, side-S supports the robot; on the other hand, when it is in the folded state, side-F is in contact with the terrain. We call the case where side-S is in contact with the ground as the *spreading mode* and the case where side-F is in contact with the ground as the *folding mode*.

In addition, we have assumed the pose of the track to be 0[°] when its straight section on side-S is in contact with the flat surface; we have also assumed the direction in which the track lifts its second pulley from 0[°] to be positive.

The geometrical relationship between the contact point and pose of the track is summarized in Table 50.1. For example, in the case of the spreading mode and the contact on the straight section (Fig. 50.15), we can derive the following equation according to the geometry of the track contacting with the surface

$$\begin{aligned}\theta_{\text{contact}} &= \theta_1 + \theta_2 \\ &= \tan^{-1}\frac{z}{x - x_{\text{support}}} \\ &\quad + \sin^{-1}\frac{r}{\sqrt{(x - x_{\text{support}})^2 + z^2}}\,. \end{aligned} \tag{50.20}$$

50.4.3 Contact with the Terrain Surface Represented by a Point Cloud

Now we are well prepared to geometrically understand contacts between the track and the terrain surface. In this section, we describe those contacts by iterating a point cloud that represents the terrain surface.

Let the shape of the terrain surface be represented by a set of points $\{u_1, u_2, \ldots, u_n\}$. Then, the contact angle of the track with the surface is determined using the following equation

$$\theta_{ref} = \begin{cases} \min(\theta_{\text{contact},1}, \ldots, \theta_{\text{contact},n}): \\ \quad \text{in spreading mode}\,, \\ \max(\theta_{\text{contact},1}, \ldots, \theta_{\text{contact},n}): \\ \quad \text{in folding mode}\,. \end{cases} \tag{50.21}$$

To calculate a contact angle of the track with a single point in the terrain surface, we need to determine which section and which side of the track makes the contact.

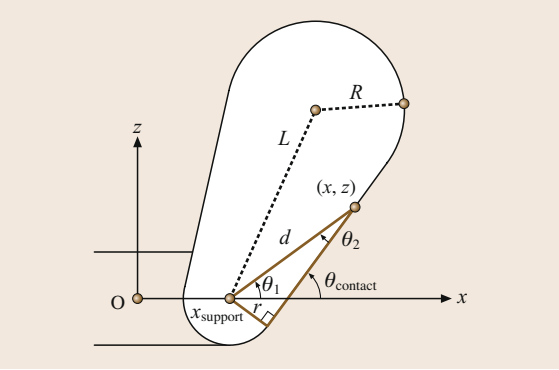


Fig. 50.15 Example of geometrical analysis contact between track and single point on the terrain surface

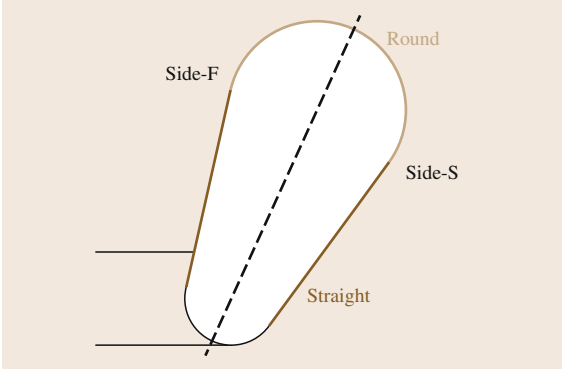


Fig. 50.16 Generic-shaped track consisted of staraight and round sections

Table 50.1 Geometric-based contact angle of subtrack

Section		θ_{contact}
Spreading	Straight	$\tan^{-1}\dfrac{z}{x-x_{\text{support}}}+\sin^{-1}\dfrac{r}{\sqrt{(x-x_{\text{support}})^2+z^2}}$
	Round	$\cos^{-1}\dfrac{d^2+L^2-R^2}{2Ld}+\tan^{-1}\dfrac{z}{x-x_{\text{support}}}-\sin^{-1}\dfrac{R-r}{L}$
Folding	Straight	$\tan^{-1}\dfrac{z}{x-x_{\text{support}}}-\sin^{-1}\dfrac{r}{\sqrt{(x-x_{\text{support}})^2+z^2}}-2\sin^{-1}\dfrac{R-r}{L}$
	Round	$\tan^{-1}\dfrac{z}{x-x_{\text{support}}}-\sin^{-1}\dfrac{R-r}{L}-\cos^{-1}\dfrac{d^2+L^2-R^2}{2Ld}$

The type of the section can be determined by checking the distance between the center of the first pulley and the point; if $d < \sqrt{L^2+R^2}$ the straight section makes the contact, else the round section does. On the other hand, the side will be determined by the choice of the application.

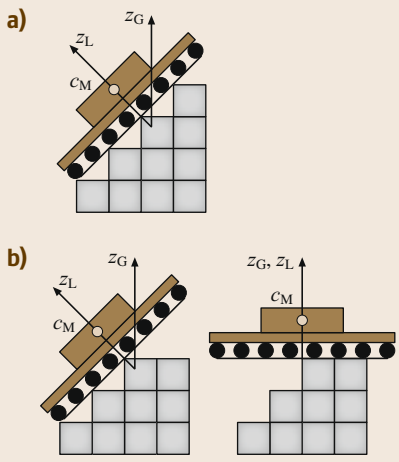
Now we understand a geometric model which describes contacts between the tracked vehicle consisted of multiple tracks and the free-formed terrain surface whose shape is represented as a point cloud. In the following subsection, we introduce an application example utilizing the contact model.

50.5 Stability Analysis of Tracked Vehicles

As shown in the previous sections, it is hardly possible to build an indention model of a rough terrain consisted of rubbles. Thus, in this research domain, stability criteria based only on geometrical parameters are often used to evaluate pose of the tracked vehicle on the rough terrain.

This section introduces two typical stability criteria which have been adopted for tracked vehicles; (1) support polygon and (2) normalized energy stability margin.

The support polygon is a simple and reasonable criterion to determine whether an object on a surface is statically stable. The support polygon for the tracked vehicle is equal to the horizontal projection of the convex hull of all the contact points between the tracks and the terrain. With this support polygon, we can consider



Label	Balance quality	Technical details
Red	Turn upside down or get stuck	Pitch > π/4 or roll > π/6 or at least one of conditions A is not satisfied
Orange	Lose balance on purpose	Two optional postures exist
Magenta	Climb up or slide down	Oscillations in posture estimation algorithm
Cyan	Jump down	Jump of CM between two stable postures < 50 mm
Yellow	Fair	NESM parameter < 1
Green	Good	NESM parameter ≥ 1

Fig.50.17a,b Example of stability analysis of tracked vehicle on a rough terrain based on support polygon and NESM (after [50.20]); **(a)** Stable posture corresponding to green label in table. **(b)** Unstable posture corresponding to orange label in table

that the vehicle is statically stable when it contains the projection of the vehicle's center of gravity.

Another stability criterion is the normalized energy stability margin, or NESM, proposed by Hirose. The NESM is a criterion that is used to evaluate the stability of a robot on the basis of the vertical distance between the initial position of its center of gravity and its highest position during tumbling. Although it is mainly used for walking robots, its evaluation only requires the positions of contact points with the ground and the center of gravity of the robot. In other words, there is no conceptual difference if this criterion is applied to the stability of tracked vehicles.

One typical application of stability criteria for the tracked vehicle is to rank possible path in a navigation. For example, Magid proposed a rating method for possible paths based on stability analysis utilizing the support polygon and NESM. This method evaluates the stability of each adjacent grid of the current position of the vehicle; paths to grids on which the support polygon condition is violated are denied. Then survived paths are ranked utilizing the NESM (Fig. 50.17).

50.6 Control of Tracked Vehicle on Rough Terrain

In this section, an approach for the autonomous control of active joints in a tracked vehicle with multiple tracks is introduced. As shown in the previous subsections, the geometric model of the tracked vehicle gives the pose of each track, which makes contact with the rough terrain on the basis of the shape of the terrain surface near the vehicle. The autonomous control utilizes the geometric model to determine the desired angles of the active joints between the tracks to perform a smooth maneuver on an unknown rough terrain.

Tracked vehicles with multiple tracks and joints typically have better locomotion than single-tracked vehicles. This is because one can change the pose of a track by actuating an adequate joint to negotiate bumps or steps in a rough terrain. Some multitracked vehicles a are equipped with a single-degree-of-freedom (1-DOF) tracked arms, often called subtracks, and the others are snake shaped.

However, multitracked vehicles are difficult to control manually because of their multiple degrees of freedom. For military and search-and-rescue robots in particular, which are likely to be teleoperated on an unknown rough terrain with limited camera views, it is quite difficult to control all of their DOFs manually.

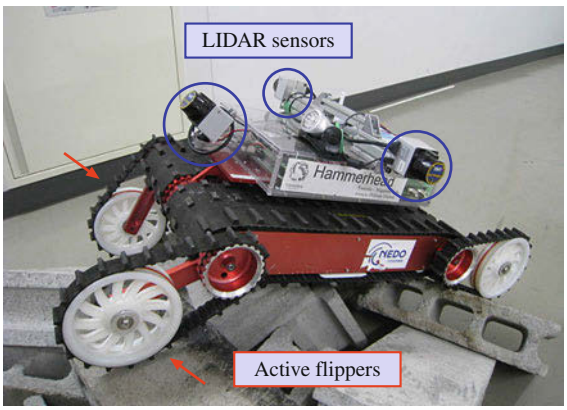


Fig. 50.18 Tracked vehicle Kenaf equipped with four subtracks enhansing mobility and three LIDAR sensors sensing shape of terrain

In the rest of this section, an approach for the autonomous control of active subtracks on the basis of the geometric model of a multitracked vehicle is introduced. In the control algorithm, the geometric model automatically determines the desired poses of active joints, which support subtracks on the basis of the terrain shape obtained in real time.

The multitracked vehicle Kenaf, which has autonomous subtrack control, is shown in Fig. 50.18. Kenaf has two main tracks covering its main body and four subtracks actuated by 1-DOF joints on each corner of the main body. It has 6-DOFs in total: 4-DOFs corresponding to joints supporting the subtracks, 1-DOF corresponding to the one main track and two subtrack belts on the right, and 1-DOF corresponding to the one main track and two subtrack belts on the left.

In addition, Kenaf is equipped with three LIDAR sensors, one on the front, one on the right side, and one on the left side of the main body, to obtain data on the shape of the terrain online. Scans from LIDARs are integrated to estimate the three-dimensional shape of the terrain near the vehicle. The aforementioned approach for autonomous control of active subtracks are described in [50.59].

Figure 50.19 shows the flowchart of the algorithm for the autonomous subtrack control, and its actual motions can be seen in VIDEO 190 and VIDEO 191. The control algorithm is divided into six steps and summarized as follows:

1. Slices of the shape of the terrain around the robot are first obtained from the LIDAR sensors attached

to the robot body and the three-dimensional terrain shape near the robot is estimated.
2. The desired posture (roll and pitch angles) of the body is then calculated on the basis of the estimated terrain shape.
3. The desired positions of the subtracks that realize the desired posture of the robot body are also determined.
4. Next, the stability of the desired posture and subtrack positions is evaluated.
5. If the desired pose (posture of the robot body and subtrack positions) are unstable, then the desired pose is redefined and steps (3)–(5) are repeated.
6. When the desired subtrack positions that realize a stable posture are generated, position control of the subtracks is finally performed.

In the following sections, we explain each step of the algorithm.

50.6.1 Ground Detection and Trimming of the Scanned Data

In this step, we obtain the scanned points U_{target} near the robot using the LIDAR sensors attached to the robot; ground detection is performed according to the approach described in the paper [50.59]. We first obtain two-dimensional terrain slices from three LIDAR sensors and integrate them according to each tagged and estimated position of the robot to generate the three-dimensional information of the terrain shape. At the end of this step, we filter distant points from the integrated terrain shape.

$$U_{\text{target}} = \{u_1, u_2, \ldots, u_m\} \tag{50.22}$$

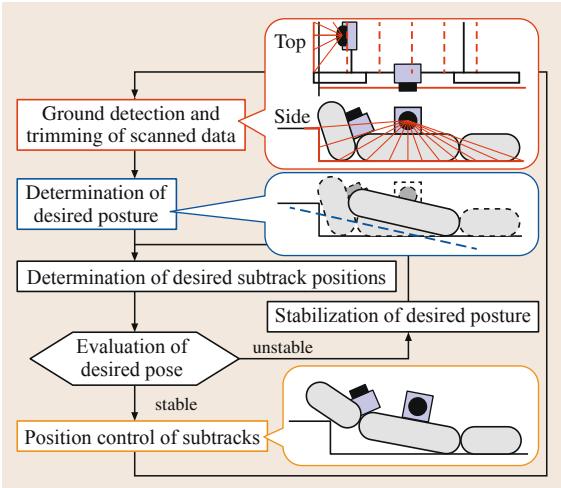


Fig. 50.19 Algorithm for autonomous control of subtracks

50.6.2 Determination of Desired Posture

We then determine the desired posture on the basis of the least-squares plane for the target terrain U_{target}. This is intended to enable smooth locomotion on rough terrain. Parameters a, b, and c of the least-squares plane $z = ax + by + c$ are determined using the following equations (an over-bar indicates the average of m scanned points in U_{target})

$$a = \frac{\alpha_{z,x}\alpha_{y,y} - \alpha_{x,y}\alpha_{y,z}}{\alpha_{x,x}\alpha_{y,y} - \alpha_{x,y}\alpha_{x,y}} , \tag{50.23}$$

$$b = \frac{\alpha_{y,z}\alpha_{x,x} - \alpha_{x,y}\alpha_{z,x}}{\alpha_{x,x}\alpha_{y,y} - \alpha_{x,y}\alpha_{x,y}} , \tag{50.24}$$

$$c = \overline{z_v} - \overline{x_v}a - \overline{y_v}b , \tag{50.25}$$

$$\alpha_{x,y} = \overline{x_u \cdot y_u} - \overline{x_u} \cdot \overline{y_u} , \tag{50.26}$$

$$\alpha_{y,z} = \overline{y_u \cdot z_u} - \overline{y_u} \cdot \overline{z_u} , \tag{50.27}$$

$$\alpha_{z,x} = \overline{z_u \cdot x_u} - \overline{z_u} \cdot \overline{x_u} , \tag{50.28}$$

$$\alpha_{x,x} = \overline{x_u \cdot x_u} - \overline{x_u} \cdot \overline{x_u} , \tag{50.29}$$

$$\alpha_{y,y} = \overline{y_u \cdot y_u} - \overline{y_u} \cdot \overline{y_u} . \tag{50.30}$$

Then, a translation to the coordinate system, if the robot body is parallel to the least-squares plane of the ground surface and makes contact with the ground, is described by the following equation in quaternion algebra:

$$\begin{pmatrix} 0 \\ x_{u_i'} \\ y_{u_i'} \\ z_{u_i'} \end{pmatrix} = q' \times \begin{pmatrix} 0 \\ x_{u_i} \\ y_{u_i} \\ z_{u_i} \end{pmatrix} \times q'^{-1} - \begin{pmatrix} 0 \\ 0 \\ 0 \\ \max(z_{u'}) \end{pmatrix} , \tag{50.31}$$

$$q' = \begin{pmatrix} \cos\dfrac{\theta_{\text{rot}}}{2} \\ b\sin\dfrac{\theta_{\text{rot}}}{2} \\ -a\sin\dfrac{\theta_{\text{rot}}}{2} \\ 0 \end{pmatrix} , \tag{50.32}$$

$$\theta_{\text{rot}} = \cos^{-1}\frac{1}{\sqrt{a^2 + b^2 + 1}} . \tag{50.33}$$

50.6.3 Determination of Desired Subtrack Positions

In this step, using the desired body posture and the integrated terrain slices, we determine the desired subtrack positions on the basis of the geometric model of contacts between the vehicle and the terrain surface.

Our application allows the operator to manually switch between the spreading and the folding control modes. Each control mode employs a corresponding geometric calculation method described in Sect. 50.4 In other words, the autonomous controller uses the corresponding calculation to determine the desired subtrack positions according to the control mode specified by the operator.

In both modes, we determine the desired subtrack positions to make contact with the ground through the desired robot posture. In particular, we calculate the angular position of each subtrack that makes contact with each point scanned by the left and the right LIDAR sensors. The desired position of each subtrack is determined for the maximum or minimum angular position of the subtrack in the spreading or folding mode, respectively.

50.6.4 Stability Evaluation of Desired Pose

In the proposed controller, we have adopted the normalized energy stability margin (NESM) [50.19] as the stability criterion for the desired pose. The stability of a desired pose is evaluated by NESM, and the pose is redefined if the stability is insufficient.

In the case of a tracked vehicle with four subtracks, four contact points (front-right, front-left, rear-right, and rear-left) can be determined by the step described in Sect. 50.6.3. In addition, the four axes of tumbling that pass through the front, rear, right, and left contact points can be assumed. Hence, the stability of a tracked vehicle with four subtracks can be determined from the minimum value of the NESM about these four axes.

50.6.5 Stabilization of Desired Pose

When the NESM of the robot is less than a predetermined threshold, we repeat the following steps until a desired, stable pose is realized. This step is intended to realize the strategy 4:

1a. If the NESM about the front or rear is adopted, reduce the pitch angle of the desired posture to zero.
1b. When the NESM about the right or left is adopted, reduce the roll angle of the desired posture to zero.
2. Redefine the desired subtrack positions by recalculating them to realize the redefined desired posture.
3. Evaluate the NESM about the redefined posture and positions of subtracks.

50.6.6 Position Control of Subtracks

Finally, we perform position control of the subtracks to realize the desired subtrack position produced through the above-mentioned steps on the basis of the strategy described in the previous subsection.

All subtracks of Kenaf and Quince are controlled by microprocessors present on the built-in motor drivers. Each desired subtrack position is sent to the corresponding microprocessor as a reference for position control using the conventional proportional–integral–derivative (PID) controller.

50.7 Summary

In this chapter, an overview of the modeling and control of robots on the rough terrain motion has been provided. The target environment was rough terrains, which includes a variety of ground conditions. In the case of weak and fine gravel, the motion of a wheeled mobile robot becomes relatively complicated because the interaction mechanics of the wheel in rough terrains differs greatly from that of an indoor robot on a flat surface. In this chapter, we introduced a terramechanics-based model for the interactions between wheels and deformable terrains. Furthermore, we explained the steering control of wheeled mobile robots to enable path-following in such environments. In the case of an environment with heaps of rubble, the approach for controlling robots is different from the terramechanics approach. Therefore, we introduced an initial attempt at a sensor-based approach that involved measuring the terrain shape.

Video-References

VIDEO 184 Mobility prediction of rovers on soft terrain
available from http://handbookofrobotics.org/view-chapter/50/videodetails/184

VIDEO 185 Experiments of wheeled rovers in a sandbox covered with loose soil
available from http://handbookofrobotics.org/view-chapter/50/videodetails/185

VIDEO 186 Terradynamics of legged locomotion for traversal in granular media
available from http://handbookofrobotics.org/view-chapter/50/videodetails/186
VIDEO 187 Interaction human-robot supervision, long range science rover for Mars exploration
available from http://handbookofrobotics.org/view-chapter/50/videodetails/187
VIDEO 188 A path-following control scheme for a four-wheeled mobile robot
available from http://handbookofrobotics.org/view-chapter/50/videodetails/188
VIDEO 189 Evaluation test of tracked vehicles on random step fields in the Disaster City
available from http://handbookofrobotics.org/view-chapter/50/videodetails/189
VIDEO 190 Autonomous sub-tracks control
available from http://handbookofrobotics.org/view-chapter/50/videodetails/190
VIDEO 191 Autonomous sub-tracks control
available from http://handbookofrobotics.org/view-chapter/50/videodetails/191

References

50.1 M. Jurkat, C. Nuttall, P. Haley: *The AMC' 74 Mobility Model, Tech. Rep. 11921* (US Army Tank Automotive Command, Warren, 1975)
50.2 R.B. Ahlvin, P.W. Haley: *NATO Reference Mobility Model Edition II, NRMM User's Guide, Tech. Rep. GL-92-19* (US Army WES, Vicksburg, 1992)
50.3 A. Gibbesch, B. Schäfer: Multibody system modelling and simulation of planetary rover mobility on soft terrain, 8th Int. Symp. Artif. Intell. Robotics Autom. Space (i-SAIRAS), Munich (2005)
50.4 R. Krenn, A. Gibbesch, G. Hirzinger: Contact dynamics simulation of rover locomotion, Proc. 9th Int. Symp. on Artif. Intell., Robotics Autom. Space, Los Angeles (2007)
50.5 D. Holz, A. Azimi, M. Teichmann, J. Kövecses: Mobility prediction of rovers on soft terrain: Effects of wheel- and tool-induced terrain deformations, Proc. 15th Int. Conf. Climbing Walk. Robots Support Technol. Mob. Mach. (CLAWAR) (2012)
50.6 J.Y. Wong: *Theory of Ground Vehicles* (Wiley, New York 1978)
50.7 M. Buehler, K. Iagnemma, S. Singh (Eds.): *The 2005 DARPA Grand Challenge: The Great Robot Race Springer Tracts Adv. Robotics Ser*, Vol. 36 (Springer, Berlin, Heidelberg 2005)
50.8 M. Buehler, K. Iagnemma, S. Singh (Eds.): *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*, Springer Tracts Adv. Robotics, Vol. 56 (Springer, Berlin, Heidelberg 2009)
50.9 C. Li, T. Zhang, D.I. Goldman: A terradynamics of legged locomotion on granular media, Science **339**, 1408–1412 (2013)
50.10 C. de Wit, H. Khennouf, C. Samson, O. Sordalen: Nonlinear control design for mobile robots. In: *Recent Trends in Mobile Robots*, World Scientific Series in Robotics and Automated System, Vol. 11, ed. by Y. Zheng (World Scientific, Singapore 1993)
50.11 A. Luca, G. Oriolo, C. Samson: Feedback control of nonholonomic car-like robots. In: *Robot Motion Planning and Control*, ed. by J. Laumond (Springer, Berlin, Heidelberg 1998) pp. 171–254
50.12 F. Rio, G. Jimenez, J. Sevillano, S. Vicente, A. Balcells: A generalization of path following for mobile robots, Proc. 1999 IEEE Int. Conf. Robotics Autom. (ICRA), Detroit (1999) pp. 7–12
50.13 S. Rezaei, J. Guivant, E. Nebot: Car-like robot path following in large unstructured environments, Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 2468–2473
50.14 P. Coelho, U. Nunes: Path-following control of mobile robots in presence of uncertainties, IEEE Trans. Robotics **21**(2), 252–261 (2005)
50.15 D. Helmick, Y. Cheng, D. Clouse, L. Matthies, S. Roumeliotis: Path following using visual odometry for a Mars rover in high-slip environments, Proc. 2004 IEEE Aerosp. Conf., Big Sky (2004) pp. 772–789
50.16 D. Helmick, S. Roumeliotis, Y. Cheng, D. Clouse, M. Bajracharya, L. Matthies: Slip-compensated path following for planetary exploration rovers, Adv. Robotics **20**(11), 1257–1280 (2006)
50.17 G. Ishigami, K. Nagatani, K. Yoshida: Slope traversal controls for planetary exploration rover on sandy terrain, J. Field Robotics **26**(3), 264–286 (2009)
50.18 D.A. Messuri, C.A. Klein: Automatic body regulation for maintaining stability of a legged vehicle during rough-terrain locomotion, IEEE J. Robotics Autom. **1**(3), 132–141 (1985)
50.19 S. Hirose, H. Tsukagoshi, K. Yoneda: Normalized energy stability margin and its contour of walking vehicles on rough terrain, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 181–186
50.20 E. Magid, T. Tsubouchi, E. Koyanagi, T. Yoshida, S. Tadokoro: Controlled balance losing in random step environment for path planning of a teleoperated crawler-type vehicle, J. Field Robotics **28**(6), 932–949 (2011)
50.21 A. Jacoff, E. Messina, B.A. Weiss, S. Tadokoro, Y. Nakagawa: Test arenas and performance metrics for urban search and rescue robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Las Vegas (2003) pp. 3396–3403
50.22 K. Ohno, V. Chun, T. Yuzawa, E. Takeuchi, S. Tadokoro, T. Yoshida, E. Koyanagi: Rollover avoidance using a stability margin for a tracked vehicle with sub-tracks, IEEE Int. Workshop Saf. Sec. Rescue Robotics (2009)
50.23 N. Vandapel, D. Huber, A. Kapuria, M. Hebert: Natural terrain classification using 3-D ladar data, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 5 (2004) pp. 5117–5122

50.24 M. Onosato, S. Yamamoto, M. Kawajiri, F. Tanaka: Digital gareki archives: An approach to know more about collapsed houses for supporting search and rescue activities, IEEE Int. Symp. Saf. Secur. Rescue Robotics (SSRR) (2012) pp. 1–6

50.25 A. Lacaze, K. Murphy, M. Del Giorno: Autonomous mobility for the demo III experimental unmanned vehicles, AUVS Int. Conf. Unnanned Veh. (2002)

50.26 K. Ohno, T. Suzuki, K. Higashi, M. Tsubota, E. Takeuchi, S. Tadokoro: Classification of 3-D point cloud data that includes line and frame objects on the basis of geometrical features and the pass rate of laser rays, Proc. 8th Int. Conf. Field Serv. Robotics (2012)

50.27 M. Onosato, T. Watasue: Two attempts at linking robots with disaster information: InfoBalloon and gareki engineering, Adv. Robotics **16**(6), 545–548 (2002)

50.28 M. Onosato: Digital GAREKI modeling for exploring knowledge of disaster-collapsed houses, IEEE Int. Workshop Saf. Secur. Rescue Robotics (SSRR) (2006)

50.29 L. Woosub, K. Sungchul, K. Munsang, P. Mignon: ROBHAZ-DT3: Teleoperated mobile platform with passively adaptive double-track for hazardous environment applications, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 33–38

50.30 B. Yamauchi: Packbot: A versatile platform for military robotics, Proc. SPIE **5422**, 228–237 (2004)

50.31 D. Inoue, K. Ohno, S. Nakamura, S. Tadokoro, E. Koyanagi: Whole-body touch sensors for tracked mobile robots using force-sensitive chain guides, IEEE Int. Workshop Saf. Secur. Rescue Robotics (SSRR) (2008) pp. 71–76

50.32 A. Jain, J. Balaram, J. Cameron, J. Guineau, C. Lim, M. Pornerantz, G. Sohl: Recent developments in the ROAMS planetary rover simulation environment, Proc. 2004 IEEE Aerosp. Conf., Big Sky (2004) pp. 861–876

50.33 K. Iagnemma, C. Senatore, B. Trease, R. Arvidson, A. Shaw, F. Zhou, L. Van Dyke, R. Lindemann: Terramechanics modeling of mars surface exploration rovers for simulation and parameter estimation, ASME Int. Des. Eng. Tech. Conf. (2011)

50.34 R. Bauer, W. Leung, T. Barfoot: Development of a dynamic simulation tool for the exomars rover, Proc. 8th Int. Symp. Artif. Intell., Robotics Autom. Space, Munich (2005)

50.35 M.G. Bekker: *Theory of Land Locomotion* (Univ. Michigan Press, Ann Arbor 1956)

50.36 M.G. Bekker: *Introduction to Terrain-Vehicle Systems* (Univ. Michigan Press, Ann Arbor 1969)

50.37 J.Y. Wong: *Theory of Ground Vehicles*, 4th edn. (Wiley, Hoboken 2008)

50.38 J.Y. Wong, A.R. Reece: Prediction of rigid wheel performance based on the analysis of soil-wheel stresses – Part I: Performance of driven rigid wheels, J. Terramechanics **4**(1), 81–98 (1967)

50.39 J.Y. Wong, A.R. Reece: Prediction of rigid wheel performance based on the analysis of soil-wheel stresses – Part II: Performance of towed rigid wheels, J. Terramechanics **4**(2), 7–25 (1967)

50.40 I.C. Schmid: Interaction of vehicle and terrain results from 10 years research at IKK, J. Terramechanics **32**(1), 3–25 (1995)

50.41 L. Ding, Z. Deng, H. Gao, K. Nagatani, K. Yoshida: Planetary rovers' wheel-soil interaction mechanics: New challenges and applications for wheeled mobile robots, Intell. Serv. Robotics **4**(1), 17–38 (2010)

50.42 H. Nakashima, H. Fujii, A. Oida, M. Momozu, Y. Kawase, H. Kanamori, S. Aoki, T. Yokoyama: Parametric analysis of lugged wheel performance for a lunar microrover by means of DEM, J. Terramechanics **44**, 153–162 (2007)

50.43 H. Nakashima, H. Fujii, A. Oida, M. Momozu, H. Kanamori, S. Aoki, T. Yokoyama, H. Shimizu, J. Miyasaka, K. Ohdoi: Discrete element method analysis of single wheel performance for a small lunar rover on sloped terrain, J. Terramechanics **47**, 307–321 (2010)

50.44 W. Li, Y. Huang, Y. Cui, S. Dong, J. Wang: Trafficability analysis of lunar mare terrain by means of the discrete element method for wheeled rover locomotion, J. Terramechanics **47**, 161–172 (2010)

50.45 K. Iagnemma: *A Laboratory single wheel testbed for studying planetary rover wheel-terrain interaction, Tech. Rep. 01-05-05* (MIT, Cambridge 2005)

50.46 S. Wakabayashi, H. Sato, S. Nishida: Design and mobility evaluation of tracked lunar vehicle, J. Terramechanics **46**(3), 105–114 (2009)

50.47 N. Patel, R. Slade, J. Clemmet: The ExoMars rover locomotion subsystem, J. Terramechanics **47**, 227–242 (2010)

50.48 G. Ishigami, A. Miwa, K. Nagatani, K. Yoshida: Terramechanics-based model for steering maneuver of planetary exploration rovers on loose soil, J. Field Robotics **24**(3), 233–250 (2007)

50.49 R. Lindemann, D. Bickler, B. Harrington, G. Ortiz, C. Voorhees: Mars exploration rover mobility development, IEEE Robotics Autom. Mag. **13**(2), 19–26 (2006)

50.50 G. Ishigami, A. Miwa, K. Nagatani, K. Yoshida: Terramechanics-based analysis on slope traversability for a planetary exploration rover, Proc. 25th Int. Symp. Space Technol. Sci. (2006) pp. 1025–1030

50.51 S. Michaud, L. Richter, T. Thueer, A. Gibbesch, T. Huelsing, N. Schmitz, S. Weiss, A. Krebs, N. Patel, L. Joudrier, R. Siegwart, B. Schäfer, A. Ellery: Rover chassis evaluation and design optimisation using the RCET, Proc. 9th ESA Workshop Adv. Space Technol. Robotics Autom. (ASTRA) (2006)

50.52 K. Nagatani, A. Ikeda, K. Sato, K. Yoshida: Accurate estimation of drawbar pull of wheeled mobile robots traversing sandy terrain using built-in force sensor array wheel, Proc. 2009 IEEE/RSJ Int. Conf. Robots Syst. (IROS), St. Loius (2009) pp. 2373–2378

50.53 G. Meirion-Griffith, M. Spenko: A Modified pressure-sinkage model for small, rigid wheels on deformable terrains, J. Terramechanics **48**(2), 149–155 (2011)

50.54 C. Senatore, K. Iagnemma: Direct shear behaviour of dry, granular soils for low normal stress with

application to lightweight robotic vehicle modeling, 17th Conf. Terrain-Veh. Syst. (ISTVS), Blacksburg (2011)
50.55 K. Iagnemma, S. Kang, H. Shibly, S. Dubowsky: Online terrain parameter estimation for wheeled mobile robots with application to planetary rovers, IEEE Trans. Robotics **20**(5), 921–927 (2004)
50.56 S. Hutangkabodee, Y. Zweiri, L. Seneviratne, K. Althoefer: Soil parameter identification for wheel-terrain interaction dynamics and traversability prediction, Int. J. Autom. Comput. **3**(3), 244–251 (2006)
50.57 D. Helmick, A. Angelova, L. Matthies, C. Brooks, I. Halatci, S. Dubowsky, K. Iagnemma: Experimental results from a terrain adaptive navigation system for planetary rovers, Proc. 9th Int. Symp. Artif. Intell., Robotics Autom. Space (i-SAIRAS), Hollywood (2008)
50.58 G. Ishigami, G. Kewlani, K. Iagnemma: A statistical approach to mobility prediction for planetary surface exploration rovers in uncertain terrain, IEEE Robotics Autom. Mag. **16**(4), 61–70 (2009)
50.59 O. Yoshito, K. Nagatani, K. Yoshida, S. Tadokoro, T. Yoshida, E. Koyanagi: Shared autonomy system for traversing and turning tracked vehicles on rough terrain based on continuous three-dimensional terrain scanning, J. Field Robotics **28**(6), 875–893 (2011)

Multimedia Contents

51. Modeling and Control of Underwater Robots

Gianluca Antonelli, Thor I. Fossen, Dana R. Yoerger

This chapter deals with modeling and control of underwater robots. First, a brief introduction showing the constantly expanding role of marine robotics in oceanic engineering is given; this section also contains some historical backgrounds. Most of the following sections strongly overlap with the corresponding chapters presented in this handbook; hence, to avoid useless repetitions, only those aspects peculiar to the underwater environment are discussed, assuming that the reader is already familiar with concepts such as fault detection systems when discussing the corresponding underwater implementation. The modeling section is presented by focusing on a coefficient-based approach capturing the most relevant underwater dynamic effects. Two sections dealing with the description of the sensor and the actuating systems are then given. Autonomous underwater vehicles require the implementation of mission control system as well as guidance and control algorithms. Underwater localization is also discussed. Underwater manipulation is then briefly approached. Fault detection and fault tolerance, together with the coordination control of multiple underwater vehicles, conclude the theoretical part of the chapter. Two final sections, reporting some successful applications and discussing future perspectives, conclude the chapter. The reader is referred to Chap. 25 for the design issues.

51.1 **The Expanding Role of Marine Robotics in Oceanic Engineering** 1285
51.1.1 Historical Background 1287
51.2 **Underwater Robotics** 1287
51.2.1 Modeling 1287
51.2.2 Sensor Systems 1293
51.2.3 Actuating Systems 1295
51.2.4 Communication Systems 1296
51.2.5 Mission Control System 1296
51.2.6 Guidance and Control 1297
51.2.7 Localization 1299
51.2.8 Underwater Manipulation 1299
51.2.9 Fault Detection/Tolerance 1300
51.2.10 Multiple Underwater Vehicles 1301
51.3 **Applications** 1302
51.4 **Conclusions and Further Reading** 1303
Video-References 1304
References 1304

The world's oceans cover 2/3 of the Earth's surface and have been critical to human welfare throughout history. As in ancient times, they enable the transport of goods between nations. Presently, the seas represent critical sources of food and other resources such as oil and gas. In the near term, we may soon see the emergence of offshore mining for metals as well as the exploitation of gas hydrates. Conversely, the ocean can also threaten human safety and damage infrastructure through natural phenomena such as hurricanes and tsunamis.

51.1 The Expanding Role of Marine Robotics in Oceanic Engineering

Our scientific understanding of the deep sea is expanding rapidly through the use of a variety of technologies. The first scientific explorations were conducted primarily through the use of diving and human-occupied

submersibles, complemented by a variety of other technologies such as towed or lowered instruments, trawls, dredges, autonomous seafloor instruments, and deep-sea drilling. More recently remotely operated and autonomous vehicles have begun to revolutionize seafloor exploration, often returning superior data at reduced costs. In the near future, seafloor observatories linked by fiber-optic cables and satellites will return massive amounts of data from coastal and deep-sea sites. These observations will complement those from conventional expeditionary investigations, and will require teleoperated or robotic intervention during installation and for service. An example of a remotely operated vehicle developed for the scientific study of the seafloor is the Jason 2 vehicle developed at the Woods Hole Oceanographic Institution (shown in Fig. 51.1), and a list of remotely operated vehicles for scientific exploration appears in Table 51.1 (the last vehicle in the table, Kaiko, was lost several years ago).

Offshore oil and gas installations are presently serviced almost exclusively by remotely operated vehicles (ROVs), physically connected via a tether to receive power and data, with human divers used only for the shallowest installations. Subsea systems require extensive work capability during installation, and need frequent inspection and intervention to support drilling operations, actuate valves, repair or replace subsea components, and to accomplish a variety of tasks required to maintain production rates and product quality. The trend toward robotic and teleoperated subsea intervention is certain to continue as offshore oil and gas production moves into deeper waters, and economic considerations push key production steps from surface platforms to the seafloor. Remotely operated manipulators enable these systems to perform complex tasks such as debris removal, cleaning using abrasive tools, and to operate a variety of nondestructive testing tools. The effectiveness of using ROVs decreases with depth mainly due to the cost increase and the difficulties of handling the long tether.

Fig. 51.1 The ROV Jason 2 (courtesy of Woods Hole Oceanographic Institution)

Table 51.1 ROVs for scientific use

Vehicle	Depth (m)	Institution	Manufacturer
Hyperdolphin	3000	JAMSTEC[a]	ISE
Dolphin 3K	3000	JAMSTEC	JAMSTEC
Quest	4000	MARUM[b]	Shilling
Tiburon	4000	MBARI[c]	MBARI
ROPOS	5000	CSSF[d]	ISE
Victor	6000	IFREMER[e]	IFREMER
Jason	6500	WHOI[f]	WHOI
ISIS	6500	NOC[g]	WHOI
UROV 7K	7000	JAMSTEC	JAMSTEC
Kaiko	11 000	JAMSTEC	JAMSTEC

[a] Japan Marine Science and Technology Center; [b] Zentrum für Marine Umweltwissenschaften; [c] Monterey Bay Aquarium Research Institute; [d] Canadian Scientific Submersile Facility; [e] Institut français de recherche pour l'exploitation de la mer; [f] Woods Hole Oceanographic Institution; [g] National Oceanography Centre

Autonomous underwater vehicles (AUVs) are free-swimming unoccupied underwater vehicles that can overcome the limitations imposed by ROV tethers for some tasks. Such vehicles carry their own energy supplies (presently batteries, perhaps fuel cells in the future) and communicate only through acoustics and perhaps optical links in the near future. Limited communications require these vehicles to operate independently of continuous human control, in many cases the vehicles operate completely autonomously. AUVs are currently used for scientific survey tasks, oceanographic sampling, underwater archeology and under-ice survey. Military applications, such as mine detection and landing site survey, are presently operational, and more ambitious applications such as long-term undersea surveillance are in engineering development. Presently, AUVs are incapable of sampling or manipulations tasks like those done routinely by ROVs, as typical work environments tend to be complex and challenging even to skilled human pilots.

Today, approximately 1000 AUVs are operational, many of them experimental. However, they are maturing rapidly. Recently several companies now offer commercial services with AUVs. As an example, for the oil and gas industry the cost reduction of a survey performed with an AUVs instead of a towed vehicle is up to 30% and the data quality is generally higher. Likewise, commercial manufacturers in several countries now offer turnkey AUV systems for specific, well-defined tasks. Currently, remotely operated manip-

ulators are standard equipment for most ROVs, while on the contrary autonomous manipulation is still a research challenge; the projects SAUVIM [51.1], ALIVE [51.2] and TRIDENT [51.3] were devoted to studying this control problem. VIDEO 88 shows a graphical rendering of a mission with the AUV Nereus, VIDEO 87 shows the perspective view from the Remus AUV, VIDEO 92 a 10-minutes long documentary of an underice mission and finally VIDEO 90 a strong interaction with a shark.

51.1.1 Historical Background

Boats have been used by humans since the start of recorded history, but vehicles able to go under water are more recent. Perhaps the first recorded idea of an underwater machine came from Aristotle; according to legend he built the: *skaphe andros* (boat-man) that allowed Alexander the Great (Alexander III of Macedonia, 356–323 BC) to stay submerged for at least half a day during the war of Tiro in 325 BC. This is probably unrealistic; if true it would precede Archimedes' law, which was first articulated in approximately 250 BC. Leonardo Da Vinci may have been the first to design an underwater vehicle. His efforts were recorded in the *Codice Atlantico* (*Codex Atlanticus*), written between 1480 and 1518. Legends say that Leonardo worked on the idea of an underwater military machine but he destroyed the results as he judged them to be too dangerous. The first use of feedback theory for marine control was probably the Northseeking device, patented in 1908, that used gyroscopic principals to develop the first autopilot [51.4]. From that point on, the use of feedback theory in marine control grew continuously; it is interesting to notice that the proportional–integral–derivative (PID) control commonly used today in numerous industrial applications was first formally analyzed in 1929 by Minorsky [51.5]. The first remotely operated underwater vehicle, POODLE, was built in 1953, and the ROV evolved through the 1960s and 1970s, mostly for military purposes. In the 1980s ROVs became established for use in the commercial offshore industry and began to emerge for scientific applications. The first tetherless, autonomous vehicles were built for experimental purposes in the 1970s. Currently, AUVs are becoming increasingly commonplace for scientific, military, and commercial applications. Turnkey AUV systems for a range of tasks are available from commercial vendors, and AUV services can be acquired from a number of companies [51.6].

51.2 Underwater Robotics

51.2.1 Modeling

A rigid body is completely described by its position and orientation with respect to a reference frame $\Sigma_i, O_i - xyz$ that is supposed to be Earth-fixed and inertial. Let us define $\boldsymbol{\eta}_1 \in \mathbb{R}^3$ as

$$\boldsymbol{\eta}_1 = (x \;\; y \;\; z)^{\mathrm{T}} ,$$

the vector of the body position coordinates in an Earth-fixed reference frame. The vector $\dot{\boldsymbol{\eta}}_1$ is the corresponding time derivative (expressed in the Earth-fixed frame). If one defines

$$\boldsymbol{\nu}_1 = (u \;\; v \;\; w)^{\mathrm{T}}$$

as the linear velocity of the origin of the body-fixed frame $\Sigma_\mathrm{b}, O_\mathrm{b} - x_\mathrm{b} \; y_\mathrm{b} \; z_\mathrm{b}$ with respect to the origin of the Earth-fixed frame expressed in the body-fixed frame (from now on: body-fixed linear velocity) the following relation between the defined linear velocities holds

$$\boldsymbol{\nu}_1 = \mathbf{R}_\mathrm{I}^\mathrm{B} \dot{\boldsymbol{\eta}}_1 , \tag{51.1}$$

where $\mathbf{R}_\mathrm{I}^\mathrm{B}$ is the rotation matrix expressing the transformation from the inertial frame to the body-fixed frame.

Let us define $\boldsymbol{\eta}_2 \in \mathrm{SO}(3)$ as

$$\boldsymbol{\eta}_2 = (\phi \;\; \theta \;\; \psi)^{\mathrm{T}}$$

the vector of body Euler angle coordinates in a Earth-fixed reference frame. In the nautical field those are commonly named roll, pitch, and yaw. Yaw is defined as rotation around the z axis of the fixed frame; pitch is defined as rotation around the y axis resulting after the yaw movement; and roll is defined as rotation around the x axis resulting after both yaw and pitch movements. The vector $\dot{\boldsymbol{\eta}}_2$ is the corresponding time derivative (expressed in the inertial frame). Let us define

$$\boldsymbol{\nu}_2 = (p \;\; q \;\; r)^{\mathrm{T}}$$

as the angular velocity of the body-fixed frame with respect to the Earth-fixed frame expressed in the body-fixed frame (from now on: body-fixed angular velocity). The vector $\dot{\boldsymbol{\eta}}_2$ does not have a physical interpretation and it is related to the body-fixed angular velocity by a proper Jacobian matrix

$$\boldsymbol{\nu}_2 = \mathbf{J}_{k,o}(\boldsymbol{\eta}_2)\dot{\boldsymbol{\eta}}_2 . \tag{51.2}$$

The matrix $\mathbf{J}_{k,o} \in \mathbb{R}^{3\times3}$ can be expressed in terms of the Euler angles as

$$\mathbf{J}_{k,o}(\boldsymbol{\eta}_2) = \begin{pmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{pmatrix}, \tag{51.3}$$

where c_α and s_α are abbreviations for $\cos(\alpha)$ and $\sin(\alpha)$, respectively. The matrix $\mathbf{J}_{k,o}(\boldsymbol{\eta}_2)$ is not invertible for every value of $\boldsymbol{\eta}_2$. In detail, it is

$$\mathbf{J}_{k,o}^{-1}(\boldsymbol{\eta}_2) = \frac{1}{c_\theta} \begin{pmatrix} 1 & s_\phi s_\theta & c_\phi s_\theta \\ 0 & c_\phi c_\theta & -c_\theta s_\phi \\ 0 & s_\phi & c_\phi \end{pmatrix}, \tag{51.4}$$

that it is singular for $\theta = (2l+1)\frac{\pi}{2}$ rad, with $l \in \mathbb{N}$, i. e., for a pitch angle of $\pm\frac{\pi}{2}$ rad.

The rotation matrix $\mathbf{R}_\mathrm{I}^\mathrm{B}$, needed in (51.1) to transform the linear velocities, is expressed in terms of the Euler angles by

$$\mathbf{R}_\mathrm{I}^\mathrm{B}(\boldsymbol{\eta}_2) = \begin{pmatrix} c_\psi c_\theta & s_\psi c_\theta & -s_\theta \\ -s_\psi c_\phi + c_\psi s_\theta s_\phi & c_\psi c_\phi + s_\psi s_\theta s_\phi & s_\phi c_\theta \\ s_\psi s_\phi + c_\psi s_\theta c_\phi & -c_\psi s_\phi + s_\psi s_\theta c_\phi & c_\phi c_\theta \end{pmatrix}. \tag{51.5}$$

Table 51.2 shows the common notation used for marine vehicles according to the Society of Naval Architects and Marine Engineers (SNAMEs) notation [51.7]; a sketch is shown in Fig. 51.2.

As for any representation of a rigid body's orientation several possibilities arise, among them, the use of a four-parameter description given by unit quaternions. The term *quaternion* was introduced by Hamilton in 1840, 70 years after the introduction of a four-parameter rigid-body attitude representation by Euler. An introduction to alternative orientation representations can be found in Chap. 2 and concerning the marine environment in [51.8].

Table 51.2 Common notation for the motion of a marine vehicle

		Forces and moments	$\boldsymbol{\nu}_1, \boldsymbol{\nu}_2$	$\boldsymbol{\eta}_1, \boldsymbol{\eta}_2$
Motion in the x-direction	Surge	X	u	x
Motion in the y-direction	Sway	Y	v	y
Motion in the z-direction	Heave	Z	w	z
Rotation about the x-axis	Roll	K	p	ϕ
Rotation about the y-axis	Pitch	M	q	θ
Rotation about the z-axis	Yaw	N	r	ψ

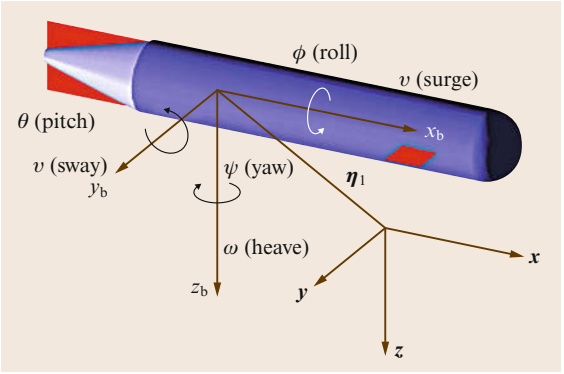


Fig. 51.2 Motion variables for an underwater vehicle

It is useful to collect the kinematic equations in six-dimensional matrix forms. Let us define the vector $\boldsymbol{\eta} \in \mathbb{R}^6$ as

$$\boldsymbol{\eta} = \begin{pmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \end{pmatrix}, \tag{51.6}$$

and the vector $\boldsymbol{\nu} \in \mathbb{R}^6$ as

$$\boldsymbol{\nu} = \begin{pmatrix} \boldsymbol{\nu}_1 \\ \boldsymbol{\nu}_2 \end{pmatrix}, \tag{51.7}$$

and, defining the matrix $\mathbf{J}_\mathrm{e}(\mathbf{R}_\mathrm{B}^\mathrm{I}) \in \mathbb{R}^{6\times6}$

$$\mathbf{J}_\mathrm{e}(\mathbf{R}_\mathrm{B}^\mathrm{I}) = \begin{pmatrix} \mathbf{R}_\mathrm{I}^\mathrm{B} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{J}_{k,o} \end{pmatrix}, \tag{51.8}$$

where the rotation matrix $\mathbf{R}_\mathrm{I}^\mathrm{B}$ is given in (51.5) and $\mathbf{J}_{k,o}$ is given in (51.3),

$$\boldsymbol{\nu} = \mathbf{J}_\mathrm{e}(\mathbf{R}_\mathrm{B}^\mathrm{I})\dot{\boldsymbol{\eta}}\,. \tag{51.9}$$

The inverse mapping, given the block-diagonal structure of $\mathbf{J}_\mathrm{e}$, is given by

$$\dot{\boldsymbol{\eta}} = \mathbf{J}_\mathrm{e}^{-1}(\mathbf{R}_\mathrm{B}^\mathrm{I})\boldsymbol{\nu} = \begin{pmatrix} \mathbf{R}_\mathrm{B}^\mathrm{I} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{J}_{k,o}^{-1} \end{pmatrix} \boldsymbol{\nu}\,, \tag{51.10}$$

where $\mathbf{J}_{k,o}^{-1}$ is given in (51.4).

Defining as

$$\boldsymbol{\tau}_\mathrm{v} = \begin{pmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \end{pmatrix},$$

the vector of generalized forces, where

$$\boldsymbol{\tau}_1 = (X \;\; Y \;\; Z)^\mathrm{T}\,, \tag{51.11}$$

the resultant forces acting on the rigid body expressed in a body-fixed frame, and

$$\boldsymbol{\tau}_2 = (K \;\; M \;\; N)^{\mathrm{T}} , \tag{51.12}$$

the corresponding resultant moment to the pole O_b, it is possible to rewrite the Newton–Euler equations of motion of a rigid body moving in the space as

$$\mathbf{M}_{\mathrm{RB}}\dot{\boldsymbol{\nu}} + \mathbf{C}_{\mathrm{RB}}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_\mathrm{v} . \tag{51.13}$$

The derivation of (51.13) can be found in Chap. 3.

The matrix $\mathbf{M}_{\mathrm{RB}}$ is constant, symmetric, and positive definite, i. e., $\dot{\mathbf{M}}_{\mathrm{RB}} = \mathbf{0}$ and $\mathbf{M}_{\mathrm{RB}} = \mathbf{M}_{\mathrm{RB}}^{\mathrm{T}} > \mathbf{0}$. Its unique parametrization is of the form

$$\mathbf{M}_{\mathrm{RB}} = \begin{pmatrix} m\mathbf{I}_3 & -m\mathbf{S}(\boldsymbol{r}_C^{\mathrm{b}}) \\ m\mathbf{S}(\boldsymbol{r}_C^{\mathrm{b}}) & \mathbf{I}_{O_\mathrm{b}} \end{pmatrix} , \tag{51.14}$$

where $\boldsymbol{r}_C^{\mathrm{b}}$ is the 3×1 distance vector to the center of gravity (CG) expressed in the body-fixed frame, $\mathbf{I}_3$ is the 3×3 identity matrix, and $\mathbf{I}_{O_\mathrm{b}}$ is the inertia tensor expressed in the body-fixed frame $\mathbf{S}(\boldsymbol{x})$ is the matrix operator performing the cross product between two (3×1) vectors

$$\mathbf{S}(\boldsymbol{x}) = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix} .$$

On the other hand, there does not exist a unique parametrization of the matrix $\mathbf{C}_{\mathrm{RB}}$, which represents the Coriolis and centripetal terms. It can be demonstrated that the matrix $\mathbf{C}_{\mathrm{RB}}$ can always be parameterized such that it is skew symmetric, i. e.,

$$\mathbf{C}_{\mathrm{RB}}(\boldsymbol{\nu}) = -\mathbf{C}_{\mathrm{RB}}^{\mathrm{T}}(\boldsymbol{\nu}) \quad \forall \boldsymbol{\nu} \in \mathbb{R}^6 ; \tag{51.15}$$

explicit expressions for $\mathbf{C}_{\mathrm{RB}}$ can be found in [51.8].

Notice that (51.13) can be greatly simplified if the origin of the body-fixed frame is chosen to be coincident with the central frame, i. e., $\boldsymbol{r}_C^{\mathrm{b}} = \mathbf{0}$.

Hydrodynamic Generalized Forces

Equation (51.13) represents the motion of a rigid body in an empty space, while dealing with ships or underwater vehicles requires the consideration of the presence of the hydrodynamics generalized forces, i. e., the forces and moments caused by the presence of the fluid. In hydrodynamics it is common to assume that the generalized hydrodynamics forces on a rigid body can be linearly superimposed [51.9]; in particular, those are separated into radiation-induced forces, environmental disturbances, and restoring forces due to gravity and buoyancy.

Radiation-induced forces are defined as the:

forces on the body when the body is forced to oscillate with the wave excitation frequency and there are no incident waves;

these can be identified as the sum of the *added mass* due to the inertia of the surrounding fluid and the *radiation-induced potential damping* due to the energy dissipated by generated surface waves.

Environmental disturbances can be identified as the generalized forces caused by the wind, the waves, and the ocean current.

The overall equations of motions can therefore be written in matrix form as [51.8, 10, 11]

$$\mathbf{M}_\mathrm{v}\dot{\boldsymbol{\nu}} + \mathbf{C}_\mathrm{v}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}_\mathrm{v}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}_\mathrm{v}(\mathbf{R}_\mathrm{B}^\mathrm{I}) = \boldsymbol{\tau}_\mathrm{v} , \tag{51.16}$$

where $\mathbf{M}_\mathrm{v} = \mathbf{M}_{\mathrm{RB}} + \mathbf{M}_\mathrm{A}$ and $\mathbf{C}_\mathrm{v} = \mathbf{C}_{\mathrm{RB}} + \mathbf{C}_\mathrm{A}$ also include the added mass terms.

In the following sections these generalized forces, specific to the marine environment, will be briefly discussed.

Added Mass and Inertia

When a rigid body is moving in a fluid, the additional inertia of the fluid surrounding the body that is accelerated by the movement of the body has to be considered. This effect can be neglected in industrial robotics since the density of the air is much lower than the density of a moving mechanical system. In underwater applications, however, the density of the water, $\rho \approx 1000\,\mathrm{kg/m^3}$, is comparable with the density of the vehicles. In particular, at 0 °C, the density of freshwater is $1002.68\,\mathrm{kg/m^3}$; for sea water with 3.5% salinity it is $\rho = 1028.48\,\mathrm{kg/m^3}$.

The fluid surrounding the body is accelerated with the body, so a force is necessary to achieve this acceleration, while the fluid exerts a reaction force which is equal in magnitude and opposite in direction. This reaction force is the added mass contribution. The added mass is not a quantity of fluid to add to the system such that it has an increased mass. Different properties hold with respect to the 6×6 inertia matrix of a rigid body due to the fact that the added mass is a function of the body's surface geometry.

The hydrodynamic force along $\boldsymbol{x}_\mathrm{b}$ due to the linear acceleration in the $\boldsymbol{x}_\mathrm{b}$-direction is defined as

$$X_\mathrm{A} := -X_{\dot{u}}\dot{u} , \quad \text{where} \quad X_{\dot{u}} := X_{\dot{u}} := \frac{\partial X}{\partial \dot{u}} ,$$

where the symbol ∂ denotes the partial derivative. In the same way it is possible to define all the remaining 35 elements that relate the six force/moment components $(X \;\; Y \;\; Z \;\; K \;\; M \;\; N)^{\mathrm{T}}$ to the six linear/angular accelerations $(\dot{u} \;\; \dot{v} \;\; \dot{w} \;\; \dot{p} \;\; \dot{q} \;\; \dot{r})^{\mathrm{T}}$. These elements can be grouped

into the added mass matrix $\mathbf{M}_\mathrm{A} \in \mathbb{R}^{6\times6}$. Usually, all the elements of the matrix are nonzero.

In general, added mass and potential damping will be frequency dependent and depend on the forward speed. This is also the case for certain viscous damping terms (skin friction, roll damping, etc.). This gives a pseudo differential equation describing the frequency response of the vehicle. Since some of the coefficients depend on the frequency this is not an ordinary differential equation (ODE). The frequency equation, however, can be transformed to the time domain using the concepts described in [51.12] and [51.13], and recently in [51.14]. The time-domain equations will in general include fluid memory effects [51.4]. However, underwater robots are usually modelled using maneuvering theory where the potential coefficients are evaluated at the zero frequency. The resulting equation is an ODE where the added inertia matrix $\mathbf{M}_\mathrm{A}$ is constant, speed independent, and positive definite

$$\mathbf{M}_\mathrm{A} = \mathbf{M}_\mathrm{A}^\mathrm{T} > \mathbf{0}\,, \quad \dot{\mathbf{M}}_\mathrm{A} = \mathbf{0}\,. \tag{51.17}$$

This result is well known from ship hydrodynamics; [51.15] for instance. The matrix $\mathbf{M}_\mathrm{A}$ can be computed using numerical programs such as WAMIT or Matlab, based on the US Air Force Digital Datcom [51.16]; in this case, the zero-frequency result should be used, that is, $\mathbf{M}_\mathrm{A} = \mathbf{A}(0)$ where $\mathbf{A}(\omega)$ is the frequency-dependent added mass matrix. The potential damping matrix will be small compared to the viscous effects and drag/lift terms. Hence, this term can be set to zero for underwater vehicles. If the added mass is computed experimentally, it is common practice to symmetrize the results such that

$$\mathbf{M}_\mathrm{A} = \frac{1}{2}\left(\mathbf{A}_\mathrm{exp} + \mathbf{A}_\mathrm{exp}^\mathrm{T}\right),$$

where $\mathbf{A}_\mathrm{exp}$ denotes the experimentally obtained added mass terms.

If the body is completely submerged in the water and is designed with port/starboard symmetry (xz-plane) as is common for underwater vehicles in six degrees-of-freedom (DOF), the following structure of the matrices $\mathbf{M}_\mathrm{A}$ can be considered

$$\mathbf{M}_\mathrm{A} = -\begin{pmatrix} X_{\dot{u}} & 0 & X_{\dot{w}} & 0 & X_{\dot{q}} & 0 \\ 0 & Y_{\dot{v}} & 0 & Y_{\dot{p}} & 0 & Y_{\dot{r}} \\ Z_{\dot{u}} & 0 & Z_{\dot{w}} & 0 & Z_{\dot{q}} & 0 \\ 0 & K_{\dot{v}} & 0 & K_{\dot{p}} & 0 & K_{\dot{r}} \\ M_{\dot{u}} & 0 & M_{\dot{w}} & 0 & M_{\dot{q}} & 0 \\ 0 & N_{\dot{v}} & 0 & N_{\dot{p}} & 0 & N_{\dot{r}} \end{pmatrix}. \tag{51.18}$$

The added mass coefficients can theoretically be derived by exploiting the geometry of the rigid body or numerically by strip theory [51.17].

In [51.18] the coefficients for the experimental AUV Phoenix of the Naval Postgraduate School (NPS) are reported. These coefficients have been derived experimentally, and the geometry gives a nondiagonal $\mathbf{M}_\mathrm{A}$ matrix. To provide an order of magnitude for the added mass terms, for the vehicle mass of about 5000 kg, the $X_{\dot{u}}$ is approximately −500 kg.

The added mass also makes an *added* Coriolis and centripetal contribution. It can be demonstrated that the matrix expression can always be parameterized such that

$$\mathbf{C}_\mathrm{A}(\boldsymbol{\nu}) = -\mathbf{C}_\mathrm{A}^\mathrm{T}(\boldsymbol{\nu})\,, \quad \forall \boldsymbol{\nu} \in \mathbb{R}^6\,,$$

whose symbolic expressions can be found in [51.4].

Hydrodynamic Damping

The hydrodynamic damping for marine vehicles is mainly caused by:

- Potential damping
- Skin friction
- Wave drift damping
- Vortex shedding damping
- Viscous damping.

The radiation-induced potential damping due to forced body oscillations is commonly known as potential damping; its dynamic contribution is usually negligible with respect to, e.g., the viscous friction for underwater vehicles while it may be significant for surface vessels.

Linear skin friction is due to laminar boundary layers and can affect the low-frequency motion of the vehicle. Together with this effect, at high frequencies it is possible to observe a quadratic, or nonlinear, skin friction phenomenon caused by turbulent boundary layers.

Wave drift damping is the dominant dynamic damping effect in surge motion of surface vessels in high sea. It can be considered as an added resistance for boats advancing in waves; its drift is proportional to the square of the significant wave height. In the sway and yaw directions, however, its dynamic contribution is negligible with respect to the effect of vortex shedding.

A body moving in a fluid causes a separation of the flow; this can still be considered as laminar in the upstream while two antisymmetric vortices can be observed in the downstream. In case that the body is a cylinder moving in a direction normal to its axis, the result is a periodic force normal to both the velocity and the axis. This effect may cause the oscillation of cables and other underwater structures. However, concerning underwater vehicles, this effect is negligible for

ROVs and may be counteracted by designing proper small control surfaces for torpedo-like AUVs.

Vortex shedding is an unsteady flow that takes place at special flow velocities (according to the size and shape of the cylindrical body). In this flow vortices are created at the back of the body, periodically from each side.

The viscosity of the fluid also causes dissipative forces. These are composed of drag and lift forces, the former being parallel to the relative velocity of the vehicle with respect to the water while the latter are normal to it. For a sphere moving in a fluid, the drag force can be modeled as [51.9]

$$F_{\text{drag}} = \frac{1}{2}\rho U^2 S C_{\text{d}}(R_{\text{n}}) \,, \tag{51.19}$$

where ρ is the fluid density, U is the velocity of the sphere, S is the frontal area of the sphere, C_{d} is the nondimensional drag coefficient, and R_{n} is the Reynolds number. For a generic body, S is the projection of the frontal area along the flow direction. The drag force can be considered as the sum of two physical effects: the frictional contribution of the surface whose normal is perpendicular to the flow velocity, and the pressure contribution of the surface whose normal is parallel to the flow velocity. For a hydrofoil moving in a fluid, the lift force can be modeled as [51.9]

$$F_{\text{lift}} = \frac{1}{2}\rho U^2 S C_{\text{l}}(R_{\text{n}}, \alpha) \,, \tag{51.20}$$

where S is now the area, C_{l} is the nondimensional lift coefficient, and α is the angle of attack, i. e., the angle between the relative velocity and the tangent to the surface. For small angles of attack, i. e., $|\alpha| < 10°$, the lift coefficient is approximatively proportional to α and rapidly decays to zero as α increases [51.19].

The drag and lift coefficients are therefore dependent on the Reynolds number, i. e., on the laminar/turbulent fluid motion

$$R_{\text{n}} = \frac{\rho |U| D}{\mu} \,,$$

where D is the characteristic dimension of the body perpendicular to the direction of U and μ is the dynamic viscosity of the fluid. Table 51.3 reports the drag coefficients as a function of the Reynolds number for a cylinder [51.20].

A common simplification considers only linear and quadratic damping terms and group these into a matrix $\mathbf{D}_{\text{v}}$ as in (51.16) such that

$$\mathbf{D}_{\text{v}}(\boldsymbol{\nu}) > \mathbf{0} \,, \quad \forall \boldsymbol{\nu} \in \mathbb{R}^6 \,.$$

Gravity and Buoyancy

When a rigid body is completely or partially submerged in a fluid under the effect of the gravity two more forces have to be considered: the gravitational force and buoyancy. The latter is the only hydrostatic effect, i. e., it is not a function of the relative movement between the body and fluid.

Let us define as

$$\boldsymbol{g}^{\text{I}} = (0 \;\; 0 \;\; 9.81)^{\text{T}} \text{m}/s^2$$

the acceleration of gravity. This effect is not constant but varies with depth, longitude, and latitude; however, this value is usually accurate enough for most applications except for inertial navigation systems.

For a completely submerged body the computation of these dynamic effects is straightforward. The submerged weight of the body is defined as $W = m||\boldsymbol{g}^{\text{I}}||$, while its buoyancy $B = \rho \nabla ||\boldsymbol{g}^{\text{I}}||$, where ∇ is the volume of the body and m is its mass. The gravity force, which acts at the center of mass $\boldsymbol{r}_{\text{C}}^{\text{B}}$, is represented in the body-fixed frame by

$$\boldsymbol{f}_{\text{G}}(\mathbf{R}_{\text{I}}^{\text{B}}) = \mathbf{R}_{\text{I}}^{\text{B}} \begin{pmatrix} 0 \\ 0 \\ W \end{pmatrix} ,$$

while the buoyancy force, acting at the center of buoyancy $\boldsymbol{r}_{\text{B}}^{\text{B}}$, is represented in the body-fixed frame by

$$\boldsymbol{f}_{\text{B}}(\mathbf{R}_{\text{I}}^{\text{B}}) = -\mathbf{R}_{\text{I}}^{\text{B}} \begin{pmatrix} 0 \\ 0 \\ B \end{pmatrix} .$$

The 6×1 vector of force/moment due to gravity and buoyancy in the body-fixed frame, included in the left-hand side of the equations of motion, is represented by

$$\boldsymbol{g}_{\text{v}}(\mathbf{R}_{\text{I}}^{\text{B}}) = - \begin{pmatrix} \boldsymbol{f}_{\text{G}}(\mathbf{R}_{\text{I}}^{\text{B}}) + \boldsymbol{f}_{\text{B}}(\mathbf{R}_{\text{I}}^{\text{B}}) \\ \boldsymbol{r}_{\text{G}}^{\text{B}} \times \boldsymbol{f}_{\text{G}}(\mathbf{R}_{\text{I}}^{\text{B}}) + \boldsymbol{r}_{\text{B}}^{\text{B}} \times \boldsymbol{f}_{\text{B}}(\mathbf{R}_{\text{I}}^{\text{B}}) \end{pmatrix} .$$

Table 51.3 Lift and drag coefficient for a cylinder

Reynolds number	Regime motion	C_{d}	C_{l}
$R_{\text{n}} < 2 \times 10^5$	Subcritical flow	1	3–0.6
$2 \times 10^5 < R_{\text{n}} < 5 \times 10^5$	Critical flow	1–0.4	0.6
$5 \times 10^5 < R_{\text{n}} < 3 \times 10^5$	Transcritical flow	0.4	0.6

In the following, the symbol $\boldsymbol{r}_{\mathrm{G}}^{\mathrm{B}} = (x_{\mathrm{G}}\ y_{\mathrm{G}}\ z_{\mathrm{G}})^{\mathrm{T}}$ (with $\boldsymbol{r}_{\mathrm{G}}^{\mathrm{B}} = \boldsymbol{r}_{\mathrm{C}}^{\mathrm{B}}$) will be used for the center of gravity. The expression for $\boldsymbol{g}_{\mathrm{v}}$ in terms of the Euler angles is

$$\boldsymbol{g}_{\mathrm{v}}(\boldsymbol{\eta}_2) = \begin{pmatrix} (W-B)s_\theta \\ -(W-B)c_\theta s_\phi \\ -(W-B)c_\theta c_\phi \\ -(y_{\mathrm{G}}W - y_{\mathrm{B}}B)c_\theta c_\phi + (z_{\mathrm{G}}W - z_{\mathrm{B}}B)c_\theta s_\phi \\ (z_{\mathrm{G}}W - z_{\mathrm{B}}B)s_\theta + (x_{\mathrm{G}}W - x_{\mathrm{B}}B)c_\theta c_\phi \\ -(x_{\mathrm{G}}W - x_{\mathrm{B}}B)c_\theta s_\phi - (y_{\mathrm{G}}W - y_{\mathrm{B}}B)s_\theta \end{pmatrix} . \tag{51.21}$$

Current

Ocean currents are mainly caused by tidal movement, the atmospheric wind system over the sea surface, heat exchange at the sea surface, salinity changes, the Coriolis force due to the Earth's rotation, nonlinear waves, the major ocean circulations such as the Gulf Stream, the effect of setup phenomena or storm surges, and strong density gradients in the upper ocean. Currents can be very different due to local climatic and/or geographic characteristics; as an example, in fjords, the tidal effect can cause currents of up to 3 m/s, moreover, specific mathematical models exist for the various components [51.8].

Let us assume that the ocean current, expressed in the inertial frame, $\boldsymbol{\nu}_{\mathrm{c}}^{\mathrm{I}}$, is constant and irrotational, i. e.,

$$\boldsymbol{\nu}_{\mathrm{c}}^{\mathrm{I}} = (\nu_{\mathrm{c},x}\ \nu_{\mathrm{c},y}\ \nu_{\mathrm{c},z}\ 0\ 0\ 0)^{\mathrm{T}} ,$$

and $\dot{\boldsymbol{\nu}}_{\mathrm{c}}^{\mathrm{I}} = \boldsymbol{0}$; its effects can be added to the dynamics of a rigid body moving in a fluid simply by considering the *relative* velocity in the body-fixed frame

$$\boldsymbol{\nu}_{\mathrm{r}} = \boldsymbol{\nu} - \mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\boldsymbol{\nu}_{\mathrm{c}}^{\mathrm{I}} \tag{51.22}$$

in the derivation of the added Coriolis and centripetal and the damping terms.

$$\mathbf{M}_{\mathrm{v}}\dot{\boldsymbol{\nu}} + \mathbf{C}_{\mathrm{RB}}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{C}_{\mathrm{A}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}} + \mathbf{D}_{\mathrm{v}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}} + \boldsymbol{g}_{\mathrm{v}}(\mathbf{R}_{\mathrm{B}}^{\mathrm{I}}) = \boldsymbol{\tau}_{\mathrm{v}} . \tag{51.23}$$

Notice that the term $\mathbf{C}_{\mathrm{A}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}}$ includes the important destabilizing effect known as the Munk moment [51.9].

If $\mathbf{D}_{\mathrm{v}}(\boldsymbol{\nu}_{\mathrm{r}})$ is unknown, quadratic surge resistance and the cross-flow drag principle can be used to describe the dissipative forces and moments in surge, sway, and yaw [51.9]. Moreover

$$\mathbf{C}_{\mathrm{A}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}} + \mathbf{D}_{\mathrm{v}}(\boldsymbol{\nu}_{\mathrm{r}})\boldsymbol{\nu}_{\mathrm{r}} \approx (X_{\mathrm{c}}\ Y_{\mathrm{c}}\ 0\ 0\ 0\ N_{\mathrm{c}})^{\mathrm{T}} \tag{51.24}$$

for large relative current angles $|\beta_{\mathrm{c}} - \psi|$, where β_{c} is the current direction, the cross-flow principles models the sway force Y_{c} and yaw moment N_{c} as

$$Y_{\mathrm{c}} = \frac{\rho}{2}\int_L H(x)C_{\mathrm{D}}(x)v_{\mathrm{r}}^x(x)|v_{\mathrm{r}}^x(x)|\mathrm{d}x \tag{51.25}$$

$$N_{\mathrm{c}} = \frac{\rho}{2}\int_L xH(x)C_{\mathrm{D}}(x)v_{\mathrm{r}}^x(x)|v_{\mathrm{r}}^x(x)|\mathrm{d}x - X_{r|r|}r|r| , \tag{51.26}$$

where L is the vehicle length, $H(x)$ is the vehicle height, $C_{\mathrm{D}}(x)$ is the two-dimensional drag coefficient, and $v_{\mathrm{r}}^x(x) = v_{\mathrm{r}} + rx$ is the relative cross-flow velocity at x. In practice $C_{\mathrm{D}}(x)$ can be chosen as a constant between 0 and 1. The proper value can be determined by curve fitting of experimental data. Along the surge direction, however, the quadratic damping contribution X_{c} is still well represented by a term proportional to the square of the relative velocity, whose symbolic expression can be written as

$$X_{\mathrm{c}} = -X_{u|u|}u_{\mathrm{r}}|u_{\mathrm{r}}| , \tag{51.27}$$

where $-X_{u|u|} > 0$ is the quadratic surge damping coefficient, which can be found by curve fitting of experimental data or relating it to the drag coefficient C_{d} as in (51.19). The approximation embedded in (51.24) does not represent sufficiently enough the dynamics at low velocities, when the quadratic terms are negligible. Hence, it is common to add a linear optional damper in surge, sway and yaw; this can be physically explained as skin friction which shows a linear behavior.

Alternatively, the computation of the quadratic surge resistance, nonlinear roll damping, and the cross-flow drag effect can be made by resorting to the Datcom database for aircraft, as shown in [51.21].

Model Properties

For completely submerged bodies in an ideal fluid moving at low velocity where there are no currents or waves, (51.16) satisfies the following properties:

- The inertia matrix is symmetric and positive definite, i. e.,
$$\mathbf{M}_{\mathrm{v}} = \mathbf{M}_{\mathrm{v}}^{\mathrm{T}} > \mathbf{0} .$$
- The damping matrix is positive definite, i. e.,
$$\mathbf{D}_{\mathrm{v}}(\boldsymbol{\nu}) > \mathbf{0} .$$
- The matrix $\mathbf{C}_{\mathrm{v}}(\boldsymbol{\nu})$ is skew symmetric, i. e.,
$$\mathbf{C}_{\mathrm{v}}(\boldsymbol{\nu}) = -\mathbf{C}_{\mathrm{v}}^{\mathrm{T}}(\boldsymbol{\nu}), \forall \boldsymbol{\nu} \in \mathbb{R}^6 .$$

Hydrodynamic Modeling

The mathematical model of an underwater robot as expressed in (51.16) is of great importance; even when

Table 51.4 Dorado payload for scientific sampling (courtesy of Monterey Bay Aquarium Research Institute [51.22])

Sensor model	Description
WHN300	300 kHz Acoustic Doppler Current Profiler/Doppler Velocity Log, manufactured by Teledyne/RD Instruments
8CB4000I	Paroscientific Digiquartz pressure sensor, 4000 m full scale
Gulpers	10 × 2 liter Water Samplers
HS2	Hobilabs 2 channel backscatter/fluorometer, 420 nm/700 nm excitation
LISST-100	Sequoia Scientific particle size spectrum instrument
ISUS	In-situ ultraviolet spectrometer for nitrate measurement
2xSBE3/SBE4	Seabird temperature/conductivity instrument, 2 pairs of instruments carried
SBE43	Seabird dissolved oxygen cell
2xSBE5	Pumps for C/T/DO/ISUS and C/T/LISST flow paths
UBAT	Wetlabs bioluminescence assessment tool (bathyphtometer for measuring plankton bio-luminsescence)
LOPC	Laser Optical Plankton Counter
ECO-CDOM	Fluorometer for measuring CDOM (colored dissolved organic matter)

simplified it captures the most important part of the dynamics. Moreover, it is in a form appropriate for control design. A wide literature exists on AUV/ROV controllers whose stability relies on the properties reported above. On the other hand, there are working conditions in which these assumptions are no longer valid, i. e., when the AUV is traveling at high speed, or close to the surface, or when its shape does not allow geometric simplifications. The latter is the case of, e.g., several ROVs. In addition, it is still common to design the controllers for AUVs based on linearized models and to control ROVs with simple PID controllers.

These considerations justify a modeling effort to calculate the hydrodynamic terms more accurately with the aim of prediction, simulation, and performance analysis rather than control design. This can be done by switching from a coefficient-based approach, such as that presented above, to a component modeling method, the latter being based on computational fluid dynamics theory. In detail, each vehicle geometry, with its specific angle of attack and sideslip, is taken into consideration when computing the hydrodynamic forces/moments. This increased computational effort makes it possible to capture some dynamic effects, such as the vortex-induced roll moment, not justifiable with the coefficient-based approach.

The control plant model is usually a simplified model that captures the most important parts of the dynamics. The most accurate model of the vehicle should be used for prediction and motion simulation.

51.2.2 Sensor Systems

Underwater vehicles are equipped with a sensor system devoted to enabling motion control as well as accomplishing the specific mission it has been commanded to complete. In the latter case, sensors developed for chemical/biological measurements or mapping may be installed, which is beyond the scope of this chapter. For seek of example, Table 51.4 reports the payload for the AUV Dorado, shown in Fig. 51.3.

AUVs need to operate underwater most of the time; one of the major problems with underwater robotics is in the localization task due to the absence of a single, proprioceptive sensor to measure the vehicle position. The global positioning system (GPS) cannot be used underwater. Redundant multisensor systems are commonly combined using state estimation or sensor fusion techniques to provide fault detection and tolerance capability to the vehicle. Table 51.5 lists the types of sensors and the corresponding variable measured commonly available for unmanned underwater vehicles (UUVs).

The sensors that can be found on an underwater vehicle are:

- *Compass*: A gyrocompass can provide an estimate of geodetic north accurate to a fraction of a degree. Magnetic compasses can provide estimates of

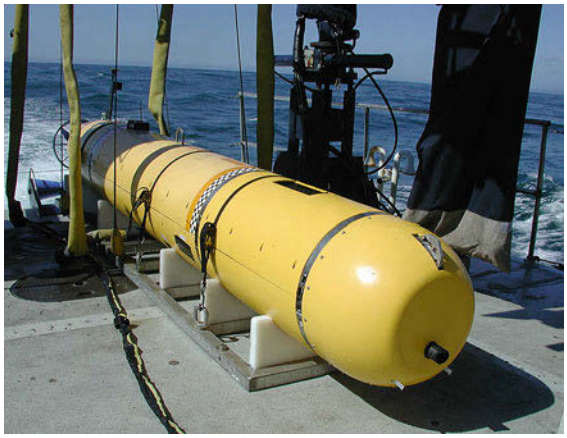


Fig. 51.3 The AUV Dorado (courtesy of Monterey Bay Aquarium Research Institute [51.22])

Table 51.5 UUVs: possible instrumentation

Sensor	Measured variable
Inertial system	Linear acceleration and angular velocity
Pressure meter	Vehicle depth
Frontal sonar	Distance from obstacles
Vertical sonar	Distance from the bottom
Doppler velocity log	Relative velocity vehicle/bottom
Current meter	Relative velocity vehicle/current
Global positioning system	Absolute position at the surface
Compass	Orientation
Acoustic positioning	Absolute position in known area
Vision systems	Relative position/velocity
Acoustic Doppler current profiler	Relative water velocity

magnetic north with an accuracy of less than 1° if carefully calibrated to compensate for magnetic disturbances from the vehicle itself. Tables or models can be used to convert from magnetic north to geodetic north.

- *Gyroscope*: The term gyroscope, or gyro, is the name given generically to any instrument measuring inertial angular rotation. Those are based on inertial properties of a vibrating mass or of light. Very accurate gyros may output signal that, when integrated, give heading information with a small enough drift for practical purposes.
- *Inertial measurement unit* (*IMU*): An IMU provides information about the vehicle's linear acceleration and angular velocity. These measurements are combined to form estimates of the vehicle's attitude including an estimate of geodetic (true) north from the most complex units. In most cases, for slow-moving underwater vehicles, an independent measurement of the vehicle's velocity is also required to produce accurate estimates of the translational velocity or relative displacement.
- *Depth sensor*: Measuring the water pressure gives the vehicle's depth. At depths beyond a few hundred meters, the equation of state of seawater must be invoked to produce an accurate depth estimate based on the ambient pressure [51.23]. With a high-quality sensor, these estimates are reliable and accurate, giving a small error of order 0.01%.
- *Altitude and forward-looking sonar*: These are used to detect the presence of obstacles and distance from the seafloor.
- *Doppler velocity log* (*DVL*): By processing reflected acoustic energy from the seafloor and the water column from three or more beams, estimates of vehicle velocity relative to the seafloor and relative water motion can be obtained. Bottom-tracking velocity estimates can be accurate to $\approx 1\,\mathrm{mm/s}$.
- *Global navigation satellite system* (*GLS*): This is used to localize the vehicle while on the surface to initialize or reduce drift of estimates from an IMU/DVL combination. GNSS, such as GPS or Galileo, only works at the surface.
- *Acoustic positioning*: A variety of schemes exist for determining vehicle position using acoustics. Long-baseline navigation can determine the position of the vehicle relative to a set of acoustic beacons anchored to the seafloor or on the surface through range estimates obtained from acoustic travel times. Ultrashort-baseline navigation uses phase information to determine direction from a cluster of hydrophones; this is most often used to determine the direction of the vehicle (in two dimensions) from a surface support vessel, which is then combined with an acoustic travel-time measurement to produce an estimate of relative vehicle position in

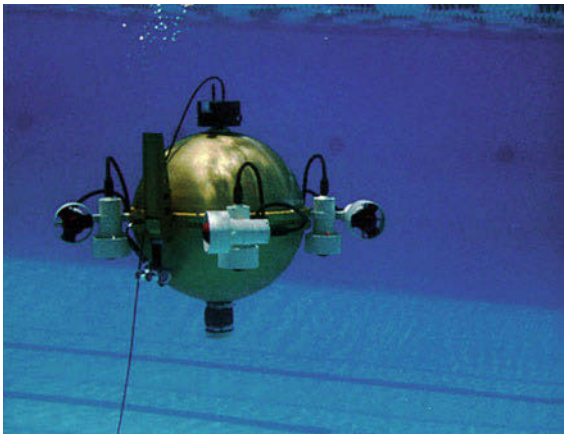


Fig. 51.4 The fully actuated AUV ODIN (courtesy of Autonomous Systems Laboratory, University of Hawaii [51.24])

Table 51.6 JHUROV instrumentation

Measured variable	Sensor	Precision	Update rate
3-DOF vehicle position	SHARP acoustic transponder	0.5 cm	10 Hz
Depth	Foxboro/ICT model n. 15	2.5 cm	20 Hz
Heading	Litton LN200 IMU Gyro	0.01°	20 Hz
Roll and pitch	KVH ADGC	0.1°	10 Hz
Heading	KVH ADGC	1°	10 Hz

Table 51.7 ODIN III sensors summary

Measured variable	Sensor	Update rate
xy vehicle position	8 sonars	3 Hz
Depth	Pressure sensor	30 Hz
Roll, pitch, and yaw	IMU	30 Hz

spherical coordinates. These techniques will be discussed later in the localization section.

- *Vision systems*: Cameras can be used to obtain estimates of relative, and in some cases absolute, motion using a type of simultaneous localization and mapping (SLAM) algorithm [51.25] and used to perform tasks such as visual tracking of pipelines, station keeping, visual servoing or image mosaicking.

As an example, Table 51.6 reports some data from the instrumentation of the ROV developed at the John Hopkins University [51.26], and Table 51.7 some data from the AUV ODIN III [51.27] shown in Fig. 51.4. *Majumder* et al. [51.28] show some data fusion results with a redundant sensorial system mounted on the AUV Oberon, while [51.29] reviews advances in navigation technology.

51.2.3 Actuating Systems

Marine vehicles are generally propelled by means of thrusters or hydrojets. In the case of ROVs with structural pitch–roll stability, there are usually four thrusters that provide holonomic mobility to the four remained DOFs, in particular, the depth is often decoupled and the vehicle is controlled on a plane in the surge, sway, and yaw DOFs. Those vehicles, being underactuated, cannot easily be used for interactive control by means of a manipulator due to the impossibility of counteracting the generalized forces exchanged with the manipulator's base; in such case, six or more thrusters are required. AUVs generally have a torpedo-like shape and are used for mapping/exploration. They are propelled using one or two thrusters parallel to the fore–aft direction and a fin and a rudder; this kind of propulsion is obviously nonholonomic and experiences a loss of mobility at low velocities. Hydrojets, also known as pump jets or water jets, are systems that create a jet of water for propulsion; they have certain advantages over thrusters such as a higher power density and usability in shallow water, but can provide thrust in one direction only.

Several efforts have been made to accurately and efficiently describe the mathematical model of a thruster; [51.30] reports a *one-state* model where the state is n, the propeller shaft speed. In [51.31] a *two-state* model is proposed to take into account the experimentally observed overshoot in the thrust; together with n, the additional state variable is u_p, the axial flow velocity in the propeller disc. In [51.32] a thruster model incorporating the effects of rotational fluid velocity and inertia on thruster responses is presented together with a method for experimentally determining nonsinusoidal lift/drag curves. A *three-state* model is described in [51.33]

$$\begin{aligned} J_\mathrm{m}\dot{n} + K_n n &= \tau - Q\,, \\ m_f\dot{u}_\mathrm{p} + d_{f0}u_\mathrm{p} + d_f|u_\mathrm{p}|(u_\mathrm{p} - u_\mathrm{a}) &= T\,, \\ (m - X_{\dot{u}})\dot{u} - X_u u - X_{u|u|}u|u| &= (1-t)T\,, \end{aligned}$$

where J_m is the moment of inertia for the dc-motor/propeller, K_n is the linear motor damping coefficient, τ is the motor control input, Q is the propeller torque, m_f is the mass of water in the propeller control volume, u_p is the axial flow velocity in the propeller disc, d_{f0} and d_f are the linear and quadratic damping coefficients for control volume, respectively, u_a is the ambient water velocity, T is the propeller thrust, and t is the thrust deduction number (Fig. 51.5). In the case of steady-state motion, i. e., $\dot{u} = 0$, the ambient water velocity u_a is related to the surge by the *wake fraction number* w as

$$u_\mathrm{a} = (1-w)u\,. \tag{51.28}$$

Notice also that the unmeasured variable u_p can be estimated using a nonlinear observer [51.33].

The *outputs* of the nonlinear three-state dynamic systems are the thrust T and the torque Q, which are functions of several variables; in the following, unsteady flow effects such as air suction, cavitation, the in-and-out-of-water (Wagner), boundary layer, and

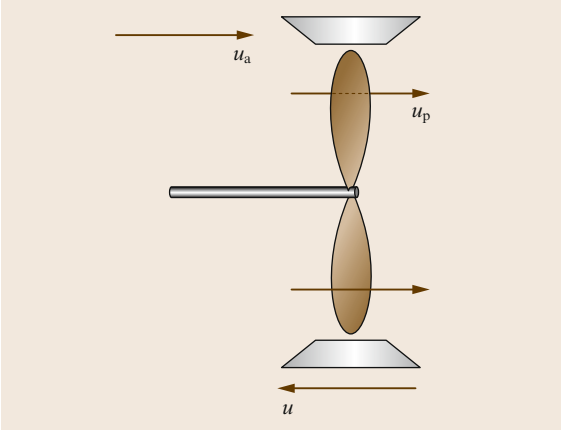


Fig. 51.5 Ambient water and axial flow velocities affecting thruster behavior

gust (Kuessner) effects will be neglected. This leads to a quasi-steady representation of the model

$$T = \rho D^4 K_T(J_0) n|n| \,, \tag{51.29}$$

$$Q = \rho D^5 K_Q(J_0) n|n| \,, \tag{51.30}$$

where D is the propeller diameter and $K_T(J_0)$ and $K_Q(J_0)$ are the thrust and torque coefficients, respectively. The latter are function of the advance ratio J_0

$$J_0 = \frac{u_a}{nD} \,. \tag{51.31}$$

The open-water propeller efficiency in undisturbed water is given as the ratio of the work done by the propeller in producing a thrust force divided by the work required to overcome the shaft torque

$$\eta_0 = \frac{u_a T}{2\pi n Q} = \frac{J_0}{2\pi} \cdot \frac{K_T}{K_Q} \,. \tag{51.32}$$

Figure 51.6 shows the values of K_T, K_Q, and η_0 as functions of the advance ratio for the Wageningen B4-70 propeller [51.34].

Controlling a marine vehicle usually requires that desired forces/moments act on the vehicle's body; these generalized forces are mapped into desired thrusts to be provided by the propellers. There is, thus, a nontrivial control problem in that the motors are required to provide the appropriate propeller shaft speed n that satisfies the nonlinear relationship with the thrust T presented above.

To enable robustness with respect to possible failures, the actuating system is often redundant. In this case, the problem of allocation of the desired force/moment acting on the vehicle among the thrusters must also be solved. Reference [51.35] reports a survey of control allocation methods of ships and underwater vehicles.

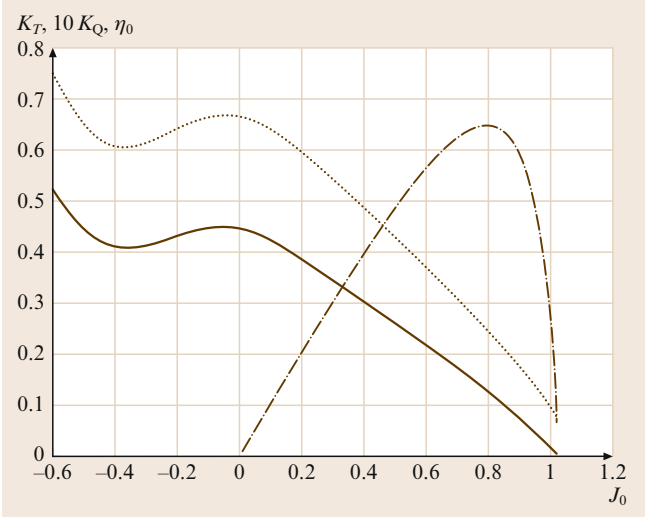


Fig. 51.6 Values of K_T (*solid*), $10K_Q$ (*dotted*), and η_0 (*dash-dotted*) as a function of J_0 (after [51.34])

51.2.4 Communication Systems

Autonomous underwater vehicles may require to communicate with a remote operator or with a base station for monitoring purposes. In case of coordinated mission among several AUVs a network of vehicles/modes communicating among them need to be developed.

Due to the physical characteristics of the water, the more diffused communication technology is based on acoustic propagation. However, the performance of acoustic modems are not as efficient as their aerial conuterparts. Three main factors affect the acoustic propagation: a low speed of sound ($\approx 1500\,\mathrm{m/s}$), the presence of time-varying multiple paths, an attenuation increasing with the frequency. Finally, the channel capacity is function of the distance and is limited [51.36]. For such a challenging communication medium, efficient communication protocols need to be properly designed [51.37].

51.2.5 Mission Control System

The mission control system (MCS) can be considered as the highest-level process running during an AUV's mission; it is responsible for achieving several control objectives. At the highest level it works as an interface between the operator, accepting his instructions in a higher-level language and decomposing those instructions into mission tasks according to the implemented software architecture. The mission tasks are generally concurrent and their handling depends on the vehicle state and environmental conditions; it is therefore the MCS that handles the tasks, eventually suppressing, sequencing, modifying, and prioritizing them. An MCS is also usually equipped with a graphical user interface (GUI) to report the mission state to the operator, see, e.g., the Neptus Command and Control Infrastructure in VIDEO 324.

As for most advanced robotics applications, an efficient MCS should allow the use of complex robotic systems by users that do not necessarily know all of their technical details. An overview relevant to underwater mission control is given in [51.38], which includes an interesting classification of the MCSs in use in several laboratories according to which four major AUV control architectures were identified: the hierarchical, heterarchical, subsumption, and hybrid.

From a mathematical point of view, the MCS generally needs to be designed in order to be able to address hybrid dynamical systems, i. e., handling both event-driven and time-driven processes. In [51.39], e.g., the MCS developed at the Portuguese Instituto Superior Técnico (IST), named CORAL, is implemented by resorting to a Petri-net-based architecture that properly handles all the necessary tasks in order to manage navigation, guidance and control, sensing, communications, etc.

The motion-oriented operating system (MOOS), designed at the Massachusetts Institute of Technology, is a software tool capable of executing and coordinating a multitude of subsea operations. The MCS developed at the Naval Postgraduate School is in the framework of the behavioral control organized in three layers [51.40]; it is based on PROLOG, an artificial intelligence language for predicate logic.

51.2.6 Guidance and Control

The terms guidance and control can be defined as [51.8]:

- *Guidance* is the action of determining the course, attitude, and speed of the vehicle, relative to some reference frame (usually the Earth), to be followed by the vehicle.
- *Control* is the development and application to a vehicle of appropriate forces and moments for operating point control, tracking, and stabilization. This involves designing the feedforward and feedback control laws.

Figure 51.7 shows the corresponding block diagram, in which the navigation component is also outlined.

Guidance of Underwater Vehicles

Guidance algorithms may benefit from a wide range of inputs, overall mission information, real-time operator input, environmental measured data such as the ocean current, environmental topological information such as a bathymetric map, exteroceptive sensors for obstacle avoidance, and obviously the vehicle state as output from the navigation system.

The vehicle may be required to follow a path, i. e., a curve geometrically represented in two or three dimensions, or a trajectory, i. e., a path with a specific time law assigned. Moreover, when the desired position is constant, the problem is called set-point regulation or maneuvering. The guidance problem is commonly decomposed into simple subtasks of lower dimension: an attitude control problem and a path control. Moreover, attitude is usually considered as a simple depth set-point with null roll and pitch and the path is usually a line in the horizontal plane.

One of the most common guidance approaches is based on the generations of way-points. Those are usually stored in a database and are used to generate the vehicle path/trajectory; a *passing* velocity, in fact, may be defined together with the Cartesian coordinates of the points. The simplest way to connect the way-points is to use the segments connecting two successive way-points. Efficient way-point-based guidance approaches need to take into account the presence of the current and the eventual nonholonomicity of the vehicle [51.41]. A technique for adaptively tracking bathymetric contours by proper generation of way-points is presented in [51.42]; environment information is acquired by mean of a single vertical sonar. An alternative method is based on *line-of-sight* guidance [51.43–45]. In this case, the heading control is computed by considering as input the angle formed by the vector from the vehicle to the next way-point rather than requiring the vehicle to

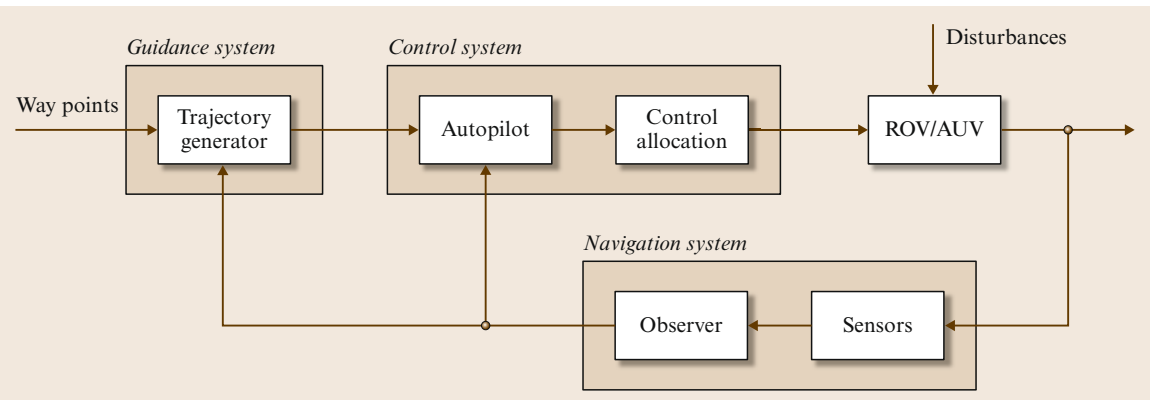


Fig. 51.7 Guidance, navigation, and control for an autonomous marine vehicle

exactly follow the line segment between the current and the following way-point. For docking maneuvers, algorithms must enable control that meets the precision and alignment constraints of the docking fixture [51.46].

By combining vision-based guidance with a neurocontroller trained by reinforcement learning, in [51.47], an algorithm aimed at a hold station on a reef or swimming along a pipe has been presented. In [51.48] guidance for AUVs specifically involved in a predeployment survey of the sea bottom and visual inspection of pipelines is given. *Hyland* and *Taylor* [51.49] report a specific guidance system aimed at mine avoidance for AUVs. Based on a three-dimensional discretization of the environment, the path-planning technique consists of computing a safe path avoiding the unsafe cells of the map. Due to the poor manoeuvrability at low speed under some conditions, the vehicle has to make a 360° turn to avoid stopping and to map the environment close to it before generating a safe path.

A deep discussion on guidance for surface and underwater vehicles can be found in [51.4, 8].

Control of Underwater Vehicles

Control of underwater vehicles needs to consider the different operating conditions and actuating configurations in which a submerged vehicle is required to operate. In particular, there are three main control problems:

- An AUV traveling at high speed ($>1\,\mathrm{m/s}$) generally equipped with at least one thruster aligned in the fore–aft direction and at least two control surfaces (stern and rudder).
- An underactuated ROV, with high metacentric stability, i. e., structurally stable in roll and pitch, and equipped with at least four thrusters.
- A fully actuated AUV equipped with at least six trusters.

AUVs equipped with control surfaces are underactuated vehicles mainly used for survey/exploration missions. Inheriting the common practice of submarine control, they are not allowed to perform arbitrary motions in six DOFs but are rather designed to perform specific movements such as: cruising along a given direction at constant depth, steering at constant depth, or diving. Marine experience and mathematical insight, in fact, demonstrate that these movements are lightly coupled in dynamic terms. For these vehicles, moreover, specific manoeuvres such as homing or docking require special capabilities [51.46]. This requires the design of vehicles that are structurally stable in the roll DOF. Cruising requires control of the surge velocity $u(t)$; steering requires control of the sway velocity $v(t)$ and the yaw DOF $r(t)$, $\psi(t)$, diving requires control of the heave DOF $\omega(t)$, $z(t)$ and the pitch DOF $q(t)$, $\theta(t)$. The simplest configuration of actuators that can control an AUV through these movements is composed of one thruster aligned along the fore–aft direction, one stern, and one rudder; the control variables, thus, are the propeller speed and the deflection of the fins. Several approaches can then be considered to solve this control problem, among them, in [51.50] the sliding mode control is proposed, while [51.51] presents an adaptive sliding mode control for the diving manoeuvre. *Healey* and *Lienard* [51.18] reports a successful implementation of multivariable sliding mode control on the NPS AUV II, later also implemented on the NPS ARIES AUV [51.52]. As the model of an AUV traveling at high speed is nonlinear and coupled, the tuning of the parameters is mainly based on a linearized model around the working conditions.

From a descriptive point of view, an ROV is mainly a box-shaped underwater vehicle equipped with tools such as a video camera or robot manipulator, while its payload is often variable depending on the task. It is remotely operated and physically connected to another vehicle, either an underwater or a surface vessel. It is mainly designed to travel at low speed and it is structurally stable in roll and pitch, while its depth, surge, sway, and yaw are independently controllable. Due to the absence of a specific shape, the varying payload, and the relatively low required performances, it is common to control a ROV by means of single-input single-output (SISO) controllers. Moreover, the PID approach is often used due to its simplicity. A two-layer guidance and control architecture for the ROV Romeo is given in [51.53].

Control of a fully actuated AUV in six DOFs is needed in the case of, e.g., an interaction task performed by a manipulator mounted on a vehicle; the latter, in fact, needs to provide all the force/moment components in order to counteract the presence of the manipulator dynamically. This problem is kinematically similar to the problem of controlling a satellite in six DOFs; the underwater environment, however, makes it significatively different from the dynamic point of view. In kinematic terms the main issue is in implementing a suitable policy for orientation control; any three-parameter representation of orientation, in fact, experiences representation singularities (Chap. 2). This problem may be overcome by resorting to a redundant representation of the orientation such as the quaternion. Most of the six-DOF controllers proposed in the literature are based on (51.16), which model the simplified effect of the hydrodynamic terms and which have very similar properties as the equations of motion of an industrial manipulator. Based on this, it is obviously possible to find a collection of approaches inherited

from classical robotics, [51.4, 8] for some examples. In [51.54], some specific considerations for the underwater environment lead to a quaternion-based, adaptive controller; it is worth noticing that adaptive control requires a suitable, and simplified, expression for the hydrodynamic terms. In [51.55] a comparison among several 6-DOF controllers is made. VIDEO 267 and VIDEO 268 show two controllers validation implemented in a pool.

51.2.7 Localization

Localization in the underwater environment can be a complex task, mainly due to the absence of a single external sensor that gives the vehicle position such as, e.g., the GPS for outdoor ground vehicles; moreover, the environment is often poorly structured.

One of the most reliable methods is based on the use of acoustic systems such as the baseline systems: the long-baseline system (LBL), the short-baseline system (SBL), and the ultrashort-baseline system (USBL). These systems are based on the presence of a transceiver mounted on the vehicle and a variable number of transponders located in known positions. The transceiver's distance from each transponder can be measured via the measurement of an echo delay; from this information the position of the vehicle can be calculated by basic triangulation operations [51.56]. The USBL can be used with a single transponder, which is usually mounted on a surface ship whose position is measured by GPS. Recent works are aimed at developing localization algorithms based on a single beacon, i. e., range-only measurement among vehicles and buoys [51.57].

Another localization system is called terrain-aided navigation and is based on the use of terrain elevation maps; bathymetric maps are available, especially in the case of well-known locations such as harbors where they usually have a resolution of ≈ 1 m. In this case, the vehicle position is obtained by filtering the information coming from a downward-looking sonar. In [51.58], a particle filter approach was used to localize an AUV in Sydney harbor.

Moving vehicles may be equipped with an IMU or DVL in order to measure its velocity and/or acceleration. This data can then be integrated to estimate the vehicle position. This kind of information is subject to the drift phenomenon and may not be reliable for long-duration runs or may become cost ineffective if accurate IMU devices are needed.

Relative localization can be obtained by resorting to any device that provides information about the relative position of the vehicle with respect to the environment, even in the absence of a map. In this case, by filtering the distance measurements taken along the motion, the vehicle's position can be measured. This is the case, e.g., of sonar or vision-based localization techniques [51.59].

Often, the techniques presented above are used together in a redundant system and the effective position is obtained by resorting to sensor fusion techniques such as the Kalman filtering approach.

Simultaneous Localization and Mapping

Simultaneous localization and mapping (SLAM), also known as concurrent mapping and localization (CML), is a wide topic in mobile robotics. The problem can be formulated as the requirement for a mobile robot to be placed in an unknown environment and progressively build a map while locating itself inside the map. Chapter 46 discusses this topic in detail. For the marine environment an additional issue arises due by the large-scale map that needs to be used for long-duration missions; [51.60] implements a decoupled stochastic mapping to handle this computational problem in an extended Kalman filter. Terrain-aided navigation with the use of a scanning sonar is implemented in [51.61]. *Newman* and *Leonard* [51.62] use long-baseline range measurements as the input for a nonlinear least-squares approach solved by the Gauss–Newton method; both the initially unknown position of the transponders and the vehicle position are estimated. An interesting survey on navigation and SLAM for underwater vehicles is given in [51.29].

51.2.8 Underwater Manipulation

A manipulator may be mounted on an AUV or a ROV in order to accomplish interaction operations. In this case, the vehicle needs to be fully actuated to counteract the forces and moments generated by the manipulator's base. By considering a manipulator with n links, thus six DOFs, the underwater vehicle manipulator system (UVMS) is a $(6+n)$-DOF robotic system whose velocity vector is

$$\boldsymbol{\zeta} = (\boldsymbol{\nu}_1^{\mathrm{T}} \ \boldsymbol{\nu}_2^{\mathrm{T}} \ \dot{\boldsymbol{q}}^{\mathrm{T}})^{\mathrm{T}} , \tag{51.33}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ is the vector collecting the manipulator joints positions.

Repeating the same considerations as for an underwater vehicle, it is possible to write the equations of motions of an UVMS in matrix form as

$$\mathbf{M}(\boldsymbol{q})\dot{\boldsymbol{\zeta}} + \mathbf{C}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \mathbf{D}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{g}(\boldsymbol{q}, \mathbf{R}_{\mathrm{B}}^{\mathrm{I}}) = \boldsymbol{\tau} , \tag{51.34}$$

where $\mathbf{M} \in \mathbb{R}^{(6+n)\times(6+n)}$ is the inertia matrix, including added mass terms, $\mathbf{C}(\boldsymbol{q}, \boldsymbol{\zeta})\boldsymbol{\zeta} \in \mathbb{R}^{6+n}$ is the vector of

the Coriolis and centripetal terms, $\mathbf{D}(\boldsymbol{q},\boldsymbol{\zeta})\boldsymbol{\zeta} \in \mathbb{R}^{6+n}$ is the vector of dissipative effects, and $\boldsymbol{g}(\boldsymbol{q},\mathbf{R}_1^{\mathrm{B}}) \in \mathbb{R}^{6+n}$ is the vector of gravity and buoyancy effects. The relationship between the generalized forces $\boldsymbol{\tau}$ and the control input is given by

$$\boldsymbol{\tau} = \begin{pmatrix} \boldsymbol{\tau}_{\mathrm{v}} \\ \boldsymbol{\tau}_q \end{pmatrix} = \begin{pmatrix} \mathbf{B}_{\mathrm{v}} & \mathbf{0}_{6\times n} \\ \mathbf{0}_{n\times 6} & \mathbf{I}_n \end{pmatrix} \boldsymbol{u} = \mathbf{B}\boldsymbol{u}\,, \qquad (51.35)$$

where $\boldsymbol{u} \in \mathbb{R}^{p_{\mathrm{v}}+n}$ is the vector of the control input. Notice that, while for the vehicle a generic number $p_{\mathrm{v}} \geq 6$ of control inputs is assumed, for the manipulator it is supposed that n joint motors are available.

Under this hypothesis, which can be considered as reasonable at low velocity, it holds that:

- The inertia matrix $\mathbf{M}$ of the system is symmetric and positive definite.
- For a suitable choice of the parametrization of $\mathbf{C}$ and if all the single bodies of the system are symmetric, $\dot{\mathbf{M}} - 2\mathbf{C}$ is skew symmetric.
- The matrix $\mathbf{D}$ is positive definite.

In [51.20] the mathematical model written with respect to the Earth-fixed-frame vehicle position and the manipulator end-effector can be found. However, it must be noted that, in this case, a six-dimensional manipulator is considered in order to have a square Jacobian to work with; moreover, kinematic singularities need to be avoided.

The equations of motion of UVMSs in matrix form presented in (51.34) are formally similar to the equations of motion of ground fixed manipulators (Chap. 3) for which a wide control literature exists. This has suggested a suitable translation/implementation of existing control algorithms. However, some differences, crucial from the control aspect, need to be underlined. UVMSs are complex systems characterized by several strong constraints:

- Uncertainty in the model knowledge, mainly due to the poor knowledge about the hydrodynamic effects
- The complexity of the mathematical model
- The kinematic redundancy of the system
- The difficulty in controlling the vehicle in hovering, mainly due to poor thruster performance
- The dynamic coupling between the vehicle and the manipulator
- The low bandwidth of the sensor readings.

In 1996 *McLain* et al. [51.63] presented a control law for UVMSs with some interesting experimental results conducted at the Monterey Bay Aquarium Research Institute (MBARI). A one-link manipulator was mounted on the OTTER vehicle controlled in all six DOFs by means of eight thrusters. A coordinated control system was then implemented to improve the tracking errors of the end effector.

Fig. 51.8 An underwater vehicle–manipulator system (courtesy of FP7-TRIDENT Project Consortium)

The monograph [51.64] is focused on the modeling and control issues of such systems, and can be considered as a reference for further reading. Moreover, interaction with the environment is also discussed.

Currently, remotely operated manipulator are standard equipment for several underwater ROVs. However, autonomous manipulation is still a research challenge. The SAUVIM vehicle, developed at the Autonomous Systems Laboratory, University of Hawaii, one of the first semiautonomous underwater vehicle manipulator systems. Similar research projects, ALIVE and TRIDENT were funded by the Framework Program of the European Community [51.2]. Figure 51.8 reports the UVMS developed under the TRIDENT project [51.3]. VIDEO 89 shows a sampling performed with the Nereus arm.

51.2.9 Fault Detection/Tolerance

Generally, AUVs must operate over long periods of time in unstructured environments in which an undetected failure could cause the loss of the vehicle. Failure detection and a fault-tolerant strategy are required to determine whether a mission must be terminated in the safest manner possible or if the vehicle can continue in a diminished capacity. An example is the case of the arctic mission of Theseus [51.65].

In the case of the use of ROVs, a skilled human operator is in charge of commanding the vehicle; a failure detection strategy is then of help in the human decision-making process. Based on the information detected, the operator can decide on vehicle rescue or to terminate the mission by, e.g., turning off a thruster.

Fault detection is the process of monitoring a system in order to recognize the presence of a failure; fault isolation or diagnosis is the capability to determine which specific subsystem is subject to failure. Often in the literature there is a certain overlap in the use of these terms. Fault tolerance is the capability to complete the mission in the case of the failure of one or more subsystems; it is also known as fault control, fault accommodation, or control reconfiguration. In the following the terms fault detection/tolerance will be used.

The characteristics of a fault detection scheme are the capability to isolate the detected failure, sensitivity in terms of the magnitude of the failure that can be detected, and robustness in the sense of the capability to continue working properly in non-nominal conditions. The requirements of a fault-tolerant scheme are reliability, maintainability, and survival. The common concept is that, to overcome the loss of capability due to a failure, a kind of redundancy is required in the system.

In this section, a survey of existing fault-detection and fault-tolerant schemes for underwater vehicles is presented. For these specific systems, if proper strategies are applied, a hardware/software (HW/SW) sensor or thruster failure can be successfully handled. In some conditions, the fault-detection scheme must also be able to diagnose some external abnormal working conditions such as a multipath phenomena affecting the echo-sounder system. It is worth noticing that, for autonomous systems such as AUVs, space systems or aircraft, a fault-tolerant strategy is necessary to safely recover the damaged vehicle and, obviously, there is no *panic button* in the sense that the choice of turning off the power or activating some kind of brake is not available.

Most fault-detection schemes are model based [51.66, 67] and consider the dynamic relationship between the actuators and vehicle behavior or the specific input–output thruster dynamics. In general, fault-detection/tolerance theory has been applied to the specific case of the underwater environment even if only a few papers report experimental results; see [51.68] for a survey on this topic.

Most fault-tolerant schemes consider a thruster-redundant vehicle that, after a fault has occurred in one of its thrusters, is still actuated in six DOFs. Based on this assumption a reallocation of the desired forces on the vehicle over the working thrusters is performed [51.69]. Of interest is also the study of reconfiguration strategies if the vehicle becomes underactuated.

Possible Failures

Underwater vehicles are currently equipped with several sensors in order to provide information about their localization and velocity. The problem is not easy. No single, reliable sensor is available that gives the required position/velocity measurement, or information about the environment such as the presence of obstacles. For this reason the use of sensor fusion by, e.g., a Kalman filtering approach, is a common technique to provide the controller with the required variables. This structural redundancy can be used to provide fault-detection capabilities to the system.

For each of the sensors listed in Sect. 51.2.2 failure can consist of an output of zero if, e.g., there is an electrical trouble, or a loss of meaning. It can be considered as sensor failure also an external disturbance such as a multipath reading of the sonar that can be interpreted as a sensor fault and correspondingly detected.

Thruster blocking occurs when a solid body is present between the propeller blades. It can be checked by monitoring the current required by the thruster. This was observed, e.g., during the Antarctic mission of Romeo [51.70], in that case caused by a block of ice. During the same mission a thruster also flooded with water. The consequence was an electrical dispersion, causing an increasing blade rotation velocity and thus a thruster force higher then desired.

A possible consequence of different failures of the thrusters is the zeroing of the blade rotation. The thruster in question thus simply stops working. This has been intentionally experienced during experiments with, e.g., ODIN [51.67, 69], Roby 2 [51.66], and Romeo [51.70].

Other failures include a hardware/software crash or the occurrence of fin sticking or loss. A very common type of failure involves the loss of electrical isolation due to seawater intrusion into underwater electrical cables or connectors. Such a condition can be detected through a technique called ground-fault monitoring. Should this occur, electrical power must be removed from the affected device.

51.2.10 Multiple Underwater Vehicles

A growing research effort has recently been devoted to developing strategies to design coordinated control for underwater vehicles. The use of multiple AUVs, in fact, might improve overall mission performance as well as provide greater tolerance to failures. Specific applications of this method in the underwater environment might include the naval mine countermeasure problem, harbor monitoring, and inspection, exploration, and mapping of large areas. AUVs might be coordinated with one or more surface vessels or connected to ground or aerial vehicles to form a coordinated network of heterogeneous autonomous robots.

Beside several institutions that have developed simulation packages for multiple-AUV operations, the use of real multiple AUVs is being considered for the adaptive sampling and forecasting plan of the Autonomous Ocean Sampling Network, formed by several research institutions such as (for the robotic components) Caltech, MBARI, Princeton, and WHOI [51.71]. Adaptive sampling is also investigated within the project Adaptive Sampling and Prediction (ASAP) [51.72]. The Australian National University is currently working on a shoal of small, autonomous robots, named Serafina [51.73]. At Instituto Superior Técnico (IST), work is ongoing on the coordination between an AUV and a catamaran [51.74], i. e., a multirobot system constituted by heterogeneous autonomous vehicles. In [51.75] the cooperation between AUVs and a sensor network is experimentally investigated. VIDEO 323 and VIDEO 94 show two examples of multi-vehicle bathymetry mission and multi-vehicle patrolling, respectively.

51.3 Applications

Underwater robots currently play prominent roles in a number of scientific, commercial, and military tasks. Remotely teleoperated vehicles are very well established in all these areas, and are becoming increasingly automated to relieve the burden on human operators and to improve performance. Increasingly, autonomous underwater vehicles are finding application in these areas as well. Presently, AUVs are used almost exclusively for survey work, but sampling and other intervention tasks are becoming more feasible. Additionally, the line between ROVs and AUVs continues to blur, as systems that have the best properties of both evolve.

The offshore oil and gas industry relies heavily on ROVs for installation, inspection, and servicing of platforms, pipelines, and subsea production facilities. As the search for oil and gas goes deeper, this trend can only continue. The Marine Technology Society estimates that there are over 435 *work-class* ROVs operating in the commercial offshore industry today. AUVs are now beginning to appear in the commercial offshore industry for survey tasks, and concepts for hybrid systems that can perform intervention tasks are now appearing. The goal is not only for these robotic vehicles to replace human divers or human-occupied vehicles, but to enable an entire new generation of subsea equipment that is serviced without intervention by drill ships or other heavy-lifting vessels. This holds the prospect of greatly reduced cost.

Scientific demand for ROVs and AUVs is also increasing dramatically. Scientific applications for ROVs include survey, inspection, and sampling tasks previously performed by human-occupied submersibles or towed vehicles. While ROVs operating for science are not nearly as numerous as those in the offshore oil and gas industry, they are becoming commonplace. Most nations involved in global seafloor studies have several vehicles. Like the vehicles for the commercial offshore sector, these vehicles are becoming increasingly automated. High-quality electronic imaging, including high-definition television, is becoming increasingly common. Scientific ROVs are now equipped with sophisticated sampling devices for sampling animals, microbes, caustic hydrothermal vent fluids, and a variety of rock samples. Moreover, ROVs are also used to deploy and operate seafloor experiments, which can involve difficult tasks such as drilling and delicate placement of instruments.

ROVs have also emerged as powerful tools for investigating underwater shipwrecks and other cultural sites. Applications include forensic investigations of modern shipwrecks to determine the cause of sinking, archaeology, and salvage. For archaeology, the goals are the same as for excavation on land: detailed mapping followed by careful excavation. Beyond diver depths, ROVs are the preferred method for these investigations. Great progress has been made in the detailed mapping phase, and capabilities for excavation are evolving. Unfortunately, the same technology also opens the possibility for shipwrecks to be looted for financial gain, which usually results in the loss of the most valuable historical information.

After a long period of skepticism, AUVs are now accepted for scientific tasks. Presently, AUVs most often perform mapping tasks while tended by a vessel. Specific mapping tasks include seafloor bathymetry, sidescan sonar imaging, magnetic field mapping, hydrothermal vent localization, and photo surveys. AUVs have been shown to improve productivity and data quality compared to towed and tethered systems. They have also operated in environments where no other means of gathering data is possible, such as under ice shelves. Likewise, the increasing availability of sophisticated in situ chemical sensors, biological sensors, and mass spectrometers now allows AUVs to build spatial and temporal maps of environmental features that could pre-

viously only be studied by bringing samples back to the laboratory. AUV systems that can dock to subsea nodes to recharge batteries, offload data and receive new instructions are presently being tested.

The military has always been a leader in the development of underwater robotic capabilities. They pioneered ROVs for tasks such as recovering test weapons and deep-sea salvage, and present-day commercial and scientific ROVs have descended directly from these early systems. Likewise, military interests are presently pushing AUV technology very hard. Many different countries operate AUVs for military surveys, gathering environmental data as well as searching for hazards such as mines. An operational success was achieved in surveying for mines in the Persian Gulf harbor of Umm Qasr using REMUS vehicles. AUVs in development will not only be able to detect mines, but to disable them. Bolder, more innovative concepts are also in development. These include networks of AUVs that can act as extensions of conventional surface vessels and submarines, enabling surveillance over wide areas for extended periods of time at costs far less than could be achieved with conventional surface vessels, submarines, and aircraft. These developments will rely on improvements in acoustic communications, energy systems, sensors, and onboard intelligence that will likely find their way into commercial and scientific practice.

51.4 Conclusions and Further Reading

The underwater environment is extremely hostile for human engineering activities. In addition to high pressures and hydrodynamic forces that are both nonlinear and unpredictable, water is not an appropriate media for electromagnetic communication except at short ranges. This pushes underwater technology to rely on acoustic communication and positioning systems that are characterized by low bandwidth. On the other hand, the ocean is extremely important for numerous human activities from the commercial, cultural, and environmental points of view.

Research on underwater robotic applications is active both from the technological and methodological aspects. The power endurance of commercially AUVs is currently up to 50 h; this will increase as energy-storage devices improve. Improved energy and power capability will enable longer missions, higher speeds, or better/additional sensors such as, e.g., more powerful lighting for underwater video/photography. The current trend for the price of AUVs prices is downward, with more and smaller research institutions building or buying AUVs to enrich their research results; moreover the setup of multiple-AUV systems is becoming cost effective. The goal is to develop fully autonomous, reliable, robust, decision-making AUVs.

There are a number of technology issues that are needed in order to improve AUV capabilities: to increase the underwater bandwidth of current acoustic modems, to increase onboard power to handle larger tools and interact more strongly with the environment, to create AUVs with significant hovering capability to allow better interaction, and to enable easier launch and recovery.

In the near future, the ROV/AUV dichotomy will likely become less prominent, with a variety of systems appearing that have attributes of both systems:

- For offshore oil and gas intervention tasks, a vehicle could transit to the work site as a self-powered, fully autonomous vehicle, then dock to the work site. Utilizing energy and communications infrastructure at the work site, the vehicle could then be operated much like a conventional ROV.
- Battery-operated ROVs can communicate with the surface by very lightweight fiber-optic links, enabling the mobility of an AUV but with a high-bandwidth connection to skilled human operators for complex intervention or scientific sampling tasks.
- Acoustic and optical data links can provide moderate to high communication bandwidths over short ranges, enabling human supervision without any tether restrictions. At longer ranges, more modest acoustic bandwidths are available.

These developments make marine robotics a challenging engineering problem with strong connections to several engineering domains. Sending an autonomous vehicle into an unknown and unstructured environment with limited online communication requires some onboard intelligence and the ability for the vehicle to react in a reliable way to unexpected situations.

A major challenge concerning underwater robotics is the interaction with the environment by means of one or more manipulators. Autonomous UVMSs are still the object of research; the current trend is in developing the first semiautonomous robotic devices, which

might be acoustically operated; moreover, if physically possible, the capability to dock to the structure where the intervention is needed might significatively simplify the control. The final aim might be to develop a completely autonomous UVMS, able to localize the intervention site, recognize the task to be performed, and act on it without docking to the station and without human intervention. This might make it possible to perform missions that are currently impossible such as autonomous archaeological intervention at deep sites. This would also enable the oil and gas industry to significatively decrease costs and risks to humans.

The Marine Systems Simulator [51.76] is a Matlab/Simulink library and simulator for marine systems. It includes models for ships, underwater vehicles, and floating structures. The library also contains guidance, navigation, and control (GNC) blocks for real-time simulation. A numerical simulator including also the presence of a manipulator has been developed under the TRIDENT project [51.77].

For further reading on the topic of underwater systems, the reader is referred to several survey articles, including [51.6, 29, 35, 38, 68]. Additionally, several journals cover oceanic engineering topics, including robotics aspects. A variety of symposia and workshops have been held on a regular basis. Some books/monographs treating marine robotics are [51.4, 8, 9, 17, 64].

Video-References

VIDEO 87 Dive with REMUS
available from http://handbookofrobotics.org/view-chapter/51/videodetails/87

VIDEO 88 Underwater vehicle Nereus
available from http://handbookofrobotics.org/view-chapter/51/videodetails/88

VIDEO 89 Mariana Trench: HROV Nereus samples the Challenger Deep seafloor
available from http://handbookofrobotics.org/view-chapter/51/videodetails/89

VIDEO 90 REMUS SharkCam: The hunter and the hunted
available from http://handbookofrobotics.org/view-chapter/51/videodetails/90

VIDEO 92 The Icebot
available from http://handbookofrobotics.org/view-chapter/51/videodetails/92

VIDEO 94 Two underwater Folaga vehicles patrolling a 3-D area
available from http://handbookofrobotics.org/view-chapter/51/videodetails/94

VIDEO 267 Adaptive L1 depth control of a ROV
available from http://handbookofrobotics.org/view-chapter/51/videodetails/267

VIDEO 268 Saturation based nonlinear depth and yaw control of an underwater vehicle
available from http://handbookofrobotics.org/view-chapter/51/videodetails/268

VIDEO 323 Multi-vehicle bathymetry mission
available from http://handbookofrobotics.org/view-chapter/51/videodetails/323

VIDEO 324 Neptus command and control infrastructure
available from http://handbookofrobotics.org/view-chapter/51/videodetails/324

References

51.1 J. Yuh, S.K. Choi, C. Ikehara, G.H. Kim, G. McMurty, M. Ghasemi-Nejhad, N.N. Sarkar, K. Sugihara: Design of a semi-autonomous underwater vehicle for intervention missions (SAUVIM), IEEE Int. Symp. Underw. Technol. (1998) pp. 63–68

51.2 P. Marty: ALIVE: An autonomous light intervention vehicle, Adv. Technol. Underw. Veh. Conf., Oceanol. Int. (2004)

51.3 M. Prats, J.C. Garcia, S. Wirth, D. Ribas, P.J. Sanz, P. Ridao, N. Gracias, G. Oliver: Multipurpose autonomous underwater intervention: A systems integration perspective, 20th IEEE Mediterr. Conf. Contr. Autom., Barcelona (2012) pp. 1379–1484

51.4 T.I. Fossen: *Handbook of Marine Craft Hydrodynamics and Motion Control* (Wiley, New York 2011)

51.5 S. Bennett: A brief history of automatic control, IEEE Control Syst. Mag. **16**(3), 17–25 (1996)

51.6 J. Yuh, M. West: Underwater robotics, J. Adv. Robotics **15**(5), 609–639 (2001)

51.7 SNAME: *Nomenclature for Treating the Motion of a Submerged Body Through a Fluid*, Techn. Res. Bull. (SNAME, New York 1952) pp. 1–5

51.8 T.I. Fossen: *Guidance and Control of Ocean Vehicles* (Wiley, New York 1994)

51.9 O.M. Faltinsen: *Sea Loads on Ships and Offshore Structures* (Cambridge Univ. Press, Cambridge 1990)

51.10 J. Yuh: Modeling and control of underwater robotic vehicles, IEEE Trans. Syst. Man Cybern. **20**, 1475–1483 (1990)

51.11 T.I. Fossen, A. Ross: *Guidance and Control of Unmanned Marine Vehicles*, IEEE Control Engineering (Wiley, Chichester 1999) pp. 23–42

51.12 W.E. Cummins: *The impulse response function and ship motions, Techn. Rep. 1661* (DTIC, Washington 1962)

51.13 T.F. Ogilvie: Recent progress towards the understanding and prediction of ship motions, 5th Symp. Nav. Hydrodyn. (1964) pp. 3–79

51.14 T. Perez, T.I. Fossen: Time-domain models of marine surface vessels for simulation and control design based on seakeeping computations, 7th Conf. Manoeuvring Control Mar. Craft, (IFAC) (2006)

51.15 T.I. Fossen: A nonlinear unified state-space model for ship maneuvering and control in a seaway, J. Bifurc. Chaos **15**(9), 2717–2746 (2005)

51.16 M. Nahon: Determination of undersea vehicle hydrodynamic derivatives using the USAF, Datcom, Proc. Ocean. Conf., Victoria (1993) pp. 283–288

51.17 J.N. Newman: *Marine Hydrodynamics* (MIT, Cambridge 1977)

51.18 A.J. Healey, D. Lienard: Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles, IEEE J. Ocean. Eng. **18**, 327–339 (1993)

51.19 B. Stevens, F. Lewis: *Aircraft Control and Simulations* (Wiley, New York 1992)

51.20 I. Schjølberg, T.I. Fossen: Modelling and control of underwater vehicle-manipulator systems, 3rd Conf. Manoeuvring Control Mar. Craft (IFAC), Southampton (1994) pp. 45–57

51.21 E.A. de Barros, A. Pascoal, E. de Sea: Progress towards a method for predicting AUV derivatives, 7th Conf. Manoeuvring Control Mar. Craft (IFAC), Lisbon (2006)

51.22 Monterey Bay Aquarium Research Institute: http://www.mbari.org

51.23 N.P. Fofonoff, R.C. Millard: *Algorithms for Computation of Fundamental Properties of Seawater, UNESCO Tech. Pap. Mar. Sci. No. 44* (UNESCO, Paris 1983)

51.24 Autonomous Systems Laboratory, University of Hawaii: http://www.eng.hawaii.edu/~asl/

51.25 R. Eustice, H. Singh, J.J. Leonard, M. Walter: Visually mapping the RMS Titanic: Conservative covariance estimates for SLAM information filters, Int. J. Robotics Res. **25**(12), 1223–1242 (2006)

51.26 D.A. Smallwood, L.L. Whitcomb: Adaptive identification of dynamically positioned underwater robotic vehicles, IEEE Trans. Control Syst. Technol. **11**(4), 505–515 (2003)

51.27 S. Zhao, J. Yuh: Experimental study on advanced underwater robot control, IEEE Trans. Robotics **21**(4), 695–703 (2005)

51.28 S. Majumder, S. Scheding, H.F. Durrant-Whyte: Multisensor data fusion for underwater navigation, Robotics Auton. Syst. **35**(2), 97–108 (2001)

51.29 J.C. Kinsey, R.M. Eustice, L.L. Whitcomb: A survey of underwater vehicle navigation: Recent advances and new challenges, 7th Conf. Manoeuvring Control Mar. Craft (IFAC), Lisbon (2006)

51.30 D.R. Yoerger, J.G. Cooke, J.J. Slotine: The influence of thruster dynamics on underwater vehicle behavior and their incorporation into control system design, IEEE J. Ocean. Eng. **15**, 167–178 (1990)

51.31 A.J. Healey, S.M. Rock, S. Cody, D. Miles, J.P. Brown: Toward an improved understanding of thruster dynamics for underwater vehicles, IEEE J. Ocean. Eng. **20**(4), 354–361 (1995)

51.32 L. Bachmayer, L.L. Whitcomb, M.A. Grosenbaugh: An accurate four-quadrant nonlinear dynamical model for marine trhusters: Theory and experimental validation, IEEE J. Ocean. Eng. **25**, 146–159 (2000)

51.33 T.I. Fossen, M. Blanke: Nonlinear output feedback control of underwater vehicle propellersusing feedback form estimated axial flow velocity, IEEE J. Ocean. Eng. **25**(2), 241–255 (2000)

51.34 W.P.A. Van Lammeren, J. van Manen, M.W.C. Oosterveld: The wageningen B-screw series, Trans. SNAME **77**, 269–317 (1969)

51.35 T.I. Fossen, T.I. Johansen: A survey of control allocation methods for ships and underwater vehicles, 14th IEEE Mediterr. Conf. Control Autom., Ancona (2006) pp. 1–6

51.36 M. Stojanovic, J. Preisig: Underwater acoustic communication channels: Propagation models and statistical characterization, IEEE Commun. Mag. **47**(1), 84–89 (2009)

51.37 D. Pompili, I. Akyildiz: Overview of networking protocols for underwater wireless communications, IEEE Commun. Mag. **47**(1), 97–102 (2009)

51.38 K.P. Valavanis, D. Gracanin, M. Matijasevic, R. Kolluru: Control architecture for autonomous underwater vehicles, IEEE Control Syst. **17**, 48–64 (1997)

51.39 P. Oliveira, A. Pascoal, V. Silva, C. Silvestre: Mission control of the MARIUS AUV: System design, implementation, and sea trials, Int. J. Syst. Sci. **29**(10), 1065–1080 (1998)

51.40 D. Brutzman, M. Burns, M. Campbell, D. Davis, T. Healey, M. Holden, B. Leonhardt, D. Marco, D. McClarin, B. McGhee: NPS Phoenix AUV software integration and in-water testing, Proc. IEEE Symp. Auton. Underw. Veh. Technol. (AUV) (1996) pp. 99–108

51.41 A.P. Aguiar, A.M. Pascoal: Dynamic positioning and way-point tracking of underactuated AUVs in the presence of ocean currents, Int. J. Control **80**(7), 1092–1108 (2007)

51.42 A.A. Bennett, J.J. Leonard: A behavior-based approach to adaptive feature detection and following with autonomous underwater vehicles, IEEE J. Ocean. Eng. **25**(2), 213–226 (2000)

51.43 M. Breivik, T.I. Fossen: Principles of guidance-based path following in 2D and 3D, 44th IEEE Conf. Decis. Control 8th Eur. Control Conf., Sevilla (2005)

51.44 F.A. Papoulias: Bifurcation analysis of line of sight vehicle guidance using sliding modes, Int. J. Bifurc. Chaos **1**(4), 849–865 (1991)

51.45 R. Rysdyk: UAV path following for constant line-of-sight, Proc. 2nd AIAA Unmanned Unltd. Syst. Technol. Oper. Aerosp., San Diego (2003)

51.46 M.D. Feezor, F.Y. Sorrel, P.R. Blankinship, J.G. Bellingham: Autonomous underwater vehicle homing/docking via electromagnetic guidance, IEEE J. Ocean. Eng. **26**(4), 515–521 (2001)

51.47 D. Wettergreen, A. Zelinsky, C. Gaskett: Autonomous guidance and control for an underwater robotic vehicle, Int. Conf. Field Serv. Robotics (1999)

51.48 G. Antonelli, S. Chiaverini, R. Finotello, R. Schiavon: Real-time path planning and obstacle avoidance for RAIS: An autonomous underwater vehicle, IEEE J. Ocean. Eng. **26**(2), 216–227 (2001)

51.49 J.C. Hyland, F.J. Taylor: Mine avoidance techniques for underwater vehicles, IEEE J. Ocean. Eng. **18**, 340–350 (1993)

51.50 D.R. Yoerger, J.J. Slotine: Robust trajectory control of underwater vehicles, IEEE J. Ocean. Eng. **10**, 462–470 (1985)

51.51 R. Cristi, F.A. Pappulias, A. Healey: Adaptive sliding mode control of autonomous underwater vehicles in the dive plane, IEEE J. Ocean. Eng. **15**(3), 152–160 (1990)

51.52 D.B. Marco, A.J. Healey: Command, control and navigation experimental results with the NPS ARIES AUV, IEEE J. Ocean. Eng. **26**(4), 466–476 (2001)

51.53 M. Caccia, G. Veruggio: Guidance and control of a reconfigurable unmanned underwater vehicle, Control Eng. Prac. **8**(1), 21–37 (2000)

51.54 G. Antonelli, F. Caccavale, S. Chiaverini, G. Fusco: A novel adaptive control law for underwater vehicles, IEEE Trans. Control Syst. Technol. **11**(2), 221–232 (2003)

51.55 G. Antonelli: On the use of adaptive/integral actions for 6-degrees-of-freedom control of autonomous underwater vehicles, IEEE J. Ocean. Eng. **32**(2), 300–312 (2007)

51.56 M. Erol-Kantarci, H.T. Mouftah, S. Oktug: A survey of architectures and localization techniques for underwater acoustic sensor networks, IEEE Commun. Surv. Tutor. **13**(3), 487–502 (2011)

51.57 A. Bahr, J.J. Leonard, M.F. Fallon: Cooperative localization for autonomous underwater vehicles, Int. J. Robotics Res. **28**(6), 714–728 (2009)

51.58 S.B. Williams, I. Mahon: A terrain-aided tracking algorithm for marine systems. In: *Field and Service Robotics*, (Springer, Berlin, Heidelberg 2006) pp. 93–102

51.59 M. Dunbabin, P. Corke, G. Buskey: Low-cost vision-based AUV guidance system for reef navigation, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2004) pp. 7–12

51.60 J.J. Leonard, H.J.S. Feder: Decoupled stochastic mapping, IEEE J. Ocean. Eng. **26**(4), 561–571 (2001)

51.61 S. Williams, G. Dissanayake, H. Durrant-Whyte: Towards terrain-aided navigation for underwater robotics, Adv. Robotics **15**(5), 533–549 (2001)

51.62 P. Newman, J. Leonard: Pure range-only sub-sea SLAM, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2003) pp. 1921–1926

51.63 T.W. McLain, S.M. Rock, M.J. Lee: Experiments in the coordinated control of an underwater arm/vehicle system, Auton. Robots **3**(2), 213–232 (1996)

51.64 G. Antonelli (Ed.): *Underwater Robots. Motion and Force Control of Vehicle-Manipulator Systems*, Springer Tracts in Advanced Robotics (Springer, Berlin, Heidelberg 2014), 3rd edn.

51.65 J.S. Ferguson, A. Pope, B. Butler, R. Verrall: Theseus AUV – Two record breaking missions, Sea Technol. Mag. **40**, 65–70 (1999)

51.66 A. Alessandri, M. Caccia, G. Veruggio: Fault detection of actuator faults in unmanned underwater vehicles, Control Eng. Prac. **7**, 357–368 (1999)

51.67 K.C. Yang, J. Yuh, S.K. Choi: Fault-tolerant system design of an autonomous underwater vehicle – ODIN: An experimental study, Int. J. Syst. Sci. **30**(9), 1011–1019 (1999)

51.68 G. Antonelli: A survey of fault detection/tolerance strategies for AUVs and ROVs. Recent advances. In: *Fault Diagnosis and Tolerance for Mechatronic Systems*, Springer Tracts in Advanced Robotics, ed. by F. Caccavale, L. Villani (Springer, Berlin, Heidelberg 2002) pp. 109–127

51.69 T.K. Podder, G. Antonelli, N. Sarkar: An experimental investigation into the fault-tolerant control of an autonomous underwater vehicle, J. Adv. Robotics **15**, 501–520 (2001)

51.70 M. Caccia, R. Bono, G. Bruzzone, G. Bruzzone, E. Spirandelli, G. Veruggio: Experiences on actuator fault detection, diagnosis and accomodation for ROVs, Int. Symp. Unmanned Untethered Submers. Technol. (2001)

51.71 E. Fiorelli, P. Bhatta, N.E. Leonard, I. Shulman: Adaptive sampling using feedback control of an autonomous underwater glider fleet, Int. Symp. Unmanned Untethered Submers. Technol. (2003)

51.72 N.E. Leonard, D.A. Paley, R.E. Davis, D.M. Fratantoni, F. Lekien, F. Zhang: Coordinated control of an underwater glider fleet in an adaptive ocean sampling field experiment in Monterey Bay, J. Field Robotics **27**(6), 718–740 (2010)

51.73 S. Kalantar, U. Zimmer: Distributed shape control of homogeneous swarms of autonomous underwater vehicles, Auton. Robots **22**(1), 37–53 (2006)

51.74 A. Pascoal, C. Silvestre, P. Oliveira: Vehicle and mission control of single and multiple autonomous marine robots. In: *Advances in Unmanned Marine Vehicles*, IEEE Control Engineering, ed. by G. Roberts, R. Sutton (Peregrinus, New York 2006) pp. 353–386

51.75 M. Dunbabin, I. Vasilescu, P. Corke, D. Rus: Experiments with cooperative networked control of underwater robots. In: *Experimental Robotics*, (Springer, Berlin, Heidelberg 2008) pp. 463–470

51.76 Marine Systems Simulator: http://www.marinecontrol.org

51.77 TRIDENT project: http://www.irs.uji.es/uwsim

Multimedia Contents

52. Modeling and Control of Aerial Robots

Robert Mahony, Randal W. Beard, Vijay Kumar

Aerial robotic vehicles are becoming a core field in mobile robotics. This chapter considers some of the fundamental modelling and control architectures in the most common aerial robotic platforms; small-scale rotor vehicles such as the quadrotor, hexacopter, or helicopter, and fixed wing vehicles. In order to control such vehicles one must begin with a good but sufficiently simple dynamic model. Based on such models, physically motivated control architectures can be developed. Such algorithms require realisable target trajectories along with real-time estimates of the system state obtained from on-board sensor suite. This chapter provides a first introduction across all these subjects for the quadrotor and fixed wing aerial robotic vehicles.

52.1 **Overview** 1307
52.2 **Modeling Aerial Robotic Vehicles** 1309
52.2.1 Rigid Body Motion of the Airframe 1309
52.2.2 Modeling for Quadrotors 1310
52.2.3 Modeling of Fixed-Wing Airplanes 1312
52.3 **Control** 1316
52.3.1 Quadrotor Control 1316
52.3.2 Control of Fixed-Wing Aircraft 1319
52.4 **Trajectory Planning** 1324
52.4.1 Trajectory Planning for Quadrotors 1324
52.4.2 Trajectory Planning for Fixed-Wing Aircraft 1325
52.5 **Estimating the Vehicle State** 1328
52.5.1 Estimating Attitude 1328
52.5.2 Estimating Velocity and Position 1329
52.6 **Conclusion** 1330
Video-References 1331
References 1331

52.1 Overview

The term *aerial robotics* is often attributed to *Robert Michelson* [52.1], as a term to capture a new class of autonomous and intelligent small flying machines. The quest for autonomy of aerial vehicles goes back almost to the origins of powered flight. The first autopilot for a fixed wing vehicle (the Sperry autopilot) was demonstrated in 1912, only a decade after the first powered flight by the Wright brothers. As early as World War I, the potential for an autonomously stabilized and remotely controlled airplane to be an effective weapon was understood and the Curtis–Sperry flying bomb first demonstrated autonomous unmanned flight in 1918 [52.2]. Autopilot technology was perfected during the 1930s as improving aircraft technology lead to longer flight times and the need to relieve the pilot from constant attention to flight stability of the vehicle. Autonomous and radio controlled target aircraft were developed in the late 1930s and were extensively used during World War II for training. The German V-1 cruise missile, commonly known as the buzz bomb or doodlebug, was a highly successful autonomous vehicle; its relatively low impact in World War II being due to more British intelligence efforts than to the vehicles capabilities.

Following the World War II, the relative maturity of autopilot technology and airplane construction meant that the construction of unmanned fixed-wing vehicles was straightforward. Remote-controlled heli-

copters, such as the Gyrodyne QH-50 DASH, were developed as early as the 1960s. The Yamaha R-50 and subsequent Yamaha R-Max (Fig. 52.1), was developed during the 1980s and provides a remote controlled commercial aerial platform, with significant onboard autonomy, used primarily for agricultural applications.

Extensive development of fixed-wing autonomous vehicles was held back in the mid-20th century by the difficulty of localizing a vehicle when far from the base station. Indeed, the requirements of navigation systems for cruise missiles developed during the cold war was one of the main drivers for the development of satellite global positioning system (GPS) technology. Full operational capability of the United States GPS system with 24 satellites was declared in April 1995 and it was mandated as a dual-use system (both commercial and military) in 1996. Smaller scale commercial unmanned aerial vehicles were also constrained by the lack of robust and small-scale avionics systems. The availability of small low power computers, the development of microelectromechanical systems (MEMS) that provide affordable and robust Inertial Measurement Units (IMUs), and access to reliable GPS, lead to the beginning of the modern era of nonmilitary UAV and aerial robotic systems in the mid-1990s.

The largest class of commercial and military robotic aerial vehicles fly predominantly in obstacle free airspace. For such vehicles, once the aircraft has taken off, there is no need to employ obstacle avoidance techniques or interact with cluttered three-dimensional environments.

Definition 52.1

An *unmanned aerial vehicle* (UAV) is a system capable of sustained flight with no direct human control and able to perform a specific task.

Fig. 52.1 Autonomous aerial robotic vehicle based on the Yamaha RMAX platform developed in the University of New South Wales, Australia (after [52.3, 4])

Navigation and control of an unmanned aerial vehicle is typically based on stabilizing a reference heading and altitude derived from the error between preset GPS way-points and the present vehicle position using a classical autopilot control system. The sensor suite used for control and navigation is usually a GPS and IMU along with barometric pressure. These sensor suites can be thought of as *proprioceptive* sensors, since they measure the internal state of the vehicle without reference to the external world. The kinds of tasks that UAVs are ideal for include high level surveillance and sensing tasks where the payload sensing suite is separate from the vehicle systems, with applications in agriculture, environmental monitoring, geophysical surveys, search and rescue, and security surveillance, as well as deployment of materials, where once again the payload is separate from the vehicle system, with applications in agriculture, search and rescue, and of course military missions. Such systems are a natural development of traditional flight technology and much of the research and development that goes into these vehicles is undertaken in aerospace departments in universities and aerospace companies.

Definition 52.2

An *aerial robotic vehicle* is an aerial vehicle capable of autonomously interacting with a complex dynamic three-dimensional environment and achieving complex environment-dependent goals.

The nature of the interaction with the environment for an aerial robotic vehicle, means that the presence of *exteroceptive* sensors, those that sense the environment around the vehicle, and how these sensor inputs are integrated into the vehicle guidance is the defining property of such a vehicle. Typical examples of exteroceptive sensors for aerial robotic vehicles are vision systems, laser range finders, acoustic sensors, etc., and external sensor systems such as VICON [52.5] and Optitrack [52.6] systems. The close coupling between the goal of the vehicle, its sensing suite, and the dynamic nature of the environment makes the simple way-point navigation control architecture of an unmanned aerial vehicle unsuitable for aerial robotic vehicles. The kinds of tasks that aerial robotic vehicles are ideal for are small-scale interactive tasks such as inspection of civil infrastructure including dam walls, girders of bridges, industrial pressure vessels; and surveillance tasks such as, inspection of damaged or burning buildings, monitoring of crowds, etc. Future applications may involve aerial manipulation including repair of infrastructure, and material handling in the construction and agricultural industries. Although the flight technology remains important, the sensing and control tasks become cru-

cial in such applications and the development that goes into these vehicles tends to be undertaken in robotics departments in universities and in the growing field of new aerial-robotics start-up companies. The ability to move in three-dimensional space brings new research challenges to the robotics community compared to the wheeled mobile robot technology that has motivated mobile robotics research over the last couple of decades.

The term *unmanned aerial system* (UAS) is useful to refer to the combination of the infrastructure, human interfaces, aerial platform, and sensing and control subsystems. It is clear that certain vehicles may spend time operating in either UAV mode or aerial robotic mode and the terms are more useful to describe the way of thinking about the operation of a system rather than classifying a vehicle.

52.2 Modeling Aerial Robotic Vehicles

There is already a vast range of aerial vehicle and UAV designs in existence, and there are even more ideas under development by companies and research groups as we write. In one short chapter, we must limit the scope of our presentation to a couple of the key examples. There are two major categories of vehicles that are used as platforms for aerial robotics: small-scale rotor vehicles such as the quadrotor, hexacopter, or helicopter, and fixed wing vehicles. The word *quad* is derived from the Latin *quadrangulum* for a four-sided figure. *Rotor* is derived from the Latin *rotationem* for a rotating object. However, *hexa* is derived directly from Greek for six. *Heli* is derived from ancient Greek *helikos* for spiral, while the term *copter* is derived from the modern term *helicopter*, a combination of *helikos* with the ancient Greek *pteron* for wing. Based on this etymology, we propose the terms *quadrotor* and *hexacopter* to describe the most common modern rotary wing aerial robotic vehicles in preference to the etymological questionable terms quadcopter and hexarotor that are also in common usage in the literature. We will present modeling and control material relevant to these two categories, focusing on the basic structure of the model. We will also focus on vehicles in 500 g–4 kg weight range as this corresponds to the primary general *robotics* applications where avionics and sensor suites take up the majority of the payload. Other classes of autonomous aerial vehicles that we are unable to cover in this chapter include lighter-than-air vehicles such as blimps and balloons, flapping wing vehicles, ducted fans, and rockets.

52.2.1 Rigid Body Motion of the Airframe

Heavier-than-air aerial robotic vehicles consist of a rigid airframe coupled with aerodynamic mechanisms to generate lift and thrust. For vehicles in the weight range considered, the compactness and structural integrity of the airframe means that there is little flexing of the airframe in normal operation and the rigid-body assumption is an effective model. In this case, the vehicle model can be developed based on rigid-body dynamics with exogenous forces and torque generated by an aerodynamic model.

Let $\{e_1, e_2, e_3\}$ be the coordinate axis unit vectors $e_1 = (1, 0, 0)^{\mathrm{T}}$, $e_2 = (0, 1, 0)^{\mathrm{T}}$, and $e_3 = (0, 0, 1)^{\mathrm{T}}$ without specification of a frame of reference. Let $\{B\}$ be a (right-hand) body fixed frame for the airframe with unit vectors $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$ where these vectors are the axes of frame $\{B\}$ with respect to frame $\{A\}$ as shown in Fig. 52.2. Note that the convention for choice of axes shown in Fig. 52.2 is the convention common in the aerial robotics community [52.7–14], but is opposite to the usual convention in the aerospace community where the $\boldsymbol{b}_2$ and $\boldsymbol{b}_3$ axes are usually reversed in order so that $\boldsymbol{b}_3$ points down in the direction of gravity. In this chapter, we have chosen to follow the robotic community convention and this will lead to some nonconventional definitions for the modeling of fixed-wing airplanes. We will discuss this further in the specific sections on fixed-wing airplane modeling and control.

The orientation of the rigid body is given by a rotation matrix ${}^{A}R_{B} = R = [\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3] \in \mathrm{SO}(3)$ in the spe-

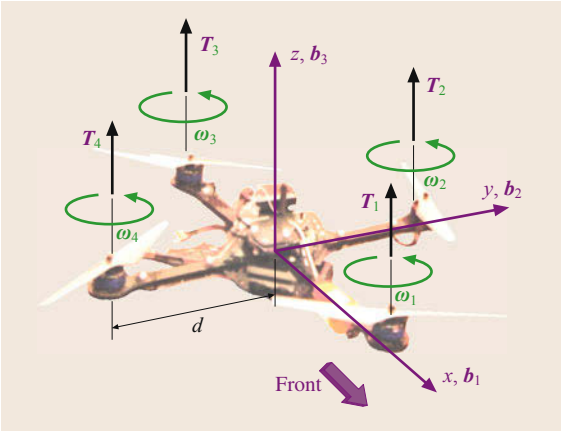


Fig. 52.2 The body-fixed frame and the directions of rotation for the propellers

cial orthogonal group. We have $\boldsymbol{b}_1 = Re_1$, $\boldsymbol{b}_2 = Re_2$, $\boldsymbol{b}_3 = Re_3$ by construction.

We will use *Z-X-Y* Euler angles to model this rotation as shown in Fig. 52.3. Note that this is not the normal roll-pitch-yaw convention used in aerospace. To get from $\{A\}$ to $\{B\}$, we first rotate about e_3 by the yaw angle, ψ, and we will call this intermediary frame $\{D\}$ with a basis $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3\}$ where $\boldsymbol{d}_i$ is expressed with respect to frame $\{A\}$. This is followed by a rotation about the x-axis in the rotated frame through the roll angle, ϕ, to intermediary frame $\{E\}$, followed by a third pitch rotation about the new y-axis through the pitch angle θ which results in the body-fixed triad $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$. The associated rotation matrix is

$$R = \begin{pmatrix} c_\psi c_\theta - s_\phi s_\psi s_\theta & -c_\phi s_\psi & c_\psi s_\theta + c_\theta s_\phi s_\psi \\ c_\theta s_\psi + c_\psi s_\phi s_\theta & c_\phi c_\psi & s_\psi s_\theta - c_\psi c_\theta s_\phi \\ -c_\phi s_\theta & s_\phi & c_\phi c_\theta \end{pmatrix} . \tag{52.1}$$

where c and s are shorthand for cosine and sine, respectively.

Let ξ denote the position of the center of mass of the airframe in frame $\{A\}$ and assume that this is also the origin of frame $\{B\}$. Let $v \in \{A\}$ denote the linear velocity of $\{B\}$ with respect to $\{A\}$ expressed in $\{A\}$. Let $\Omega \in \{B\}$ denote the angular velocity of $\{B\}$ with respect to $\{A\}$, this time expressed in $\{B\}$. Let m denote the mass of the rigid object and $\mathbf{I} \in \mathbb{R}^{3\times3}$ denote the constant inertia matrix (expressed in the body fixed frame $\{B\}$). The rigid body equations of motion of the airframe are [52.15, 16]

$$\dot{\xi} = v \,, \tag{52.2a}$$

$$m\dot{v} = -mge_3 + RF \,, \tag{52.2b}$$

$$\dot{R} = R\Omega_\times \,, \tag{52.2c}$$

$$\mathbf{I}\dot{\Omega} = -\Omega \times \mathbf{I}\Omega + \tau \,. \tag{52.2d}$$

The notation $\Omega_\times$ denotes the skew-symmetric matrix such that $\Omega_\times v = \Omega \times v$ for the vector cross-product $\times$ and any vector $v \in \mathbb{R}^3$. The vectors $F, \tau \in \{B\}$ combine the principal nonconservative forces and torques applied to the airframe by the aerodynamics of the propulsion systems and lifting surfaces of the vehicle.

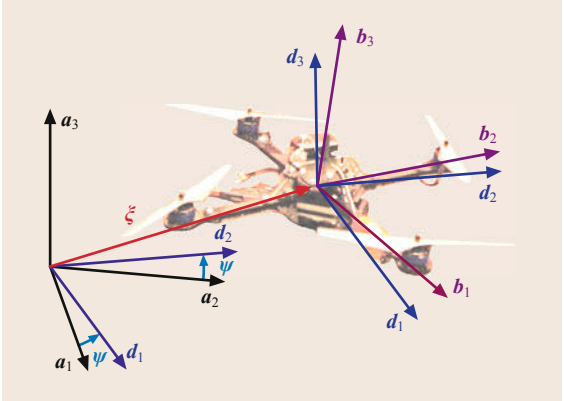


Fig. 52.3 The vehicle model. The position and orientation of the robot in the global frame are denoted by $\boldsymbol{\xi}$ and R, respectively

52.2.2 Modeling for Quadrotors

Quadrotors are currently the most popular research aerial robotics research platform because they are highly maneuverable and enable safe and low-cost experimentation in mapping, navigation, and control strategies in three dimensions. They are also arguably the simplest of the aerial robotics platforms to model for robotics applications. Early work in aerial robotics modeling and control dates back to the late 1990s and the field has remained active since [52.15–21].

In its most common form, the quadrotor vehicle is a very simple machine. It consists of four individual rotors attached to a rigid cross airframe as shown in Fig. 52.4. Control of a quadrotor is achieved by differential control of the thrust generated by each rotor. Pitch, roll, and heave (total thrust) control is straightforward to conceptualize. As shown in Fig. 52.2, rotor i rotates anticlockwise (positive about the z-axis) if i is even and clockwise if i is odd. Yaw control is obtained by adjusting the average speed of the clockwise and anticlockwise rotating rotors. The system is under-actuated and the remaining degrees of freedom corresponding to the translational velocity in the $\boldsymbol{b}_1$–$\boldsymbol{b}_2$ plane, must be controlled through the system dynamics.

The aerodynamics of rotors was extensively studied during the mid, 1900s with the development of manned helicopters and detailed models of rotor aerodynamics

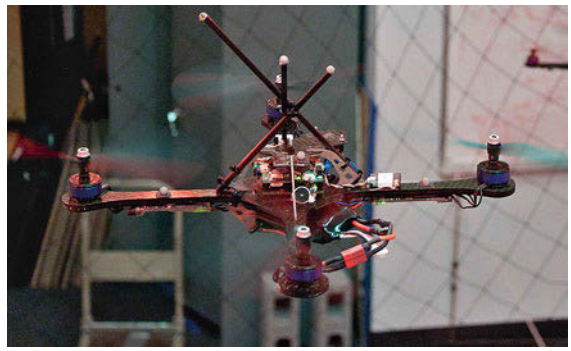


Fig. 52.4 A quadrotor made by Ascending Technologies at the University of Pennslvania with Vicon markers for state estimation

are available in the literature [52.22, 23]. Much of the detail in these aerodynamic models is useful for the design of rotor systems where the whole range of parameters (rotor geometry, profile, hinge mechanism, and much more) are fundamental to the design problem. For a typical robotic quadrotor vehicle, the rotor design is a question of choosing one among the several available rotors from the hobby shop and most of the complexity of aerodynamic modeling is best ignored. Nevertheless, a basic level of aerodynamic theory is important to understand the particularities of the control design.

The steady-state thrust generated by a hovering rotor (i. e., a rotor that is not translating horizontally or vertically) in free air may be modeled [52.23, Sect. 2.26] as

$$T_i := C_\mathrm{T} \rho A_{r_i} r_i^2 \varpi_i^2 \,, \tag{52.3}$$

where for rotor i, A_{r_i} is the rotor disk area, r_i is the radius, ϖ_i is the angular velocity, C_T is the thrust coefficient that depends on rotor geometry and profile, and ρ is the density of air. In practice, a simple lumped parameter model

$$T_i = c_\mathrm{T} \varpi_i^2 \,, \tag{52.4}$$

is used where $c_\mathrm{T} > 0$ is modeled as a constant that can be easily determined from static thrust tests.

The reaction torque (due to rotor drag) acting on the airframe generated by a hovering rotor in free air may be modeled as [52.23, Sect. 2.30]

$$Q_i := c_\mathrm{Q} \varpi_i^2 \,, \tag{52.5}$$

where the coefficient c_Q (which also depends on A_{r_i}, r_i, and ρ) can be determined by static thrust tests.

As a first approximation, assume that each rotor thrust is oriented in the $\boldsymbol{b}_3$-axis of the vehicle, although we note that this assumption does not hold exactly for a rotor translating through the air.

The total thrust at hover T applied to the airframe is the sum of the thrusts from each individual rotor (Fig. 52.2)

$$T = \sum_{i=1}^{4} |T_i| = c_\mathrm{T} \left(\sum_{i=1}^{4} \varpi_i^2 \right) . \tag{52.6}$$

The hover thrust is the primary component of the exogenous force

$$F = Te_3 + \Delta \tag{52.7}$$

in (52.2b) where Δ comprises secondary aerodynamic forces that are induced when the assumption that the rotor is in hover is violated.

The net moment arising from the aerodynamics (the combination of the individual rotor forces) applied to the quadrotor vehicle in directions $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, and $\boldsymbol{b}_3$, respectively, are

$$\begin{aligned} \tau_1 &= c_\mathrm{T} d (\varpi_2^2 - \varpi_4^2) \,, \\ \tau_2 &= -c_\mathrm{T} d (\varpi_1^2 - \varpi_3^2) \,, \\ \tau_3 &= c_\mathrm{Q} \sum_{i=1}^{4} \sigma_i \varpi_i^2 \,, \end{aligned} \tag{52.8}$$

where d is the arm length of the quadrotor. A consequence of the structure of the thrust generation and the lack of aerodynamic lifting surfaces for a quadrotor means that one can solve for the heave and torque in a single equation in terms of the motor inputs

$$\begin{pmatrix} T \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{pmatrix} = \underbrace{\begin{pmatrix} c_\mathrm{T} & c_\mathrm{T} & c_\mathrm{T} & c_\mathrm{T} \\ 0 & dc_\mathrm{T} & 0 & -dc_\mathrm{T} \\ -dc_\mathrm{T} & 0 & dc_\mathrm{T} & 0 \\ -c_\mathrm{Q} & c_\mathrm{Q} & -c_\mathrm{Q} & c_\mathrm{Q} \end{pmatrix}}_{\Gamma} \begin{pmatrix} \varpi_1^2 \\ \varpi_2^2 \\ \varpi_3^2 \\ \varpi_4^2 \end{pmatrix} . \tag{52.9}$$

Inverting this relationship provides a mapping from desired control input for the rigid-body dynamics to rotor speed set-points for the motor control.

In practice, there are additional second-order aerodynamic effects present in the thrust generation for quadrotor vehicles [52.17, 19–21]. The principle second-order aerodynamic effects that are present at low speeds are *inflow variation*, *rotor flapping*, and *induced drag*. The first of these effects reduces the heave thrust when the quadrotor is ascending and increases the thrust when it is descending due to variation in rotor inflow velocity caused by the quadrotor motion. This acts as a damping term in the vertical direction of motion for the quadrotor. The remaining two effects generate forces that oppose horizontal translation of the quadrotor; blade flapping by inclining the rotor plane of flexible blade rotors and tilting the thrust vector away from the direction of motion, and induced drag by increasing the drag on the blade of a rigid rotor advancing in the direction of motion of the quadrotor [52.21]. In practice, it is difficult to separate the difference between the various effects and it is sufficient to model them all by a single damping force

$$\Delta = -Dv \,, \tag{52.10}$$

where D is a diagonally dominant positive definite matrix. For small quadrotors with relatively rigid rotor blades designed for hover performance, that is with

close to ideal chord and ideal twist, these effects are far more significant than is usual in classical helicopters and the effect must be modeled to achieve good control performance. These aerodynamic effects are also important in providing key low-frequency excitation to attitude and velocity estimation in the absence of GPS [52.21, 24–26].

At higher translation speeds, a quadrotor also experiences *translational lift*, *translational drag*, and *parasitic drag*. These aerodynamic effects are associated with efficiency gains in the rotor operation associated with increased inflow velocity generated by the forward velocity of the vehicle. Such effects are extremely important in large scale manned helicopters, which spend much of their time in forward flight. For many robotics applications, however, the vehicle is in quasi-stationary or near hover flight almost all the time and these secondary aerodynamic effects may be ignored. We will not discuss them further in this chapter but refer the interested reader to recent work [52.18, 20] and any of the classical helicopter texts [52.22, 23, 27].

52.2.3 Modeling of Fixed-Wing Airplanes

The conventional model for fixed-wing airplane uses a north-east-down frame of reference. The frame of reference used in this chapter, corresponding to the common aerial robotics convention inherited from the mobile-robotics community, is a north-west-up convention. We will still need to define the usual auxiliary angles associated with fixed-wing modeling and control, in particular; the angle-of-attack, side-slip, aerodynamic-bank, flight-path, and course angles. We will use the normal conventions to define these angles. The north-west-up convention now means that flight-path angle is positive when descending and negative when ascending, and positive roll leads course angle to decrease. The careful reader can easily transform between the conventions used here and the conventional airplane modeling approach and the authors apologize for any confusion that is caused by using a single convention for the whole chapter.

The key difference in modeling fixed-wing airplanes to quadrotors is that they rely on lift generated by airflow over the wings to support them in flight. The angle of incidence of the wind on the wing is a key variable in the dynamics of the vehicle and must be modeled in the dynamics. Since the main lift force is generated by an aerodynamic process, rather than directly by a controlled input as was the case for quadrotors and blimps, the modeling and control of an airplane is somewhat more complicated.

The aerodynamics of a fixed wing vehicle are defined relative to the local wind frame $\{W\}$. Frame $\{W\}$ is chosen co-linear with the inertial frame $\{A\}$, but with linear velocity equal to the average extrinsic wind velocity with respect to the inertial frame. The wind frame is a Galilean frame (moving with constant velocity) with the property that it has zero extrinsic wind on average. Let $v_a \in \{W\}$ denote the aerodynamic velocity of the vehicle, that is the linear velocity of the vehicle in the wind frame $\{W\}$. Let $v_w \in \{A\}$ denote the average wind velocity as measured in the inertial frame $\{A\}$. Since the orientation of frames $\{A\}$ and $\{W\}$ are equal, the inertial velocity of the vehicle can be written as

$$v = v_a + v_w \in \{A\} .$$

To model the incident wind as seen by the vehicle as it flies through the air we introduce the flow frame (also termed the air frame) denoted by $\{F\} = (\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3)$ where the orientation vectors $\{\boldsymbol{f}_i\}$ are expressed in the wind frame $\{F\}$ which is co-linear with the inertial frame $\{A\}$. The first axis of the flow frame $\boldsymbol{f}_1$ is oriented in the direction of the incident wind on the vehicle, $\boldsymbol{f}_3$ is chosen to lie in the plane of symmetry of the vehicle orthogonal to $\boldsymbol{b}_2$, and $\boldsymbol{f}_2$ makes up the right-hand frame. By construction, the velocity of the vehicle in the flow frame always lies in the direction $\boldsymbol{f}_1$.

We define two orientation matrices associated with the wind frame. The angle-of-attack (AOA) matrix ${}^B R_F = R_{\alpha,\beta}$ describes the orientation of the airflow frame $\{F\}$ with respect to the body-fixed-frame $\{B\}$ in terms of the *angle-of-attack* α, and the *side-slip angle* β [52.28] (Fig. 52.5). Using the Z–Y–X Euler angle convention (yaw by β, pitch by α, and there is no roll) we have

$$R_{\alpha,\beta} = \begin{pmatrix} c_\alpha c_\beta & -c_\alpha s_\beta & -s_\alpha \\ s_\beta & c_\beta & 0 \\ s_\alpha c_\beta & s_\alpha s_\beta & c_\alpha \end{pmatrix} . \qquad (52.11)$$

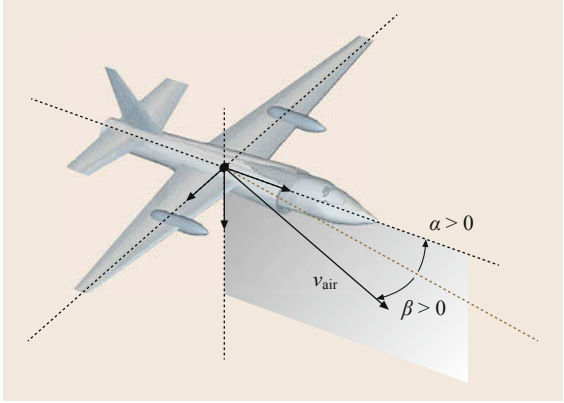


Fig. 52.5 Representation of angle-of-attack, side-slip angle, and airspeed. Here v_{air} is denoted by v_a in the text to save space

The second orientation matrix is the flight-path matrix ${}^{W}R_F = R_{\mu,\gamma,\chi}$ that denotes the orientation of the flow frame $\{F\}$ with respect to the wind frame $\{W\}$ in terms of the *aerodynamic bank angle* μ, the *flight path angle* γ, and the *course angle* χ. Using the *Z-Y-X* Euler angle convention (yaw by χ, pitch by γ, and roll by μ) we have

$$R_{\mu,\gamma,\chi} = \begin{pmatrix} \mathrm{c}_\gamma \mathrm{c}_\chi & \mathrm{s}_\gamma \mathrm{s}_\mu \mathrm{c}_\chi - \mathrm{c}_\mu \mathrm{s}_\chi & \mathrm{s}_\gamma \mathrm{c}_\mu \mathrm{c}_\chi + \mathrm{s}_\mu \mathrm{s}_\chi \\ \mathrm{c}_\gamma \mathrm{s}_\chi & \mathrm{s}_\gamma \mathrm{s}_\mu \mathrm{s}_\chi + \mathrm{c}_\mu \mathrm{c}_\chi & \mathrm{s}_\gamma \mathrm{c}_\mu \mathrm{s}_\chi - \mathrm{s}_\mu \mathrm{c}_\chi \\ -\mathrm{s}_\gamma & \mathrm{c}_\gamma \mathrm{s}_\mu & \mathrm{c}_\gamma \mathrm{c}_\mu \end{pmatrix}. \tag{52.12}$$

Note that the frames $\{W\}$ and $\{A\}$ are co-linear so ${}^{A}R_F = {}^{W}R_F$. It follows that ${}^{W}R_F = {}^{A}R_F = {}^{A}R_B\, {}^{B}R_F$.

Aerodynamic forces generated by lifting surfaces are proportional to the dynamic pressure [52.28]

$$\bar{Q} = \frac{1}{2}\rho |v_\mathrm{a}|^2 , \tag{52.13}$$

where ρ is the air density and $|v_\mathrm{a}|$ represents the airspeed or norm of the velocity v_a. Consider the situation where the aircraft is in normal flight, that is cruising at subsonic velocity while avoiding stall phenomenon. Let S denote the surface area of the wing (or lifting surface of the vehicle). The lift $L = F_Z^\mathrm{a}$ generated by the lifting surface is defined as the aerodynamic force oriented in the $\boldsymbol{f}_3$ direction in flow frame $\{F\}$ and can be modeled as

$$\begin{aligned} L &= \bar{Q}SC_\mathrm{L}^\alpha(\alpha + \alpha_0) + \bar{Q}SC_\mathrm{L}^\beta \beta \\ &\approx \bar{Q}S(C_\mathrm{L} + C_\mathrm{L}^\alpha \alpha) \end{aligned} \tag{52.14}$$

for sufficiently small angle of attack α (typically less that 15°). The terms $C_\mathrm{L}^\alpha(\alpha + \alpha_0)$ and $C_\mathrm{L}^\beta \beta$ are linear approximations to the lift-coefficient curves for small α and β [52.28]. The offset angle of attack α_0 is thought of as the effective angle of attack of the wing in normal level flight conditions. In particular, at normal cruising speed in level flight, $\bar{Q}SC_\mathrm{L}^\alpha \alpha_0 = mg$ is the lift required to sustain the vehicle in level flight, so that $\alpha, \beta = 0$ in level flight. The term $\bar{Q}SC_\mathrm{L}^\beta \beta$ is small and can be discarded in many applications.

The drag $D = -F_X^\mathrm{a}$ is a combination of parasitic and induced drag and is oriented in the $-\boldsymbol{f}_1$ direction.

$$\begin{aligned} D &= \bar{Q}S\left[C_\mathrm{D}^p + \kappa (C_\mathrm{L}^\alpha)^2(\alpha + \alpha_0)^2\right] \\ &\approx \bar{Q}S(C_\mathrm{D} + C_\mathrm{D}^\alpha \alpha) , \end{aligned} \tag{52.15}$$

where κ is the *Oswald coefficient* [52.28] and the second equation is obtained by ignoring α^2 terms. The constant C_D combines the parasitic drag coefficient C_D^p along with the induced drag $\kappa(C_\mathrm{L}^\alpha)^2\alpha_0^2$ due to α_0, while $C_\mathrm{D} = 2\kappa\alpha_0(C_\mathrm{L}^\alpha)^2$ is the linear coefficient for the dependence of induced drag on the angle-of-attack. It is also possible to model a contribution $\bar{Q}SC_\mathrm{D}^\beta \beta$ to the drag force, however, this is sufficiently small that it is normally ignored. There is a final lateral aerodynamic force component F_Y^a that is due to sideslip

$$F_Y^\mathrm{a} = \bar{Q}SC_Y^\beta \beta . \tag{52.16}$$

The propeller or thrust mechanism of a fixed wing vehicle generates thrust T in direction $\boldsymbol{b}_x$ in the body-fixed-frame. The total linear force $F \in \{B\}$ applied to the vehicle due to aerodynamic effects and thrust effects is

$$\begin{aligned} F &= R_{\alpha,\beta}(F_X^\mathrm{a}, F_Y^\mathrm{a}, F_Z^\mathrm{a})^\mathrm{T} + (T, 0, 0)^\mathrm{T} \\ &= \begin{pmatrix} T \\ 0 \\ 0 \end{pmatrix} + \bar{Q}S \begin{pmatrix} -C_\mathrm{D}\mathrm{c}_\alpha \mathrm{c}_\beta - C_\mathrm{L}\mathrm{s}_\alpha \\ -C_\mathrm{D}\mathrm{s}_\beta \\ -C_\mathrm{D}\mathrm{s}_\alpha \mathrm{c}_\beta + C_\mathrm{L}\mathrm{c}_\alpha \end{pmatrix} \\ &\quad + \bar{Q}S \begin{pmatrix} -C_\mathrm{D}^\alpha \mathrm{c}_\alpha \mathrm{c}_\beta - C_\mathrm{L}^\alpha \mathrm{s}_\alpha \\ -C_\mathrm{D}^\alpha \mathrm{s}_\beta \\ -C_\mathrm{D}^\alpha \mathrm{s}_\alpha \mathrm{c}_\beta + C_\mathrm{L}^\alpha \mathrm{c}_\alpha \end{pmatrix} \alpha \\ &\quad + \bar{Q}S \begin{pmatrix} -C_\mathrm{D}^\beta \mathrm{c}_\alpha \mathrm{c}_\beta - C_Y^\beta \mathrm{c}_\alpha \mathrm{s}_\beta - C_\mathrm{L}^\beta \mathrm{s}_\alpha \\ -C_\mathrm{D}^\beta \mathrm{s}_\beta + C_Y^\beta \mathrm{c}_\beta \\ -C_\mathrm{D}^\beta \mathrm{s}_\alpha \mathrm{c}_\beta + C_Y^\beta \mathrm{s}_\alpha \mathrm{s}_\beta + C_\mathrm{L}^\beta \mathrm{c}_\alpha \end{pmatrix} \beta \end{aligned} \tag{52.17}$$

expressed in the flow frame.

The aerodynamic moments $\tau \in \{B\}$ in the body-fixed frame are modeled by

$$\begin{pmatrix} \tau^x \\ \tau^y \\ \tau^z \end{pmatrix} = \bar{Q}Sl \begin{pmatrix} C_x^\beta \beta + C_x^p \dfrac{c\Omega_x}{2|v_\mathrm{a}|} + C_x^r \dfrac{c\Omega_z}{2|v_\mathrm{a}|} + C_x^\delta \delta_x \\ C_y + C_y^\alpha \alpha + C_y^q \dfrac{b\Omega_y}{2|v_\mathrm{a}|} + C_y^\delta(\delta_y + \delta_y^0) \\ C_z^\beta \beta + C_z^r \dfrac{c\Omega_z}{2|v_\mathrm{a}|} + C_z^\delta \delta_z \end{pmatrix}, \tag{52.18}$$

where $(\Omega_x, \Omega_y, \Omega_z)$ are the components of the angular velocity $\Omega \in \{B\}$ and represent the roll, pitch and yaw rates of body-fixed frame, respectively; b is the wing span, c is the mean cord of the wing,

and $(\delta_x, \delta_y, \delta_z)$ are deflections of the airplane control surfaces (ailerons, elevator, and rudder). The constants $\{C_x^p, C_x^r, C_x^\delta, C_y, C_y^\alpha, C_y^q, C_y^\delta, C_z^\beta, C_z^r, C_z^\delta\}$ are dimensionless aerodynamic coefficients [52.28]. In normal trim conditions, the trimmed elevator deflection δ_y^0 is chosen to cancel the static moment C_y generated by the wing in level flight, $C_y^\delta \delta_y^0 = -C_y$.

Combining (52.17) and (52.18) with (52.2) yields a dynamic model for a fixed wing vehicle with minimal approximations. The resulting equations of motion are complex and can be difficult to work with directly. The most common approach to control fixed-wing vehicles based on the full model is to take this model and linearize the state equations, either along a trajectory, or in regions of state space. The resulting model is either a linear time-varying or linear parameter varying system and can be controlled using classical linear system control techniques [52.29]. Although this approach is well established in the aerospace industry it tends to hide the underlying structure of the dynamics and requires good models of the various aerodynamic parameters, a challenge for many aerial robots with airframes that are always subject to being fiddled with and having sensor packages attached externally. For aerial robotic applications it is of interest to consider a simplified but still structurally consistent model of fixed wing dynamics and use robust and simple control strategies.

Bank-to-Turn Flight Regime

In this section, we propose a model that is suitable for a wide range of fixed-wing applications where the vehicle is in normal flight mode. The approach taken uses a flight regime where the lateral acceleration required to turn the vehicle is obtained by tilting the lift generated by the wings, a strategy termed *bank-to-turn* or *coordinated turn* in the aerospace literature [52.30, 31]. A bank-to-turn maneuver is characterized by zero sideslip, $\beta = 0$, and leads to considerable simplification of the equations of motion. This is a very common mode of flying for any fixed-wing vehicle and, unless the task requires aerobatic maneuvers or the vehicle lacks control surface actuation, is the natural mode to control a UAV.

Given that accurate measurements of the roll rates Ω are available through an onboard IMU system and most UAV systems have large control surfaces relative to their size, it is possible to use high gain to dominate the attitude dynamics (52.2d) and (52.18). That is, the angular velocity $\Omega \approx \Omega^\star$ can be thought of as an input to the reduced order model (52.2a), (52.2b), (52.2c). The above discussion motivates a dynamic reduction of the system equations with new inputs $\Omega^\star$, as well as T the propeller thrust.

Based on the assumption that the vehicle is flying using bank-to-turn control and for small angle-of-attack then the approximation $\beta \approx 0$ holds and simplified dynamic equations can be derived. Using this approximation and canceling all second-order terms in α and β allows one to rewrite (52.17) as

$$F = \begin{pmatrix} T \\ 0 \\ 0 \end{pmatrix} + \bar{Q}S \begin{pmatrix} -C_\mathrm{D} \\ 0 \\ C_\mathrm{L} \end{pmatrix} + \bar{Q}S \begin{pmatrix} -C_\mathrm{D}^\alpha - C_\mathrm{L} \\ 0 \\ C_\mathrm{L}^\alpha - C_\mathrm{D} \end{pmatrix} \alpha \,. \tag{52.19}$$

From the construction of $\{F\}$ we have ${}^v F_a = |v_\mathrm{a}| \boldsymbol{f}_1$ by definition. It follows that (52.2a) can be written as

$$\dot{\xi} = v_\mathrm{a} = |v_\mathrm{a}| R_{\mu,\gamma,\chi} e_1 \,.$$

Note that $R_{\mu,\gamma,\chi} e_1 = (\mathrm{c}_\chi \mathrm{c}_\gamma, \mathrm{s}_\chi \mathrm{c}_\gamma, -\mathrm{s}_\gamma)^\mathrm{T}$ does not depend on the aerodynamic bank angle μ. Thus, $(|v_\mathrm{a}|, \gamma, \chi)$ can be used as generalized coordinates for the velocity of the vehicle

$$\dot{\xi} = |v_\mathrm{a}| (\mathrm{c}_\chi \mathrm{c}_\gamma, \mathrm{s}_\chi \mathrm{c}_\gamma, -\mathrm{s}_\gamma)^\mathrm{T} \,. \tag{52.20}$$

Differentiating $v_\mathrm{a} = \dot{\xi}$ and rearranging yields

$$\dot{v}_\mathrm{a} = R_{\mu,\gamma,\chi} \begin{pmatrix} 1 & 0 & 0 \\ 0 & |v_\mathrm{a}| \mathrm{c}_\gamma \mathrm{s}_\mu & -|v_\mathrm{a}| \mathrm{s}_\mu \\ 0 & -|v_\mathrm{a}| \mathrm{c}_\gamma \mathrm{s}_\mu & -|v_\mathrm{a}| \mathrm{c}_\mu \end{pmatrix} \begin{pmatrix} |\dot{v}_\mathrm{a}| \\ \dot{\chi} \\ \dot{\gamma} \end{pmatrix} . \tag{52.21}$$

Consider the $(3,2)$ element of (52.12) and recall that $R_{\mu,\gamma,\chi} = R_{\phi,\theta,\psi} R_{\alpha,\beta}$ and applying $\beta \equiv 0$ then $\mathrm{s}_\mu \mathrm{c}_\gamma = \mathrm{s}_\phi$ and we have

$$\mathrm{s}_\mu = \frac{\mathrm{s}_\phi}{\mathrm{c}_\gamma} \tag{52.22}$$

for normal flight conditions where $\mathrm{c}_\gamma \neq 0$ and $\mathrm{s}_\phi < \mathrm{c}_\gamma$. By simple geometry,

$$\mathrm{c}_\mu = \frac{1}{\mathrm{c}_\gamma} \sqrt{\mathrm{c}_\gamma^2 - \mathrm{s}_\phi^2} \,,$$

and the dependence on the bank angle can be removed entirely from kinematics of $(|v_\mathrm{a}|, \gamma, \chi)$ to be replaced by the roll angle ϕ.

Inverting (52.21) and substituting for (52.2b), (52.19) and for (52.22) we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}|v_\mathrm{a}| = -g\mathrm{s}_\gamma + \frac{T}{m}\mathrm{c}_\alpha - \alpha\frac{\bar{Q}S}{m}(C_\mathrm{D}^\alpha + C_\mathrm{L}) - \frac{\bar{Q}S}{m}C_\mathrm{D}\,, \tag{52.23a}$$

$$\dot{\chi} = -\frac{\mathrm{s}_\phi}{\mathrm{c}_\gamma}\left(\frac{\bar{Q}S}{mc_\gamma}C_\mathrm{L} + (C_\mathrm{L}^\alpha - C_\mathrm{D})\alpha\right), \tag{52.23b}$$

$$\dot{\gamma} = \frac{g\mathrm{c}_\gamma}{|v_\mathrm{a}|} + \alpha\frac{\bar{Q}S}{m}(C_\mathrm{L}^\alpha - C_\mathrm{D}) - C_\mathrm{L}\frac{\bar{Q}S\sqrt{\mathrm{c}_\gamma^2 - \mathrm{s}_\phi^2}}{m|v_\mathrm{a}|\mathrm{c}_\gamma}\,, \tag{52.23c}$$

where one should recall that $\bar{Q} = \frac{1}{2}\rho|v_\mathrm{a}|^2$ (52.13) depends on $|v_\mathrm{a}|$.

Note that for α small, T provides a free input to stabilize the speed of the vehicle in (52.23a), although we have seen in Sect. 52.3.2 that a different approach is advantageous in practice. In (52.23b), there is no direct input variable, however, the roll ϕ will play this role in the control design. Similarly, in (52.23c), the angle-of-attack α, which itself has dynamics, must be used to control γ. In fact, both (52.23b) and (52.23c) depend on a complex mix of variables including dynamic pressure (and consequently $|v_\mathrm{a}|$) as well as ϕ, α and γ. Nevertheless, the identification of roll ϕ with control of heading and angle-of-attack α with control of flight path is a natural approach in the development of aircraft control.

The kinematics of ϕ are straightforward to compute by differentiating (52.1). Differentiating the $(3,2)$ entry of $R_{\phi,\theta,\psi}$ (52.1) we have

$$\dot{\phi} = \Omega_x\mathrm{c}_\theta + \Omega_z\mathrm{s}_\theta\,. \tag{52.24}$$

The primary control for $\dot{\phi}$ is the roll angular velocity Ω_x and this control will be used to cancel the disturbance created by Ω_z as discussed in Sect. 52.3.2.

To model the dynamics of α, recall the relationship $R_{\mu,\gamma,\chi} = R_{\phi,\theta,\psi}R_{\alpha,\beta}$ again. This time considering the $(3,1)$ element of $R_{\mu,\gamma,\chi}$ we obtain

$$\begin{aligned}\dot{\alpha}\,(\mathrm{c}_\phi\mathrm{s}_\theta\mathrm{s}_\alpha + \mathrm{c}_\alpha\mathrm{c}_\phi\mathrm{c}_\theta) = {}& \\ &-\mathrm{s}_\phi\mathrm{c}_\alpha\Omega_x + \mathrm{s}_\phi\mathrm{s}_\alpha\Omega_z \\ &+(\mathrm{c}_\phi\mathrm{s}_\theta\mathrm{s}_\alpha + \mathrm{c}_\phi\mathrm{c}_\theta\mathrm{c}_\alpha)\Omega_y \\ &-\mathrm{c}_\gamma\dot{\gamma}\,.\end{aligned} \tag{52.25}$$

Back substituting from (52.23c) and rearranging yields

$$\dot{\alpha} = \Omega_y + \frac{\mathrm{s}_\phi\mathrm{s}_\alpha\Omega_z - \mathrm{s}_\phi\mathrm{c}_\alpha\Omega_x}{(\mathrm{c}_\phi\mathrm{s}_\theta\mathrm{s}_\alpha + \mathrm{c}_\alpha\mathrm{c}_\phi\mathrm{c}_\theta)} - \mathrm{c}_\gamma\left[\frac{g\mathrm{c}_\gamma}{|v_\mathrm{a}|} + \alpha\frac{\bar{Q}S}{m}(C_\mathrm{L}^\alpha - C_\mathrm{D}) - \frac{\bar{Q}Sc_\mu}{m|v_\mathrm{a}|}C_\mathrm{L}\right], \tag{52.26}$$

where the denominators are always well defined for normal flight conditions. The angular velocity Ω_y enters (52.26) as a free input providing the control for AOA.

Equations (52.23a), (52.23c) and (52.26) lead to a cascade nonlinear system

$$\dot{\chi} = -\mathrm{s}_\phi\left(\frac{|v_\mathrm{a}|^2}{\mathrm{c}_\gamma^2}A_2 + \frac{\alpha}{\mathrm{c}_\gamma}A_3\right), \tag{52.27a}$$

$$\dot{\phi} = u_\phi\,, \tag{52.27b}$$

$$\dot{h} = -|v_\mathrm{a}|\mathrm{s}_\gamma\,, \tag{52.27c}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}|v_\mathrm{a}| = TA_1 - g\mathrm{s}_\gamma + f_1(\gamma,\alpha,|v_\mathrm{a}|)\,, \tag{52.27d}$$

$$\dot{\gamma} = \alpha|v_\mathrm{a}|^2A_4 - \frac{|v_\mathrm{a}|}{\mathrm{c}_\gamma}(\mathrm{c}_\gamma^2 - \mathrm{s}_\phi^2)^{\frac{1}{2}}A_5 + f_2(\gamma,|v_\mathrm{a}|)\,, \tag{52.27e}$$

$$\dot{\alpha} = u_\alpha + f_3(\gamma,\alpha,|v_\mathrm{a}|;t)\,, \tag{52.27f}$$

with the following constants

$$A_1 = \frac{\mathrm{c}_\alpha}{m}\,,\quad A_2 = \frac{C_\mathrm{L}\rho S}{2m}\,,\quad A_3 = (C_\mathrm{L}^\alpha - C_\mathrm{D})\,,$$
$$A_4 = \frac{\rho S}{2m}(C_\mathrm{L}^\alpha - C_\mathrm{D})\,,\quad A_5 = C_\mathrm{L}\frac{\rho S}{2m}\,,$$

functions;

$$f_1(\gamma,\alpha,|v_\mathrm{a}|) = -\alpha\frac{\bar{Q}S}{m}(C_\mathrm{D}^\alpha + C_\mathrm{L}) - \frac{\bar{Q}S}{m}C_\mathrm{D}\,,$$
$$f_2(\gamma,|v_\mathrm{a}|) = \frac{g\mathrm{c}_\gamma}{|v_\mathrm{a}|}\,,$$
$$f_3(\gamma,\alpha,|v_\mathrm{a}|) = -\mathrm{c}_\gamma\left[\frac{g\mathrm{c}_\gamma}{|v_\mathrm{a}|} + \alpha\frac{\bar{Q}S}{m}(C_\mathrm{L}^\alpha - C_\mathrm{D}) - \frac{\bar{Q}Sc_\mu}{m|v_\mathrm{a}|}C_\mathrm{L}\right],$$

and the input functions;

$$u_\phi = \Omega_x\mathrm{c}_\theta + \Omega_z\mathrm{s}_\theta\,,$$
$$u_\alpha = \Omega_y + \frac{\mathrm{s}_\phi\mathrm{s}_\alpha\Omega_z - \mathrm{s}_\phi\mathrm{c}_\alpha\Omega_x}{(\mathrm{c}_\phi\mathrm{s}_\theta\mathrm{s}_\alpha + \mathrm{c}_\alpha\mathrm{c}_\phi\mathrm{c}_\theta)}.$$

Here the height h kinematics have been added to complete the bank-to-turn dynamics. The course (52.27a), and the height (52.27c) dynamics have negative signs compared to conventional fixed-wing modeling developments due to the nonstandard north-west-up convention chosen.

These equations are fully nonlinear equations based on the sliding mode simplification associated with holding $\beta \equiv 0$. The condition on side-slip angle will can be enforced using the free degree of freedom in Ω_z as will be discussed in Sect. 52.3.2. We note that the dynamics presented in this section are closely related to the *longitudinal dynamics* considered in classical aeronautical texts that treat the angle of attack, glide angle, and velocity of the vehicle. The more classical expressions for longitudinal control of a fixed-wing vehicle are obtained from the above model by setting $\dot{\chi} = \mu = \phi = 0$. An additional simplifying assumption often made for the course kinematics (52.27a) is that lift is compensating gravity during level-flight. With these assumptions, (52.27a) can be replaced by

$$\dot{\chi} = -\frac{g}{|v_a|}\tan(\phi)$$

based on centripetal force balance.

52.3 Control

Control for aerial vehicles is challenging for a number of reasons. First, most aerial vehicles are under actuated and the control design must exploit interactions between dynamic states to control the vehicle. Second, aerial vehicles use aerodynamic effects for thrust and lift generation, and regulation of these forces is by nature approximate, leading to significant modeling error. Third, external effects such as wind, turbulence, and vortex generation, lead to high levels of load disturbance in the control loops. Finally, it is often difficult or impossible to measure the vehicle and aerodynamic state directly making it necessary to use an observer or design the controller from first principles to use explicit measurements.

Most aerial vehicles are controlled using a hierarchical control structure with nested feedback loops based on three levels; planning, guidance, and control:

- *Planning*: It is the outermost loop in aerial vehicle control and is associated with path planning, setting way-points, etc.
- *Guidance*: The guidance level of control concerns tracking trajectories to achieve local goals. This control loop is typically designed using the attitude reference as a virtual input, leading to a fully actuated control problem.
- *Control*: It is the high gain innermost loop of the control hierarchy and is concerned primarily with attitude and flight stability of the vehicle. This control problem is typically fully actuated and can be tackled using standard control techniques.

The more dynamic and aggressive the task considered is, the more these levels of control will interact and an integrated control design must be considered. However, in most real-world situations, splitting the control into three hierarchical levels leads to a simpler design problem that achieves the desired performance. For example, the recent aggressive maneuvers undertaken and demonstrated in quadrotor control are based on hierarchical control strategies that use trajectory planning processes for the outer loop that specify achievable trajectories that then are stabilized in real-time using attitude as a virtual input, with high-gain feedback on the inner-loop attitude stability control. Most of this work has been done with the use of external motion capture systems to provide the inertial frame position and velocity feedback [52.7–12, 32]. However, more recently there have been a few attempts to do this with onboard cameras and inertial measurement units [52.13, 14].

52.3.1 Quadrotor Control

The quadrotor control and guidance problem is conceptually more straightforward than that of fixed-wing vehicles. In its simplest form, the goal is to design control algorithms to track smooth feasible trajectories $(R^\star(t), \xi^\star(t)) \in \mathrm{SE}(3)$. We will assume that a planner specifies the full desired trajectory, including the higher order derivative terms $(\Omega^\star(t), \dot{\xi}^\star(t))$ and $(\dot{\Omega}^\star(t), \ddot{\xi}^\star(t))$. A quadrotor vehicle is an under-actuated system – there are four inputs $\mathbf{u} = (T, \tau^{\mathrm{T}})^{\mathrm{T}}$, while the trajectory lives in SE(3) and is six-dimensional. The hierarchical control structure proposed leads to an inner loop regulating the attitude using the moments τ as the control input. The guidance level utilizes the attitude R and heave T to regulate the trajectory $\xi(t)$ to track $\xi^\star(t)$. In high performance control of a quadrotor, the major limitation to performance comes from the limits of motor response and the control hierarchy must be augmented by a low level motor regulation system. The proposed control architecture forms nested feedback loops as shown in Fig. 52.6.

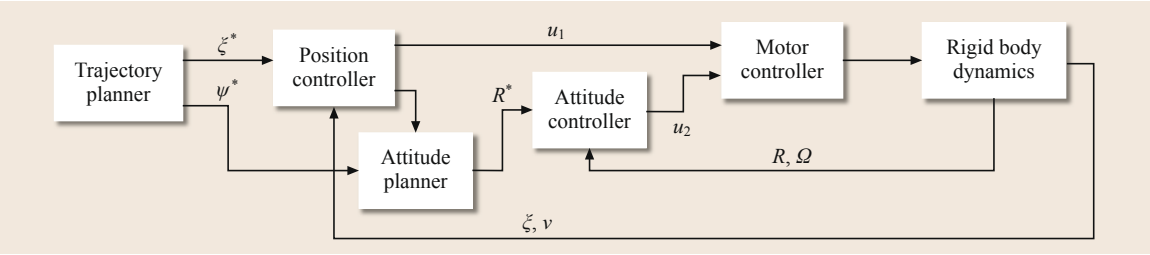


Fig. 52.6 Typical control architecture. A base-level motor regulation, an inner attitude control loop, an intermediate position guidance loop, with an outer trajectory planner

Motor Regulation

The aerodynamics of thrust generation by rotors is a subject that has been studied in detail in the classical rotorcraft literature [52.22, 23, 27]. Using a full model of rotor aerodynamics has significant potential in the design of the motor control for high-performance rotor control for aerial robotic applications [52.33]. Despite this, it is sufficient in most cases to consider a static thrust model that holds for a rotor in hover and is a reasonable approximation for most robotic applications. In such a model, the thrust generated is proportional to the square of the rotor speed. Since most quadrotor vehicles are equipped with brushless DC motors that use back EMF sensing for rotor commutation, making it possible to measure rotor angular velocity, the control of the rotor thrust is usually implemented as a local control loop on the electronic speed controller (ESC) regulating rotor speed.

Most ESC use PWM (pulse width modulation) regulation of voltage bus to provide voltage control of the motors. A typical rise time for the uncontrolled system is in the order of 200 ms and it is necessary to include a local control loop to improve the system response. The aerodynamic drag of the rotor means that the system is naturally heavily damped and there is no need for derivative control. Similarly, an integral term is rarely deemed necessary at the rotor control level since the thrust model used is not particularly accurate anyway and it will always be necessary to have integral control at higher levels in the control hierarchy. Since an integral term is inadvisable, it is important to use the best model of the thrust generation possible as a feedforward term $V_{\mathrm{ff}}(\varpi_i^\star)$ to minimize the requirement on the proportional regulation. Thus, a typical ESC rotor control is given by saturated proportional control with feedforward [52.34]

$$V_i = \mathrm{sat}\left[k(\varpi_i^\star - \varpi_i) + V_{\mathrm{ff}}(\varpi_i^\star)\right], \tag{52.28}$$

where V_i is the applied motor voltage, $\varpi_i^\star$ is the desired speed and the actual motor speed ϖ_i is measured from the electronic commutation in the embedded speed controller. The performance of the motor controllers is ultimately limited by the current that can be supplied from the batteries [52.34] and a saturation on the demanded voltage is necessary. Without the saturation, extreme maneuvers may cause the voltage bus to drop excessively, destroying the rotor regulation performance and unless care is taken, causing the onboard electronics to brownout.

Attitude Control

The control problem considered in attitude regulation is to use full actuation of τ in (52.2d) to control (52.2c) to track a desired attitude trajectory $R^\star(t)$ along with its velocity $\Omega^\star(t)$ such that $\dot{R}^\star(t) = R^\star(t)\Omega^\star(t)$. The approach described uses a global stability design based on minimizing the matrix error

$$\tilde{R} = (R^\star)^{\mathrm{T}} R \tag{52.29}$$

similar to [52.7, 35]. Driving $\tilde{R} \to I_3$ ensures that $R \to R^\star$. The kinematics of the tracking error is given by

$$\begin{aligned}\dot{\tilde{R}} &= -\Omega_\times^\star \tilde{R} + \tilde{R}\Omega_\times \\ &= [\tilde{R}, \Omega_\times^\star] + \tilde{R}(\Omega_\times - \Omega_\times^\star) \\ &= [\tilde{R}, \Omega_\times^\star] + \tilde{R}\tilde{\Omega}_\times \,,\end{aligned} \tag{52.30}$$

where $[A, B] = AB - BA$ is the matrix commutator and

$$\tilde{\Omega} := \Omega - \Omega^\star \,. \tag{52.31}$$

We assume that derivative $\dot{\Omega}^\star$ of the target angular velocity is available for the purposes of designing a feedforward torque for the angular velocity dynamics

$$\tau^\star := \mathbf{I}\dot{\Omega}^\star + \Omega_\times^\star \mathbf{I}\Omega \,, \tag{52.32}$$

where $\mathbf{I}$ is the inertia matrix of the airframe (52.2d). Feed-forward control for the angular velocity of the attitude is important for high-performance maneuvers, especially when the motor response is pushed to near performance limits. The higher derivative information

for the angular velocity is available from path planning algorithms such as those discussed in Sect. 52.4.1 or using predictive control algorithms [52.8–10, 36]. Choose the control input

$$\tau := \tau^\star + \mathbf{u}_2 \tag{52.33}$$

where $\mathbf{u}_2 \in \mathbb{R}^3$ is the free control that will be used to stabilize the error dynamics (Fig. 52.6). The error dynamics for the angular velocity error are now

$$\mathbf{I}\dot{\tilde{\Omega}} = -\tilde{\Omega}_\times \mathbf{I}\Omega + \mathbf{u}_2 \,. \tag{52.34}$$

The goal is to choose $\mathbf{u}_2$ in order to stabilize the error dynamics (52.30) and (52.34) robustly. Define

$$\mathbb{P}(\tilde{R}) := \frac{1}{2}\left(\tilde{R} - \tilde{R}^{\mathrm{T}}\right)$$

to be the skew-symmetric projection of the error matrix. It can be verified that $\mathbb{P}(\tilde{R}) = \sin(\theta) a_\times$ where (a, θ) is the angle-axis representation of $\tilde{R}$. That is the skew-symmetric projection is the axis of rotation that rotates R to $R^\star$ (equivalently rotates $\tilde{R}$ to the identity) scaled by the sin of the angle between the two orientations. Choose a proportional–derivative (PD) control

$$\mathbf{u}_2 = -k_{\mathrm{P}} \mathrm{vex}(\mathbb{P}(\tilde{R})) - k_{\mathrm{D}} \tilde{\Omega} \,,$$

where vex: $\mathbb{R}^{3\times 3} \to \mathbb{R}^3$ is the inverse of the skew-symmetric operator applied to skew-symmetric matrices, $\mathrm{vex}(\Omega_\times) = \Omega$. The proportional term applies a torque associated with a nonlinear spring associated with a potential energy $\mathrm{tr}(\tilde{R}^{\mathrm{T}}\tilde{R})$ while the derivative $k_{\mathrm{D}}\tilde{\Omega}$ provides damping. It is straightforward to demonstrate that the Lyapunov function

$$\mathcal{L} := k_{\mathrm{P}} \mathrm{tr}(\tilde{R}^{\mathrm{T}}\tilde{R}) + \frac{1}{2}\tilde{\Omega}^{\mathrm{T}}\mathbf{I}\tilde{\Omega}$$

is decreasing

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{L} := -k_{\mathrm{D}} \|\tilde{\Omega}\|^2 \,.$$

Invoking Barbalat's lemma, with some care, it follows that the system is almost globally asymptotically stable. The zero measure exception set for the basin of attraction is associated with situations where the quadrotor is completely flipped, $\theta = \pi$ rad, and $\mathbb{P}(\tilde{R}) = 0$ even though $\tilde{R} \neq I$.

To verify the local exponential convergence and provide guidance for the gain tuning, we consider the linearization of (52.30) and (52.34). Write $\tilde{R} \approx I + (z_R)_\times$ where the skew-symmetric matrix $(z_R)_\times$ (for $z_R \in \mathbb{R}^3$) is the linear approximation of $\tilde{R}$ around the identity matrix I. Write $\tilde{\Omega} \approx z_\Omega$ for the linear approximation of $\tilde{\Omega}$ around the origin $\tilde{\Omega} = 0$. The linearization of the error $z = (z_R, z_\Omega)$ system is given by

$$\begin{pmatrix} \dot{z}_R \\ \dot{z}_\Omega \end{pmatrix} = \left[\begin{pmatrix} (\Omega^\star)_\times & 0 \\ 0 & \mathbf{I}^{-1}(\mathbf{I}\Omega)_\times \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ -k_{\mathrm{P}}\mathbf{I}^{-1} & -k_{\mathrm{D}}\mathbf{I}^{-1} \end{pmatrix} \right] \begin{pmatrix} z_R \\ z_\Omega \end{pmatrix} \tag{52.35}$$

or in more compact notation $\dot{z} = A(t)z$ where $A(t) = A_1(t) + A_2$ comprises the sum of two matrices in (52.35). The matrix $A_1(t)$ is a time-varying matrix depending on the exogenous system signals Ω and $\Omega^\star$ while the second matrix A_2 is a time-invariant Hurwitz second-order linear system matrix. It is straightforward to show that this system is asymptotically stable using a Lyapunov function $\ell = k_{\mathrm{P}}/2|z_R|^2 + 1/2 z_\Omega^{\mathrm{T}}\mathbf{I}z_\Omega$, using an analog of the nonlinear argument made earlier. We have that $\dot{\ell} = -k_{\mathrm{D}}|z_\Omega|^2$. Define $C = \begin{pmatrix} 0 & I_3 \end{pmatrix} \in \mathbb{R}^{3\times 6}$ then if the pair $(A(t), C)$ is uniformly completely observable (UCO) asymptotic stability implies exponentially stability [52.37, pp. 626–628]. It is intuitively clear that the system matrices $(A(t), C)$ associated with (52.35) are UCO, although providing the algebraic proof is beyond the scope of the this chapter. Tuning the gains is then a process of choosing k_{P} and k_{D} to assign the eigenvalues of the Hurwitz system matrix, with an eye to dominate the oscillatory, but bounded energy, disturbance introduced by $A_1(t)$.

Trajectory Tracking Control

The trajectory planning algorithm, discussed later in Sect. 52.4.1, provides a full trajectory specification as well as the feedforward inputs consisting of $(\xi^p, \dot{\xi}^p, R^p, \Omega^p, \dot{\Omega}^p, u_1^p, u_2^p)$ and their derivatives. The goal of trajectory tracking control is to regulate the linear dynamics of a quadrotor to track a specified goal trajectory $\xi^\star(t)$. Trajectory tracking control operates at a higher level in the hierarchy of the quadrotor control architecture (Fig. 52.6) and the desired attitude $R^\star$ used in the attitude control loop is designed as an input in the trajectory loop control design. In addition to specifying the attitude goal, $R^\star$, the trajectory tracking control specifies the heave (total thrust), T, input reference that is used along with (52.33) for the low-level motor control reference (52.9).

The linear trajectory goal $\xi^\star = \xi^p$ and $\dot{\xi}^\star = \dot{\xi}^p$ are used directly from the path planning design as the goal of the trajectory planner. The actuation for the linear dynamics, however, depends on the attitude of the vehicle. Thus, it is necessary to define the goal for the attitude control $R^\star$ as a part of the control design

rather than rely on the planned trajectory set-point; in particular $R^\star \neq R^p$ in general. Note that the feedforward control in Sect. 52.3.1 requires $\Omega^\star$ and $\dot{\Omega}^\star$ to plan the feedforward torque input for the attitude dynamics. Feed-forward in the attitude regulation loop is a critical component of high-performance control and without this input, there will not be sufficient gain in the attitude dynamics to track aggressive attitude trajectories. However, computing the angular velocity goal that corresponds to a feedback dependent attitude control goal $R^\star$ in real-time requires forward prediction of the control trajectory using tools such as model predictive control (MPC) [52.8–10, 36]1. A simpler approach for trajectory tracking is to use the planned angular velocity $\Omega^\star = \Omega^p$ even though it does not exactly match the variation of $R^\star$. As long as $(\xi, \dot{\xi})$ is close to $(\xi^\star, \dot{\xi}^\star) = (\xi^p, \dot{\xi}^p)$ then $R^\star$ is close to R^p and $(\Omega^\star, \dot{\Omega}^\star) = (\Omega^p, \dot{\Omega}^p)$ will provide a good estimate of the critical feedforward attitude input.

We will discuss here the linear trajectory tracking algorithm for which the specified trajectory is required to have roll and pitch values that are small thus justifying a linear approximation of the dynamics. Accordingly, we assume that the Euler angles associated with the rotation R^p (52.1), are given by $\theta^p = \phi^p = 0$, while ψ^p is a specified function of t. For a given arbitrary yaw angle, we can linearize the dynamics about the hover position ($\theta = 0$, $\phi = 0$, $\psi = \psi^p$) and the nominal inputs $u_1 = mg$, $\mathbf{u}_2 = 0$. Linearizing (52.2a), we obtain

$$\begin{aligned}
\ddot{\xi}_1 &= g(\Delta\theta \cos\psi^p + \Delta\phi \sin\psi^p) \,, \\
\ddot{\xi}_2 &= g(\Delta\theta \sin\psi^p - \Delta\phi \cos\psi^p) \,, \\
\ddot{\xi}_3 &= \frac{1}{m} u_1 - g \,,
\end{aligned} \tag{52.36}$$

where $\Delta\theta = \theta^\star$ and $\Delta\phi = \phi^\star$ represent the small deviations from the hover position. In order to exponentially drive all three components of error in the position part of the trajectory, we want the acceleration vector $\ddot{\xi}$ to satisfy

$$(\ddot{\xi}^\star - \ddot{\xi}) + K_d(\dot{\xi}^\star - \dot{\xi}) + K_p(\xi^\star - \xi) = 0 \,.$$

From (52.36), we can immediately write

$$u_1 = m\left[g + \ddot{\xi}_3^\star + k_{d,z}(\dot{\xi}_3^\star - \dot{\xi}_3) + k_{p,z}(\xi_3^\star - \xi_3)\right], \tag{52.37}$$

to guarantee $(\xi_3(t) - \xi_3^\star(t)) \to 0$. We choose the appropriate desired roll, pitch, and yaw angles for $\theta^\star$, $\phi^\star$, and $\psi^\star$ to guarantee exponential convergence:

$$\phi^\star = \frac{1}{g}(\ddot{\xi}_1^\star \sin\psi - \ddot{\xi}_2^\star \cos\psi) \,, \tag{52.38a}$$

$$\theta^\star = \frac{1}{g}(\ddot{\xi}_1^\star \cos\psi + \ddot{\xi}_2^\star \sin\psi) \,, \tag{52.38b}$$

$$\psi^\star = \psi^p \,. \tag{52.38c}$$

Now $(\psi^\star, \phi^\star, \theta^\star)$ define the rotation matrix $R^\star$ (52.1) provided as a set point for the exponentially convergent attitude controller discussed in the previous section. Thus, as shown in Fig. 52.6, the control problem is addressed by decoupling the position control and attitude control subproblems and the position control loop provides the attitude set points for the attitude controller.

The position controller can also be obtained without linearization. This is done by projecting the position error (and its derivatives) along $\mathbf{b}_3$ and applying the input u_1 that cancels the gravitational force and provides the appropriate proportional plus derivative feedback

$$u_1 = m\boldsymbol{b}_3^{\mathrm{T}}\left[\ddot{\xi}^\star + K_d(\dot{\xi}^\star - \dot{\xi}) + K_p(\xi^\star - \xi) + ge_3\right]. \tag{52.39}$$

Note that the projection operation is a nonlinear function of the roll and pitch angles and thus this is a nonlinear controller. In [52.7, 35], it is shown that this approach results in exponential stability and allows the robot to track trajectories in SE(3).

52.3.2 Control of Fixed-Wing Aircraft

An aircraft is a highly coupled nonlinear dynamical system and it may seem most natural to treat the control problem as an integrated multi-input multi-output (MIMO) nonlinear design [52.29, 38–40]. However, despite the nonlinear coupling of aircraft states there are still clear dependencies between specific actuators and certain dynamic modes of the open-loop response. A disadvantage of the MIMO approach is that it tends to obscure insight between control design and specific open-loop mode response. Moreover, and as a consequence, it tends to be less robust to large changes in model parameters (such as caused by using a different set of wings, or bolting on a completely new sensor package, not an uncommon situation for a small UAV system) than a classical control design. The classical, and more intuitive, control architecture for fixed wing vehicles is to consider a set of separate, but interconnected, single-input single-output (SISO) control loops. Such an approach has the advantage of being highly intuitive and leading to a modular design methodology

based on successive loop closure, providing a straightforward gain tuning regime [52.31]. The robustness and simplicity of the SISO architecture make it the preferred choice for most small UAV control systems. A modern implementation, however, uses the total energy control system (TECS) architecture and input decoupling to overcome some of the limitations of classical fixed-wing control design.

The main control actions for a fixed wing aircraft are associated with the throttle, and the three aerodynamic control surfaces; rudder, ailerons, and elevators. The throttle regulates power to the motor and governs thrust T. The rudder, ailerons and elevators actuate the angular velocities Ω_z, Ω_x, and Ω_y, respectively, through the angular dynamics of the vehicle. For a small fixed-wing UAV in normal flight, a simple high-gain proportional control scheme will effectively regulate and decouple the variables $(T, \Omega_x, \Omega_y, \Omega_z)$ [52.31]. The control problem can then be posed as one where $(T, \Omega_x, \Omega_y, \Omega_z)$ are inputs for regulation of the remainder of the vehicle dynamics. Since there is weak physical coupling between the lateral variables (roll rate, bank angle, yaw rate, and course angle) and the longitudinal variables (airspeed, pitch rate, pitch angle, altitude rate, and altitude), and since the elevator and rudder most strongly effect the lateral variables and the throttle and aileron most strongly influence the longitudinal variables, most control architectures separate functionality into a lateral control system and a longitudinal control system. The input to the lateral control system is the desired course angle, and the input to the longitudinal control system is the desired airspeed and the desired altitude.

Lateral Control System

There are three open-loop dynamic modes associated with the response of the lateral dynamics of a fixed-wing aircraft: the roll mode, the spiral mode, and the dutch roll mode. The *roll mode* is associated with the first-order response of the roll rate of the vehicle to aileron input. Usually the wings and airframe are designed (e.g., with a positive dihedral angle Chap. 26.4.4) so that this mode is highly stable and is regulated using high-gain in the aileron input – indeed, this mode is subsumed into the control simplification (regulation of Ω_x) proposed above.

The spiral mode is associated with the natural turning motion of the vehicle. That is, if the vehicle is banked into a turn, it will continue to turn until control action is applied to flatten the vehicle out. The spiral mode is a linearization of the bank-to-turn dynamics and is a single real pole close to the origin (Chap. 26, Fig. 26.22). Typically the spiral mode is slightly unstable due to the tendency of the vehicle to side-slip into a turn. In manned flight, the spiral mode can be dangerous since there is very little external perception of turning. The nature of the banked turn dynamics ensure that the body-fixed-frame acceleration remains oriented through the central axis of the vehicle and rate of rotation is slow and nearly constant. The mode becomes dangerous if the pilot is not aware as the vehicle slowly side-slips down into the turn, progressively diving more and more steeply and increasing airspeed in what is known as a *spiral-dive*. For autonomous flight control, the spiral mode is relatively benign since autopilots never stop looking at the instruments and take immediate action if the vehicle is drifting off course.

The *dutch-roll* mode, is associated with the oscillation of the vehicle around a heading. In this motion, the vehicle yaws away from the heading and then rolls due to the dihedral effect, before the tilted lift vector and rudder action force the airplane back toward the incident wind. If the rudder is small and does not provide sufficient damping, the vehicle can overshoot to yaw and roll in the other orientation before repeating the oscillatory sequence. Dutch role motion is less dangerous for manned flight since it is usually stable (except for vehicles with strongly swept back wings or negative dihedral) and produces unpleasant sideways acceleration with the side-slip motion that is immediately apparent to a pilot. Damping is increased by large tail plane surfaces and moderate to small dihedral. A figure of the typical pole positions of an aerobatic RC airplane is shown in Chap. 26, Fig. 26.22.

To understand the control problem for lateral dynamics, consider first the angular yaw velocity Ω_z induced by rudder actuation. The primary effect of Ω_z will yaw the vehicle in the air and generate nonzero side-slip β. In the *bank-to-turn* control paradigm described later, we will use Ω_z to regulate $\beta \equiv 0$, however, it is instructive first to consider the effect of nonzero side-slip in order to build an intuitive understanding of the lateral dynamics. If the side-slip $\beta \neq 0$, the airplane will be sliding sideways or crabbing through the air. In particular, applying rudder input (nonzero Ω_z) acts to twist the airplane in the air; it does not in itself cause the trajectory of the airplane to curve. For a typical airplane with dihedral, this will lead the aircraft to bank away from the side-slip, and if the dutch-roll mode is stable, the airplane will settle into a banked turn. The causality of the system signals is given by

$$\Omega_z \rightarrow \beta \rightarrow \phi \rightarrow \chi \,.$$

Here the actuation from $\beta \rightarrow \phi$ relies on the inherent stability of the dutch roll mode of the lateral open-loop dynamics of the airframe. Indeed, small-scale vehicles with strong dihedral and large tail planes can be flown

without aileron surfaces. For example, the aerial robotic vehicle as shown in Fig. 52.7, designed for slow low-level flight and equipped for visual terrain tracking using optic flow computed from a pair of downward pointing web cameras [52.41], does not have ailerons and the lateral motion is controlled using only the rudder.

This control paradigm has several negatives. It is rarely used for manned flight because side-slip motion generates sideways accelerations that do not feel natural. More importantly, the control response is limited by the low pass response of the open loop dutch roll dynamics of the aircraft. Although this mode is stable, there is often a noticeable damped oscillatory response that causes the vehicle trajectory to wander around the desired heading and degrades course tracking performance.

The alternative is to explicitly use the ailerons to impose the desired roll angle ϕ on the vehicle and actuate the heading dynamics (52.27a) without using the rudder. The rudder then becomes a separate control input that can be used to regulate β. That is

$$\Omega_z \to \beta \,,$$
$$\Omega_x \to \phi \to \chi \,.$$

Although, this may seem counter intuitive, (the rudder is not used to steer the airplane) it is, however, the most effective heading control strategy. Using this control strategy and regulating the side-slip $\beta \equiv 0$ ensures that the bank-to-turn simplifications can be applied to the model as was shown in Sect. 52.2.3.

Specifically, in the bank-to-turn strategy where side-slip β is regulated to zero using the rudder, the lateral equations of motion are given by (52.24) and (52.27a). In level flight where the pitch angle is small and the velocity and angle of attack are roughly a constant, these equations simplify to the cascade structure

$$\dot{\phi} = \Omega_x + d_\phi \,,$$
$$\dot{\chi} = -A_\chi \phi + d_\chi \,,$$

where A_χ is a constant and d_ϕ and d_χ are disturbance signals introduced through the modeling process. From classical control theory, we know that a step in the heading command $\chi^\star$ can be tracked while simultaneously rejecting low-frequency disturbances d_ϕ and d_χ using a nested PI control law given by

$$\phi^\star = k_{\mathrm{P}_\chi}(\chi^\star - \chi) + k_{I_\chi} \int_{-\infty}^{t} (\chi^\star - \chi)\,\mathrm{d}\tau \,, \quad (52.40)$$

$$\Omega_x = k_{\mathrm{P}_\phi}(\phi^\star - \phi) + k_{I_\phi} \int_{-\infty}^{t} (\phi^\star - \phi)\,\mathrm{d}\tau \,, \quad (52.41)$$

where k_{P_χ}, k_{I_χ}, k_{P_ϕ}, and k_{I_ϕ} are positive control gains. The proportional control

$$\Omega_z = -k_{\mathrm{P}_\beta}\beta, \quad (52.42)$$

is usually effective at regulating the side-slip angle to zero, where k_{P_β} is a positive control gain.

Fig. 52.7 An autonomous fixed-wing aerial robot for experimental work on vision-based terrain tracking at ANU (after [52.41]). Note the upswept wings to generate dihedral and the lack of ailerons

Longitudinal Control System

The longitudinal dynamics concern the thrust T and elevator (regulating Ω_y) inputs, along with the angle of attack α, the flight path angle γ, the airspeed $|v_\mathrm{a}|$, and the altitude h. There are two primary open-loop dynamic modes associated with the open loop response of the longitudinal dynamics, the short-period and phugoid modes. The *short-period* mode is associated with variation of pitch of the vehicle for constant velocity, constant altitude flight. It is a consequence of the linear dependence of lift of the primary wing to angle-of-attack as the vehicle pitches up and down around its center of the mass. The center of mass of an airplane is always designed to lie in front of the center of force of the primary wing surface to ensure that the short-period mode oscillation is stable, (placing the mass further forward increases the stability margin,) while damping provided by the tail plane of the vehicle ensures asymptotic stability (Chap. 26.4.2).

The *phugoid mode* is associated with the interplay of aerodynamic lift, velocity, glide path, and height of the vehicle for constant angle-of-attack flight. As a vehicle starts to descend gently, the airspeed increases as the loss of potential energy in height is transferred into kinetic energy. As the vehicle accelerates, the increased airspeed leads to increased primary lift on the wings leading the vehicle to pull out of the dive and to start to ascend again. The vehicle will slow as the

stored kinetic energy is traded back into potential energy. The resulting open-loop unstabilized motion is a repeated swooping cycle. Since aircraft are designed with low drag coefficients, the phugoid mode is usually very lightly damped, however, it is slow and easily stabilized. A key point to note in phugoid motion is that it is a constant energy oscillation; potential energy is transformed into kinetic energy and is then transformed back into potential energy.

Both thrust and pitch rate inputs affect both velocity and angle-of-attack of an airplane. In turn, the glide-path angle can be controlled through regulation of either angle-of-attack or velocity of the vehicle. The coupling associated with these dependencies mean that simple thrust to velocity or pitch-rate to glide-path SISO control loops cannot cover the full flight regime of a typical aircraft, a limitation of many existing autopilot systems even in commercial airliners. The industry standard approach to address this issue is to have different control modes for different flight conditions, take-off and climb, level flight, and descent, that exploit different input to state control mappings [52.31]. The total energy control system (TECS) design paradigm [52.42, 43] partially avoids these difficulties for a wide range of operating conditions and has the additional advantage of minimizing the control activity undertaken by the throttle.

Due to the energy preserving nature of the phugoid mode, small variations in velocity and height tend to be stable and energy conserving. Based on this insight, it is desirable to allow small variations of the airspeed of the vehicle as long as it is associated with the phugoid mode response. The key characteristic of the phugoid mode is that it preserve *total energy*

$$E := \frac{1}{2}m|v_a|^2 + mgh \tag{52.43}$$

of the aircraft. The TECS control architecture uses the total energy as an output and uses the throttle to regulate this output to a desired energy set point. The set point

$$E^\star(t) := \frac{1}{2}m|v_a^\star|^2 + mgh^\star$$

is chosen to correspond to the desired trajectory. The energy error is

$$\tilde{E} := E^\star(t) - E(t)\,,$$

and the throttle input is assigned as

$$\delta_t = k_{P_E}\tilde{E}(t) + k_{I_E}\int_{-\infty}^{t}\tilde{E}(\tau)\,\mathrm{d}\tau\,, \tag{52.44}$$

where k_{P_E} and k_{I_E} are positive control gains.

Once the throttle input is assigned, the remaining longitudinal dynamics must be controlled using the elevator to servo the pitch rate Ω_y. The primary effect of pitch rate is to rotate the wing section of the aircraft, directly effecting the angle-of-attack. Changing the angle-of-attack effects lift and drag generated by the wing, with lift most strongly effected since it depends linearly on α (52.14) while drag is small and only varies with the square of the angle-of-attack (52.15). By definition, the lift L is the component of airfoil force that lies perpendicular to the velocity. As a consequence the lift force L *does not contribute directly to changes in the total energy* E of the vehicle. In particular, actuation of the pitch dynamics of the vehicle is (almost) decoupled from total energy regulation loop. It is now possible to choose an output for the elevator input and have confidence that the resulting SISO loop response will only marginally disturb the total energy regulation loop. Rather than simply regulate the height or velocity, both variables that have been considered as outputs for elevator regulation loops in classical fixed-wing control architectures [52.31], it is of interest to regulate a balance between kinetic and potential energy [52.42].

Consider the kinetic energy and potential energy of the vehicle separately

$$K = \frac{1}{2}m|v_a|^2\,, \quad U := mgh\,.$$

A given flight trajectory corresponding to desired values of $(|v_a^\star(t)|, h^\star(t))$ defines desired set points for kinetic and potential energy $K^\star(t) = \frac{1}{2}m|v_a^\star(t)|^2$ and $U^\star = mgh^\star(t)$. Define the errors in the kinetic and potential energy along the trajectory as

$$\begin{aligned}\tilde{K} &= K^\star(t) - K(t) = \frac{1}{2}m\left[|v_a^\star(t)|^2 - |v_a(t)|^2\right],\\ \tilde{U} &= U^\star(t) - U(t) = mg\left[h^\star(t) - h(t)\right],\end{aligned} \tag{52.45}$$

and define the combined energy balance error $\tilde{B}$ by

$$\tilde{B} = \tilde{K} - \tilde{U}\,.$$

The proposed control strategy is to use the commanded pitch angle $\theta^\star$ to regulate $\tilde{B} \to 0$.

To understand this control architecture, note that if $\tilde{E} = 0$ then $\tilde{K} = \tilde{E} - \tilde{U} = -\tilde{U}$ and hence

$$\tilde{B} = 2\tilde{K} = -2\tilde{U}\,.$$

It follows that driving $\tilde{B}$ to zero forces both $|v_a| \to |v_a^\star|$ and $h \to h^\star$. The real advantage of the balanced energy control architecture occurs when $\tilde{E} \neq 0$. In this case, driving $\tilde{B} \to 0$ will not force either $|v_a|$ or h to its reference set point at all costs, but rather, it will balance the excess or deficit energy between kinetic and potential energy set points around the desired trajectory.

Balancing the energy error in this way is a highly robust control strategy that maximizes the ability of the throttle to maintain good control of the total energy.

An additional insight is that the difference of the energy terms $\tilde{B}$ is maximally sensitive to excitation in the phugoid mode. In particular, if power flows between potential and kinetic energy levels, then the relative variation in $\tilde{B}$ is maximal compared to any other combined measure incorporating $|v_a|$ and h subject to scaling factors. Thus, the balanced energy control strongly directly damps the phugoid mode that is uncontrolled by the thrust control.

The desired pitch angle is therefore given by

$$\theta^\star = k_{P_B}\tilde{B}(t) + k_{I_B}\int_{-\infty}^{t}\tilde{B}(\tau)\,\mathrm{d}\tau\,, \qquad (52.46)$$

where k_{P_B} and k_{I_B} are positive control gains. In practice, the pitch angle actuates stable angle-of-attack dynamics that in turn lead to regulation of the flight-path angle and finally control the tradeoff encoded in $\tilde{B}$. The stability of the internal dynamics for the proposed SISO control loop are handled by suitable tuning of the PI gains, avoiding the need to explicitly model these dynamics. The effectiveness and robustness of this approach is well established in the literature [52.31].

Control Implementation and Gain Tuning

The lateral control strategy for fixed wing aircraft is based on (52.40), (52.41), and (52.42). In using these commands, we are assuming high-bandwidth feedback loops regulating Ω_x and Ω_z using the ailerons and rudder, respectively. Practical implementation of the lateral control loops required saturating the roll command $\phi^\star$ to $\pm\bar{\phi}$, where, for most airframes, a reasonable value is $\bar{\phi} = 30°$. Commanding higher roll angles usually results in large side-slip angles, which leads to unacceptably large drops in altitude during turns. The roll rate command Ω_x is also saturated to a fraction of the limits of the rate gyro sensor to ensure adequate convergence of the attitude estimator. Practical implementation of (52.40) also requires careful wrapping of the angles χ and $\chi^\star$ around $\pm 180°$ to avoid large heading errors that result in not associating $180°$ with $-180°$.

The gains of the lateral control loops can be tuned in flight by selectively enabling one loop at a time. A strategy that has proven success over numerous flight test is to tune the gains in the following order:

1. Attitude rate loops regulating Ω_x and Ω_z.
2. Side slip gain k_{P_β}.
3. The proportional gain on roll error k_{P_ϕ}, followed by the integral gain k_{I_ϕ}. The integral gain k_{I_ϕ} is often set to zero, since the integrator on the heading loop provides suitable robustness.
4. The proportional gain on the heading loop k_{P_χ}, followed by the integral gain k_{I_χ}.

The use of (52.46) assumes a high-bandwidth pitch control loop using the elevator as an actuator. Practical implementation of (52.44) requires saturating δ_t above by $\delta_t^{\max}$ and below by $\delta_t^{\min} \geq 0$. Saturating the pitch angle is also critical to avoid stall conditions that occur at large angles of attack. An effective method to do this is to saturate the altitude error in computing the error in potential energy, replacing (52.45) with

$$\tilde{U} = mg\,\mathrm{sat}\left[h^\star(t) - h(t), \bar{h}\right], \qquad (52.47)$$

where

$$\mathrm{sat}(x,\ell) = \begin{cases} \ell & \text{if } x \geq \ell\,, \\ -\ell & \text{if } x \leq -\ell\,, \\ x & \text{otherwise}\,. \end{cases}$$

Saturating the altitude error will cause a steady climb rate when $h^\star - h$ is large. To understand why, assume that velocity is well regulated, and that the altitude error is large, then $\tilde{B} \approx -mg\,\mathrm{sat}(h^\star - h, \bar{h}) = -mg\bar{h}$. Ignoring the integrator, this will cause a constant pitch angle command from (52.46). However, simply saturating the pitch angle will not result in good performance since driving

$$\tilde{B} = \tilde{K} - \tilde{U} = \frac{1}{2}m\left(|v_a^\star(t)|^2 - |v_a(t)|^2\right) - mg[h^\star(t) - h(t)]$$

to zero will cause errors in kinetic and potential energy to balance. Therefore, a large unsaturated altitude error will result in the controller increasing the airspeed error. A sustained altitude error will cause the integrator in (52.46) to wind up. Therefore, an integrator antiwindup scheme is essential.

Similar to the lateral control loop, the gains for the longitudinal controller can be tuned in flight by selectively enabling one loop at time. The gains should be tuned in the following order:

1. The pitch attitude loops is first tuned. Pitch attitude is typically controlled using a PID controller. The derivative gain can be tuned to provide adequate damping on the RC controller. The proportional gain is then added and adjusted to provide adequate transient response. The integrator is added afterward to remove steady state error in pitch.
2. The throttle gains k_{P_E} and k_{I_E} in (52.44) are then adjusted. We have found that normalizing the energy error $\tilde{E}$ by the reference kinetic energy $K^{\mathrm{ref}} = \frac{1}{2}m|v_a^{\mathrm{ref}}|$ makes these gains particularly easy to tune,

and that the scaling results in similar gains working across different platforms.

3. The energy balance gains k_{P_B} and k_{I_B} are tuned last. Again, scaling $\tilde{B}$ by K^{ref} seems to simplify the tuning process for different airframes.

An advantage of the TECS is that it reduces the need for different control modes for the longitudinal autopilot. In particular, the *altitude hold mode*, the *climb mode*, and the *descend mode* described in [52.31] are reduced to one mode using TECS. However, we should note that the *take-off mode* described in [52.31] is still needed for this scheme. For small aircraft that are hand launched, or any aircraft whose velocity at take off is significantly below the commanded velocity, the TECS control in (52.46) will cause the vehicle to pitch down to gain airspeed when the airspeed error is large. This pitch-down behavior will cause the airframe to crash if it is executed immediately after take-off. Therefore, the best strategy is to apply full throttle immediately after take-off, and to set the commanded pitch angle to fixed value that is adequate for a climb rate that avoids stall.

52.4 Trajectory Planning

There are many different control tasks that aerial robotic vehicles will be required to undertake in the coming years. In this chapter, it is impossible to cover the whole range of potential tasks along with their associated planning problems. We will consider only basic trajectory or path planning problems for both quadrotor and fixed-wing vehicles. This problem is fundamental to fixed-wing navigation where the distances involved are such that path and trajectory planning is sufficient for almost all goals. For small aerial robotic vehicles flying in cluttered 3-D-spaces, there is a much wider selection of goals including things such as, physical interaction with the environment, obstacle avoidance, etc., as well as simply planning and following a trajectory to a given way-point. However, the trajectory planning provides a basic building block in achieving a wide range of goals, and is a necessary component of achieving high-performance tasks.

52.4.1 Trajectory Planning for Quadrotors

The quadrotor is underactuated and this makes it difficult to plan trajectories in 12-dimensional state space (6DOF position and velocity). However, the problem is considerably simplified if we use the fact that the quadrotor dynamics are differentially flat [52.44]. To see this, we consider the output position $\boldsymbol{\xi}$ and the yaw angle ψ. We show that we can write all state variables and inputs as functions of the outputs (ξ, ψ) and their derivatives. Derivatives of $\boldsymbol{\xi}$ yield the velocity v, and the acceleration,

$$\dot{v} = \frac{1}{m} u_1 \boldsymbol{b}_3 + g e_3 \,.$$

From Fig. 52.3, we see that

$$\boldsymbol{d}_1 = [\cos\psi, \sin\psi, 0]^{\mathrm{T}}$$

and the unit vectors for the body-fixed frame, can be written in terms of the variables ψ and $\dot{v}$ as follows

$$\boldsymbol{b}_3 = \frac{\dot{v} - g e_3}{\|\dot{v} - g e_3\|}, \; \boldsymbol{b}_2 = \frac{\boldsymbol{b}_3 \times \boldsymbol{d}_1}{\|\boldsymbol{b}_3 \times \boldsymbol{d}_1\|}, \; \boldsymbol{b}_1 = \boldsymbol{b}_2 \times \boldsymbol{b}_3$$

provided $\boldsymbol{b}_3 \times \boldsymbol{d}_1 \neq 0$. This defines the rotation matrix ${}^{A}R_B$ as a function of $\dot{v}$ (the second derivative of ξ) and ψ. In this way, we write the angular velocity and the four inputs as functions of the position, velocity ($v = \dot{\xi}$), acceleration ($a = \ddot{\xi}$), jerk ($\gamma = \xi^{(iii)}$), and snap or the derivative of jerk ($\sigma = \xi^{(iv)}$). From these equations, it is possible to verify that there is a diffeomorphism between the 18×1 vector

$$\boldsymbol{X} = \left(\xi^{\mathrm{T}}, v^{\mathrm{T}}, a^{\mathrm{T}}, \gamma^{\mathrm{T}}, \sigma^{\mathrm{T}}, \psi^{\mathrm{T}}, \dot{\psi}^{\mathrm{T}}, \ddot{\psi}^{\mathrm{T}}\right)^{\mathrm{T}}$$

and the state augmented with the inputs and their derivatives

$$\left(\xi^{\mathrm{T}}, \dot{\xi}^{\mathrm{T}}, R, \Omega^{\mathrm{T}}, u_1, \dot{u}_1, \ddot{u}_1, \mathbf{u}_2^{\mathrm{T}}\right)^{\mathrm{T}} .$$

This property of differential flatness makes it easy to design trajectories that respect the dynamics of the underactuated system. Any four-times-differentiable trajectory in the space of flat outputs, $(\xi^{\mathrm{T}}(t), \psi(t))^{\mathrm{T}}$, corresponds to a feasible trajectory, one that satisfies the equations of motion. All inequality constraints of states and inputs can be expressed as functions of the flat outputs and their derivatives. This mapping to the space of flat outputs can be used to generate trajectories that minimize a cost functional formed by a weighted combination of the different flat outputs and

their derivatives

$$(\xi^p(t), \psi^p(t)) = \arg \min_{\xi(t), \psi(t)} \int_0^T L(\xi, v, a, \gamma, \sigma, \psi, \dot{\psi}, \ddot{\psi})\, \mathrm{d}t \,, \quad g(\xi(t), \psi(t)) \leq 0 \,. \tag{52.48}$$

In [52.7], minimum snap trajectories were generated by minimizing a cost functional derived from the snap and the angular yaw acceleration with

$$L(\xi, \dot{\xi}, \ddot{\xi}, \dddot{\xi}, \ddddot{\xi}\psi, \dot{\psi}, \ddot{\psi}) = (1-\alpha)(\dddot{\xi})^4 + \alpha(\ddot{\psi})^2 \,.$$

By suitable parameterizing trajectories with basis functions in the flat space and by considering linear inequalities in the flat space to model constraints on states and inputs (e.g., $u_1 \geq 0$), it is possible to turn this optimization into a quadratic program which can be solved in real-time for planning.

Finally, as shown in [52.45], it is possible to combine this controller with attitude only controllers to fly through vertical windows or land on inclined perches with close to zero normal velocity. A trajectory controller is used by the robot to build up momentum while the attitude controller enables reorientation when coasting with the generated momentum.

52.4.2 Trajectory Planning for Fixed-Wing Aircraft

For fixed wing vehicles, the critical constraints that influence path planning is the minimum turn radius and the maximum flight path angle. As shown in [52.29, 31], a good approximation for the minimum turn radius is given by

$$R_{\min} = \frac{|v_a|^2}{g} \tan \bar{\phi} \,,$$

where g is the acceleration of gravity at sea level, and $\bar{\phi}$ is the maximum allowable bank angle. From (52.27c), we see that a good approximation for the maximum flight path angle is

$$\bar{\gamma} = \sin^{-1}\left(\frac{\dot{h}_{\max}}{|v_a|}\right),$$

where $\dot{h}_{\max}$ is the maximum climb/descent rate of the airframe. Fixed wing aircraft are typically designed to fly at a specific airspeed where the lift-to-drag ratio is maximized, thereby minimizing fuel expenditure. Therefore path planners for fixed wing vehicles commonly assume a constant airspeed. In this section, we will assume zero wind conditions where airspeed equals ground speed. Therefore, the path planning problem for fixed wing vehicles is naturally posed as planning a constant velocity path between two configurations that satisfy the minimum turn radius and maximum flight path angle constraints.

If the maneuver is executed at a constant altitude, then a traditional Dubins car planner can be used to find a constant velocity path satisfying the turn rate constraints [52.46]. Using the Dubins car model for UAV path planning is a common practice in the literature [52.31, 47–50]. However, the constant altitude limitation can be removed by extending the traditional Dubins car model to the Dubins airplane model given by

$$\begin{aligned} \dot{n} &= V \cos\chi \,, \\ \dot{w} &= V \sin\chi \,, \\ \dot{\chi} &= u_1 \,, \\ \dot{h} &= -V \sin u_2 \,, \end{aligned}$$

where (n, w) is the inertial North–West position, V is the speed of the Dubins airplane, and in the case of zero ambient wind is $V = |v_a|$, and where $|u_1| \leq V/R_{\min}$, and $|u_2| \leq \bar{\gamma}$ [52.51].

The configuration of a Dubins airplane is given by $C = (n, w, h, \chi)$, where n and w are North–West inertial coordinates, h is the altitude, and χ is the course angle. The Dubins airplane planning problem is to find a minimum distance path between a start configuration C_s and an end configuration C_e that satisfies the constraints $|u_1| \leq V/R_{\min}$, and $|u_2| \leq \bar{\gamma}$.

Recall that Dubins car paths are composed of three segments: a turn segment that follows a circle with the minimum turn radius $R_{\min}$, a straight line segment, and finally another turn segment that again follows a circle with minimum turn radius [52.46]. The planning problem for the Dubins airplane is closely related to the planning problem for the Dubins car, with some variation due to the altitude component. As explained in [52.51, 52], Dubins airplane paths can be separated into three categories based on the altitude gain/loss between the start and end configurations, i. e., $\Delta h = |h_e - h_s|$.

If Δh is small enough, then the Dubins airplane can follow the standard Dubins car path while gaining or losing altitude at a constant rate that satisfies the flight path angle constraint and that allows the altitude difference to be realized. Such paths are termed *low-altitude paths* in [52.51]. If on the other hand, Δh is so large that flying the Dubins car path at the flight path angle constraints does not result in sufficient altitude gain/loss,

then the Dubins car path must be suitably modified. One alternative is to fly at the flight path angle constraint while making multiple complete orbits on the minimum turn radius helix at the beginning of the Dubins car path before proceeding along the straight line segment of the path. Of course, a similar strategy would be to make multiple complete orbits on the minimum turn radius helix at the end of the Dubins car path. Such paths are termed *high-altitude paths* in [52.51]. The middle case is when the flight path angle constraint does not allow sufficient altitude gain/loss following the Dubins car path, but where one complete orbit along the start or end circle of the Dubins car path at the maximum flight path angle results in more altitude gain/loss than is needed. In this case, a deviation can be placed in the Dubins car path to extend the path length to just the right amount so that the altitude gain/loss can be achieved while flying at the flight path angle constraint. These paths are termed *medium-altitude paths* in [52.51].

Following the notation defined in [52.52] let $L_{\text{car}}(C_\text{s}, C_\text{e}, R)$ denote the path length of the Dubins car path between the projection of the start configuration C_s onto the North–West plane, and the projection of the end configuration C_e onto the North–West plane, using R as the turn radius of the vehicle.

Low-Altitude Dubins Paths

The altitude gain between the start and end configuration is said to be *low altitude* if

$$|h_\text{e} - h_\text{s}| \leq L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) \tan\bar{\gamma} \,,$$

where the term on the right is the maximum altitude gain that can be obtained by flying at flight-path angle $\pm\bar{\gamma}$ for a distance of $L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min})$.

In the low-altitude case, the altitude gain between the start and end configurations can be achieved by flying the Dubins car path with a flight-path angle satisfying $|u_2| \leq \bar{\gamma}$. Therefore, the optimal flight-path angle can be computed by

$$u_2^* = \tan^{-1}\left(\frac{|h_\text{e} - h_\text{s}|}{L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min})}\right).$$

Medium-Altitude Dubins Paths

The altitude gain between the start and the end configuration is said to be *medium altitude* if

$$\begin{aligned} &L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) \tan\bar{\gamma} < |h_\text{e} - h_\text{s}| \\ &\quad \leq [L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) + 2\pi R_{\min}] \tan\bar{\gamma} \,, \end{aligned} \tag{52.49}$$

where the addition of the term $2\pi R_{\min}$ accounts for adding one orbit at radius $R_{\min}$ to the path length.

In the medium-altitude case, the altitude difference between the start and end configurations is too large to obtain by flying the Dubins car path at the flight-path angle constraint, but small enough that adding a full turn on the helix at the beginning or end of the path and flying so that $\gamma = \pm\bar{\gamma}$ results in more altitude gain/loss than is needed. As shown in [52.51], the minimum distance path is achieved by setting $\gamma = \text{sign}\,(h_\text{e} - h_\text{s})\,\bar{\gamma}$ and inserting an extra maneuver in the Dubins car path that extends the path length so that the altitude gain when $\gamma = \pm\bar{\gamma}$ is exactly $|h_\text{e} - h_\text{s}|$. While there are numerous possible ways to extend the path length, the method proposed in [52.52] is to add an additional intermediate arc to the start or end of the path, as shown in Figs. 52.8 and 52.9. If the start altitude is lower than the end altitude, then the intermediate arc is inserted immediately after the start helix. If on the other hand, the start altitude is higher than the end altitude, then the intermediate arc is inserted immediately before the end helix.

High-Altitude Dubins Paths

The altitude gain between the start and end configurations is said to be *high altitude* if

$$|h_\text{e} - h_\text{s}| > [L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) + 2\pi R_{\min}] \tan\bar{\gamma} \,.$$

In the high-altitude case, the altitude gain cannot be achieved by flying the Dubins car path within the flight-path angle constraints. As shown in [52.51], the minimum distance path is achieved when the flight-path angle is set at its limit of $\pm\bar{\gamma}$, and the Dubins car path is extended to facilitate the altitude gain. While there are many different ways to extend the Dubins car path, [52.52] suggests extending the path by spiraling a certain number of turns at the beginning or end of the path, and then by increasing the turn radius by the appropriate amount.

For UAV scenarios, the most judicious strategy is typically to spend most of the trajectory at as high an altitude as possible. Therefore, if the altitude at the end configuration is higher than the altitude at the start configuration, then the path will be extended by a climbing helix at the beginning of the path. If on the other hand, the altitude at the start configuration is higher than the end configuration, then the path will be extended by a descending helix at the end of the path. If multiple turns around the helix are required, then the turns could be split between the start and end helices and still result in the same path length. For high-altitude Dubins paths, the required number of turns in the helix will be the smallest integer k such that

$$\begin{aligned} &[L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) + 2\pi k R_{\min}] \tan\bar{\gamma} \leq |h_\text{e} - h_\text{s}| \\ &\quad < [L_{\text{car}}(C_\text{s}, C_\text{e}, R_{\min}) + 2\pi (k+1) R_{\min}] \tan\bar{\gamma} \,, \end{aligned} \tag{52.50}$$

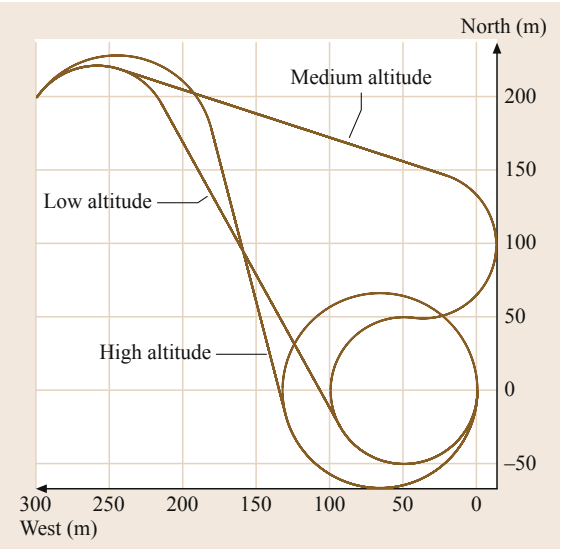


Fig. 52.8 A top down view of low altitude, medium altitude, and high-altitude Durbin airplane paths. The initial and end configurations are identical except for the final altitude

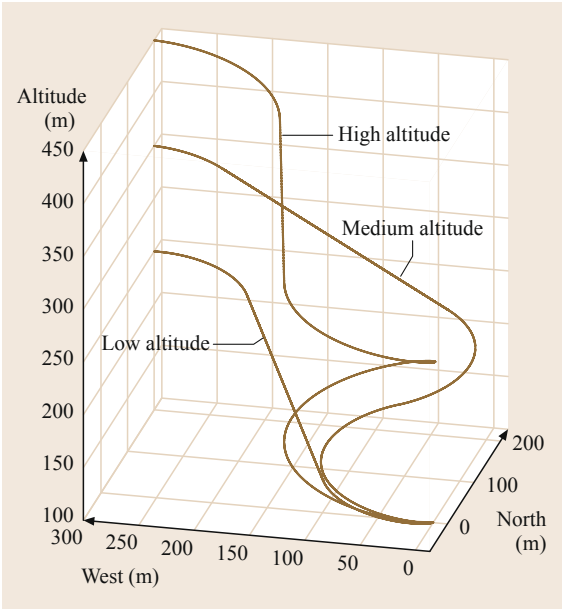


Fig. 52.9 A three-dimensional view of low altitude, medium altitude, and high-altitude Durbin airplane paths. The initial and end configurations are identical except for the final altitude

or in other words

$$k = \left\lfloor \frac{1}{2\pi R_{\min}} \left[\frac{|h_e - h_s|}{\tan \bar{\gamma}} - L_{car}(C_s, C_e, R_{\min}) \right] \right\rfloor ,$$

where $\lfloor x \rfloor$ is the floor function that rounds x down to the nearest integer. The radius of the start and end helices is then increased to $R^\star$ so that

$$\left[L_{car}(C_s, C_e, R^\star) + 2\pi k R^\star \right] \tan \bar{\gamma} = |h_e - h_s| \ . \tag{52.51}$$

Figures 52.8 and 52.9 show two perspective of three different Dubins airplane paths that start and end in the same configuration with the exception of the final altitude, which changes from 250 m for the low-altitude path, to 350 m for the medium-altitude path, to 450 m for the high-altitude path. A top down view is shown in Fig. 52.8. Note that for both the low-altitude and medium-altitude paths, that the minimum turn radius constraint is active, whereas for the high-altitude path, the turn radius in the turns is larger than the minimum turn radius constraint. A three-dimensional view is shown in Fig. 52.9. Note that for both the medium- and high-altitude paths, the flight path angle constraint is active, but that it is not active for low-altitude paths.

There are a variety of techniques that allow the control architecture described in Sect. 52.3.2 to follow Dubins airplane paths. A simple method is to parameterize the path using a path parameter σ. Suppose that the resulting parameterized path is given by $p(\sigma)$. At each sample time, the path parameter is advanced along the path so as to minimize the distance to the path from the aircraft

$$\sigma_{t+1} = \arg \min_{s \geq \sigma_t} \| \xi(t) - p(s) \| \ ,$$

where $\xi(t)$ is the inertial position of the aircraft. The commanded airspeed $|v_a^\star|$, course angle $\chi^\star$, and altitude $h^\star$ at time t are then given by the parameterized Dubins airplane path $p(\sigma_{t+1})$. This technique is similar to the guidance strategy suggested in [52.53]. Alternative method based on vector field methods are described in [52.54–56].

52.5 Estimating the Vehicle State

A critical aspect in implementation of real world aerial robotic vehicles is providing good estimates of the state of the vehicle. The key state estimates required for control of an aerial vehicle are associated with the rigid-body dynamics of its airframe, height, attitude, angular velocity, and linear velocity. Moreover, fixed-wing vehicle dynamics depend on the aerodynamics states of angle-of-attack and side-slip angle. Of these states, the attitude and angular velocity are the most important as they are the primary variable used in attitude control and flight-regulation of the vehicle. Angle-of-attack and side-slip angles are rarely estimated explicitly and are dealt with as internal dynamics in the control design rather than explicit outputs or disturbances.

The ubiquitous instrumentation carried by any aerial vehicle is an inertial measurement unit (IMU) often augmented by some form of height measurement, either acoustic, infra red, barometric, or laser based. Vehicles that fly in outside environments carry a GPS system, most of which now provide velocity estimation as well as position. Many fixed wing vehicles also include a pitot tube (for dynamic pressure) or an anemometer to measure forward velocity. Indoor aerial robotic vehicles in research laboratories are often flown in flight environments equipped with motion capture systems like VICON or Optitrack [52.5, 6]. Finally, many aerial robotic systems have also been equipped with exteroceptive sensor systems including Kinect 3D-range cameras, scanning laser rangefinder, integrated stereo camera systems, or simple monocular cameras. There is an active research field in developing 3-D-SLAM (simultaneous localization and mapping; Chap. 46.4) algorithms for mapping and localization of aerial robotic vehicles.

The rigid-body dynamic state is crucial to the control performance of an aerial robotic system and we will concentrate on this estimation problem in this chapter. A natural approach to state-estimation is to apply classical filter design to a coordinate representation of the rigid-body state. A good development of this approach for small-scale aerial robotic systems is presented in [52.31]. An alternative approach is to exploit the nonlinear structure of the rigid-body kinematics to develop nonlinear observers. This approach has been shown to generate simple, robust, and highly effective filters for small-scale aerial robotic systems, especially for the key problem of attitude estimation.

52.5.1 Estimating Attitude

A typical IMU includes a three-axis rate gyro, three-axis accelerometer, and three-axis magnetometer. The rate gyro measures angular velocity of $\{B\}$ relative to $\{A\}$ expressed in the body-fixed-frame of reference $\{B\}$

$$\Omega_{\text{IMU}} = \Omega + b_{\Omega} + \eta \in \{B\} ,$$

where η denotes additive measurement noise and b_{Ω} denotes a constant (or slowly time-varying) gyro bias. Generally, the gyroscopes installed on quadrotor vehicles are lightweight MEMS devices that are reasonably robust to noise and quite reliable.

The accelerometers (in a *strap down IMU configuration*) measure the instantaneous linear acceleration of $\{B\}$ due to exogenous force

$$a_{\text{IMU}} = R^{\text{T}}(\dot{v} + ge_3) + b_{\text{a}} + \eta_{\text{a}} \in \{B\} , \qquad (52.52)$$

where b_{a} is a bias term, η_{a} denotes additive measurement noise, and $\dot{v}$ is in the inertial frame. Accelerometers are highly susceptible to vibration and mounted on a typical aerial robotic platform they require significant low-pass mechanical and/or electrical filtering to be reliable. Most vehicles avionics will incorporate an analogue anti-aliasing filter on a MEMS accelerometer before the signal is sampled.

The magnetometers provide measurements of the ambient magnetic field

$$m_{\text{IMU}} = R^{\text{T}}\,{}^{A}m + B_m + \eta_b \in \{B\} ,$$

where ${}^{A}m$ is the Earth's magnetic field vector (expressed in the inertial frame), B_m is a body-fixed-frame expression for the local magnetic disturbance and η_b denotes measurement noise. The noise η_b is usually quite low for magnetometer readings, however, the local magnetic disturbance B_m can be very significant, especially if the sensor is placed near power cables for the motors.

In outdoor environments, the GPS is typically used to provide position, inertial velocity, and course angle. The GPS system consists of a constellation of 24 satellites that continuously orbit the earth at an altitude of 20 180 km. The position of a GPS receiver is determined by observing the time-of-flight of signals sent from the satellites and detected by the receiver. If accurate timing information is available at the receiver, then a minimum of three satellites signals are necessary to resolve the position of the receiver. However, for low-cost GPS receivers, timing information must also be received and this requires a minimum of four satellite signals. A variety of factors effect the accuracy of GPS estimates. The dominant sources of error include inaccurate satellite orbital data, inaccuracy in the satellite clocks, variable signal delay as it passes through

the ionosphere, weather conditions near the earth surface, and multipath reflections from nearby buildings and mountains. Total GPS bias error is approximately 5–10 m. Modern GPS receivers exploit doppler shift in the carrier phase of the received signals to estimate the course angle χ and the ground speed.

The accelerometers and magnetometers can be used to provide absolute attitude information on the vehicle while the rate gyroscope provides complementary angular velocity measurements. The attitude information in the magnetometer signal is straightforward to understand; in the absence of noise and bias then m_{IMU} provides a body-fixed frame measurement of $R^{\text{T}A}m$, and consequently constrains two degrees of freedom in the rotation R. The accelerometer can also be used as long as the component of acceleration associated with the inertial motion of the vehicle is compensated. The simplest way in which to do this is to use an absolute external signal such as GPS to estimate the vehicle acceleration

$$a_{\text{CTD}} = a_{\text{IMU}} - \hat{R}^{\text{T}}\ddot{\xi}_{\text{GPS}} \,,$$

where a_{CTD} is the corrected acceleration. In this case, $a_{\text{CTD}} \approx R^{\text{T}}e_3$ and this provides attitude information. It is important to note that in the complementary filter proposed below (52.53), only the low-frequency component of a_{CTD} is required. As such, the potential noise in the second derivative of a GPS signal $\ddot{\xi}_{\text{GPS}}$ is not necessarily the problem that may be initially feared. The key to good filter performance is to ensure that the cross over frequency of the complimentary sensitivity is low enough that any phase distortion associated with low-pass filtering of $\ddot{\xi}_{\text{GPS}}$ does not effect the estimate.

Remark 52.1
There may be issues in applying the correction $\hat{R}^{\text{T}}\ddot{\xi}_{\text{GPS}}$ if there is a delay in receiving the GPS signals. This can be addressed, at the cost of storing IMU data, by combining a short time prediction along with a time-lagged observer [52.57].

In the case where GPS or an external measure of the vehicles motion is not available, there are a number of alternative techniques available to use the accelerometer information for attitude estimation. For fixed-wing vehicles, the most important disturbance is due to centripetal acceleration due to the vehicle turning. It is possible to model the centripetal acceleration using airspeed and angular velocity and compensate using a feedforward term [52.58, 59]. In the case of rotorcraft in mostly hover flight conditions, it turns out that using the raw IMU accelerometer readings still works extremely well [52.60–62] due to the disturbance caused by the aerodynamic drag term $\Delta = -Dv$ (52.10) [52.21, 24, 25, 63]. In this case the corrected accelerometer a_{CTD} used in the filter (52.53) below is just the raw IMU accelerometer measurement.

The attitude kinematics of the quadrotor are given by (52.2c). Let $\hat{R}$ denote an estimate for attitude R of the quadrotor vehicle. The following observer [52.61, 62] fuses accelerometer, magnetometer, and gyroscope data as well as other direct attitude estimates R_{E} (such as provided by a VICON or other external measurement system) should they be available

$$\dot{\hat{R}} := \hat{R}\left(\Omega_{\text{IMU}} - \hat{b}\right)_{\times} - \Upsilon \,, \tag{52.53a}$$

$$\dot{\hat{b}} := k_b \Upsilon \,, \tag{52.53b}$$

$$\begin{aligned}\Upsilon := \Bigg(&\frac{k_a}{g^2}\left[(\hat{R}^{\text{T}}e_3)\times a_{\text{CTD}}\right] \\ &+\frac{k_m}{{}_A m^2}\left[(\hat{R}^{\text{T}A}m)\times m_{\text{IMU}}\right]\Bigg)_{\times} \\ &+ k_E \mathbb{P}_{\mathfrak{so}(3)}\left(\hat{R}R_{\text{E}}^{\text{T}}\right),\end{aligned} \tag{52.53c}$$

where k_a, k_m, k_E, and k_b are arbitrary nonnegative observer gains and $\mathbb{P}_{\mathfrak{so}(3)}(M) = (M - M^{\text{T}})/2$ is the Euclidean matrix projection onto the skew-symmetric matrices. If any one of the measurements in the innovation Υ are not available or unreliable, then the corresponding gain should be set to zero in the observer. Note that both the attitude $\hat{R}$ and the bias corrected angular velocity $\hat{\Omega} = \Omega_{\text{IMU}} - \hat{b}$ are estimated by this observer. The observer (52.53) has been extensively studied in the literature [52.61, 62] and shown to converge exponentially (both theoretically and experimentally) to the desired attitude estimate of attitude with $\hat{b}$ converging to the gyroscope bias b. The filter has a complementary nature, using the high-frequency part of the gyroscope signal and the low-frequency parts of the magnetometer, accelerometer, and external attitude measurements [52.61]. The roll off frequencies associated with each of these signals is given by the gains k_a, k_m, and k_E in rad/s.

52.5.2 Estimating Velocity and Position

Estimating velocity and position is a straightforward process if the vehicle is equipped with a GPS system. In this case, the dynamics (52.2a) and (52.2b) are linear, and a linear filter can be used. Let $\hat{\xi}$ denote the estimate of position and $\hat{v}$ denote the estimate of velocity. A sim-

ple linear filter is given by

$$\dot{\hat{\xi}} = \hat{v} - k_x(\hat{\xi} - \xi_{\text{GPS}}) ,$$
$$\dot{\hat{v}} = \hat{R}^{\text{T}} a_{\text{IMU}} - g e_3 - k_v(\hat{\xi} - \xi_{\text{GPS}}) ,$$

for gains $k_x, k_v > 0$. As long as the attitude estimate $\hat{R}$ is accurate, this filter design is highly robust. It is easily possible to add bias estimates for the accelerometer if desired. Equally well it is straightforward to use Kalman filter techniques to tune the gains k_x and k_v in real-time if that is considered advisable. In practice, for small aerial robotic systems, the noise characteristics of the measurements are so poor that it is often best to use a constant gain filter rather than introduce the additional complexity and potential instability of the Riccati equation associated with a Kalman filter.

In the absence of GPS, estimating position depends on the availability of additional sensor systems. Since most aerial robots are equipped with a barometer to provide an estimate of height and this can be utilized for an estimate of the vertical motion of the vehicle

$$\dot{\hat{h}} = \hat{v}_z - k_h(\hat{h} - h) , \tag{52.54a}$$
$$\dot{\hat{v}}_z = e_3^{\text{T}} \hat{R}^{\text{T}} a_{\text{IMU}} - g - k_{v_z}(\hat{h} - h) , \tag{52.54b}$$

where $k_h, k_{v_z} > 0$ are positive gains.

For quadrotor vehicles, there is also the possibility of exploiting the linear drag aerodynamic forces (52.10) associated with blade flapping and induced drag to estimate horizontal velocity for a vehicle in near hover conditions [52.24]. Define a projector matrix

$$\mathbb{P}_h := \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \tag{52.55}$$

that takes the first two components of a vector. Then the horizontal component of the *inertial* acceleration can be measured by

$${}^{A}a_h := \mathbb{P}_h \, {}^{A}a = \mathbb{P}_h R a \approx \mathbb{P}_h \hat{R} a , \tag{52.56}$$

where we assume that the estimate $\hat{R}$ is close to R. If we assume the vehicle is only slowly varying in height one has $v_z \approx 0$ in comparison to horizontal velocity

$$v_h \approx \mathbb{P}_h^{\text{T}} v .$$

Furthermore, the thrust $T \approx mg$ must compensate the weight of the vehicle. Recalling (52.10) and taking just the horizontal component one has

$${}^{A}a_h \approx -g\mathbb{P}_h \hat{R} e_3 - g\mathbb{P}_h \hat{R} D R^{\text{T}} \mathbb{P}_h^{\text{T}} v_h . \tag{52.57}$$

If the attitude filter estimate is good and the assumptions about vehicle motion hold then (52.56) and (52.57) can be solved for an estimate of v_h

$$v_h \approx -\frac{1}{g} \left(\mathbb{P}_h \hat{R} D \hat{R}^{\text{T}} \mathbb{P}_h^{\text{T}} \right)^{-1} \left({}^{A}a_h + g\mathbb{P}_h \hat{R} e_3 \right) . \tag{52.58}$$

Equation (52.58) provides a measurement of the horizontal velocity, however, since it directly incorporates the unfiltered accelerometer readings it is generally too noisy to be of much use. Its low-frequency content can, however, be used to drive a velocity complementary observer that uses the attitude estimate and the system model (52.2b). Let $\hat{v}_h$ be an estimate of the horizontal component of the inertial velocity of the vehicle then an observer is given by

$$\dot{\hat{v}}_h = -g\mathbb{P}_h^{\text{T}} \left(\hat{R} e_3 + \hat{R} D \hat{R}^{\text{T}} \mathbb{P}_h^{\text{T}} \hat{v}_h \right) - k_{\text{w}}(\hat{v}_h - v_h) , \tag{52.59}$$

where v_h is given by (52.58). The gain $k_{\text{w}} > 0$ provides a tuning parameters that indicates the roll-off frequency for the information from $\hat{v}_h$ that is used in the filter. It also uses estimated velocity $\hat{v}_h$ to provide an approximation of the more correct $RDR^{\text{T}} \mathbb{P}_h^{\text{T}} v_h$ term in the feedforward velocity estimate, however, since the underlying dynamics associated with this term is stable, then the observer is stable even with this approximation. It is found that this observer works remarkably well in practice for quadrotor vehicles that are not engaged in acrobatic maneuvers [52.21, 24, 25, 63].

52.6 Conclusion

This chapter has focused on the fundamental technology associated with making an aerial robotic system operate effectively and efficiently. In particular, we have focused on the design of control and navigation algorithms, and along with that the associated questions of modeling and state estimation. Once a robust and reliable aerial robotic platform is available, the range of potential applications for unmanned aerial systems is vast. An indication of the potential is given by a list of application categories that are already under consideration [52.64, 65]: *Remote sensing* such as pipeline spotting, power-line monitoring, volcanic sampling, mapping, meteorology, geology, and agriculture [52.66, 67], as well as unexploded mine detection [52.68].

Disaster response such as chemical sensing, flood monitoring, and wildfire management. *Surveillance* such as law enforcement, traffic monitoring, coastal and maritime patrol, and border patrols [52.69]. *Search and rescue* in low-density or hard-to-reach areas. *Transportation* including small and large cargo transport, and possibly passenger transport. *Communications* as permanent or *ad hoc* communication relays for voice and data transmission, as well as broadcast units for television or radio. *Payload delivery* for a wide range of applications including agriculture, firefighting, and even logistics of product delivery. *Image acquisition* for cinematography and real-time entertainment. It is no surprise that aerial robotics is one of the most dynamic and exciting fields of robotics research at the moment.

Video-References

VIDEO 436 Autopilot using total energy control
available from http://handbookofrobotics.org/view-chapter/52/videodetails/436

VIDEO 437 Dubins airplane
available from http://handbookofrobotics.org/view-chapter/52/videodetails/437

References

52.1 R.C. Michelson: International aerial robotics competition – The world's smallest intelligent flying machines, 13th RPVs/UAVs International Conference (1998), pp. 31.1–31.9

52.2 L.R. Newcome: *Unmanned Aviation, a Brief History of Unmanned Aerial Vehicles* (American Institute of Aeronautics and Astronautics, Reston 2004)

52.3 M.A. Garratt, J.S. Chahl: Vision-based terrain following for an unmanned rotorcraft, J. Field Robotics **25**(4/5), 284–301 (2008)

52.4 M. Garratt, H. Pota, A. Lambert, S. Eckersley-Maslin, C. Farabet: Visual tracking and lidar relative positioning for automated launch and recovery of an unmanned rotorcraft from ships at sea, Naval Eng. J. **121**(2), 99–110 (2009)

52.5 Vicon Motion Systems Ltd: http://www.vicon.com/

52.6 NaturalPoint Inc.: OptiTrack, http://www.naturalpoint.com/optitrack/

52.7 D. Mellinger, V. Kumar: Minimum snap trajectory generation and control for quadrotors, Proc. Int. Conf. Robotics Autom. (ICRA) (2011)

52.8 M. Hehn, R. D'Andrea: Quadrocopter trajectory generation and control, IFAC World Congress (2011) pp. 1485–1491

52.9 M.W. Mueller, R. D'Andrea: A model predictive controller for quadrocopter state interception, European Control Conference (2013) pp. 1383–1389

52.10 M.W. Mueller, M. Hehn, R. D'Andrea: A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013) pp. 3480–3486

52.11 J. Thomas, J. Polin, K. Sreenath, V. Kumar: Avian-inspired grasping for quadrotor micro uavs, ASME Int. Des. Eng. Tech. Conf. (IDETC) (2013)

52.12 M. Turpin, N. Michael, V. Kumar: Trajectory design and control for aggressive formation flight with quadrotors, Auton. Robots **33**(1/2), 143–156 (2012)

52.13 S. Shen, Y. Mulgaonkar, N. Michael, V. Kumar: Vision-based state estimation for autonomous rotorcraft MAVs in complex environments, Proc. IEEE Int. Conf. Robotics Autom. (2013)

52.14 S. Shen, Y. Mulgaonkar, N. Michael, V. Kumar: Initialization-free monocular visual-inertial estimation with application to autonomous mavs, Int. Symp. Exp. Robotics (ISER) (2014)

52.15 T. Hamel, R. Mahony, R. Lozano, J. Ostrowski: Dynamic modelling and configuration stabilization for an X4-flyer, Proc. Int. Fed. Autom. Control Symp. (IFAC) (2002)

52.16 S. Bouabdallah, P. Murrieri, R. Siegwart: Design and control of an indoor micro quadrotor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 5 (2004) pp. 4393–4398

52.17 P.-J. Bristeau, P. Martin, E. Salaün, N. Petit: The role of propeller aerodynamics in the model of a quadrotor uav, Proc. Eur. Control Conf. (2009) pp. 683–688

52.18 H. Huang, G.M. Hoffmann, S.L. Waslander, C.J. Tomlin: Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering, IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 3277–3282

52.19 P. Pounds, R. Mahony, P. Corke: Modelling and control of a large quadrotor robot. Control Eng, Pract. **18**(7), 691–699 (2010)

52.20 M. Bangura, R. Mahony: Nonlinear dynamic modeling for high performance control of a quadrotor, Australas. Conf. Robotics Autom. (2012)

52.21 R. Mahony, V. Kumar, P. Corke: Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor, Robotics Autom. Mag. **19**(3), 20–32 (2012)

52.22 R.W. Prouty: *Helicopter Performance, Stability and Control* (Krieger, Malabar 1995), reprint with additions

52.23 J.G. Leishman: *Principles of helicopter aerodynamics*, Cambridge Aerospace Series (Cambridge University Press, Cambridge 2000)

52.24 P. Martin, E. Salaun: The true role of accelerometer feedback in quadrotor control, Proc. IEEE Int. Conf. Robotics Autom. (2010) pp. 1623–1629

52.25 D. Abeywardena, S. Kodagoda, G. Dissanayake, R. Munasinghe: Improved state estimation in quadrotor mavs: A novel drift-free velocity estimator, IEEE Robotics Autom. Mag. **20**(4), 32–39 (2013)

52.26 R. Leishman, J. Macdonald, R.W. Beard, T.W. McLain: Quadrotors and accelerometers: State estimation with an improved dynamic model. IEEE Control Syst, Mag. **34**(1), 28–41 (2014)

52.27 A. Bramwell, G. Done, D. Balmford: *Bramwell's Helicopter Dynamics* (Butterworth Heinenmann, Woburn 2001)

52.28 J.-L. Boiffier: *The Flight Dynamics: The Equations* (Wiley, Chichester 1998)

52.29 B.L. Stevens, F.L. Lewis: *Aircraft Control and Simulation*, 2nd edn. (Wiley, Hoboken 2003)

52.30 L.H. Carter, J.S. Shamma: Gain-scheduled bank-to-turn autopilot design using linear parameter varying transformations. J. Guid, Control Dyn. **19**(5), 1056–1063 (1996)

52.31 R.W. Beard, T.W. McLain: *Navigation, Guidance, and Control of Small Unmanned Aircraft* (Princeton University Press, Princeton 2012)

52.32 M.-D. Hua, T. Hamel, P. Morin, C. Samson: A control approach for thrust-propelled underactuated vehicles and its application to vtol drones, IEEE Trans. Autom. Control **54**(8), 1837–1853 (2009)

52.33 M. Bangura, H. Lim, H. Jin Kim, R. Mahony: Aerodynamic power control for multirotor aerial vehicles, Proc. IEEE Int. Conf. Robotics Autom. (2014), Paper MoB03.2.

52.34 P. Pounds, R. Mahony, P. Corke: Design of a static thruster for micro air vehicle rotorcraft, ASCE J. Aerosp. Eng. **22**(1), 85–94 (2009)

52.35 T. Lee, M. Leok, N.H. McClamroch: Geometric tracking control of a quadrotor UAV on $se(3)$, Proc. IEEE Conf. Decis. Control (2010)

52.36 M. Bangura, R. Mahony: Real-time model predictive control for quadrotors, Proc. IFAC World Conf. (2014)

52.37 H.K. Khalil: *Nonlinear Systems*, 2nd edn. (Prentice Hall, Upper Saddle River 1996)

52.38 D. Pucci: Flight dynamics and control in relation to stall, IEEE Am. Control Conf. (ACC) (2012) pp. 118–124

52.39 S. Devasia, D. Chen, B. Paden: Nonlinear inversion-based output tracking, IEEE Trans. Autom. Control **41**(7), 930–942 (1996)

52.40 J.-F. Magni, S. Bennani, J. Terlouw: *Robust Flight Control: A Design Challenge*, Lecture Notes in Control and Information Sciences, Vol. 224 (Springer, Berlin, Heidelberg 1997)

52.41 E. Slatyer, R. Mahony, P. Corke: Terrain following using wide field optic flow, Proc. Australas. Conf. Robotics Autom. (ACRA) (2010)

52.42 A.A. Lambregts: Vertical fight path and speed control autopilot design using total energy principles, Proc. AIAA Guid. Control Conf. (1983) pp. 559–569

52.43 L.F. Faleiro, A.A. Lambregts: Analysis and tuning of a total energy control system control law using eigenstructure assignment, Aerosp. Sci. Technol. **3**(3), 127–140 (1999)

52.44 M.J. Van Nieuwstadt, R.M. Murray: Real-time trajectory generation for differentially flat systems, Int. J. Robust Nonlinear Control **8**, 995–1020 (1998)

52.45 D. Mellinger, N. Michael, V. Kumar: Trajectory generation and control for precise aggressive maneuvers with quadrotors, Int. J. Robotics Res. **31**, 664–674 (2012)

52.46 L.E. Dubins: On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents, Am. J. Math. **79**, 497–516 (1957)

52.47 G. Yang, V. Kapila: Optimal path planning for unmanned air vehicles with kinematic and tactical constraints, Proc. IEEE Conf. Decis. Control (2002) pp. 1301–1306

52.48 A. Rahmani, X.C. Ding, M. Egerstedt: Optimal motion primitives for multi-UAV convoy protection, Proc. Int. Conf. Robotics Autom. (2010) pp. 4469–4474

52.49 S. Hosak, D. Ghose: Optimal geometrical path in 3D with curvature constraint, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 113–118

52.50 H. Yu, R.W. Beard: A vision-based collision avoidance technique for micro air vehicles using local-level frame mapping and path planning, Auton. Robots **34**(1/2), 93–109 (2013)

52.51 H. Chitsaz, S.M. LaValle: Time-optimal paths for a Dubins airplane, Proc. 46th IEEE Conf. Decis. Control (2007) pp. 2379–2384

52.52 M. Owen, R.W. Beard, T.W. McLain: Implementing dubins airplane paths on fixed-wing UAVs. In: *Handbook of Unmanned Aerial Vehicles*, ed. by G.J. Vachtsevanos, K.P. Valavanis (Springer, Berlin, Heidelberg 2014) pp. 1677–1702

52.53 S. Park, J. Deyst, J.P. How: Performance and lyapunov stability of a nonlinear path-following guidance method, AIAA J. Guid. Control Dyn. **30**(6), 1718–1728 (2007)

52.54 D.R. Nelson, D.B. Barber, T.W. McLain, R.W. Beard: Vector field path following for miniature air vehicles, IEEE Trans. Robotics **37**(3), 519–529 (2007)

52.55 D.A. Lawrence, E.W. Frew, W.J. Pisano: Lyapunov vector fields for autonomous unmanned aircraft flight control, AIAA J. Guid. Control Dyn. **31**(5), 1220–12229 (2008)

52.56 V.M. Goncalves, L.C.A. Pimenta, C.A. Maia, B.C.O. Durtra, G.A.S. Pereira: Vector fields for robot navigation along time-varying curves in n-dimensions, IEEE Trans. Robotics **26**(4), 647–659 (2010)

52.57 A. Khosravian, J. Trumpf, R. Mahony, T. Hamel: Velocity aided attitude estimation on $so(3)$ with sensor delay, Proc. Conf. Decis. Control (2014)

52.58 M. Euston, P. Coote, R. Mahony, J. Kim, T. Hamel: A complementary filter for attitude estimation of a fixed-wing UAV, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 340–345

52.59 R. Mahony, M. Euston, J. Kim, P. Coote, T. Hamel: A nonlinear observer for attitude estimation of a fixed-wing UAV without GPS measurements, Trans. Inst. Meas. Control **33**(6), 699–717 (2011)
52.60 R. Mahony, T. Hamel, J.-M. Pflimlin: Complementary filter design on the special orthogonal group *SO*(3), Proc. IEEE Conf. Decis. Control (CDC) (2005) pp. 1477–1484
52.61 R. Mahony, T. Hamel, J.-M. Pflimlin: Non-linear complementary filters on the special orthogonal group, IEEE Trans. Autom. Control **53**(5), 1203–1218 (2008)
52.62 S. Bonnabel, P. Martin, P. Rouchon: Non-linear symmetry-preserving observers on lie groups, IEEE Trans. Autom. Control **54**(7), 1709–1713 (2009)
52.63 J. Macdonald, R. Leishman, R. Beard, T. McLain: Analysis of an improved imu-based observer for multirotor helicopters, J. Intell Robotic Syst. **74**(3/4), 1049–1061 (2014)
52.64 D. Hughes: Uavs face hurdles in gaining access to civil airspace, Aviation Week (2007)
52.65 K.C. Wong, C. Bil, G. Gordon, P.W. Gibbens: *Study of the Unmanned Aerial Vehicle (UAV) Market in Australia*, Tech. Rep. (Aerospace Technology Forum Report, Sydney 1997)
52.66 R. Sugiura, N. Noguchi, K. Ishii, H. Terao: Development of remote sensing system using an unmanned helicopter, J. Jap. Soc. Agricult. Mach. **65**(1), 53–61 (2003)
52.67 R. Sugiura, T. Fukagawa, N. Noguchi, K. Ishii, Y. Shibata, K. Toriyama: Field information system using an agricultural helicopter towards precision farming, IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (2003) pp. 1073–1078
52.68 K. Schutte, H. Sahli, D. Schrottmayer, F.J. Varas: Arc: A camcopter based mine field detection system, 5th Int. Airborne Remote Sens. Conf. (2001)
52.69 S.E. Wright: Uavs in community police work, AIAA Infotechs., Aerospace (2005)

Multimedia Contents

53. Multiple Mobile Robot Systems

Lynne E. Parker, Daniela Rus, Gaurav S. Sukhatme

Within the context of multiple *mobile, and networked* robot systems, this chapter explores the current state of the art. After a brief introduction, we first examine architectures for multirobot cooperation, exploring the alternative approaches that have been developed. Next, we explore communications issues and their impact on multirobot teams in Sect. 53.3, followed by a discussion of networked mobile robots in Sect. 53.4. Following this we discuss swarm robot systems in Sect. 53.5 and modular robot systems in Sect. 53.6. While swarm and modular systems typically assume large numbers of homogeneous robots, other types of multirobot systems include heterogeneous robots. We therefore next discuss heterogeneity in cooperative robot teams in Sect. 53.7. Once robot teams allow for individual heterogeneity, issues of task allocation become important; Sect. 53.8 therefore discusses common approaches to task allocation. Section 53.9 discusses the challenges of multirobot learning, and some representative approaches. We outline some of the typical application domains which serve as test beds for multirobot systems research in Sect. 53.10. Finally, we conclude in Sect. 53.11 with some summary remarks and suggestions for further reading.

53.1 **History** 1336

53.2 **Architectures for Multirobot Systems** ... 1337
- 53.2.1 The Nerd Herd 1337
- 53.2.2 The ALLIANCE Architecture 1338
- 53.2.3 The Distributed Robot Architecture 1339

53.3 **Communication** 1339

53.4 **Networked Mobile Robots** 1340
- 53.4.1 Overview 1341
- 53.4.2 State of the Art and Potential .. 1342
- 53.4.3 Research Challenges 1344
- 53.4.4 Control 1346
- 53.4.5 Communication for Control 1347
- 53.4.6 Communication for Perception . 1347
- 53.4.7 Control for Perception 1349
- 53.4.8 Control for Communication 1350

53.5 **Swarm Robots** 1351

53.6 **Modular Robotics** 1354
- 53.6.1 Chain Systems 1354
- 53.6.2 Lattice Systems 1354
- 53.6.3 Truss Systems 1356
- 53.6.4 Free-Form Systems 1356
- 53.6.5 Self-Assembling Systems 1356

53.7 **Heterogeneity** 1357

53.8 **Task Allocation** 1359
- 53.8.1 Taxonomy for Task Allocation ... 1359
- 53.8.2 Representative Approaches 1360

53.9 **Learning** 1361

53.10 **Applications** 1362
- 53.10.1 Foraging and Coverage 1362
- 53.10.2 Flocking and Formations 1362
- 53.10.3 Object Transportation and Cooperative Manipulation . 1363
- 53.10.4 Multitarget Observation 1364
- 53.10.5 Traffic Control and Multirobot Path Planning 1365
- 53.10.6 Soccer 1365

53.11 **Conclusions and Further Reading** 1366

Video-References 1366

References 1367

Researchers generally agree that multirobot systems have several advantages over single-robot systems [53.1, 2]. The most common motivations for developing multirobot system solutions are that:

1. The task complexity is too high for a single robot to accomplish.
2. The task is inherently distributed.
3. Building several resource-bounded robots is much easier than having a single powerful robot.
4. Multiple robots can solve problems faster using parallelism.
5. The introduction of multiple robots increases robustness through redundancy.

The issues that must be addressed in developing multirobot solutions are dependent upon the task requirements and the sensory and effector capabilities of the available robots.

The types of robots considered in the study of multiple *mobile* robot systems are those robots that move around in the environment, such as ground vehicles, aerial vehicles, or underwater vehicles. This chapter focuses specifically on the interaction of multiple *mobile* robots, as distinguished from other types of multirobot interaction. For example, a special case of multiple mobile robot systems are the *reconfigurable* or *modular* robots that interconnect with each other for the purposes of navigation or manipulation. Algorithmic aspects of these systems are covered in Sect. 53.5, while hardware aspects are covered in Chap. 22. *Networked* robotics is very closely related to multiple mobile robot systems; the focus in networked robotics is on systems of robots, sensors, embedded computers, and human users that are all connected by networked communication. Another variant of multirobot cooperation is multiple manipulator arm cooperation. Chapter 39 describes these systems in detail.

53.1 History

Since the earliest work on multiple mobile robot systems in the 1980s, the field has grown significantly, and covers a large body of research. At the most general level, approaches to multiple mobile robot systems fall into one of two broad categories: *collective swarm* systems and *intentionally cooperative* systems. *Collective swarm* systems are those in which robots execute their own tasks with only minimal need for knowledge about other robot team members. These systems are typified by the assumption of a large number of homogeneous mobile robots, in which robots make use of local control laws to generate globally coherent team behaviors, with little explicit communication among robots. On the other hand, robots in *intentionally cooperative* systems have knowledge of the presence of other robots in the environment and act together based on the state, actions, or capabilities of their teammates in order to accomplish the same goal. Intentionally cooperative systems vary in the extent to which robots take into account the actions or state of other robots, and can lead to either strongly or weakly cooperative solutions [53.3]. *Strongly cooperative* solutions require robots to act in concert to achieve the goal, executing tasks that are not trivially serializable. Typically, these approaches require some type of communication and synchronization among the robots. *Weakly cooperative* solutions allow robots to have periods of operational independence, subsequent to coordinating their selection of tasks or roles. Intentionally cooperative multirobot systems can deal with heterogeneity in the robot team members, in which team members vary in their sensor and effector capabilities. In these teams, the coordination of robots can be very different from collective swarm approaches, since robots are no longer interchangeable.

Most of the work specific to multiple mobile robot cooperation can be categorized into a set of key topics of study. These topics, which are the foci of this chapter, include *architectures*, *communication*, *swarm robots*, *heterogeneity*, *task allocation*, and *learning*. *Architectures* and *communication* in multirobot systems are relevant for all types of multirobot systems, as these approaches specify how the robot team members are organized and interact. *Swarm robots* is a particular type of multirobot system, typified by large numbers of homogeneous robots that interact implicitly with each other. Such systems are often contrasted with *heterogeneous* robots, in which team members may vary significantly in their capabilities. When robots vary in capabilities, challenges arise in determining which robots should perform which tasks – a challenge commonly referred to as *task allocation*. Finally *learning* in multirobot teams is of particular interest in designing teams that are adaptive over time and can learn new behaviors. Illustrating the advances in each of these areas often takes place in a set of representative application domains; these *applications* are the final major topic of discussion in this chapter.

53.2 Architectures for Multirobot Systems

The design of the overall control architecture for the multirobot team has a significant impact on the robustness and scalability of the system. Robot architectures for multirobot teams are composed of the same fundamental components as in single-robot systems, as described in Chap. 12. However, they also must address the interaction of robots and how the group behavior will be generated from the control architectures of the individual robots in the team. Several different philosophies for multirobot team architectures are possible; the most common are *centralized*, *hierarchical*, *decentralized*, and *hybrid*.

Centralized architectures that coordinate the entire team from a single point of control are theoretically possible [53.4], although often practically unrealistic due to their vulnerability to a single point of failure, and due to the difficulty of communicating the entire system state back to the central location at a frequency suitable for real-time control. Situations in which these approaches are relevant are cases in which the centralized controller has a clear vantage point from which to observe the robots, and can easily broadcast group messages for all robots to obey [53.5].

Hierarchical architectures are realistic for some applications. In this control approach, each robot oversees the actions of a relatively small group of other robots, each of which in turn oversees yet another group of robots, and so forth, down to the lowest robot, which simply executes its part of the task. This architecture scales much better than centralized approaches, and is reminiscent of military command and control. A point of weakness for the hierarchical control architecture is recovering from failures of robots high in the control tree.

Decentralized control architectures are the most common approach for multirobot teams, and typically require robots to take actions based only on knowledge local to their situation. This control approach can be highly robust to failure, since no robot is responsible for the control of any other robot. However, achieving global coherency in these systems can be difficult, because high-level goals have to be incorporated into the local control of each robot. If the goals change, it may be difficult to revise the behavior of individual robots.

Hybrid control architectures combine local control with higher-level control approaches to achieve both robustness and the ability to influence the entire team's actions through global goals, plans, or control. Many multirobot control approaches make use of hybrid architectures.

A plethora of multirobot control architectures have been developed over the years. We focus here on three early approaches that illustrate the spectrum of control architectures. The first, the Nerd Herd, is representative of a pure *swarm* robotics approach using large numbers of homogeneous robots. The second, ALLIANCE, is representative of a behavior-based approach that enables coordination and control of possibly heterogeneous robots without explicit coordination. The third, distributed robot architecture (DIRA), is a hybrid approach that enables both robot autonomy and explicit coordination in possibly heterogeneous robot teams.

53.2.1 The Nerd Herd

One of the first studies of social behaviors in multirobot teams was conducted by *Matarić* [53.6], with results being demonstrated on the Nerd Herd team of 20 identical robots (shown in Fig. 53.1). This work is an example of swarm robotic systems, as described further in Sect. 53.4. The decentralized control approach was based on the subsumption architecture (Chap. 12), and assumed that all robots were homogeneous, but with relatively simple individual capabilities, such as detecting obstacles and *kin* (i. e., other robot team members). A set of basic social behaviors were defined and demonstrated, including obstacle avoidance, homing, aggregation, dispersion, following, and safe wandering. These basic behaviors were combined in various ways to yield more composite social behaviors, including flocking (composed of safe wandering, aggregation, and dispersion), surrounding (composed of safe wandering, following, and aggregation), herding (composed of safe wandering, surrounding, and flocking), and foraging (composed of safe wandering, dispersion, following, homing, and flocking). The behaviors were

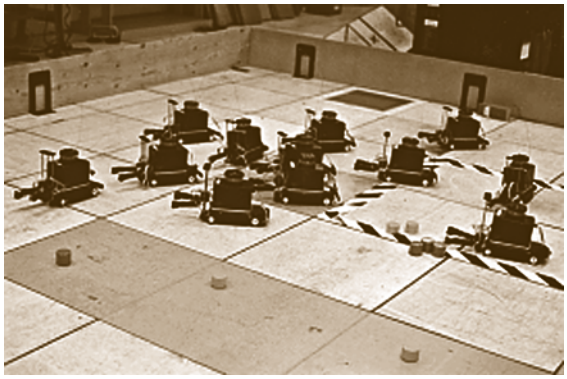


Fig. 53.1 The Nerd Herd robots

implemented as rules, such as the following rule for aggregate:

```
Aggregate:
  If agent is outside aggregation
  distance
    turn toward aggregation centroid
    and go.
  Else
    stop.
```

This work showed that collective behaviors could be generated through the combination of lower-level basic behaviors. Related work on this project studied issues such as using bucket brigades to reduce interference [53.7], and learning [53.8].

53.2.2 The ALLIANCE Architecture

Another early work in multirobot team architectures is the ALLIANCE architecture (Fig. 53.2), developed by *Parker* [53.9] for fault-tolerant task allocation in heterogeneous robot teams. This approach builds on the subsumption architecture by adding behavior sets and motivations for achieving action selection without explicit negotiations between robots. Behavior sets group low-level behaviors together for the execution of a particular task. The motivations consist of levels of impatience and acquiescence that can raise and lower a robot's interest in activating a behavior set corresponding to a task that must be accomplished.

In this approach, the initial motivation to perform a given behavior set is set to zero. Then, at each time step, the motivation level is recalculated based on:

1. The previous motivation level
2. The rate of impatience
3. Whether the sensory feedback indicates the behavior set is needed
4. Whether the robot has another behavior set already activated
5. Whether another robot has recently begun work on this task
6. Whether the robot is willing to give up the task, based on how long it has been attempting the task.

Effectively, the motivation continues to increase at some positive rate unless one of four situations occurs:

1. The sensory feedback indicates that the behavior set is no longer needed.
2. Another behavior set in the robot activates.
3. Some other robot has just taken over the task for the first time.
4. The robot has decided to acquiesce the task.

In any of these four situations, the motivation returns to zero. Otherwise, the motivation grows until it crosses a threshold value, at which time the behavior set is activated and the robot can be said to have selected an action. When an action is selected, cross-inhibition within that robot prevents other tasks from being activated within that same robot. When a behavior set is active in a robot, the robot broadcasts its current activity to other robots at a periodic rate.

The L-ALLIANCE extension [53.10] allows a robot to adapt the rate of change of the impatience and ac-

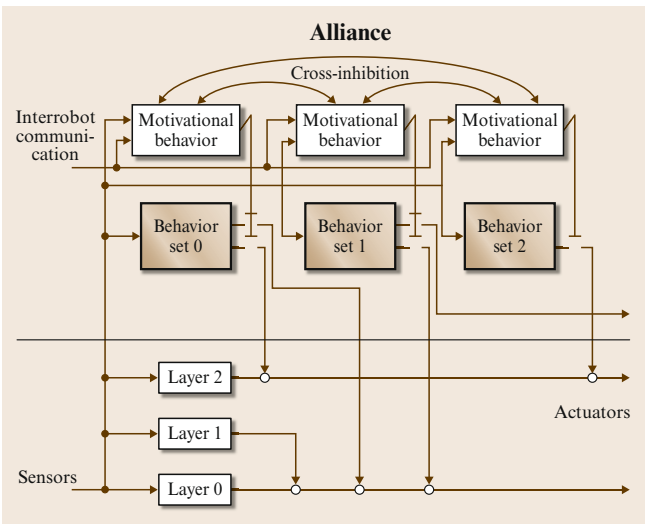


Fig. 53.2 The ALLIANCE architecture

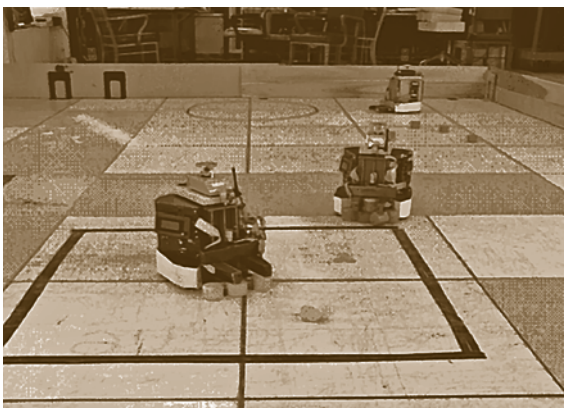


Fig. 53.3 Robots using the ALLIANCE architecture for a mock clean-up task

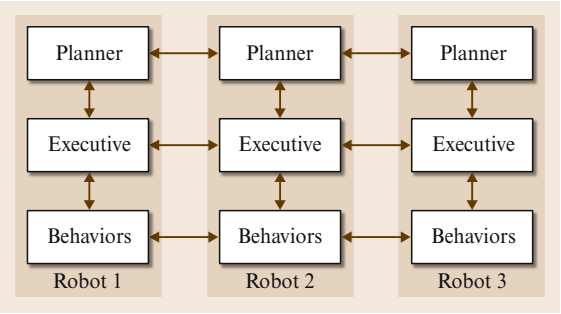


Fig. 53.4 The distributed robot architecture

quiescence values depending on the quality with which that robot is expected to accomplish a given task. The result is that robots that have demonstrated their ability to better accomplish certain tasks are more likely to choose those tasks in the future. Additionally, if problems occur during team performance, then robots may dynamically reallocate their tasks to compensate for the problems. This approach was demonstrated on a team of three heterogeneous robots performing a mock clean-up task, two robots performing a box-pushing task, and four robots performing a cooperative target observation problem. The approach has also been demonstrated in the simulation of a janitorial service task and a bounding overwatch task. Figure 53.3 shows robots using ALLIANCE to perform the mock clean-up task.

53.2.3 The Distributed Robot Architecture

Simmons et al. [53.11] have developed a hybrid architecture called the distributed robot architecture (DIRA). Similar to the Nerd Herd and ALLIANCE approaches, the DIRA approach allows autonomy in individual robots. However, unlike the previous approaches, DIRA also facilitates explicit coordination among robots. This approach is based on layered architectures that are popular for single-robot systems (Chap. 12). In this approach (Fig. 53.4), each robot's control architecture

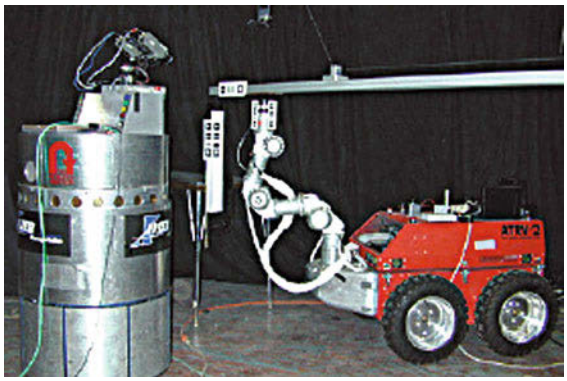


Fig. 53.5 Robots using the distributed robot architecture for assembly tasks

consists of a *planning* layer that decides how to achieve high-level goals; an *executive* layer that synchronizes agents, sequences tasks, and monitors task execution; and a *behavioral* layer that interfaces to the robot's sensors and effectors. Each of these layers interacts with those above and below it. Additionally, robots can interact with each other via direct connections at each of the layers.

This architecture has been demonstrated in a team of three robots – a crane, a roving eye, and a mobile manipulator – performing a construction assembly task (Fig. 53.5). This task requires the robots to work together to connect a beam at a given location. In these demonstrations, a *foreman* agent decides which robot should move the beam at which times. Initially, the crane moves the beam to the vicinity of the emplacement based on encoder feedback. The foreman then sets up a behavioral loop between the roving eye and the crane robot to servo the beam closer to the point of emplacement. Once the beam is close enough, the foreman tasks the roving eye and the mobile manipulator to servo the arm to grasp the beam. After contact is made, the foreman tasks the roving eye and the mobile manipulator to coordinate to servo the beam to the emplacement point, thus completing the task.

53.3 Communication

A fundamental assumption in multirobot systems research is that globally coherent and efficient solutions can be achieved through the interaction of robots lacking complete global information. However, achieving these globally coherent solutions typically requires robots to obtain information about their teammates' states or actions. This information can be obtained in a number of ways; the three most common techniques are:

1. The use of implicit communication *through the world* (called *stigmergy*), in which robots sense the effects of teammate's actions through their effects on the world [53.6, 12–16]

2. Passive action recognition, in which robots use sensors to directly observe the actions of their teammates [53.17]
3. Explicit (intentional) communication, in which robots directly and intentionally communicate relevant information through some active means, such as radio [53.9, 18–21] – an area widely studied in the field of networked robot systems.

Each of these mechanisms for exchanging information between robots has its own advantages and disadvantages [53.22]. *Stigmergy* is appealing because of its simplicity and its lack of dependence upon explicit communications channels and protocols. However, it is limited by the extent to which a robot's perception of the world reflects the salient states of the mission the robot team must accomplish. *Passive action recognition* is appealing because it does not depend upon a limited-bandwidth, fallible communication mechanism. As with implicit cooperation, however, it is limited by the degree to which a robot can successfully interpret its sensory information, as well as the difficulty of analyzing the actions of robot team members. Finally, the *explicit communication* approach is appealing because of its directness and the ease with which robots can become aware of the actions and/or goals of its teammates. The major uses of explicit communication in multirobot teams are to synchronize actions, exchange information, and to negotiate between robots. Explicit communication is a way of dealing with the *hidden-state* problem [53.23], in which limited sensors cannot distinguish between different states of the world that are important for task performance. However, explicit communication is limited in terms of fault tolerance and reliability, because it typically depends upon a noisy, limited-bandwidth communications channel that may not continually connect all members of the robot team. Thus, approaches that make use of explicit communications must also provide mechanisms to handle communication failures and lost messages.

Selecting the appropriate use of communication in a multirobot team is a design choice dependent upon the tasks to be achieved by the multirobot team. One needs to carefully consider the costs and benefits of alternative communications approaches to determine the method that can reliably achieve the required level of system performance. Researchers generally agree that communication can have a strong positive impact on the performance of the team. One of the earliest illustrations of this impact was given in the work of *MacLennan* and *Burghardt* [53.24], which investigates the evolution of communication in simulated worlds and concludes that the communication of local robot information can result in significant performance improvements. Interestingly, for many representative applications, researchers have found a nonlinear relationship between the amount of information communicated and its impact on the performance of the team. Typically, even a small amount of information can have a significant impact on the team, as found in the study of *Balch* and *Arkin* [53.25]. However, more information does not necessarily continue to improve performance, as it can quickly overload the communications bandwidth without providing an application benefit. The challenge in multirobot systems is to discover the *optimal* pieces of information to exchange that yield these performance improvements without saturating the communications bandwidth. Currently, no general approaches to identifying this critical information are available; thus, the decision of what to communicate is an application-specific question to be answered by the system designer. *Dudek* et al.'s taxonomy of multirobot systems [53.26] includes axes related to communication, including communication range, communication topology, and communication bandwidth. These characteristics can be used to compare and contrast multirobot systems.

Several related issues of active research in communications for multirobot teams deal with dynamic network connectivity and topologies; for example, robot teams must either be able to maintain communications connectivity as they move, or employ recovery strategies that allow the robot team to recover when the communications connectivity is broken. These concerns may require robots to adapt their actions in response to the anticipated effects on the communications network, or in response to knowledge of the anticipated propagation behavior of information through the dynamic network. These and related issues are discussed next in the context of *networked robot systems*.

53.4 Networked Mobile Robots

Networked robots are multiple robots operating together coordinating and cooperating by *networked communication* to accomplish a specified task. Multiple robots enable new capabilities and the communication network enables new approaches and solutions that are difficult with just perception and control. Communication enables new control and perception capabilities in the system (e.g., access to information outside the perception range of the robot system). Conversely, control enables solutions for problems that are difficult with-

out mobility (e.g., localization). Section 53.4.1 defines the field, examines the benefits of networking in robot coordination, and discusses applications. Section 53.4.2 highlights a few projects focused on networked robotics and discusses the application potential of the field. Section 53.4.3 discusses the research challenges at the intersection of control, communication, and perception. Section 53.4.4 defines a model for the control of a networked system which is used in Sects. 53.4.5–53.4.8 to examine specific research issues and opportunities facilitated by the interplay between communication, control, and perception.

53.4.1 Overview

The term *networked robots* refers to multiple robots operating together coordinating and cooperating by *networked communication* to accomplish a specified task. Communication between entities is fundamental to cooperation (and coordination), hence there is a central role for the communication network in networked robots. Networked robots may also involve coordination and cooperation with stationary sensors, embedded computers, and human users. The central feature of networked robots is the ability of the system to perform tasks that are well beyond the abilities of a single robot or multiple uncoordinated robots.

The IEEE (Institute of Electrical and Electronics Engineers) *Technical Committee on Networked Robots* has adopted the following definition of a networked robot:

A networked robot is a robotic device connected to a communications network such as the Internet or local-area network (LAN). The network could be wired or wireless, and based on any of of a variety of protocols such as the transmission control protocol (TCP), the user datagram protocol (UDP), or 802.11. Many new applications are now being developed ranging from automation to exploration. There are two subclasses of networked robots:

1. Teleoperated, where human supervisors send commands and receive feedback via the network. Such systems support research, education, and public awareness by making valuable resources accessible to broad audiences.
2. Autonomous, where robots and sensors exchange data via the network. In such systems, the sensor network extends the effective sensing range of the robots, allowing them to communicate with each other over long distances to coordinate their activity. Sensing, actuation, and computation need no longer be collocated. A broad challenge is to develop a science base that couples communication, perception, and control to enable such new capabilities.

This definition of autonomous networked robots also includes a third class of distributed systems, mobile sensor networks, which is a natural evolution of sensor networks. Robot networks allow robots to measure spatially and temporally distributed phenomena more efficiently. The robots in turn can deploy, repair, and maintain the sensor network to increase its longevity, and utility. The focus of this chapter is *autonomous networked robots*.

Embedded computers and sensors are becoming ubiquitous in homes and factories, and increasingly wireless ad hoc networks or plug-and-play wired networks are becoming commonplace. Human users interact with embedded computers and sensors to perform tasks ranging from monitoring (e.g., supervising the operation of a factor and surveillance in a building) to control (e.g., running an assembly line consisting of sensors, actuators, and material-handling equipment). In most of these cases, the human users, embedded computers, and sensors are not collocated and the coordination and communication happens through a network. *Networked robots* extends this vision to multiple robots functioning in a wide range of environments performing tasks that require them to coordinate with other robots, cooperate with humans, and act on information derived from multiple sensors.

Figure 53.6 shows prototype concepts derived from academic laboratories and industry. In all these exam-

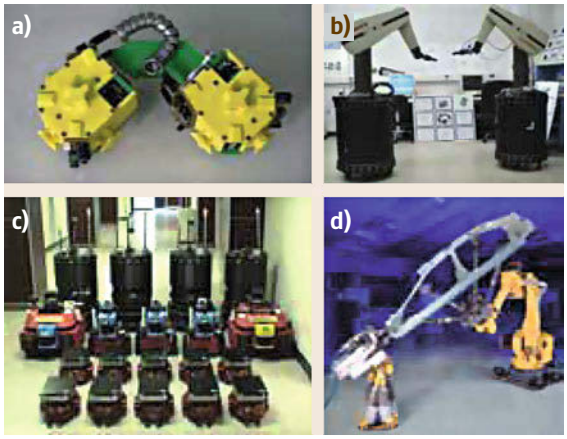


Fig.53.6a–d Small modules (after [53.27]) can automatically connect and communicate information to perform locomotion tasks **(a)**; robot arms (after [53.28]) on mobile bases can cooperate to perform household chores **(b)**; swarms of robots (after [53.29]) can be used to explore an unknown environment **(c)**; and industrial robots can cooperate in welding operations **(d)**

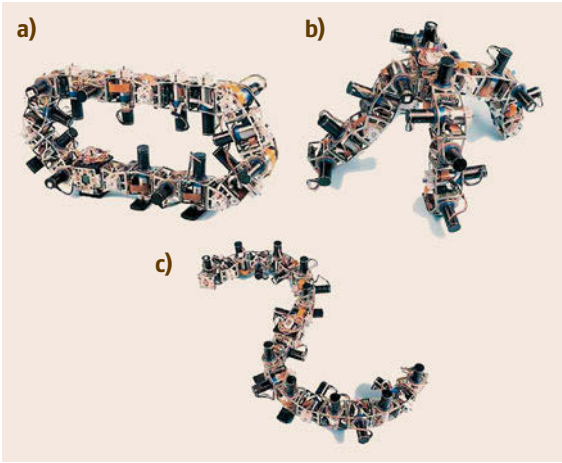


Fig.53.7a–c Robotic modules (after [53.30]) can be reconfigured to morph into different locomotion systems including **(a)** a wheel-like rolling system, **(b)** a snake-like undulatory locomotion system, and **(c)** a four-legged walking system

ples, independent robot or robotic modules can cooperate to perform tasks that a single robot (or module) cannot perform. Robots can automatically couple to perform locomotion tasks (also Fig. 53.7) and manipulation tasks that either a single robot cannot perform, or that would require a special-purpose larger robot to perform. They can also coordinate to perform search and reconnaissance tasks exploiting the efficiency that is inherent in parallelism. They can also perform independent tasks that need to be coordinated. Examples in the manufacturing industry include, for example, fixturing and welding.

Besides being able to perform tasks that individual robots cannot perform, networked robots also result in improved efficiency. Networking gives each robot access to information outside its perception range. Tasks such as searching or mapping can, in principle, be performed faster with an increase in the number of robots. A speed up in manufacturing operations can be achieved by deploying multiple robots performing operations in parallel but in a coordinated fashion.

Another advantage of using the network to connect robots is the ability to connect and harness physically removed assets. Mobile robots can react to information sensed by other mobile robots at a remote location. Industrial robots can adapt their end-effectors to new parts being manufactured upstream in the assembly line. Human users can use machines that are remotely located via the network.

The ability to network robots also enables fault tolerance in design. If robots can dynamically reconfigure themselves using the network, they are more tolerant to robot failures. This is seen in the Internet where multiple gateways, routers, and computers provide for a fault-tolerant system (although the Internet is not robust in other ways). Similarly, robots that can *plug* and *play* can be swapped in and out automatically to provide for a robust operating environment.

Finally, networked robots have the potential to provide great synergy by bringing together components with complementary benefits and making the whole greater than the sum of the parts.

Applications for networked robots abound. The US military routinely deploys unmanned vehicles that are reprogrammed remotely based on intelligence gathered by other unmanned vehicles, sometimes automatically. The deployment of satellites in space, often by astronauts in a shuttle with the shuttle robot arm, requires the coordination of complex instrumentation onboard the space shuttle, human operators on a ground station, the shuttle arm, and a human user on the shuttle. Home appliances now contain sensors and are becoming networked. As domestic and personal robots become more commonplace, it is natural to see these robots working with sensors and appliances in the house while cooperating with one or more human users. Networked robots will likely be used as critical ingredients in the environmental observatories of the future. Large-scale ecological monitoring precludes the use of monolithic infrastructure, and is envisioned to be built as a distributed, networked robotic system.

53.4.2 State of the Art and Potential

The growth in networked robot systems is broad-based, across many industries. There is a strong connection between this industry and the industry connected to sensor networks. Sensor networks have been projected to grow dramatically in terms of commercialization and market value [53.31]. Robot networks are analogous to sensor networks except that they allow sensors to have mobility and allow the geographical distribution of the sensors to be adapted based on the information acquired.

A system of robots, embedded computers, actuators, and sensors has tremendous potential in civilian, defense, and manufacturing applications. Nature provides the proof of concept of what is possible [53.32]. Group behaviors in nature can be found in organisms that are only microns to those that are several meters in length. There are numerous examples of simple animals that execute simple behaviors with modest sensors and actuators but communicate with and sense nearest neighbors to enable complex emergent behaviors that are fundamental to navigation, foraging, hunting, constructing nests, survival, and eventually growth. As seen

Fig. 53.8 Ants are able to cooperatively manipulate and transport objects often in large groups, without identified or labeled neighbors, and without centralized coordination

in Fig. 53.8, relatively small agents are able to manipulate objects that are significantly larger in terms of size and payload by cooperating with fairly simple individual behaviors. The coordination between agents is completely decentralized, allowing scaling up to large numbers of robots and large objects [53.33]. Individuals do not recognize each other. In other words, there is no labeling or identification of robots. The number of agents in the team is not explicitly encoded. Agents are identical, enabling robustness to failures and modularity. There is minimal communication, and even that which is present is only between neighbors. Furthermore, the optimal mode of group coordination may be scale dependent. Studies of wasps show strong evidence of centralized coordination among species with small colony sizes, but a distributed, decentralized coordination in larger colonies [53.34]. All these attributes are relevant to networked robots.

Biology has shown how simple decentralized behaviors in unidentified individuals (e.g., insects and birds exhibiting swarming behaviors) can exhibit a wide array of seemingly intelligent group behaviors. Similarly networked robots can potentially communicate and cooperate with each other, and even though individual robots may not be sophisticated, it is possible for networked robots to provide a range of intelligent behaviors that are beyond the scope of intelligent robots.

The significance and potential impact of networked robots is apparent from the following examples.

The manufacturing industry has always relied on integration between sensors, actuators, material-handling equipment, and robots. Today companies are finding it easier to reconfigure existing infrastructure by networking new robots and sensors with existing robots via wireless networks. There is also an increasing trend toward robots interacting with each other in operations like welding and machining, and robots cooperating with humans in assembly and material-handling tasks. Workcells consist of multiple robots, numerous sensors and controllers, automated guided vehicles, and one or two human operators working in a supervisory role. However, in most of these cells, the networked robots operate in a structured environment with very little variation in configuration and/or operating conditions.

There is a growing emphasis on networking robots in applications of field robotics, for example, in the mining industry. Like the manufacturing industry, operating conditions are often unpleasant and the tasks are repetitive. However, these applications are less structured and human operators play a more important role.

In the health care industry, networks allow health care professionals to interact with their patients, other professionals, expensive diagnostic instruments, and surgical robots. Telemedicine is expected to provide a major growth impetus for remote networked robotic devices that will take the place of today's stand-alone medical devices.

There are already many commercial products, notably in Japan, where robots can be programmed via and communicate with cellular phones. For example, the MARON robot developed by Fujitsu lets a human user dial up their robot and instruct it to conduct simple tasks including sending pictures back to the user via a cellular phone. Indeed these robots will interact with other sensors and actuators in the home – door openers equipped with Bluetooth cards and actuators and computer-controlled lighting, microwaves, and dishwashers. Indeed the Network Robot Forum [53.35] is already setting standards for how stationary sensors and actuators can interact with other robots in domestic and commercial settings.

Environmental monitoring is a key application for networked robots. By exploiting mobility and communication, robotic infrastructure enables observation and data collection at unprecedented scales in various aspects of ecological monitoring. This is significant for environmental regulatory policies (e.g., clean air and water legislation), as well as an enabler of new scientific discovery. For example, it is possible to obtain maps of salinity gradients in oceans, temperature and humidity variations in forests, and chemical composition of air and water in different ecological systems [53.36]. In addition to mobile sensor networks, it is also possible to use robots to deploy sensors and to retrieve information from the sensors. Mobile platforms allow the same sensor to collect data from multiple locations while communication allows the coordinated control and aggregation of information. Examples include systems built for aquatic [53.37], terrestrial [53.38], and subsoil monitoring [53.39]. There are many efforts to developed networked underwater platforms [53.40–42].

Networks of static and robotic devices have been developed for aquatic monitoring [53.37] and to obtain high-resolution information on the spatial and temporal distributions of plankton assemblages and concomitant environmental parameters. The RiverNet project [53.43] at Rensselaer Polytechnic Institute (RPI) has focused on the development of robotic sensor networks for monitoring a river ecosystem. Recent work at University of California, Los Angeles (UCLA), University of Southern California (USC), University of California, Riverside, and University of California, Merced on the networked infomechanical system project [53.38] has focused on the development of robotic networks for monitoring the forest canopy, with a view to providing data for modeling canopy and undercover growth. Networked robotic mini-rhizotrons [53.39] are being deployed in the forest to monitor root growth in the soil.

In the defense industry, countries like the USA have invested heavily in the concept of networked, geographically distributed assets. Unmanned aerial vehicles like the Predators are operated remotely. Information from sensors on the Predators triggers the deployment of other vehicles and weapon systems at a different remote location and allows commanders in a third location to control and command all these assets. The US military engaged in the large Future Combat Systems initiative to develop network-centric approaches to deploying autonomous vehicles. The network-centric tactical paradigms for modern warfare have created networked robots for defense and homeland security. While networked robots are already in operation, current approaches are limited to human users commanding a single vehicle or sensor system. However, it takes many human operators (between 2–10 depending on the complexity of the system) to deploy complex systems like unmanned aerial vehicles. A Predator unmanned aerial vehicle (UAV) is operated from a tactical control station, which may be on an aircraft carrier, with a basic crew of 3–10 operators.

The eventual goal, however, is to enable a single human user to deploy networks of unmanned aerial, ground, surface, and underwater vehicles. There have been several recent demonstrations of multirobot systems exploring urban environments [53.44, 45] and the interiors of buildings [53.46, 47] to detect and track intruders, and transmit all of the above information to a remote operator. These examples show that it is possible to deploy networked robots using an off-the-shelf 802.11b wireless network and have the team be remotely tasked and monitored by a single operator. An example of a project with heterogeneous vehicles in an urban setting is shown in Fig. 53.9. An example of a project with heterogeneous vehicles in an indoor setting is shown in Fig. 53.10 wherein robots map an environment and deploy themselves to form a sensor network to detect intruders.

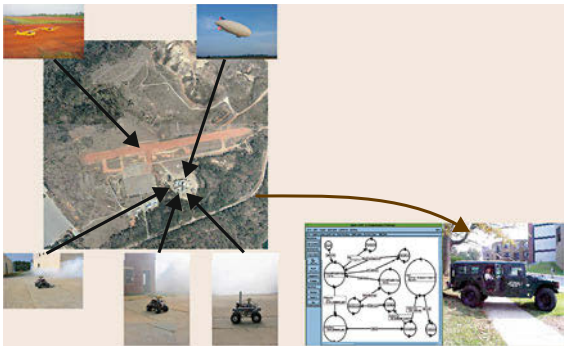


Fig. 53.9 A single operator commanding a network of aerial and ground vehicles from a command and control vehicle in an urban environment for scouting and reconnaissance in a recent demonstration by the University of Pennsylvania, Georgia Tech. and University of Southern California (after [53.48])

Many research projects are addressing group behaviors or collective intelligence by realizing swarming behaviors observed in nature. For example, the European Union (EU) funded several EU-wide coordinated projects on collective intelligence or swarm intelligence. The I-Swarm project in Karlsruhe [53.49] and the Swarm-Bot project at Ecole Polytechnique Fédérale de Lausanne (EPFL) [53.50] are examples of swarm intelligence. The Laboratory for Analysis and Architecture of Systems (LAAS) has a strong group in robotics and artificial intelligence. This group has had a long history of basic and applied research in multirobot systems. The integration of multiple unmanned vehicles for applications such as terrain mapping and fire-fighting is addressed in [53.51]. A multi-university US project addressed the development of networked vehicles for swarming behaviors [53.52]. Projects such as these are exploring the scalability of the basic concepts to large numbers of robots, sensors, and actuators.

53.4.3 Research Challenges

While there are many successful embodiments of networked robots with applications to manufacturing industry, the defense industry, space exploration, domestic assistance, and civilian infrastructure, there are significant challenges that have to be overcome.

The problem of coordinating multiple autonomous units and making them cooperate creates problems at the intersection of communication, control, and perception. Who should talk to whom and what information should be conveyed, and how? How does each unit

Fig. 53.10 Under the DARPA SDR program, a team from the University of Southern California, the University of Tennessee, and Science Applications and International Corporation (SAIC) demonstrated mapping, and intruder detection by a team of networked robots (after [53.46])

move in order to accomplish the task? How should the team members acquire information? How should the team aggregate information? These are all basic questions that need basic advances in control theory, perception, and networking. In addition, because humans are part of the network (as in the case of the Internet), we have to devise an effective way for multiple humans to be embedded in the network and command/control/monitor the network without worrying about the specificity of individual robots in the network. Thus the underlying research challenges lie at the intersection of control theory, perception, and communication/networks, as shown in Fig. 53.11.

It is also worth noting that robot networks are dynamic, unlike networks of sensors, computers or machines which might be networked together in a fixed topology. When a robot moves, its neighbors change and its relationship to the environment changes. As a consequence, the information it acquires and the actions it executes must change. Not only is the network topology dynamic, but the robot's behavior also changes as the topology changes. It is very difficult to predict the performance of such dynamic robot networks, yet it is this analysis problem that designers of robot networks must solve before deploying the network.

This notion of a changing topology inevitably leads us to complicated mathematical models. Traditionally, models of group behavior have been built on continuous models of dynamics of individuals, including local interactions with neighbors, and models of control and sensing with a fixed set of neighbors. While dynamics at the level of individual units may be adequately described by differential equations, the interactions with neighbors are best described by edges on a graph. Modeling, analysis, and control of such systems will require a comprehensive theoretical framework and new representational tools. New mathematical tools that marry dynamical system theory, switched systems, discrete mathematics, graph theory, and computational geometry are needed to solve the underlying problems. We need a design methodology for solving the *inverse*

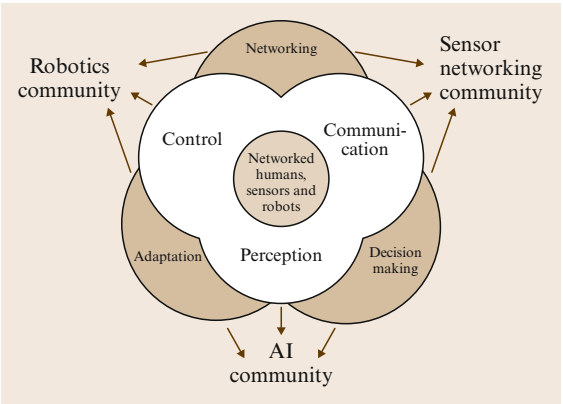


Fig. 53.11 The paradigm of *networked robots* introduces fundamental challenges at the intersection of control, perception, and communication that is of interest to the robotics, sensor networks, and artificial intelligence communities

problem in navigation – behaviors for controlling individuals to achieve a specified aggregate *motion* and *shape* of the group, and the application to active perception and coverage. An overview of some of these methods is provided in Sect. 53.4.4.

Problems of perception have been studied extensively in the robotics community. However, the perception problems in a system of networked, mobile sensor platforms bring a new set of challenges; for example, consider the problem of estimating the state of the network. State estimation requires the estimation of the state of robots and the environment based on local, limited-range sensory information. Localization of n vehicles in an m-dimensional configuration space requires $O((nm)^k)$ computations, where k is somewhere between 3 and 6, depending on the algorithm and domain-specific assumptions. The estimation problem is further exacerbated by the fact that not all robots in the network may be able to get the necessary information in a time-critical fashion. There are deep issues of representation and algorithmic development, which are discussed in Sect. 53.4.6.

The paradigm of active perception [53.53] links the control of sensor platforms to perception, bringing control theory and perception together in a common framework. Extending this paradigm to networked robots requires approaches of distributed control to be merged with decentralized estimation. Robots can move in order to localize themselves with respect to their neighbors, to localize their neighbors, and also to identify, localize, and track features in the environment. These problems are discussed in Sect. 53.4.7.

As discussed earlier, the communication network is central to the functioning of a network of robots.

However, if the network consists of mobile agents with transmitters and receivers with finite power, there is no guarantee that all agents can talk to teach other. Unlike a static sensor network, robots in a network can move toward each other to facilitate communication and adaptively maintain a communication network. Some basic algorithmic problems and several pertinent results are provided in Sect. 53.4.8.

53.4.4 Control

The control of individual robots is critical to the performance and scope of robot networks. Indeed motion coordination algorithms have been proposed for the purpose of improving communication performance [53.54, 55], localization [53.56, 57], information integration, deployment [53.58], and coverage [53.59–61], among other tasks. Mobility allows the group of robots to self-deploy, and self-organize by relocating themselves in support of communication, sensing, or task needs; for example, they can reconfigure to guarantee a desired communication bandwidth, k-hop connectivity, or algebraic connectivity, enabling message delivery from one robot to another. The group can also self-organize to position sensors so as to cover a desired area and adapt to shifts in the focus of monitoring activities. Controlling sensor position also supports map making, tracking of objects and events, and goal-directed navigation for users of the network. Finally, mobility allows robots to accomplish tasks such as navigation, reconnaissance, transportation, and search and rescue.

Given a group of mobile sensors, we would like to have distributed control capabilities that realize desirable global specifications. Thus, it is necessary to be able to automatically determine the necessary position and orientation of the group members and/or the distribution of group members, and their motion to achieve the desired task. At a lower level, the robots must be able to use information from the communication network and from their own sensors to derive local estimates, reason about the spatial network (their neighbors and their relationship to the environment), and then use the appropriate control policies to achieve the desired group specifications. We briefly outline the simplest mathematical model that is necessary to formulate such problems in order to provide a better sense of the underlying challenges.

In a robot network, we have multiple agents or nodes in which each agent is a physical entity that can be a robot, a vehicle with actuators and sensors, a sensor platform (possibly static) or even a communication relay node. Each agent A_i is characterized by an identifier, $i \in I \subset Z$, a state $x_i \in X_i \subset \mathbb{R}^n$, and control inputs $u_i \in U_i \subset \mathbb{R}$, with $f_i : X_i \times U_i \to TX_i$ specifying the dynamics

$$\dot{x}_i = f_i(x_i, u_i) \,. \tag{53.1}$$

The state x_i will consist of the position (and orientation), r_i in some d-dimensional space, and its velocity, $\dot{r}_i : x_i = \left(r_i^{\mathrm{T}}, \dot{r}_i^{\mathrm{T}}\right)^{\mathrm{T}}$, with $n = 2d$. $\mathcal{N}^{\mathrm{c}}(r_i)$ and $\mathcal{N}^{\mathrm{s}}(r_i)$ are neighborhoods of r that define the range and field of view of the communication hardware and sensors, respectively.

A network of robots S consists of N agents with a *sensing graph* and a *communications graph* that is defined by the physical distribution of the agents. The sensing graph (and similarly the communications graph) is defined by a map $E^{\mathrm{s}} : X^1 \times X^2 \cdots X^N \to I \times I$, where the edges of the graph are formed dynamically depending on the physical proximity of pairs of agents. Specifically, the $N \times N$ adjacency matrix, $\mathcal{A}^{\mathrm{s}}$ (and similarly $\mathcal{A}^{\mathrm{c}}$) has entries

$$\mathcal{A}^{\mathrm{s}}{}_{ij} = \begin{cases} 1, & \text{if } r_j \in \mathcal{N}^{\mathrm{s}}(r_i) \,, \\ 0, & \text{otherwise} \,. \end{cases} \tag{53.2}$$

Agent A_i has estimates of its own state and the states of neighbors (e.g., A_j), and these estimates are derived from information associated with edges in the sensing and communication graph

$$\hat{x}_j^{(i)} = h(x_i, z_{ij}) \,, \tag{53.3}$$

where z_{ij} represents measurements of the state of agent A_j available to A_i by sensing or communication channels and h is the estimator used by A_i. Note that z_{ij} may have dimension less than n and may therefore not contain complete information about $x_{ij} = x_i - x_j$. Clearly the relative position vector denoted by $r_{ij} = r_i - r_j$ and its magnitude are important quantities that may need to be estimated for biological and artificial agents.

Finally, A_i can encode n_{b_i} behaviors, which we will denote by $\mathcal{B}_i = B_1, B_2, \ldots, B_{n_{b_i}}$. Each behavior B_j is a controller, a function $k_j : \mathbb{R} \times X_i \to U_i$. All agents can be assigned identical or different behaviors. Each behavior represents a set of unsynchronized, locally executed computations (for control or estimation) being carried out for some collective purpose, with each processor using in its computations only data from its *neighboring* processors. Furthermore, even for a fixed assignment of behaviors, each processor's neighbors typically *change* with time because the processors are moving in and out of the sets $\mathcal{N}^{\mathrm{c}}$ and $\mathcal{N}^{\mathrm{s}}$. Thus the methodology for modeling and analyzing such systems will require the merging of graph theory and dynamical system theory at a fundamental level.

The reader is directed to the many survey articles on this subject for further information. An overview of challenges for the controls community is presented in [53.62]. The underlying theory for networked mobile systems has been explored in the context of automated highway systems [53.63], cooperative robot reconnaissance [53.46] and manipulation [53.64], formation flight control [53.65], and the control of groups of unmanned vehicles [53.45]. Our goal in the following sections is to explore the connections between communication, perception, and control.

53.4.5 Communication for Control

Communication networks allow physically disconnected entities to exchange information. At the lowest level, when groups of vehicles coordinate their actions, communication allows vehicles to exchange state information [53.66–68]. At a higher level, robots can plan navigation and exploration tasks based on an integrated map of the world derived from information acquired from different robots [53.52].

The use of communication for control in the multivehicle context has been addressed in the PATH project where formations of inline vehicles were studied [53.63]. Problems of the stability of the formation [53.69], the convergence of the formation to shapes [53.70], and the overall performance of the system [53.71] are of great interest. The performance of the system is directly influenced by the interconnections between agents. In addition to impacting on stability [53.63], feedback of states from different agents and feedforward information from the plans of different agents affects the rates at which the system of agents can respond to external stimuli [53.71] or to commands from human operators [53.72].

In addition, communication can be used for high-level control and planning of robots. There is great interest in using static sensor nodes as beacons to guide robot navigation. In [53.73], the problem of coverage and exploration of an unknown dynamic environment using a mobile robot is considered. An algorithm is presented which assumes that global information is not available (neither a map, nor global positioning system (GPS) information). The algorithm deploys a network of radio beacons that assists the robot in coverage. The network is also used by the robot for navigation. The deployed network can also be used for applications other than coverage (such as multirobot task allocation). A similar idea was presented using potential-field-based navigation in [53.52]. In this work the notion of no-go or danger areas was incorporated into the navigation cost function. Recent work along these lines with experimental data from sensor nodes is reported in [53.74].

In such communication-enabled cooperative control and planning (see also [53.75]), the communication network plays an important role in the creation of a shared representation of information. This notion of a shared representation is important to the scaling of coordinated control algorithms to large numbers of devices. For example, in [53.67], the information form of the Kalman filter is used to derive a framework for decentralizing estimation and fusion algorithms. This approach was shown to be applicable to multiple heterogenous ground and aerial platforms [53.56]. Such methodologies are transparent to the specificity and identity of the cooperating vehicles. This is because vehicles share a common representation, which consists of a certainty grid that contains information about the probability of detection of targets, and an information vector–matrix pair that is used in the information form of the Kalman filter [53.45]. Observations are propagated through the network by changing both the certainty grid and the information vector/matrix. This allows each vehicle to choose the action that maximizes a utility function, which is the combined mutual information gain from onboard sensors towards the detection and localization of features in the environment.

Thus, in summary, at the lowest level, communication enables either partial or complete state feedback of the network and allows agents to exchange information for feedforward control. At the higher levels, agents can share information for planning and for control. This is also discussed in Sect. 53.4.6 where the communication network is shown to enable a network-centric approach to perception.

53.4.6 Communication for Perception

While individual robots have sensors and the ability to build maps and models by integrating sensory information, networked robots can exchange information and leverage sensory data, maps, and models from other robots. The challenge is to exploit communication for perception in tasks such as distributed mapping in the presence of the delays, limited bandwidth, and disruption that are typical of communication networks.

Distributed localization is the term used to describe the merging of communication and perception for state estimation. Localization is an essential tool for the development of low-cost robot networks for use in location-aware applications and ubiquitous networking [53.76]. Location information is needed to track the placement of the nodes and to correlate the values measured by the node with their physical location. Distributed computation and robustness in the presence of measurement noise are key ingredients for a practical

localization algorithm that will give reliable results over a large-scale network.

The methods for distributed localization can be classified into two broad classes: algorithms that rely on anchor nodes for localization and algorithms that use no beacons. Localization may be computed using range information between nodes, bearing information, or both.

In [53.54] a theoretical foundation for network localization in terms of graph rigidity theory is provided. The problem is solved when nodes have perfect range information and it is show that a network has a unique localization if and only if its underlying graph is *generically globally rigid*. In [53.77] the Cramér–Rao lower bound (CRLB) for network localization is derived. This work computes the expected error characteristics for an ideal algorithm, and compares this to the actual error in an algorithm based on multilateration, drawing the important conclusion that the error introduced by the algorithm is just as important as the measurement error in assessing end-to-end localization accuracy. In [53.78] a distributed algorithm that uses no beacons and is guaranteed to compute correct location information under measurement noise for nodes that can range to neighbors is presented. This algorithm relies on the notion of robust quadrilaterals to compute robustly a global system of coordinates among the nodes. The computation supports moving nodes. Extensions of this work to passive tracking have been discussed in [53.79]. Localization based on the propagation of location information from known reference nodes based on connectivity includes [53.80, 81]. Mobility-assisted localization is introduced in [53.82]. Other techniques use distributed propagation of location information using multilateration [53.77, 83]. In a recent paper [53.84] the problem of evaluating the rigidity of a planar network is treated while satisfying common objectives of real-world systems: decentralization, asynchronicity, and parallelization.

Two approaches for cooperative relative localization of mobile robot teams are given in [53.85, 86]. Neither method uses GPS, landmarks, or maps of any kind; instead, robots make direct measurements of the relative pose of nearby robots and broadcast this information to the team as a whole. In [53.85], each robot processes this information independently to generate an egocentric estimate for the pose of other robots using a Bayesian formalism with a particle filter implementation. In [53.86], maximum-likelihood estimation (MLE) and numerical optimization is used to achieve a similar result.

A key issue is to be able to scale these computations for building a shared representation to large numbers of robots and sensors. This problem was studied in experiments under the US Defense Advanced Research Projects Agency (DARPA-funded software for distributed robotics (SDR) program. The goal of these experiments was to develop and demonstrate a multi-robot system capable of carrying out a specific mission. This required the ability to deploy a large number of robots into an unexplored building, map the building interior, detect and track intruders, and transmit all of the above information to a remote operator. A report on one set of experiments is presented in [53.46]. A tiered strategy for deploying the robots is described, where highly capable robots formed the first wave to enter and map a building, followed by a second wave which used the resulting map to self-deploy and monitor the environment for intruders. Both approaches relied extensively on networking the robots using commercial 802.11b wireless technology. This task involved both communication for building a shared representation as well control for perception.

Another important set of problems arises when robot networks are used for identifying, localizing, and then tracking targets in a dynamic setting. An embedded stationary wireless sensor network is like a virtual sensor spread over a large geographical area. Such a network can provide information to mobile robots about remote locations. Robot networks allow this virtual sensor to move in response to external stimuli and to track moving targets. Indeed, it is possible to cast this scenario as a pursuit-evasion game with robotic sensor networks [53.87]. For example, the Tenet project at USC addresses the design of network primitives and abstractions for tiered network architectures, with robotic pursuit evasion as one of the target applications. Algorithms for guiding the sampling strategy of a robotic boat to model and locate phenomena of interest (e.g., hotspots) in aquatic environments are discussed in [53.37]. The networked infomechanical systems (NIMS) project has focused on sensor-assisted techniques for mobile robot-based adaptive sampling for event response [53.88] and field reconstruction [53.89].

The information collected by the nodes in a sensor network can be processed at a central location or in a decentralized fashion. Such in-network data processing techniques make better use of network communication and computation resources than centralized processing. This also enables the network to compute accurate and up-to-date global pictures of the global perception landscape that are available to all the robots in the system. Methods for in-network data processing with static nodes include artificial potential-field computation, gradient computations, particle filters, Bayesian inference, and signal processing. Algorithms have been developed for computing maps, paths, and predictors [53.52, 73, 90].

A recent DARPA demonstration showed how communication networks can be used effectively in perception tasks involving heterogenous robots [53.44]. In cooperative search, identification, and localization unmanned aerial vehicles (UAVs) can be used to cover large areas, searching for targets. However, sensors on UAVs are typically limited in their accuracy of localization of targets on the ground. On the other hand, ground robots can be deployed to accurately locate ground targets but have the disadvantage of not being able to move rapidly and see through obstacles such as buildings or fences. In [53.56], the synergy between these two devices is exploited by creating a seamless network of UAVs and unmanned ground vehicles (UGVs). As discussed in Sect. 53.2, the key to such network-centric approaches for search and localization is a shared representation of state information, which in this case is easily scalable to large numbers of UAVs and UGVs and is transparent to the specificity of individual platforms. However, how to do this more generally and for more unstructured information remains an issue for future research.

53.4.7 Control for Perception

Networked mobile robots enable the exploration of dynamic environments and the recovery of three-dimensional information via distributed active perception [53.53]. Since the nodes are mobile, a natural question is: where should the nodes be placed in order to ensure successful integration of information from multiple nodes, and to maximize the quality of the estimates returned by the team? Since there is a cost associated with transmitting and processing data, it is important to consider which sensor readings should be used in the state estimation and what information should be communicated to the rest of the system. The quality of the information computed by the network depends on the locations of the sensor platforms both in an absolute and relative sense. The quality also depends on the noise characteristics of each sensor, and the communication network.

A robot network goes well beyond a fixed sensor network, which can only collect data at fixed positions in space; for example, when an event is detected at a specific location it is possible to direct more sensors toward the location of observation of the event for more information (for example, higher-resolution data or higher sampling frequency). Reconfiguring the node locations for adaptive resolution sampling relies on distributed control strategies.

Various strategies have been introduced for controlling mobile sensor network coverage. Mobile sensing agents are controlled using gradients of information-based objective functions [53.91]. Stability results are derived without concerns for the optimality of the network configuration, but local guarantees are provided. Topology aware coordinated behavior is treated in [53.92]. A body of results reported in [53.93] and [53.94] describes decentralized control laws for positioning mobile sensor networks optimally with respect to a known event distribution density function. This approach is advantageous because it guarantees that the network (locally) minimizes a cost function relevant to the coverage problem. However, the control strategy requires that each agent have a complete knowledge of the event distribution density, thus it is not reactive to the sensed environment. The work by [53.95, 96] generalizes these results to situations in which the nodes estimate rather than know ahead of time the event distribution density function. A local (decentralized) control law requires that each agent can measure the value and gradient of the distribution density function at its own position. This results in a sensor network that is reactive to its sensed environment while maintaining or seeking a near-optimal sensing configuration. In addition, the distribution density function approximation yields a closed-form expression for the control law in terms of the vertices of an agent's Voronoi region. This eliminates the need for the numerical integration of a function over a polynomial domain at every time step, thereby providing a significant reduction in computational overhead for each agent. Other work in event monitoring for unknown distributions includes [53.59]. *Krause* et al. [53.97] have recently proposed an approach for sensor placement that considers both the sensing quality and communication cost of imperfect sensing and communication components. They use a parametric model for link reception rate that assumes no acknowledgement and no temporal correlation of lossy links.

Beginning with the art gallery problem, there have been multiple efforts to determine an optimal configuration of sensors to cover a given region [53.98–100]. A variant which allows the use of mobile sensors is known as the *watchmen tours* problem. In these approaches the sensor model is abstract and not well suited to real environments and cameras. Distributed geometric optimization methods [53.94] have also been used for mobile sensor network reconfiguration. A related class of methods is the use of estimation-theoretic optimization metrics and the application of information filters to coordinate network-wide motion [53.56]. There are other distributed optimization methods which use a distributed control law and show that it optimizes a global metric of interest, such as using a potential field or other linear control law based only on local neighbor interactions [53.101]. Research focusing on

the control of cameras with pan, tilt, and zoom capabilities is due to [53.60, 102, 103]. In [53.102] an approach is developed to calibrate a pan–tilt–zoom camera automatically over its full zoom range and to build very high-resolution panoramas. In [53.60], the cameras are constantly moved to track observed targets, using a factor graph. A recent algorithm due to [53.104] significantly improves on this by positioning cameras to make the network better suited to detect and classify targets as they emerge. Pan–tilt–zoom cameras allow the construction of far more flexible vision systems than static cameras.

53.4.8 Control for Communication

In Sect. 53.4.5, we briefly discussed the benefits of using the communication network to synthesize and improve controller design. Conversely, the movement of robots affects the network and data transmission in the network. This gives rise to many challenges. If the controllers for individual robots are known, can we provide guarantees about communication in the network and can we develop robust information routing and networking algorithms in the presence of robot motion? Another challenge concerns how information propagates and diffuses in these networks. If the robots move under a given control model, how does a piece of information propagate through the network and what can we say about when and where that piece of information will be heard? If we know the answers to such questions, it may be possible to design controllers to realize desired communication network characteristics.

One simple control strategy that can affect network performance is to control the robot motion to ensure messages are transmitted between designated nodes. The movement of robots in a network of robots and sensors may cause network partitioning when nodes go out of range. However, the ability of the robots to move in a controlled way also leads to an opportunity to address the information routing problem in disconnected networks by turning the robots into relay nodes. The key idea here is to enable the robot holding a current message to an unavailable destination to modify their trajectory in order to relay a message. This problem has been formulated as an optimization problem. The goal is to minimize the trajectory modifications necessary to send a message to its destination. Several solutions have been proposed depending on the information that is available to the robots. If the robots' trajectories are known, path planning techniques can be used to compute who moves where to relay what. If the robots' trajectories are not known, a distributed spanning tree can be created to enable robots to keep track of each other. Each robot is assigned a region of movement and a parent in the spanning tree. When the robot leaves its region, the parent is informed. When the robot moves too far away, the spanning tree is modified.

Mobile robots can be used to create desired network topologies under suitable models of network communication. If a robot is used to emplace nodes in an environment (or if sensor nodes robotically self-deploy) to build a network, the problem is referred to as deployment. It is possible to control the motion of individual nodes to guarantee that a specified topology is maintained [53.55]. It is also possible to reposition nodes with the explicit aim of changing the network topology – the so-called mobility-based topology control problem.

A distributed algorithm for the deployment of mobile robot teams has been described by [53.105] using the concept of virtual pheromones: localized messages from one robot to another. These messages are used to generate either a gas expansion or a guided growth deployment model. Similar algorithms based on artificial potential fields are described in [53.106, 107], where the latter incorporates a connectivity constraint. An incremental deployment algorithm for mobile sensor networks is given in [53.58]; nodes are deployed one at a time into an unknown environment, with each node making use of information gathered by previously deployed nodes to determine its deployment location. The algorithm is designed to maximize network coverage while ensuring that nodes retain line of sight with one another.

Most work on network topology control has dealt with uncontrolled deployments, where there is no explicit control of the positions of individual nodes. The primary mechanisms proposed are power control and sleep scheduling. These methods involve pruning an already existing, well-connected communication graph in order to save power while ensuring that the resultant subgraph preserves connectivity. Given a network that is connected when all nodes are operating at maximum power, the aim of power control is to use the minimum power level at each node for which the network remains connected [53.108]. Given an overdeployed network, sleep scheduling seeks to activate a minimal subset of nodes to maintain connectivity and achieve other desired metrics [53.109]. In contrast, controlled deployments are feasible when the positions of individual nodes can be altered. Such deployments are interesting for two reasons. First, network topology with wireless communication relates directly to proximity relations and hence the position of the nodes. Second, there is increasing evidence that a large number of deployments are likely to involve careful, nonrandom placement of nodes. The positioning of nodes is controlled either by the nodes themselves or by external agents. Such net-

works present a different and interesting scenario for topology control since it is possible to exploit control of the motion and placement of the nodes to build efficient topologies. A local, completely decentralized technique for topology control using mobility is given in [53.110].

An important application for networked robots is in monitoring and surveillance, where it is important that the robots cover the space while remaining within communication range [53.111–113]. In a recent development [53.114] the problem of how to design communication models and scheduling protocols for choosing the appropriate path planning algorithms for robotic data collection is discussed. Probing environment and adaptive sleeping protocol (PEAS) was one of the first attempts to address communication connectivity and sensing coverage simultaneously using heuristic algorithms [53.115]. *Wang* et al. [53.116] proposed a new coverage configuration protocol (CCP) to produce an approach that simultaneously optimizes coverage and connectivity while maximizing the number of nodes that are placed into sleep mode. Furthermore, they also identified three different classes of coverage–connectivity problems with respect to the ratio of radio and sensing ranges and recognized the critical ratio where the former range is twice as long as the latter. *Zhang* and *Hou* proved that, if the communication range is at least twice the sensing range, complete coverage of a convex area guarantees network communication connectivity, and then used this theorem as a basis for a localized density control algorithm [53.109]. This was subsequently generalized to show that the condition that the communication range is twice the sensing range is sufficient and is the tight lower bound to guarantee that complete coverage preservation implies communication connectivity among nodes if the original network topology is connected [53.117].

In summary, if the state of the communication network and the desired state of the communication network is known to each agent, it should be possible to synthesize distributed controllers to move agents to attain desired network characteristics. However, the assumptions on the global state are clearly not justified. Also, the desired motion to optimize network characteristics will conflict with the motion that is required to perform the desired task. However, as the brief discussion above illustrates, there are many interesting studies that point to promising directions for future work in this very fertile research field.

53.5 Swarm Robots

Historically, some of the earliest work in multirobot systems [53.12, 13, 118–125] dealt with large numbers of homogeneous robots, called *swarms*; swarm robotics continues to be a very active area of research. Most swarm approaches obtain inspiration from biological societies – particularly ants, bees, and birds – to develop similar behaviors in multirobot teams. Because biological societies are able to accomplish impressive group capabilities, such as the ability of termites to build large complex mounds, or the ability of ants to collectively carry large prey, robotics researchers aim to reproduce these capabilities in robot societies.

Swarm robotics systems are often called *collective* robotics, indicating that individual robots are often unaware of the actions of other robots in the system, other than information on proximity. These approaches aim to achieve a desired team-level global behavior from the interaction dynamics of individual robots following relatively simple local control laws. Swarm robotic systems typically involve very little explicit communication between robots, and instead rely on stigmergy (i. e., communication through the world) to achieve emergent cooperation. Individual robots are assumed to have minimal capabilities, with little ability to solve meaningful tasks on their own. However, when grouped with other similar robots, they are collectively able to achieve team-level tasks. Ideally, the entire team should be able to achieve much more than individual robots working alone (i. e., it is *superadditive*, meaning that the whole is bigger than the sum of the parts). These systems assume very large numbers of robots (at least dozens, and often hundreds or thousands) and explicitly address issues of scalability. Swarm robotic approaches achieve high levels of redundancy because robots are assumed to be identical, and thus interchangeable with each other.

Many types of swarm behaviors have been studied, such as foraging, flocking, chaining, search, herding, aggregation, and containment. The majority of these swarm behaviors deal with spatially distributed multirobot motions, requiring robots to coordinate motions either:

1. Relative to other robots
2. Relative to the environment
3. Relative to external agents
4. Relative to robots and the environment
5. Relative to all (i. e., other robots, external agents, and the environment).

Table 53.1 categorizes swarm robot behaviors according to these groupings, citing representative examples of relevant research.

Table 53.1 Categories of swarm behaviors

Relative motion requirements	Swarm behaviors
Relative to other robots	Formations [53.126, 127], flocking [53.121], natural herding (as in herds of cattle), schooling, sorting [53.14], clumping [53.14], condensation, aggregation [53.128], dispersion [53.129]
Relative to the environment	Search [53.130], foraging [53.131], grazing, harvesting, deployment [53.58], coverage [53.132], localization [53.133], mapping [53.134], exploration [53.135]
Relative to external agents	Pursuit [53.136], predator–prey [53.137], target tracking [53.138], forced herding/shepherding (as in shepherding sheep)
Relative to other robots and the environment	Containment, orbiting, surrounding, perimeter search [53.139]
Relative to other robots, external agents, and the environment	Evasion, tactical overwatch, soccer [53.140]

Much of the current research in swarm robotics is aimed at developing specific solutions to one or more of the swarm behaviors listed in Table 53.1 (VIDEO 214). Some of these swarm behaviors have received particular attention, notably formations, flocking, search, coverage, and foraging. Section 53.10 discusses these behaviors in more detail. In general, most current work in the development of swarm behaviors is aimed not just at demonstrating group motions that are similar to biological systems, but also at understanding the formal control theoretic principles that can predictably converge to the desired group behaviors, and remain in stable states.

Demonstration of physical robot swarms is both a hardware and a software challenge. As discussed in Sect. 53.2, the first demonstrations were by *Matarić* [53.6], involving about 20 physical robots performing aggregation, dispersion, and flocking. This work defined composable *basis behaviors* as primitives for structuring more complex systems. In later research, *McLurkin* [53.141] (VIDEO 215) developed an extensive catalog of swarm behavior software, and demonstrated these behaviors on about 100 physical robots (called the SwarmBot robots), developed by iRobot, as shown in Fig. 53.12. He created several group behaviors, such as avoidManyRobots, disperseFromSource, disperseFromLeaves, disperseUniformly, computeAverageBearing, avoidManyRobots, followTheLeader, orbitGroup, navigateGradient, clusterOnSource, and clusterIntoGroups. A swarm of 108 robots used the developed dispersion algorithms in an empty schoolhouse of area of about 300 m^2, and were able to locate an object of interest and lead a human to its location [53.129].

The European Union has sponsored many swarm robot projects, leading toward decreasingly smaller-sized individual robots, and increasingly larger-sized robot teams. The I-SWARM project, for instance, aimed at developing millimeter-sized robots with full onboard sensing, computation, and power for performing biologically inspired swarming behaviors, as well as collective perception tasks. This project was both a hardware and a software challenge, in that developing microscale robots that are fully autonomous and can perform meaningful cooperative behaviors will require significant advances in the current state of the art.

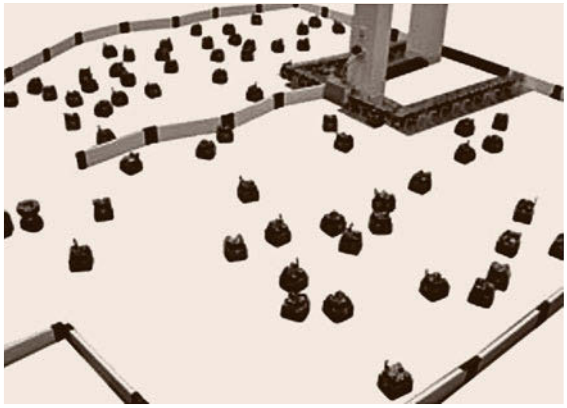


Fig. 53.12 The SwarmBot robots

The EU-based SWARM-BOTS project studied new concepts in the design and implementation of self-organizing and self-assembling robots (VIDEO 195). In this work [53.142], *s-bot* robots are developed that have grippers enabling the robots to create physical links with other s-bots or objects, thus creating assemblies of robots. These assemblies can then work together for navigation across rough terrain, or to collectively transport objects. The s-bots are cylindrical, with a flexible arm and toothed gripper that can connect one s-bot to another. An interesting application of object transport using SWARM-BOTs is shown in Figure 53.13, which shows the robots self-assembling into four chains in order to pull a child across the floor (VIDEO 212).

Another notable effort in swarm robotics research is the US multi-university SWARMS initiative led by the University of Pennsylvania. Research in this project aimed at developing a new system-theoretic frame-

Fig. 53.13 SWARM-BOTs self-assembling to move a child across the floor

work for swarming, developing models of swarms and swarming behavior, analyzing swarm formation, stability, and robustness, synthesizing emergent behaviors for active perception and coverage, and developing algorithms for distributed localization.

Besides the hardware challenges of dealing with large numbers of small robots, there are many important software challenges that remain to be solved. From a practical perspective, a common approach to creating homogeneous multirobot swarms is to hypothesize a possible local control law (or laws), and then study the resulting group behavior, iterating until the desired global behavior is obtained. However, the longer-term objective is to be able to both predict group performance based on known local control laws, and to generate local control laws based upon a desired global group behavior.

More recent research has focused on the development of analytical techniques that can synthesize distributed controllers that achieve the desired macro-level system behaviors. *Mather* and *Hsieh* [53.143] address this challenge by proposing a technique that first identifies robot–robot interactions at the macroscopic level; they then use this analysis to improve local robot control policies by filtering out spurious robot–robot interactions. Another top-down design approach is presented by *Chen* et al. [53.144] who show how to automatically synthesize control and communication strategies for a robot team based on global specifications of the desired system-level behavior, stated using regular expressions. The resulting control strategies are formally proven to correctly achieve the desired global behavior. The work of *Melo* and *Veloso* [53.145] takes a different approach to the synthesis of local decision policies by making use of the decentralized sparse-interaction Markov decision process. This technique allows agents to recognize states when interactions with other robots might occur, thus enabling them to choose better motions based on possible future inter-robot actions. *Tsiotras* and *Castro* [53.146] synthesize local controllers by generalizing the standard consensus algorithm, applied to geometric pattern formation.

A common theme in most of the works cited above is that they first make use of formal methods to describe the desired macro-level behavior, and then show how to use this macro-level goal to synthesize individual robot controllers. Another use of formal methods is to show how the individual goals of robot team members can be considered collectively, with the objective of maximizing the system's achievement of individual goals. Toward this end, game-theoretic techniques have been shown useful in a variety of distributed robot formulations. *Cheng* and *Dasgupta* [53.147] make use of the game-theoretic technique called Weighted Voting Games to address the problem of multi-robot team formation control amidst obstacles. *Taheri* et al. [53.148] also make use of game-theoretic principles, building upon the Local Interaction Game diffusion model to investigate how a small number of agents can influence the global society's behavior through local interactions.

An interesting question in the design of distributed robot coordination mechanisms is the extent to which identical controllers can lead to diversity, specialization, or changes in robot behavior. *Hsieh* et al. [53.149] study the emergence of specialization in robot swarms by making use of a distributed adaptation algorithm. They present a top-down analytical approach that defines the system equilibrium using waiting time parameters, and then present adaptive optimization strategies that converge to the optimal configurations that achieve system equilibrium. Temporal changes in system-level swarm behavior are addressed by *Hoff* et al. [53.150] who show how a swarm can change and improve its foraging behavior by switching between algorithms based on the environment in which the swarm finds itself.

Once the distributed controller is synthesized, most of the works mentioned above presume that individual robots execute their controller successfully. The typical presumption is that large swarms of interchangeable robots automatically result in robust and scalable swarm behavior. However, this presumption is challenged by *Winfield* and *Nembrini* [53.151], who illustrate that overall swarm reliability quickly falls in the presence of worst-case, partially failed robots. They conclude that future large scale swarm systems must develop new approaches for achieving high levels of fault tolerance. One example approach is shown in VIDEO 194.

53.6 Modular Robotics

The modular robotics field began with a paper presented at International Conference on Robotics and Automation in the Spring of 1988 by *Fukuda* and *Nakagawa* [53.152] describing the abstract concept of a reconfigurable robotic system that can assume different shapes and envisioned a robot system composed of different types of modules that can combine to accomplish a variety of tasks. Over the past twenty years, modular robotics research developed many facets: hardware design; planning and control algorithms; the trade-off between hardware and algorithmic complexity; efficient simulation; and system integration.

Modular robots are collections of physically connected, electromechanically active modules that, as a whole, form robotic systems that exhibit capabilities greater than those of the individual modules. Typically modular robots can change their shape or configuration in order to adapt to a variety of different tasks. For example, a collection of modules could reconfigure from a closed chain that rolls quickly over open ground to a legged robot that more easily traverses rough terrain. Modular robots are typically touted for their adaptability, their fault-tolerance, and the relative simplicity of the unit modules. Modular robotic systems can be described and classified on several axes using a variety of properties. In what follows, we choose the traditional route of classifying modular robotic systems by the geometry of the system: chain, lattice, truss, or free form. For a more detailed history of the modular robotics field, consult [53.153, 154].

53.6.1 Chain Systems

The defining characteristic of chain-type modular robot systems is the fact that the modules, when connected to their neighbors, are arranged in a chain. These chains may be one-dimensional, or two-dimensional, but three-dimensional chains are not as common. The fact that a chain-type modular robot is two-dimensional, or even one-dimensional, does not mean that it cannot operate in three dimensions. In fact, snake-like modular robots composed of segments with orthogonal joints are quite common.

One of the first chain-type modular robotic systems was the polypod system developed by *Yim* [53.155, 156] (VIDEO 196). The polypod system was composed of two types of modules: segments and nodes. It could form a variety of shapes including rolling loops and hexapods, and it went on to inspire many other chain-based systems. One was the CONRO system [53.157–159] in which each module was composed of two orthogonal servo motors controlling each module's pitch and yaw.

Murata et al. developed the M-TRAN modular robotic system [53.160–163] which has undergone multiple revisions and improvements. In [53.161], Kamimura et al. employ a set of interconnected, out of phase oscillators to achieve walking gaits in the M-TRAN system. *Marbach* and *Ijspeert* improved upon the ability of systems like M-TRAN to generate gaits in real-time by applying function optimization to their modular system, YaMoR [53.164]. *Murata* et al. added cameras to the M-TRAN system so that a set of M-TRAN modules could separate, perform independent tasks, and then rejoin into a larger structure [53.163].

The ATRON system [53.165, 166] was developed to improve upon the M-TRAN. Lund et al. wanted to keep M-TRAN's ability to form dense lattices while taking advantage of the two orthogonal degrees of freedom, (pitch and yaw), found in the CONRO system. The Superbot system [53.167] also builds upon on the mechanical design of M-TRAN by adding an additional degree of rotational freedom between the two existing rotation axes.

The PolyBot is chain-type modular robot [53.168, 169] with a single rotational degree of freedom. PolyBot evolved into CKBot which has demonstrated the ability to reassemble itself after being accidentally or intentionally destroyed [53.170]. The Molecube system [53.171], developed by *Lipson* et al. is another example of a chain-type modular system with only one degree of freedom but still able to achieve interesting three-dimensional (3-D) configurations. Lipson et al. showed that a short chain of Molecube modules, along with some free modules, can self-replicate.

Yim et al. designed another unique chain-type system named RATChET [53.172] which uses a connected chain of inter-latching right angle tetrahedrons to form structures. Neighboring RATChET modules latch together when the angle between them passes some critical value, and they unlatch through the use of shape memory alloy (SMA) springs when heated beyond 70 °C. Interestingly, the RATChET modules possess no intelligence. Instead, they rely on an intelligent external actuator which rotates to control one end of the dangling chain. One unique property of the RATChET system is its relatively strength.

53.6.2 Lattice Systems

Lattice-type modular robot systems are collections of interconnected robotic modules in which the units are situated at the intersection points of a two or three dimensional grid. (A 1-D (one-dimensional) lattice system is simply a chain-type robot.) The main characteris-

tic separating a lattice system from a densely configured chain-type robot is the density of the interconnections between the modules. In a lattice-type system, each module is typically connected to all of its neighbors. In a dense chain-type system, two modules may be neighbors, but they will not be physically connected.

Additionally, lattice-type systems tend to be built with modules that contain no rotational degrees of freedom. While the modules in a lattice system typically have mechanisms which enable the modules to move relative to, and bond with, their neighbors, they generally cannot bend themselves. In comparison, chain-type systems are often built from modules that contain one or more rotational degrees of free so that the modules can flex like links in a chain. There is some overlap between between the two types of system.

Chirikjian et al. developed one of the first lattice-based modular robotic systems [53.173–175] (VIDEO 198) in which the modules are deformable hexagons capable of bonding with their neighbors. Others, such as *Walter* et al. [53.176] further analyzed these hexagonal type systems to create distributed motion planners capable of reconfiguring the system from one state to another.

Murata et al. were also early contributors to the development of lattice-based modular robotic systems with their development of a roughly hexagonal module capable of rolling around its neighbors in two dimensions [53.177, 178]. *Kurokawa* et al. presented a three dimensional adaptation [53.179] composed of cubes with six protruding arms capable of rotation. *Yoshida* et al. improved on this system with a new design that used shape-memory alloy actuators to rotate one robot module around the perimeter of a neighbor [53.180].

One of the simplest lattice systems is the the Digital Clay project [53.181]. The system was a set of completely passive modules that relied on the user to make changes to it topology. The 2.5 cm rhombic dodecahedrons were able to sense and communicate with their neighbors in order to create a virtual model of the physical arrangement of modules.

Rus et al. also explored the idea of 3-D modules capable reconfiguration through a series of latchings, rotations, and unlatchings with the Molecule system [53.182–185]. In [53.186, 187], *Vona* and *Rus* describe a different type of deformable lattice system. The Crystal system is composed of square modules able to expand and contract by a factor of two in the x–y plane. *Suh* et al. expanded on the Crystal concept with the Telecubes [53.188] that could move in three dimensions by expanding all six faces.

Chiang and *Chirikjian* analyzed how to perform motion planning in a lattice of rigid cubic modules able to slide past each other [53.189]. The CHOBIE robot developed by *Koseki* [53.190] is able to actually perform the sliding motion assumed by *Chiang* and *Chirikjian* in [53.189]. More recently, *An* developed the EM-Cube system [53.191] which is also capable of sliding motion.

Another unique lattice is the I-Cube developed by *Khosla* et al. [53.192, 193]. The 3-D I-Cube system consists of passive cubes which are connected by active links with three rotational degrees of freedom that are able to grab, reposition, and release the cubes. The 3-D I-Cube system was an improvement of the two-dimensional (2-D) system [53.194] developed by *Hosokawa* et al. for rearranging cubic modules in a vertical plane.

Goldstein et al. initiated the Claytronics project by publishing several papers [53.195, 196] proposing lattice-based *claytronic atoms* or catoms. These vertically-oriented cylindrical robots, which were incapable of independent motion, used 24 electromagnets around their perimeters to achieve rolling locomotion about their neighbors. Goldstein et al. envisioned a system in which millions of smaller catoms could form arbitrary shapes using a randomized algorithm that avoided conveying a complete description of the shape to each module in the system.

The catoms continue to evolve. One of the newest instantiations [53.197] employs hollow cylinders rolled from SiO_2 rectangles patterned with aluminum electrodes. The authors hope that two of these cylinders, when placed in close proximity with their axes aligned, will be able to rotate with respect to one another using electrostatic forces. Specifically, the electrodes, (which reside on the inside of each cylinder and are electrically isolated by the SiO_2), will be charged so that they attract and repel mirror charges on the neighboring cylinder in a way that causes rotation. Currently, the system appears to be constrained to form 2-D structures. The authors claim the completed system will have a yield strength similar to that of plastic and that the modules will be able to transfer power and communication signals capacitively from neighbor to neighbor.

The Claytronics project has proposed, but not yet demonstrated with hardware, the use of sub-millimeter intelligent particles as sensing and replication devices [53.198]. In particular, Pillai et al. present a theoretical 3-D fax machine in which the object to be *faxed* is immersed in a container of intelligent particles that sense and encode the object's dimensions. At the receiving end, these same Claytronic particles decode the shape description sent by the transmitter and bond together to replicate the original object. Unlike our approach, Pillai's approach is completely centralized and relies on an external computer for computation.

White et al. developed hardware and algorithms for several 2-D stochastically-driven self-assembling sys-

tems [53.199]. To form specific shapes, each module is provided with a representation of the desired shape and decides, based on its location in the structure, whether to allow other modules to bond to its faces. *Lipson* et al. extended their 2-D system to 3-D [53.200–202] by using cubic modules suspended in turbulent fluid to achieve self-assembly and reconfiguration. As the free modules circulate in the fluid, they pass by a growing structure of assembled modules. When they come close enough, they are accreted onto the structure. The modules attract or repel each other with fluid suction or positive pressure. Early versions of the system used modules with interval values that could redirect these suction forces. More recently, *Lipson*'s group has worked to move the intelligence and actuation capabilities from the modules to the tank in which the modules circulate [53.203].

The Miche system [53.204] consisting of 45 mm cubic modules capable of mating with their neighbors using mechanically switchable permanent magnets. Each module contains three switchable magnets, each of which mated with a steel face on a neighboring module. Because the connectors were gendered, any collection of modules had to be assembled by hand so that the connectors were always oriented correctly, but the system was capable of self-disassembling to form 3-D structures. The Robot Pebbles (VIDEO 211) are based, at least in principle, on the Miche modules.

One of the newest lattice-type modular robotics is an aerial system composed of identical, hexagonal, single-rotor modules [53.205]. A group of modules may connect to form a flying platform with an arbitrary arrangement of multiple rotors. In addition to the ability to fly, each module contains wheels so that the system may self-reconfigure on the ground for the specific task at hand.

53.6.3 Truss Systems

Truss systems, as their name implies, are modular robotic systems in which the modules are nodes and edges in a truss structure. Both the trusses and connectors may be active in such systems. Unlike the lattice-based systems, truss-based systems do not need to operate on any regular lattice. Most truss-based systems under development employ struts that expand or contract to achieve structural deformation. One of the first such system to do so was Tetrobot [53.206]. The Odin system, conceived by *Lyder* et al. [53.207, 208] consists of three physically different types of modules: active strut modules capable of changing their length; passive strut modules of fixed length; and joint modules. The biologically inspired Morpho system [53.209] developed by *Nagpal* et al. is similar to Odin. It also uses active links, passive links, and connector cubes.

53.6.4 Free-Form Systems

Free-form systems are able to aggregate modules in at least semi-arbitrary positions. One such system is the Slimebot [53.210, 211]. The system consists of identical vertical cylindrical modules that move on a horizontal plane. The perimeter of each module is covered by six gender-less hook and loop patches used to bond with neighboring modules. These patches oscillate radially in and out from the center of the body. By controlling the frequency and phase of the oscillations between neighbors, the system can achieve aggregate motion in a given direction.

Researchers are also developing algorithms for free-form systems. *Funiak* et al. developed a localization algorithm [53.212] that is capable of localizing tens-of-thousands of irregularly packed modules in 3-D. Rubenstein and Shen developed a number of shape formation algorithms for collections of two-dimensional modules. These algorithms allow an arbitrary-sized collection of modules to form arbitrary scale-independent shapes [53.213, 214]. Once the shape is formed, modules can be added to or removed from the system, and the system will reconfigure itself to incorporate the new modules. The resulting shape will grow or shrink, but its basic form will remain unchanged. Recently, *Rubenstein* et al. developed a 1000 modules hardware platform on which to deploy these algorithms [53.215].

Researchers have also explored the use of folding to create reconfigurable foldable systems [53.216, 217]. These systems use flexible wiring and shape memory alloy actuators embedded in composite sheets to programmatically create origami-inspired shapes. By controlling which actuators are energized, the system can form multiple different shapes.

53.6.5 Self-Assembling Systems

Self-assembling modular robotic systems are collection of modules that are capable of autonomously coalescing and bonding with their neighbors to form a greater structure. The result is often robotic, but it need not be. Whether a system is capable of self-assembling is independent of whether it is free-form, a chain, a lattice, or a truss-based system. Almost all of aforementioned modular robot systems rely on human intervention to assemble. In an attempt to automate the process of creating intricate modular robotic systems, researchers have attempted to mimic and improve upon natural self-assembling systems. *Whitesides* et al. investigated a wide variety of engineered self-assembling systems [53.218–220].

Miyashita et al. performed a more theoretical analysis of self-assembly using pie-shaped pieces

to form complete circles [53.221] from pie-shaped pieces. In the process, they followed *Hosokawa* et al.'s lead [53.222] and modeled the system as a chemical reaction. *Shimizu* and *Suzuki* developed a system of passive modules capable of self-repair when placed on a vibrating table [53.223].

Computer scientists have also investigated theoretical aspects of self-assembly in the context of 2-D tiles which selectively bond with their neighbors to form simple well-defined shapes like squares [53.224–226]. Each side of every tile in the system has an associated bonding strength. When two tiles collide, they remain attached only if their cumulative bond strength exceeds a globally defined system entropy. To form a specific shape, one must design a set of tiles with the appropriate bonding strengths.

Klavins et al. worked to develop intelligent self-assembling systems that employ triangular modules driven by oscillating fans on an air table to self-assemble different shapes [53.227]. The authors employ knowledge of the module's local topology and internal module state so that each module decides, in a distributed fashion, when to maintain or break a connection with its immediate neighbors. *Griffith* et al. also worked with intelligent modules capable of selective bonding to show that self-assembling systems may self-replicate [53.228].

Rus et al. [53.185] present the first generic rule-based approach to self-assembly, shape formation, and locomotion by reconfiguration. The rules can be used on any modular robot system that can implement the sliding cube model of relative motion. The result is an abstract set of rules for each of these tasks, that can be compiled down to module motions, taking into account how the physical module implementes translation and convex and concave transitions.

Jones and *Matarić* [53.229] presented rule-based approach to self-assembly termed transition rule sets. In particular, they present a method that, given a goal structure, produces a set of rules shared among all modules that govern when and where new modules are allowed to attach to the growing structure. *Kelly* and *Zhang* [53.230] expanded on this work by optimizing the size of the rule sets used to form a specific shape. *Werfel* [53.231] also applied the idea of a transition rule set when studying the use of swarms to assemble complex structures from passive materials.

Other groups have attempted to make self-assembly more deterministic. The MEMS (microelectromechanical system) robots developed by *Donald* et al. [53.232, 233] consists of thin ($7-20\,\mu m$), rectangular ($\approx 260\,\mu m \times 60\,\mu m$), scratch-drive devices capable of moving on an insulating substrate embedded with electrodes. The authors used four of these robots to build larger composite structures. The Sitti group has developed a similar system of micro-meter sized robots [53.234]. Instead of using a scratch drive for locomotion, the robots are manipulated by external magnetic fields. The authors can electrostatically clamp any number of robots to the stage on which they move. With all but one robot immobilized, the remaining robot may be moved independently. The system naturally self-assembles because the robots contain permanent magnets that attract their neighbors.

The majority of existing self-assembly systems aim to form structures in one of two ways. Some systems such as [53.221, 223–226] use a collection of application specific differentiated modules, that are only capable of assembling in a particular fashion to form a specific shape. In contrast, other systems such as [53.199–201, 227, 229–231, 235] use completely generic modules with more computation and communication ability embedded in each module. Both types of systems aim to form complex shapes in a direct manner: as these structures grow from a single module, new modules are only allowed to attach to the structure in specific locations.

An alternative approach eliminates many of the complexities of shape formation by active assembly by using dissasemly for shape formation. The Smart Pebble system [53.154, 204, 236, 237] employs a set of distributed algorithms to perform two discrete steps: 1) rely on stochastic forces to self-assemble a close-packed crystalline lattice of modules and 2) use the process of self-disassembly to remove the extra material from this block leaving behind the goal structure. By approaching shape formation in this manner, the entire shape-formation process is sped-up and the robustness of the system is increased.

53.7 Heterogeneity

Robot heterogeneity can be defined in terms of variety in robot behavior, morphology, performance quality, size, and cognition. In most large-scale multirobot systems work, the benefits of parallelism, redundancy, and solutions distributed in space and time are obtained through the use of homogeneous robots, which are completely interchangeable (i. e., the swarm approach, as described in Sect. 53.4). However, certain complex applications of large-scale robot teams may require the simultaneous use of multiple types of sensors and

robots, all of which cannot be designed into a single type of robot. Some robots may need to be scaled to smaller sizes, which will limit their payloads, or certain required sensors may be too expensive to duplicate across all robots on the team. Other robots may need to be large to carry application-specific payload or sensors, or to navigate long distances in a limited time. These applications, therefore, require the collaboration of large numbers of heterogeneous robots.

The motivation for developing heterogeneity in multirobot teams is thus twofold: heterogeneity may be a design feature beneficial to particular applications, or heterogeneity may be a necessity. As a design feature, heterogeneity can offer economic benefits, since it can be easier to distribute varying capabilities across multiple team members rather than to build many copies of monolithic robots. Heterogeneity can also offer engineering benefits, as it may simply be too difficult to design individual robots that incorporate all of the sensing, computational, and effector requirements of a given application. Heterogeneity in behavior may also arise in an emergent manner in physically homogeneous teams, as a result of behavior specialization.

A second compelling reason to study heterogeneity is that it may be a necessity, in that it is nearly impossible in practice to build a truly homogeneous robot team. The realities of individual robot design, construction, and experience will inevitably cause a multirobot system to drift to heterogeneity over time. This is recognized by experienced roboticists, who have seen that several copies of the same model of robot can vary widely in capabilities due to differences in sensor tuning, calibration, etc. Over time, even minor initial differences among robots will grow due to individual robot drift and wear and tear. The implication is that, to employ robot teams effectively, we must understand diversity, predict how it will impact performance, and enable robots to adapt to the diverse capabilities of their peers. In fact, it is often advantageous to build diversity explicitly into the design of a robot team.

There are a variety of research challenges in heterogeneous multirobot systems. A particular challenge to achieving efficient autonomous control is when overlap in team member capabilities occurs, thus affecting task allocation or role assignments [53.238]. Techniques as described in Sect. 53.6 can typically deal with heterogeneous robots for the purposes of task allocation. Another important topic in heterogeneity is how to recognize and quantify heterogeneity in multirobot teams. Some types of heterogeneity can be evaluated quantitatively, using metrics such as the social entropy metric developed by *Balch* [53.239]. Most research in heterogeneous multirobot systems assumes that robots have a common language and a common understanding of symbols in their language; developing a common understanding of communicated symbols among robots with different physical capabilities is a fundamental challenge, addressed by *Jung* and *Zelinsky* [53.240] (VIDEO 200).

As discussed in Sect. 53.2, one of the earliest research demonstrations of heterogeneity in physical robot teams was in the development of the ALLIANCE architecture by *Parker* [53.9]. This work demonstrated the ability of robots to compensate for heterogeneity in team members during task allocation and execution. *Murphy* has studied heterogeneity in the context of marsupial robot deployment, where a *mothership* robot assists smaller robots in applications such as search and rescue [53.241] (VIDEO 206). *Grabowski* et al. [53.134] developed modular millibots for surveillance and reconnaissance that could be composed of interchangeable sensor and effector components, thus creating a variety of different heterogeneous teams. *Simmons* et al. [53.11] demonstrated the use of heterogeneous robots for autonomous assembly and construction tasks relevant to space applications. *Sukhatme* et al. [53.242] demonstrated a helicopter robot cooperating with two ground robots in tasks involving marsupial-inspired payload deployment and recovery, cooperative localization, and reconnaissance and surveillance tasks, as shown in Fig. 53.14. *Parker* et al. [53.243] demonstrated assistive navigation for sensor network deployment using a more intelligent leader robot for guiding navigationally challenged simple sensor robots to goal locations, as part of a larger demonstration by *Howard* et al. [53.244] of 100 robots performing exploration, mapping, deployment, and detection. *Chaimowicz* et al. [53.245] demonstrated a team of aerial and ground robots cooperating for surveillance applications in urban environments. *Parker* and *Tang* [53.246] developed ASyMTRe (automated

Fig. 53.14 Heterogeneous team of an air and two ground vehicles that can perform cooperative reconnaissance and surveillance

synthesis of multirobot task solutions through software reconfiguration), which enables heterogeneous robots to share sensory resources to enable the team to accomplish tasks that would be impossible without tightly coupled sensor sharing.

Many open research issues remain to be solved in heterogeneous multirobot teams; for example, the issue of optimal team design is a very challenging problem. Clearly, the required behavioral performance in a given application dictates certain constraints on the physical design of the robot team members. However, it is also clear that multiple choices may be made in designing a solution to a given application, based upon cost, robot availability, ease of software design, flexibility in robot use, and so forth. Designing an optimal robot team for a given application requires significant analysis and consideration of the tradeoffs in alternative strategies.

53.8 Task Allocation

In many multirobot applications, the mission of the team is defined as a set of tasks that must be completed. Each task can usually be worked on by a variety of different robots; conversely, each robot can usually work on a variety of different tasks. In many applications, a task is decomposed into independent subtasks [53.9], hierarchical task trees [53.247], or roles [53.11, 245, 248, 249] either by a general autonomous planner or by the human designer. Independent subtasks or roles can be achieved concurrently, while subtasks in task trees are achieved according to their interdependence. Once the set of tasks or subtasks have been identified, the challenge is to determine the preferred mapping of robots to tasks (or subtasks). This is the *task allocation* problem.

The details of the task allocation problem can vary in many dimensions, such as the number of robots required per task, the number of tasks a robot can work on at a time, the coordination dependencies among tasks, and the time frame for which task assignments are determined. *Gerkey* and *Matarić* [53.250] defined a taxonomy for task allocation that provides a way of distinguishing task allocation problems along these dimensions, which is referred to as the multirobot task allocation (MRTA) taxonomy.

53.8.1 Taxonomy for Task Allocation

Generally, tasks are considered to be of two principal types: *single-robot tasks* (SR, according to the MRTA taxonomy) are those that require only one robot at a time, while *multirobot tasks* (MR) are those that require more than one robot working on the same task at the same time. Commonly, single-robot tasks that have minimal task interdependencies are referred to as *loosely coupled tasks*, representing a *weakly cooperative* solution. On the other hand, multirobot tasks are often considered to be sets of subtasks that have strong interdependencies. These tasks are therefore often referred to as *tightly coupled tasks* that require a *strongly cooperative* solution. The subtasks of a loosely coupled multirobot task require a high level of synchronization or coordination between subtasks, meaning that each task must be aware of the current state of the coordinated subtasks within a small time delay. As this time delay becomes progressively larger, coordinated subtasks become more loosely coupled, representing weakly cooperative solutions.

Robots can also be categorized as either *single-task robots* (ST), which work on only one task at a time or *multitask robots* (MT), which are able to make progress on more than one task at a time. Most commonly, task allocation problems assume robots are single-task robots, since more capable robots that perform multiple tasks in parallel are still beyond the current state of the art.

Tasks can either be assigned to optimize the instantaneous allocation of tasks (IA), or to optimize the assignments into the future (TA, for time-extended assignment). In the case of instantaneous assignment, no consideration is made for the effect of the current assignment on future assignments. Time-extended assignments attempt to assign tasks so that the performance of the team is optimized for the entire set of tasks that may be required, not just the current set of tasks that need to be achieved at the current time step.

Using the MRTA taxonomy, triples of these abbreviations are used to categorize various task allocation approaches, such as SR-ST-IA, which refers to an assignment problem in which single-robot tasks are assigned once to single-task robots. Different variations of the task allocation problem have different computational complexities. The easiest variant is the ST-SR-IA problem, which can be solved in polynomial time since it is an instance of the optimal assignment problem [53.251]. Other variants are much more difficult, and do not have known polynomial time solutions. For example, the ST-MR-IA variant can be shown to be an instance of the set partitioning problem [53.252], which is strongly *NP*-hard. The ST-MR-TA, MT-SR-IA, and MT-SR-TA variants have also all been shown to be *NP*-hard problems. Because these problems are computationally complex, most approaches to task allocation in multirobot teams generate approximate solutions.

53.8.2 Representative Approaches

Approaches to task allocation in multirobot teams can be roughly divided into behavior-based approaches and market-based (sometimes called negotiation-style or auction-based) approaches. The following subsections describe some representative architectures for each of these general approaches. Refer to [53.250] for a comparative analysis of some of these approaches, in terms of computation and communications requirements and solution quality.

Behavior-Based Task Allocation

Behavior-based approaches typically enable robots to determine task assignments without explicitly discussing individual tasks. In these approaches, robots use knowledge of the current state of the robot team mission, robot team member capabilities, and robot actions to decide, in a distributed fashion, which robot should perform which task.

One of the earliest architectures for multirobot task allocation that was demonstrated on physical robots was the behavior-based ALLIANCE architecture [53.9] and the related L-ALLIANCE architecture [53.10]. ALLIANCE addresses the ST-SR-IA and ST-SR-TA variants of the task allocation problem without explicit communication among robots about tasks. As described in Sect. 53.2.2, ALLIANCE achieves adaptive action selection through the use of motivational behaviors, which are levels of impatience and acquiescence within each robot that determine its own and its teammates' relative fitness for performing certain tasks. These motivations are calculated based upon the mission requirements, the activities and capabilities of teammates, and the robots' internal states. These motivations effectively calculate utility measures for each robot–task pair.

Another behavior-based approach to multirobot task allocation is broadcast of local eligibility (BLE) [53.253], which addresses the ST-SR-IA variant of task allocation. BLE uses a subsumption style behavior control architecture [53.254] that allows robots to efficiently execute tasks by continuously broadcasting locally computed eligibilities and only selecting the robot with the best eligibility to perform the task. In this case, task allocation is achieved through behavior inhibition. BLE uses an assignment algorithm that is very similar to *Botelho* and *Alami*'s M+ architecture [53.255].

Market-Based Task Allocation

Market-based (or negotiation-based) approaches typically involve explicit communications between robots about the required tasks, in which robots bid for tasks based on their capabilities and availability. The negotiation process is based on market theory, in which the team seeks to optimize an objective function based upon individual robot utilities for performing particular tasks. The approaches typically greedily assign subtasks to the robot that can perform the task with the highest utility.

Smith's contract net protocol (CNP) [53.256] was the first to address the problem of how agents can negotiate to collectively solve a set of tasks. The use of a market-based approach specifically for multirobot task allocation was first developed by *Botelho* and *Alami* with their M+ architecture [53.255]. In the M+ approach, robots plan their own individual plans for the task they have been assigned. They then negotiate with other teammates to incrementally adapt their actions to suit the team as a whole, through the use of social rules that facilitate the merging of plans.

Since these early developments, many alternative approaches to market-based task allocation have been developed. A thorough survey on the current state of the art in market-based techniques for multirobot task allocation is given in [53.257], comparing alternative approaches in terms of solution quality, scalability, dynamic events and environments, and heterogeneous teams.

Most of the current approaches in market-based task allocation address the ST-SR problem variant, with some approaches (e.g., [53.11, 258–260]) dealing with instantaneous assignment (IA), and others (e.g., [53.135, 261–263]) addressing time-extended assignments (TA). More recent methods are beginning to address the coaltion formation problem, which is the allocation of multirobot tasks (i. e., the MR-ST problem variant), including [53.246, 264–269]. An example approach to the MR-MT problem variant is found in [53.270].

Some representative market-based techniques include MURDOCH [53.258], TraderBots [53.247, 263], and Hoplites [53.265]. The MURDOCH approach [53.258] employs a resource-centric, publish–subscribe communication model to carry out auctions, which has the advantage of anonymous communication. In this approach a task is represented by the required resources, such as the environmental sensors. The methods for how to use such a sensor to generate satisfactory results is preprogrammed into the robot.

The TraderBots approach [53.247, 263] applies market economy techniques for generating efficient and robust multirobot coordination in dynamic environments. In a market economy, robots act based on selfish interests. A robot receives revenue and incurs cost when trying to accomplish a task. The goal is for robots to

trade tasks through auctions/negotiations such that the team profit (revenue minus cost) is optimized.

The Hoplites approach [53.265] focuses on the selection of an appropriate joint plan for the team to execute by incorporating joint revenue and cost into the bid. This approach couples planning with passive and active coordination strategies, enabling robots to change coordination strategies as the needs of the task change. Strategies are predefined for a robot to accomplish a selected plan.

Some alternative approaches formulate the objects to be assigned as *roles*, which typically package a set of tasks and/or behaviors that a robot should undertake when acting in a particular role. Roles can then be dynamically assigned to robots in a similar manner as in the auction-based approaches [53.11, 248].

53.9 Learning

Multirobot learning is the problem of learning new cooperative behaviors, or learning in the presence of other robots. The other robots in the environment, however, have their own goals and may be learning in parallel [53.271]. The challenge is that having other robots in the environment violates the Markov property that is a fundamental assumption of single-robot learning approaches [53.271]. The multirobot learning problem is particularly challenging because it combines the difficulties of single-robot learning with multiagent learning. Particular difficulties that must be considered in multirobot learning include continuous state and action spaces, exponential state spaces, distributed credit assignment, limited training time and insufficient training data, uncertainty in sensing and shared information, nondeterministic actions, difficulty in defining appropriate abstractions for learned information, and difficulty of merging information learned from different robot experiences.

The types of applications that have been studied for multirobot learning include multitarget observation [53.272, 273], air fleet control [53.274], predator–prey [53.137, 275, 276], box pushing [53.277], foraging [53.23], and multirobot soccer [53.140, 278]. Particularly challenging domains for multirobot learning are those tasks that are *inherently* cooperative. Inherently cooperative tasks are those that cannot be decomposed into independent subtasks to be solved by individual robots. Instead, the utility of the action of one robot is dependent upon the current actions of the other team members. This type of task is a particular challenge in multirobot learning, due to the difficulty of assigning credit for the individual actions of the robot team members.

The credit assignment problem is a particular challenge, since it is difficult for a robot to determine whether the fitness (either good or bad) is due to its own actions, or due to the actions of another robot. As discussed by *Pugh* and *Martinoli* in [53.279], this problem can be especially difficult in situations where robots do not explicitly share their intentions. Two different variations of the credit assignment problem are common in multirobot learning. The first is when robots are learning individual behaviors in the presence of other robots that can affect their performance. The second is when robots are attempting to learn a task with a shared fitness function. It can be difficult to determine how to decompose the fitness function to appropriately reward or penalize the contributions of individual robots.

While learning has been explored extensively in the area of single-robot systems (see, for example, the discussion of learning in behavior-based systems in Chap. 13, and a discussion of fundamental learning techniques in Chap. 15) and in multi*agent* systems [53.280], much less work has been done in the area of multi*robot* learning, although the topic is gaining increased interest. Much of the work to date has focused on reinforcement learning approaches. Some examples of this multirobot learning research include the work by *Asada* et al. [53.281], who propose a method for learning new behaviors by coordinating previously learned behaviors using Q-learning, and apply it to soccer-playing robots. *Matarić* [53.8] introduces a method for combining basic behaviors into higher-level behaviors through the use of unsupervised reinforcement learning, heterogeneous reward functions, and progress estimators. This mechanism was applied to a team of robots learning to perform a foraging task. *Kubo* and *Kakazu* [53.282] proposed another reinforcement learning mechanism that uses a progress value for determining reinforcement, and applied it to simulated ant colonies competing for food. *Fernandez* and *Parker* [53.272] apply a reinforcement learning algorithm that combines supervised function approximation with generalization methods based on state-space discretization, and apply it to robots learning the multi-object tracking problem. *Bowling* and *Veloso* [53.271] developed a general-purpose, scalable learning algorithm called GraWoLF (gradient-based win or learn fast), which combines gradient-based policy learning

techniques with a variable learning rate, and demonstrated the results in the adversarial multirobot soccer application.

Other multirobot learning approaches not based on reinforcement include *Parker*'s L-ALLIANCE architecture [53.10], which uses parameter tuning, based on statistical experience data, to learn the fitness of different heterogeneous robots in performing a set of tasks. *Pugh* and *Martinoli* [53.279] apply particle swarm optimization techniques to distributed unsupervised robot learning in groups, for the task of learning obstacle avoidance.

53.10 Applications

Many real-world applications can potentially benefit from the use of multiple mobile robot systems. Example applications include container management in ports [53.283], extraplanetary exploration [53.284], search and rescue [53.241], mineral mining, transportation, industrial and household maintenance, construction [53.11], hazardous waste cleanup [53.9], security [53.285, 286], agriculture, and warehouse management [53.287] (VIDEO 210). Multiple robot systems are also used in the domain of localization, mapping, and exploration; Chap. 46 mentions some of the work in multirobot systems applied to these problems. Parts F and G of this handbook outline many application areas that are relevant not only to single-robot systems, but also to multiple mobile robot systems. To date, relatively few real-world implementations of these multirobot systems have occurred, primarily due to the complexities of multiple robot systems and the relative newness of the supporting technologies. Nevertheless, many proof-of-principle demonstrations of physical multirobot systems have been achieved, and the expectation is that these systems will find their way into practical implementations as the technology continues to mature.

Research in multiple mobile robot systems is often explored in the context of common application test domains. While not yet elevated to the level of benchmark tasks, these common domains do provide opportunities for researchers to compare and contrast alternative strategies to multirobot control. Additionally, even though these common test domains are usually just laboratory experiments, they do have relevance to real-world applications. This section outlines these common application domains; see also [53.2] and [53.288] for a discussion of these domains and a more detailed listing of related research.

53.10.1 Foraging and Coverage

Foraging is a popular testing application for multirobot systems, particularly for those approaches that address swarm robotics, involving very large numbers of mobile robots. In the foraging domain, objects such as pucks or simulated food pellets are distributed across the planar terrain, and robots are tasked with collecting the objects and delivering them to one or more gathering locations, such as a home base. Foraging lends itself to the study of weakly cooperative robot systems, in that the actions of individual robots do not have to be tightly synchronized with each other. This task has traditionally been of interest in multirobot systems because of its close analogy to the biological systems that motivate swarm robotics research. However, it also has relevance to several real-world applications, such as toxic waste cleanup, search and rescue, and demining. Additionally, since foraging usually requires robots to completely explore their terrain in order to discover the objects of interest, the *coverage* domain has similar issues to the foraging application. In coverage, robots are required to visit all areas of their environment, perhaps searching for objects (such as landmines) or executing some action in all parts of the environment (e.g., for floor cleaning). The coverage application also has real-world relevance to tasks such as demining, lawn care, environmental mapping, and agriculture.

In foraging and coverage applications, a fundamental question is how to enable the robots to explore their environments quickly without duplicating actions or interfering with each other. Alternative strategies can include basic stigmergy [53.14], forming chains [53.120], and making use of heterogeneous robots [53.131]. Other research demonstrated in the foraging and/or coverage domain includes [53.23, 132, 289–294].

53.10.2 Flocking and Formations

Coordinating the motions of robots relative to each other has been a topic of interest in multiple mobile robot systems since the inception of the field. In particular, much attention has been paid to the flocking and formation control problems (VIDEO 217, VIDEO 293). The flocking problem could be viewed as a subcase of the formation control problem, requiring robots to move together along some path in the aggregate, but with only minimal requirements for paths taken by specific robots. Formations are stricter, requiring robots to

maintain certain relative positions as they move through the environment. In these problems, robots are assumed to have only minimal sensing, computation, effector, and communications capabilities. A key question in both flocking and formation control research is determining the design of local control laws for each robot that generate the desired emergent collective behavior. Other issues include how robots cooperatively localize themselves to achieve formation control [53.133, 295], and how paths can be planned for permutation-invariant multirobot formations [53.296].

Early solutions to the flocking problem in artificial agents were generated by *Reynolds* [53.297] using a rule-based approach. Similar behavior- or rule-based approaches have been used physical robot demonstrations and studies, such as in [53.121, 298]. These earlier solutions were based on human-generated local control rules that were demonstrated to work in practice. More recent work is based on control theoretic principles, with a focus on proving stability and convergence properties in multirobot team behaviors. Examples of this work include [53.128, 299–307]. Refer to [53.308, 309] for surveys of relevant control theoretic work.

53.10.3 Object Transportation and Cooperative Manipulation

Some of the earliest work in swarm robotics was aimed at the object transportation task [53.13, 123, 310–313], which requires a team of robots to move an object from its current position in the environment to some goal destination (VIDEO 193). The primary benefit of using collective robots for this task is that the individual robots can combine forces to move objects that are too heavy for individual robots working alone or in small teams. However, the task is not without its challenges; it is non-trivial to design decentralized robot control algorithms that can effectively coordinate robot team members during object transportation. A further complication is that the interaction dynamics of the robots with the object can be sensitive to certain object geometries [53.314, 315] and object rotations during transportation [53.315], thus exacerbating the control problem.

Object transportation and cooperative manipulation are popular domains for demonstrating multirobot cooperation, because they offer a clear domain where close coordination and cooperation is required. A common type of object transportation – box pushing – requires robot teams to move boxes from their starting positions to defined goal configurations, sometimes along specified paths. Typically, box pushing operates in the plane, and the assumption is made that the boxes are too heavy or too long to enable single robots to push alone. Sometimes there are several boxes to be moved, with ordering dependencies constraining the sequence of motions. Cooperative manipulation is similar, except it requires robots to lift and carry objects to a destination. This test bed domain lends itself to the study of strongly cooperative multirobot strategies, since robots often have to synchronize their actions to successfully execute these tasks. The domain of box pushing and cooperative manipulation is also popular because it has relevance to several real-world applications [53.288], including warehouse stocking, truck loading and unloading, transporting large objects in industrial environments, and assembly of large-scale structures.

Researchers usually emphasize different aspects of their cooperative control approach in the box pushing and cooperative manipulation domain. For example, *Kube* and *Zhang* [53.13] (VIDEO 199) demonstrate how swarm-type cooperative control techniques could achieve box pushing (Fig. 53.15), *Parker* [53.10, 316] illustrates aspects of adaptive task allocation and learning, *Donald* et al. [53.317] (VIDEO 208) illustrates concepts of information invariance and the interchangeability of sensing, communication, and control, and *Simmons* et al. [53.11] demonstrate the feasibility of cooperative control for building planetary habitats.

In general, the manipulation techniques used for collective object transportation can be grouped into three primary methods [53.318]: pushing, grasping, and caging. The pushing approach [53.10, 11, 13, 316, 317] requires contact between each robot and the object, in order to impart force in the goal direction; however, the robots are not physically connected with the object. In the grasping approach [53.123, 142, 310–312, 319–322], each robot in the team physically attaches to the object being transported. See for example Fig. 53.16

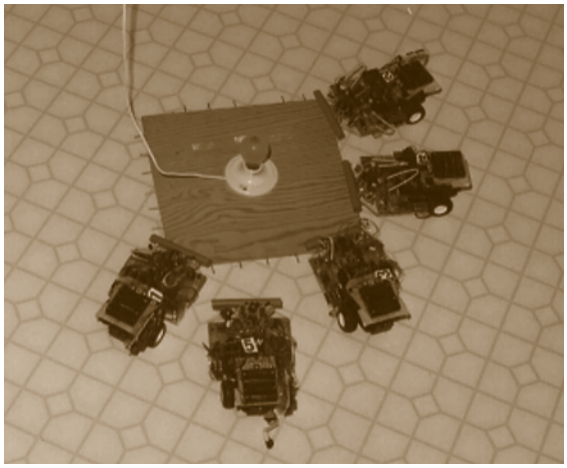


Fig. 53.15 Collective pushing of lighted box

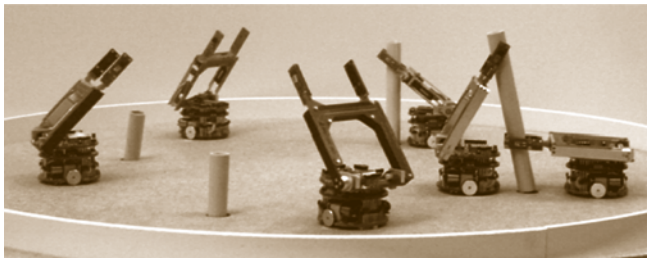


Fig. 53.16 Cooperative stick pulling

Fig. 53.17 Collective transport via caging

for cooperative stick pulling work of [53.320]. Finally, the caging approach [53.323–326] (VIDEO 292) involves robots encircling the object so that the object moves in the desired direction, even without the constant contact of all the robots with the object. See Fig. 53.17 for an example of collective transport via caging, from [53.323].

A closely related task is that of collective construction and wall building. The objective of the collective construction and wall building task is for robots to build structures of a specified form, in either 2-D or 3-D. This task is distinguished from self-reconfigurable robots, whose bodies themselves serve as the dynamic structure. *Werfel* and *Nagpal* have extensively explored this topic (VIDEO 216), developing distributed algorithms that enable simplified robots to build structures based on provided blueprints, both in 2-D [53.327–329] and in 3-D [53.330]. In their 3-D approach, the system consists of idealized mobile robots that perform the construction, and smart blocks that serve as the passive structure. The robots' job is to provide the mobility, while the blocks' role is to identify places in the growing structure at which an additional block can be placed that is on the path toward obtaining the desired final structure. The goal of their work is to be able to deploy some number of robots and free blocks into a construction zone, along with a single block that serves as a seed for the structure, and then have the construction to proceed autonomously according to the provided blueprint of the desired structure.

Hardware challenges of collective robot construction are addressed by *Terada* and *Murata* [53.331]. In this work, a hardware design is proposed that defines passive building blocks, along with an assembler robot that constructs structures with the robots. Other related work on the topic of collective construction includes the work of *Wawerla* et al. [53.332], in which robots use a behavior-based approach to build a linear wall using blocks equipped with either positive or negative Velcro, distinguished by block color. Their results show that adding 1 bit of state information to communicate the color of the last attached block provides a significant improvement in the collective performance. The work by *Stewart* and *Russell* [53.333, 334] proposes a distributed approach to building a loose wall structure with a robot swarm.

Another type of construction is called blind bulldozing, which is inspired by a behavior observed in certain ant colonies. Rather than constructing by accumulating materials, this approach achieves construction by removing materials. This task has practical application in site clearing, such as would be needed for planetary exploration [53.335]. Early ideas of this concept were discussed by *Brooks* et al. in [53.336], which argues for large numbers of small robots to be delivered to the lunar surface for site preparation. *Parker* et al. [53.337], further develop this idea by proposing robots using force sensors to clear an area by pushing material to the edges of the work site.

A significant body of additional research has been illustrated in this domain; representative examples include [53.3, 6, 123, 258, 284, 310, 323, 338–344].

53.10.4 Multitarget Observation

The domain of multitarget observation requires multiple robots to monitor and/or observe multiple targets moving through the environment. The objective is to maximize the amount of time, or the likelihood, that the targets remain in view by some team member throughout task execution. The task can be especially challenging if there are more targets than robots. This application domain can be useful for studying strongly cooperative task solutions, since robots have to coordinate their motions or the switching of targets to follow in order to maximize their objective. In the context of multiple mobile robot applications, the planar version of this test bed was first introduced

in [53.345] as cooperative multirobot observation of multiple moving targets (CMOMMT). Similar problems have been studied by several researchers, and extended to more complex problems such as environments with complex topography or three-dimensional versions for multiple aerial vehicle applications. This domain is also related to problems in other areas, such as art gallery algorithms, pursuit evasion, and sensor coverage. This domain has practical application in many security, surveillance, and reconnaissance problems. Research applied to the multitarget observation problem in multirobot systems includes [53.138, 253, 346–353].

53.10.5 Traffic Control and Multirobot Path Planning

When multiple robots are operating in a shared environment, they must coordinate their actions to prevent interference. These problems typically arise when the space in which robots operate contains bottlenecks, such as networks of roadways, or when the robots take up a relatively large portion of the navigable space. In these problems, the open space can be viewed as a resource that robots must share as efficiently as possible, avoiding collisions and deadlocks. In this domain, robots usually have their own individual goals, and must work with other robots to ensure that they receive use of the shared space to the extent needed to achieve their goals. In some variants, the entire paths of multiple robots need to be coordinated with each other; in other variants, robots must simply avoid interfering with each other.

A variety of techniques have been introduced to address this problem, including traffic rules, subdividing the environment into single-ownership sections, and geometric path planning ([53.354] for an overview). Many of the earliest research approaches to this problem were based on heuristic approaches, such as predefining motion control (or traffic) rules that were shown to prevent deadlock [53.355–358], or using techniques similar to mutual exclusion in distributed computing [53.359, 360]. These approaches have the benefit of minimizing the planning cost for obtaining a solution. Other, more formal, techniques view the application as a geometric multirobot path planning problem that can be solved precisely in configuration space–time. Chapter 7 includes a discussion of motion planning for multiple robots relevant to this domain. While geometric motion planning approaches provide the most general solutions, they can often be too computationally intensive for practical application, impractical due to the dynamic nature of the environment, or simply unnecessary for the problem at hand. In these cases, approximation approaches may be sufficient, such as centralized techniques that limit the search space through roadmapping [53.361, 362], and decoupled approaches that use either prioritized planning [53.363–365] (i. e., generating robot paths one by one) or path coordination (i. e., first planning individual paths for robots, then handling collision avoidance).

Fig. 53.18 Legged robot teams competing in robot soccer

53.10.6 Soccer

Since the inception of the RoboCup multirobot soccer domain as a proposed challenge problem for studying coordination and control in multirobot systems [53.366], research in this domain has grown tremendously. This domain incorporates many challenging aspects of multirobot control, including collaboration, robot control architectures, strategy acquisition, real-time reasoning and action, sensor fusion, dealing with adversarial environments, cognitive modeling, and learning. Annual competitions show the ever-improving team capabilities of the robots in a variety of settings, as shown in Fig. 53.18. A key aspect of this domain that is not present in the other multirobot test domains is that robots must operate in *adversarial* environments. This domain is also popular because of its educational benefits, as it brings together students and researchers from across the world in competitions to win the RoboCup challenges. The RoboCup competitions have added an additional search-and-rescue category to the competition [53.367], which has also become a significant area of research (Chap. 66 for more details on this field). Annual proceedings of the RoboCup competitions document much of the research that is incorporated into the multirobot soccer teams. Some representative research works include [53.368–372] (VIDEO 202, VIDEO 209).

53.11 Conclusions and Further Reading

This chapter has surveyed the current state of the art in multirobot systems, examining architectures, communications issues, swarm robot systems, heterogeneous teams, task allocation, learning, and applications. Clearly, significant advances have been made in the field in the last decade. The field is still an active area of research, however, since many open research issues still remain to be solved. Key open research questions remain in the broad areas of system integration, robustness, learning, scalability, generalization, and dealing with heterogeneity.

For example, in the area of system *integration*, an open question is how to effectively allow robot teams to combine a spectrum of approaches toward achieving complete systems that can perform more than a limited set of tasks. We are a long way from creating robust robot networks that can perform physical tasks in the real world. In the area of *robustness*, multirobot teams still need improvements in the ability to degrade gracefully, to reason for fault tolerance, and to achieve complexity without escalating failure rates. The area of *learning* in multirobot teams is still in its infancy, with open questions including how to achieve continual learning in multirobot teams, how to facilitate the use of complex representations, and how to enable humans to influence and/or understand the results of the team learning. *Scalability* is still a challenging problem, in terms of more complex environments as well as ever-larger numbers of robots. We do not yet have a methodology for creating self-organizing robot networks that are robust to labeling (or numbering), with completely decentralized controllers and estimators, and with provable emergent response. This requires basic research at the intersection of control, perception, and communication. Open issues in *generalization* include enabling the robot team to reason about context and increasing the versatility of systems so that they can operate in a variety of different applications. In dealing with *heterogeneity*, open questions include determining theoretical approaches to predicting system performance when all robots are not equal, and determining how to design a robot team optimally for a given application. Advances over the last decade have provided human users with the ability to interact with hundreds or thousands of computers on the Internet. It is necessary to develop similar network-centric approaches to interfacing, both for control and for monitoring. Finally, a major challenge is to create systems that are proactive and anticipate our needs and commands rather than reacting (with delays) to human commands.

For further reading on the topic of multiple mobile robot systems, the reader is referred to survey articles in the field, including [53.2, 288, 373, 374]. Additionally, several special journal issues on this topic have appeared, including [53.1, 375–377]. Some taxonomies of multirobot systems are given in [53.26, 288, 378]. A variety of symposia and workshops have been held on a regular basis on the topic of multirobot systems, in particular the DARS (distributed autonomous robotic systems) series of symposia. Recent proceedings of these workshops and symposia include [53.379–388]. Additional texts on this topic include [53.389–391]. For some excellent further background on networked robotics we direct the reader to [53.31, 35, 46, 52, 55, 75, 90, 104].

Video-References

VIDEO 192 Agents at play: Off-the-shelf software for practical multi-robot applications
available from http://handbookofrobotics.org/view-chapter/53/videodetails/192

VIDEO 193 Handling of a single object by multiple mobile robots based on caster-like dynamics
available from http://handbookofrobotics.org/view-chapter/53/videodetails/193

VIDEO 194 Synchronization and fault detection in autonomous robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/194

VIDEO 195 Self-assembly and morphology control in a swarm-bot
available from http://handbookofrobotics.org/view-chapter/53/videodetails/195

VIDEO 196 CKBOTS reconfigurable robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/196

VIDEO 197 Biologically inspired multi vehicles control algorithm
available from http://handbookofrobotics.org/view-chapter/53/videodetails/197

VIDEO 198 Metamorphic robotic system
available from http://handbookofrobotics.org/view-chapter/53/videodetails/198

VIDEO 199 Multi-robot box pushing
available from http://handbookofrobotics.org/view-chapter/53/videodetails/199

VIDEO 200 Elements of cooperative behavior in autonomous mobile robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/200
VIDEO 201 Coordination of multiple mobile platforms for manipulation and transportation
available from http://handbookofrobotics.org/view-chapter/53/videodetails/201
VIDEO 202 Robots in games and competition
available from http://handbookofrobotics.org/view-chapter/53/videodetails/202
VIDEO 203 A robotic reconnaissance and surveillance team
available from http://handbookofrobotics.org/view-chapter/53/videodetails/203
VIDEO 204 MARS (multiple autonomous robots)
available from http://handbookofrobotics.org/view-chapter/53/videodetails/204
VIDEO 205 A method for transporting a team of miniature robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/205
VIDEO 206 Reconfigurable multi-agents with distributed sensing for robust mobile robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/206
VIDEO 207 Miniature air vehicle cooperative timing missions
available from http://handbookofrobotics.org/view-chapter/53/videodetails/207
VIDEO 208 Distributed manipulation with mobile robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/208
VIDEO 209 Autonomous robot soccer – Through the wormhole with Morgan Freeman
available from http://handbookofrobotics.org/view-chapter/53/videodetails/209
VIDEO 210 A day in the life of a Kiva robot
available from http://handbookofrobotics.org/view-chapter/53/videodetails/210
VIDEO 211 Robot Pebbles – MIT developing self-sculpting smart sand robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/211
VIDEO 212 Transport of a child by swarm-bots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/212
VIDEO 213 Towards a swarm of nano quadrotors
available from http://handbookofrobotics.org/view-chapter/53/videodetails/213
VIDEO 214 Swarm robotics at CU-Boulder
available from http://handbookofrobotics.org/view-chapter/53/videodetails/214
VIDEO 215 Swarm robot system
available from http://handbookofrobotics.org/view-chapter/53/videodetails/215
VIDEO 216 Swarm construction robots
available from http://handbookofrobotics.org/view-chapter/53/videodetails/216
VIDEO 217 Multi robot formation control – Khepera team
available from http://handbookofrobotics.org/view-chapter/53/videodetails/217
VIDEO 292 Experiments of escorting a target
available from http://handbookofrobotics.org/view-chapter/53/videodetails/292
VIDEO 293 Formation control via a distributed controller-observer
available from http://handbookofrobotics.org/view-chapter/53/videodetails/293

References

53.1 T. Arai, E. Pagello, L.E. Parker: Editorial: Advances in multi-robot systems, IEEE Trans. Robotics Autom. **18**(5), 655–661 (2002)
53.2 Y. Cao, A. Fukunaga, A. Kahng: Cooperative mobile robotics: Antecedents and directions, Auton. Robots **4**, 1–23 (1997)
53.3 R.G. Brown, J.S. Jennings: A pusher/steerer model for strongly cooperative mobile robot manipulation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (1995) pp. 562–568
53.4 D. Milutinović, P. Lima: Modeling and optimal centralized control of a large-size robotic population, IEEE Trans. Robotics **22**(6), 1280–1285 (2006)
53.5 B. Khoshnevis, G.A. Bekey: Centralized sensing and control of multiple mobile robots, Comput. Ind. Eng. **35**(3/4), 503–506 (1998)
53.6 M.J. Matarić: Issues and approaches in the design of collective autonomous agents, Robotics Auton. Syst. **16**, 321–331 (1995)
53.7 E. Ostergaard, G.S. Sukhatme, M.J. Matarić: Emergent bucket brigading, 5th Int. Conf. Auton. Agents (2001)
53.8 M. Matarić: Reinforcement learning in the multi-robot domain, Auton. Robots **4**, 73–83 (1997)
53.9 L.E. Parker: ALLIANCE: An architecture for fault-tolerant multi-robot cooperation, IEEE Trans. Robotics Autom. **14**(2), 220–240 (1998)
53.10 L.E. Parker: Lifelong adaptation in heterogeneous teams: Response to continual variation in individual robot performance, Auton. Robots **8**(3), 239–267 (2000)
53.11 R. Simmons, S. Singh, D. Hershberger, J. Ramos, T. Smith: First results in the coordination of het-

erogeneous robots for large-scale assembly, Proc. ISER 7th Int. Symp. Exp. Robotics (2000)
53.12 J. Deneubourg, S. Goss, G. Sandini, F. Ferrari, P. Dario: Self-organizing collection and transport of objects in unpredictable environments, Jpn.-USA Symp. Flex. Autom. (1990) pp. 1093–1098
53.13 C.R. Kube, H. Zhang: Collective robotics: From social insects to robots, Adapt. Behav. **2**(2), 189–219 (1993)
53.14 R. Beckers, O. Holland, J. Deneubourg: From local actions to global tasks: Stigmergy and collective robotics, Proc. 14th Int. Workshop Synth. Simul. Living Syst. (1994) pp. 181–189
53.15 S. Onn, M. Tennenholtz: Determination of social laws for multi-agent mobilization, Artif. Intell. **95**, 155–167 (1997)
53.16 B.B. Werger: Cooperation without deliberation: A minimal behavior-based approach to multi-robot teams, Artif. Intell. **110**(2), 293–320 (1999)
53.17 M.J. Huber, E. Durfee: Deciding when to commit to action during observation-based coordination, Proc. 1st Int. Conf. Multi-Agent Syst. (1995) pp. 163–170
53.18 H. Asama, K. Ozaki, A. Matsumoto, Y. Ishida, I. Endo: Development of task assignment system using communication for multiple autonomous robots, J. Robotics Mechatron. **4**(2), 122–127 (1992)
53.19 N. Jennings: Controlling cooperative problem solving in industrial multi-agent systems using joint intentions, Artif. Intell. **75**(2), 195–240 (1995)
53.20 M. Tambe: Towards flexible teamwork, J. Artif. Intell. Res. **7**, 83–124 (1997)
53.21 R.T. Vaughan, K. Stoy, G.S. Sukhatme, M.J. Matarić: LOST: Localization-space trails for robot teams, IEEE Trans. Robotics Autom. **18**(5), 796–812 (2002)
53.22 L.E. Parker: The Effect of action recognition and robot awareness in cooperative robotic teams, IEEE/RSJ Int. Conf. Intell. Robots Syst. (1995) pp. 212–219
53.23 M. Matarić: Behavior-based control: Examples from navigation, learning, and group behavior, J. Exp. Theor. Artif. Intell. **19**(2/3), 323–336 (1997)
53.24 B. MacLennan, G.M. Burghardt: Synthetic ethology and the evolution of cooperative communication, Adapt. Behav. **2**, 161–188 (1993)
53.25 T. Balch, R.C. Arkin: Communiation in reactive multiagent robotic systems, Auton. Robots **1**(1), 27–52 (1995)
53.26 G. Dudek, M. Jenkin, E. Milios, D. Wilkes: A taxonomy for multi-agent robotics, Auton. Robots **3**, 375–397 (1996)
53.27 Z. Butler, K. Kotay, D.L. Rus, M. Vona: Self-reconfiguring robots, Commun. ACM **45**(3), 39–45 (2002)
53.28 O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal: Coordination and decentralized cooperation of multiple mobile manipulators, J. Robotic Syst. **13**(11), 755–764 (1996)
53.29 L.E. Parker: The effect of heterogeneity in teams of 100+ mobile robots. In: *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata*, ed. by A. Schultz, L.E. Parker, F. Schneider (Kluwer, Dordrecht 2003)
53.30 M. Yim, Y. Zhang, D. Duff: Modular robots, IEEE Spectrum **39**(22), 30–34 (2002)
53.31 D. Estrin: *Embedded, Everywhere* (National Academies Press, Washington 2001)
53.32 J. Parrish, S. Viscido, D. Grünbaum: Self-organized fish schools: An examination of emergent properties, Biol. Bull. **202**, 296–305 (2002)
53.33 N. Franks, S. Pratt, E. Mallon, N. Britton, D. Sumpter: Information flow, opinion polling and collective intelligence in house-hunting social insects, Philos. Trans. R. Soc. B **357**, 1567–1584 (2002)
53.34 R.L. Jeanne: Group size, productivity, and information flow in social wasps. In: *Information Processing in Social Insects*, ed. by C. Detrain, J.M. Pasteels, J.L. Deneubourg (Birkhauser, Basel 1999)
53.35 T. Akimoto, N. Hagita: Introduction to a network robot system, IEEE Int. Symp. Intell. Signal Proc. Comm. (2006)
53.36 Argo Floats: *A global array of 3000 Free-Drifting Profiling Floats for Environmental Monitoring* (Argo Information Center, Ramonville 2007)
53.37 G.S. Sukhatme, A. Dhariwal, B. Zhang, C. Oberg, B. Stauffer, D.A. Caron: The design and development of a wireless robotic networked aquatic microbial observing system, Environ. Eng. Sci. **24**(2), 205–215 (2006)
53.38 W. Kaiser, G. Pottie, M. Srivastava, G.S. Sukhatme, J. Villasenor, D. Estrin: Networked infomechanical systems (NIMS) for ambient intelligence. In: *Ambient Intelligence*, ed. by W. Weber, J.M. Rabaey, E. Aarts (Springer, Berlin, Heidelberg 2005)
53.39 Amarss: *Networked Minirhizotron Planning and Initial Deployment* (Center for Embedded Networkedsensing, Los Angeles 2007)
53.40 H. Singh, J. Catipovic, R. Eastwood, L. Freitag, H. Henriksen, F.F. Hover, D. Yoerger, J.G. Bellingham, B.A. Moran: An integrated approach to multiple AUV communications, navigation and docking, MTS/IEEE Conf. Proc. OCEANS (1996) pp. 59–64
53.41 I. Vasilescu, M. Dunbabin, P. Corke, K. Kotay, D. Rus: Data collection, storage, and retrieval with an underwater sensor network, Proc. ACM Sens. Syst. (2005)
53.42 N. Leonard, D. Paley, F. Lekien, R. Sepulchre, D.M. Fratantoni, R. Davis: Collective motion, sensor networks and ocean sampling, Proc. IEEE **95**(1), 48–74 (2006)
53.43 D.O. Popa, A.S. Sanderson, R.J. Komerska, S.S. Mupparapu, D.R. Blidberg, S.G. Chappell: Adaptive sampling algorithms for multiple autonomous underwater vehicles, IEEE/OES AUV2004: A Workshop Multiple Auton. Underw. Veh. Oper. (2004)
53.44 L. Chaimowicz, A. Cowley, B. Grocholsky, M.A. Hsieh, J.F. Keller, V. Kumar, C.J. Taylor: Deploying air-ground multi-robot teams in urban environments, 3rd Multi-Robot Syst. Workshop (2005)

53.45 B. Grocholsky, R. Swaminathan, J. Keller, V. Kumar, G. Pappas: Information driven coordinated air-ground proactive sensing, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

53.46 A. Howard, L.E. Parker, G.S. Sukhatme: Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection, Int. J. Robotics Res. **25**(5/6), 431–447 (2006)

53.47 D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, B. Stewart: Distributed multirobot exploration and mapping, Proc. IEEE **94**(7), 1325–1339 (2006)

53.48 M.A. Hsieh, A. Cowley, J.F. Keller, L. Chaimowicz, B. Grocholsky, V. Kumar, C.J. Talyor, Y. Endo, R. Arkin, B. Jung, D. Wolf, G. Sukhatme, D.C. MacKenzie: Adaptive teams of autonomous aerial and ground robots for situational awareness, J. Field Robotics **24**(11/12), 991–1014 (2007)

53.49 J. Seyfied, M. Szymanski, N. Bender, R. Estana, M. Theil, H. Worn: The I-swarm project: Intelligent small world autonomous robots for micromanipulation, SAB 2004 Int. Workshop (2004)

53.50 F. Mondada, G.C. Pettinaro, A. Guignard, I.W. Kwee, D. Floreano, J.-L. Deneubourg, S. Nofli, L.M. Gambardella, M. Dorigo: SwarmBot: A new distributed robotic concept, Auton. Robots **17**, 193–221 (2004)

53.51 A. Ollero, S. Lacroix, L. Merino, J. Gancet, J. Wiklund, V. Remuss, I. Veiga, L.G. Gutierrez, D.X. Viegas, M.A. Gonzalez, A. Mallet, R. Alami, R. Chatila, G. Hommel, F.J. Colmenero, B. Arrue, J. Ferruz, R. Martinez de Dios, F. Caballero: Architecture and perception issues in the COMETS multi-UAV project, IEEE Robotics Autom. Mag. **12**(2), 46–57 (2005)

53.52 Q. Li, D. Rus: Navigation protocols in sensor networks, ACM Trans. Sens. Netw. **1**(1), 3–35 (2005)

53.53 R. Bajcsy: Active perception, Proc. IEEE **76**, 996–1005 (1988)

53.54 T. Eren, D. Goldenberg, W. Whitley, Y.R. Yang, S. Morse, B.D.O. Anderson, P.N. Belhumeur: Rigidity, computation, and randomization of network localization, Proc. IEEE INFOCOM (2004)

53.55 A. Hsieh, A. Cowley, V. Kumar, C.J. Taylor: Towards the deployment of a mobile robot network with end-to-end performance guarantees, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

53.56 B. Grocholsky, S. Bayraktar, V. Kumar, C.J. Taylor, G. Pappas: Synergies in feature localization by air-ground teams, Proc. 9th Int. Symp. Exp. Robotics (2004)

53.57 B. Grocholsky, E. Stump, V. Kumar: An extensive representation for range-only SLAM, Int. Symp. Exp. Robotics (2006)

53.58 A. Howard, M.J. Matarić, G.S. Sukhatme: An incremental self-deployment algorithm for mobile sensor networks, Auton. Robots **13**(2), 113–126 (2002)

53.59 Z. Butler, D. Rus: Controlling mobile networks for monitoring events with coverage constraints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003)

53.60 M. Chu, J. Reich, F. Zhao: Distributed attention for large video sensor networks, Intell. Distrib. Surveill. Syst. Semin (2004)

53.61 B. Jung, G.S. Sukhatme: Tracking targets using multiple robots: The effect of environment occlusion, Auton. Robots **13**(3), 191–205 (2002)

53.62 R.M. Murray, K.J. Åström, S.P. Boyd, R.W. Brockett, G. Stein: Future directions in control in an information-rich world, IEEE Control Syst. Mag. (2003)

53.63 A. Pant, P. Seiler, K. Hedrick: Mesh stability of look-ahead interconnected systems, IEEE Trans. Autom. Control **47**, 403–407 (2002)

53.64 T. Sugar, J. Desai, V. Kumar, J.P. Ostrowski: Coordination of multiple mobile manipulators, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 3022–3027

53.65 R.W. Beard, J. Lawton, F.Y. Hadaegh: A coordination architecture for spacecraft formation control, IEEE Trans. Control Syst. Technol. **9**, 777–790 (2001)

53.66 A. Das, J. Spletzer, V. Kumar, C. Taylor: Ad hoc networks for localization and control, Proc. IEEE Conf. Decis. Control (2002) pp. 2978–2983

53.67 J. Manyika, H. Durrant-Whyte: *Data Fusion and Sensor Management: An Information-Theoretic Approach* (Prentice Hall, Upper Saddle River 1994)

53.68 H.G. Tanner, A. Jadbabaie, G.J. Pappas: Stable flocking of mobile agents, Part I: Fixed topology, Proc. IEEE Conf. Decis. Control (2003) pp. 2010–2015

53.69 J.M. Fowler, R. D'Andrea: Distributed control of close formation flight, Proc. IEEE Conf. Decis. Control (2002) pp. 2972–2977

53.70 J.P. Desai, J.P. Ostrowski, V. Kumar: Modeling and control of formations of nonholonomic mobile robots, IEEE Trans. Robotics Autom. **17**(6), 905–908 (2001)

53.71 H.G. Tanner, V. Kumar, G.J. Pappas: Leader-to-formation stability, IEEE Trans. Robotics Autom. **20**(3), 443–455 (2004)

53.72 S. Loizou, V. Kumar: Relaxed input to state stability properties for navigation function based systems, Proc. IEEE Conf. Decis. Control (2006)

53.73 M. Batalin, G.S. Sukhatme: Coverage, exploration and deployment by a mobile robot and communication network, Telecommun. Syst. J. **26**(2), 181–196 (2004)

53.74 K.J. O'hara, V. Bigio, S. Whitt, D. Walker, T.R. Balch: Evaluation of a large scale pervasive embedded network for robot path planning, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2072–2077

53.75 V. Kumar, N. Leonard, A.S. Morse (Eds.): *Cooperative control*, Lecture Notes in Control and Information Sciences, Vol. 309 (Springer, Berlin, Heidelberg 2004)

53.76 J. Chen, S. Teller, H. Balakrishnan: Pervasive pose-aware applications and infrastructure, IEEE Comput. Graph. Appl. **23**(4), 14–18 (2003)

53.77 A. Savvides, C.-C. Han, M. Srivastava: Dynamic fine-grained localization in ad-hoc networks of sensors, Proc. 7th Annu. Int. Conf. Mobile Comput. Netw. (MOBICOM-01) (2001) pp. 166–179

53.78 D. Moore, J. Leonard, D. Rus, S.J. Teller: Robust distributed network localization with noisy range measurements, Proc. 2nd Int. Conf. Embed. Netw. Sens. Syst. (SenSys) (2004) pp. 50–61

53.79 C. Detweiler, J. Leonard, D. Rus, S. Teller: Passive mobile robot localization within a fixed beacon field, Proc. Int. Workshop Algorithmic Found. Robotics (2006)

53.80 N. Bulusu, J. Heidemann, D. Estrin: Adaptive beacon placement, Proc. 21st Int. Conf. Distrib. Comput. Syst. (ICDCS-01) (2001) pp. 489–498

53.81 S.N. Simic, S. Sastry: Distributed Localization in Wireless ad hoc Networks, Tech. Rep. UCB/ERL M02/26 (2001), http://www.eecs.berkeley.edu/Pubs/TechRpts/2002/4010.html

53.82 P. Corke, R. Peterson, D. Rus: Communication-assisted localization and navigation for networked robots, Int. J. Robotics Res. **4**(9), 116 (2005)

53.83 R. Nagpal, H.E. Shrobe, J. Bachrach: Organizing a global coordinate system from local information on an ad hoc sensor network, Lect. Notes Comput. Sci. **2634**, 333–348 (2003)

53.84 R.K. Williams, A. Gasparri, A. Priolo, G.S. Sukhatme: Evaluating network rigidity in realistic systems: Decentralization, asynchronicity, and parallelization, IEEE Trans. Robotics **30**(4), 950–965 (2014)

53.85 A. Howard, M.J. Matarić, G.S. Sukhatme: Putting the 'i' in 'team': An ego-centric approach to cooperative localization, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 868–892

53.86 A. Howard, M.J. Matarić, G.S. Sukhatme: Localization for mobile robot teams using maximum likelihood estimation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2002) pp. 434–459

53.87 R. Vidal, O. Shakernia, H.J. Kim, D.H. Shim, S. Sastry: Probabilistic pursuit-evasion games: Theory, implementation and experimental evaluation, IEEE Trans. Robotics Autom. **18**(5), 662–669 (2002)

53.88 M. Batalin, M.H. Rahimi, Y. Yu, D. Liu, A. Kansal, G.S. Sukhatme, W. Kaiser, M. Hansen, G. Pottie, M. Srivastava, D. Estrin: Call and response: Experiments in sampling the environment, Proc. 2nd Int. Conf. Embed. Netw. Sens. Syst. (SenSys) (2004) pp. 25–38

53.89 M.H. Rahimi, W. Kaiser, G.S. Sukhatme, D. Estrin: Adaptive sampling for environmental field estimation using robotic sensors, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2005) pp. 747–753

53.90 F. Zhao, L. Guibas: *Wireless Sensor Networks: An Information Processing Approach* (Morgan Kaufmann, New York 2004)

53.91 F. Zhang, B. Grocholsky, V. Kumar: Formations for localization of robot networks, IEEE Int. Conf. Robotics Autom. (2004)

53.92 R.K. Williams, G.S. Sukhatme: Constrained interaction and coordination in proximity-limited multiagent systems, IEEE Trans. Robotics **29**(4), 930–944 (2013)

53.93 J. Cortes, S. Martinez, T. Karatas, F. Bullo: Coverage control for mobile sensing networks, IEEE Trans. Robotics Autom. **20**(2), 243–255 (2004)

53.94 J. Cortes, S. Martinez, F. Bullo: Spatially-distributed coverage optimization and control with limited-range interactions, ESAIM Control Optim. Calc. Var. **11**, 691–719 (2005)

53.95 M. Schwager, J. McLurkin, D. Rus: Distributed coverage control with sensory feedback for networked robots, Proc. Robotics Sci. Syst. (RSS) (2006)

53.96 M. Schwager, J.-J. Slotine, D. Rus: Decentralized adaptive control for coverage for networked robots, Proc. Int. Conf. Robotics Autom. (2007)

53.97 A. Krause, C. Guestrin, A. Gupta, J. Kleinberg: Near-optimal sensor placements: Maximizing information while minimizing communication cost, 5th Int. Conf. Inf. Process Sens. Netw. (IPSN) (2006)

53.98 V. Chvatal: A combinatorial theorem in plane geometry, J. Comb. Theory Ser. **18**, 39–41 (1975)

53.99 J. O'Rourke: *Art Gallery Theorems and Algorithms* (Oxford Univ. Press, New York 1987)

53.100 S. Fisk: A short proof of Chvatal's watchmen theorem, J. Comb. Theory Ser. **24**, 374 (1978)

53.101 A. Jadbabaie, J. Lin, A.S. Morse: Coordination of groups of mobile autonomous agents using nearest neighbor rules, IEEE Trans. Autom. Control **48**(6), 988–1001 (2003)

53.102 S. Sinha, M. Pollefeys: Camera network calibration from dynamic silhouettes, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2004)

53.103 R. Collins, A. Lipton, H. Fujiyoshi, T. Kanade: Algorithms for cooperative multisensor surveillance, Proc. IEEE **89**(10), 1456–1477 (2001)

53.104 A. Kansal, W. Kaiser, G. Pottie, M. Srivastava, G.S. Sukhatme: Reconfiguration methods for mobile sensor networks, ACM Trans. Sens. Networks **3**(4), 1–28 (2007)

53.105 D. Payton, M. Daily, R. Estkowski, M. Howard, C. Lee: Pheromone robotics, Auton. Robots **11**, 319–324 (2001)

53.106 A. Howard, M.J. Matarić, G.S. Sukhatme: Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem, Proc. Int. Symp. Distrib. Auton. Robotic Syst. (2002) pp. 299–308

53.107 S. Poduri, G.S. Sukhatme: Constrained coverage for mobile sensor networks, IEEE Int. Conf. Robotics Autom. (2004) pp. 165–172

53.108 R. Wattenhofer, L. Li, P. Bahl, Y.M. Wang: A cone-based distributed topology-control algorithm for wireless multi-hop networks, IEEE/ACM Trans. Netw. **13**(1), 147–159 (2005)

53.109 H. Zhang, J.C. Hou: On deriving the upper bound of alphalifetime for large sensor networks, ACM Trans. Sens. Netw. **1**(6), 272–300 (2005)

53.110 S. Poduri, S. Pattem, B. Krishnamachari, G.S. Sukhatme: Using local geometry for tunable topology control in sensor networks, IEEE Trans. Mobile Comput. **8**(2), 218–230 (2009)

53.111 M.A. Hsieh, V. Kumar: Pattern generation with multiple robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006)

53.112 R.N. Smith, M. Schwager, S.L. Smith, B.H. Jones, D. Rus, G.S. Sukhatme: Persistent ocean mon-

itoring with underwater gliders: Adapting spatiotemporal sampling resolution, J. Field Robotics **28**(5), 714–741 (2011)
53.113 J. Das, F. Py, T. Maughan, T. O'Reilly, M. Messié, J. Ryan, G.S. Sukhatme, K. Rajan: Coordinated sampling of dynamic oceanographic features with AUVs and drifters, Int. J. Robotics Res. **31**(5), 626–646 (2012)
53.114 G.A. Hollinger, S. Choudhary, P. Qarabaqi, C. Murphy, U. Mitra, G.S. Sukhatme, M. Stojanovic, H. Singh, F. Hover: Underwater data collection using robotic sensor networks, IEEE J. Sel. Areas Commun. **30**(5), 899–911 (2012)
53.115 F. Ye, G. Zhong, J. Cheng, L. Zhang, S. Lu: Peas: A robust energy conserving protocol for long-lived sensor networks, Int. Conf. Distrib. Comput. Syst. (2003)
53.116 X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, C. Gill: Integrated coverage and connectivity configuration in wireless sensor networks, 1st ACM Conf. Embed. Netw. Sens. Syst. (SenSys) (2003)
53.117 D. Tian, N.D. Georganas: Connectivity maintenance and coverage preservation in wireless sensor networks, Ad Hoc Networks **3**(6), 744–761 (2005)
53.118 G. Theraulaz, S. Goss, J. Gervet, J.-L. Deneubourg: Task differentiation in Polistes wasp colonies: A model for self-organizing groups of robots, Proc. 1st Int. Conf. Simul. Adapt. Behav. (1990) pp. 346–355
53.119 L. Steels: Cooperation Between Distributed Agents Through Self-Organization, Proc. IEEE Int. Workshop Intell. Robots Syst. (IROS) (1990)
53.120 A. Drogoul, J. Ferber: From tom thumb to the dockers: Some experiments with foraging robots, Proc. 2nd Int. Conf. Simul. Adapt. Behav. (1992) pp. 451–459
53.121 M.J. Matarić: Designing emergent behaviors: From local interactions to collective intelligence, Proc. 2nd Int. Conf. Simul. Adapt. Behav. (1992) pp. 432–441
53.122 G. Beni, J. Wang: Swarm intelligence in cellular robotics systems, Proc. NATO Adv. Workshop Robots Biol. Syst. (1989)
53.123 D. Stilwell, J. Bay: Toward the development of a material transport system using swarms of ant-like robots, Proc. IEEE Int. Conf. Robotics Autom. (1993) pp. 766–771
53.124 T. Fukuda, S. Nakagawa, Y. Kawauchi, M. Buss: Self organizing robots based on cell structures – CEBOT, Proc. IEEE Int. Workshop Intell. Robots Syst. (1988) pp. 145–150
53.125 J.H. Reif, H.Y. Wang: Social Potential fields: A distributed behavior control for autonomous robots, Robotics Auton. Syst. **27**(3), 171–194 (1999)
53.126 L.E. Parker: Designing control laws for cooperative agent teams, Proc. IEEE Int. Cont. Robotics Autom. (1993) pp. 582–587
53.127 K. Sugihara, I. Suzuki: Distributed algorithms for formation of goemetric patterns with many mobile robots, J. Robotic Syst. **13**(3), 127–139 (1996)
53.128 V. Gazi: Swarm aggregations using artificial potentials and sliding-mode control, IEEE Trans. Robotics **21**(6), 1208–1214 (2005)
53.129 J. McLurkin, J. Smith: Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots, Symp. Distrib. Auton. Robots Syst. (2004)
53.130 D. Gage: Randomized search strategies with imperfect sensors, Proc. SPIE Mobile Robots VIII (1993) pp. 270–279
53.131 T. Balch: The impact of diversity on performance in robot foraging, Proc. 3rd Ann. Conf. Auton. Agents (1999) pp. 92–99
53.132 Z.J. Butler, A.A. Rizzi, R.L. Hollis: Cooperative coverage of rectilinear environments, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000)
53.133 A.I. Mourikis, S.I. Roumeliotis: Performance analysis of multirobot cooperative localization, IEEE Trans. Robotics **22**(4), 666–681 (2006)
53.134 R. Grabowski, L.E. Navarro-Serment, C.J. Paredis, P.K. Khosla: Heterogeneous teams of modular robots for mapping and exploration, Auton. Robots **8**(3), 271–298 (2000)
53.135 M. Berhault, H. Huang, P. Keskinocak, S. Koenig, W. Elmaghraby, P. Griffin, A. Kleywegt: Robot exploration with combinatorial auctions, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 1957–1962
53.136 J. Kim, J.M. Esposito, V. Kumar: An RRT-based algorithm for testing and validating mulit-robot controllers, Proc. Robotics Sci. Syst. I (2005)
53.137 Z. Cao, M. Tin, L. Li, N. Gu, S. Wang: Cooperative hunting by distributed mobile robots based on local interaction, IEEE Trans. Robotics **22**(2), 403–407 (2006)
53.138 R.W. Beard, T.W. McLain, M. Goodrich: Coordinated target assignment and intercept for unmanned air vehicles, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 2581–2586
53.139 J. Clark, R. Fierro: Cooperative hybrid control of robotic sensors for perimeter detection and tracking, Proc. Am. Control Conf. (2005) pp. 3500–3505
53.140 P. Stone, M. Veloso: A layered approach to learning client behaviors in the RoboCup soccer server, Appl. Artif. Intell. **12**, 165–188 (1998)
53.141 J. McLurkin: Stupid Robot Tricks: Behavior-Based Distributed Algorithm Library for Programming Swarms of Robots, Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 2004)
53.142 F. Mondada, L.M. Gambardella, D. Floreano, S. Nolfi, J.L. Deneuborg, M. Dorigo: The cooperation of swarm-bots: Physical interactions in collective robotics, IEEE Robotics Autom. Mag. **12**(2), 21–28 (2005)
53.143 T.W. Mather, M.A. Hsieh: Macroscopic modeling of stochastic deployment policies with time delays for robot ensembles, Int. J. Robotics Res. **30**(5), 590–600 (2011)
53.144 X.C. Ding, M. Kloetzer, Y. Chen, C. Belta: Automatic deployment of robot teams, IEEE Robotics Autom. Mag. **18**(3), 75–86 (2022)

53.145 F.S. Melo, M. Veloso: Decentralized MDPs with sparse interactions, Artif. Intell. **175**(11), 1757–1789 (2011)

53.146 P. Tsiotras, L.I.R. Castro: Extended multi-agent consensus protocols for the generation of geometric patterns in the plane, Am. Controls Conf. (2011) pp. 3850–3855

53.147 K. Cheng, P. Dasgupta: Weighted voting game based multi-robot team formation for distributed area coverage, Proc. 3rd Int. Symp. Pract. Cognit. Agents Robots (2010) pp. 9–15

53.148 A.A. Taheri, M. Afshar, M. Asadpour: Influence maximization for informed agents in collective behavior, Distrib. Auton. Robotic Syst. (2013) pp. 389–402

53.149 M.A. Hsieh, A. Halász, E.D. Cubuk, S. Schoenholz, A. Martinoli: Specialization as an optimal strategy under varying external conditions, Proc. IEEE Int. Conf. Robotics Autom. (2009) pp. 1941–1946

53.150 N. Hoff, R. Wood, R. Nagpal: Distributed colony-level algorithm switching for robot swarm foraging, Distrib. Auton. Robotic Syst. (2013) pp. 417–430

53.151 A. Winfield, J. Nembrini: Safety in numbers: Fault-tolerance in robot systems, Int. J. Model. Identif. Control **1**(1), 30–37 (2006)

53.152 T. Fukuda, S. Nakagawa: Dynamically reconfigurable robotic system, IEEE Int. Conf. Robotics Autom. (1988) pp. 1581–1586

53.153 M. Yim, W.-M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, G.S. Chirikjian: Modular self-reconfigurable robot systems: Challenges and opportunities for the future, IEEE Robotics Autom. Mag. **14**(1), 43–52 (2007)

53.154 K. Gilpin, D. Rus: Modular robot systems: From self-assembly to self-disassembly, IEEE Robotics Autom. Mag. **17**(3), 38–53 (2010)

53.155 M. Yim: A reconfigurable modular robot with many modes of locomotion, JSME Int. Conf. Adv. Mechatron. (1993) pp. 283–288

53.156 M. Yim: New locomotion gaits, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 2508–2514

53.157 A. Castano, P. Will: Mechanical design of a module for reconfigurable robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 2203–2209

53.158 W.-M. Shen, P. Will: Docking in self-reconfigurable robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 1049–1054

53.159 A. Castano, A. Behar, P. Will: The conro modules for reconfigurable robots, IEEE Trans. Mechatron. **7**(4), 403–409 (2002)

53.160 S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, S. Kokaji: M-TRAN: Self-reconfigurable modular robotic system, IEEE/ASME Trans. Mechatron. **7**(4), 431–441 (2002)

53.161 A. Kamimura, H. Kurokawa, E. Yoshida, S. Murata, K. Tomita, S. Kokaji: Automatic locomotion design and experiments for a modular robotic system, IEEE/ASME Trans. Mechatron. **10**(3), 314–325 (2005)

53.162 H. Kurokawa, K. Tomita, A. Kamimura, E. Yoshida, S. Kokahji, S. Murata: Distributed self-reconfiguration control of modular robot M-TRAN, IEEE Int. Conf. Mechatron. Autom. (2005) pp. 254–259

53.163 S. Murata, K. Kakomura, H. Kurokawa: Docking experiments of a modular robot by visual feedback, Proc. IEEE/RSJ Int. Conf. Intell. Robots Systems (IROS) (2006) pp. 625–630

53.164 D. Marbach, A.J. Ijspeert: Online optimization of modular robot locomotion, IEEE Int. Conf. Mechatron. Autom. (2005) pp. 248–253

53.165 E.H. Østergaard, H.H. Lund: Evolving control for modular robotic units, IEEE Int. Symp. Comput. Intell. Robotics Autom. (2003) pp. 886–892

53.166 M.W. Jørgensen, E.H. Østergaard, H.H. Lund: Modular ATRON: Modules for a self-reconfigurable robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 2068–2073

53.167 B. Salemi, M. Moll, W.-M. Shen: SUPERBOT: A deployable, multi-functional, and modular self-reconfigurable robotic system, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 3636–3641

53.168 M. Yim, D.G. Duff, K.D. Roufas: PolyBot: A modular reconfigurable robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 514–520

53.169 M. Yim, Y. Zhang, K. Roufas, D. Duff, C. Eldershaw: Connecting and disconnecting for self–reconfiguration with PolyBot, IEEE/ASME Trans. Mechatron. **7**(4), 442–451 (2003)

53.170 M. Yim, B. Shirmohammadi, J. Sastra, M. Park, M. Dugan, C.J. Taylor: Towards robotic self-reassembly after explosion, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2007) pp. 2767–2772

53.171 V. Zykov, E. Mytilinaios, M. Desnoyer, H. Lipson: Evolved and designed self-reproducing modular robotics, IEEE Trans. Robotics **23**(2), 308–319 (2007)

53.172 P.J. White, M.L. Posner, M. Yim: Strength analysis of miniature folded right angle tetrahedron chain programmable matter, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 2785–2790

53.173 G.S. Chirikjian: Kinematics of a metamorphic robotic system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 449–455

53.174 G. Chirikjian, A. Pamecha, I. Ebert-Uphoff: Evaluating efficiency of self-reconfiguration in a class of modular robots, J. Robotic Syst. **13**(5), 317–388 (1996)

53.175 A. Pamecha, I. Ebert-Uphoff, G.S. Chirikjian: Useful metrics for modular robot motion planning, IEEE Trans. Robotics Autom. **13**(4), 531–545 (1997)

53.176 J.E. Walter, E.M. Tsai, N.M. Amato: Algorithms for fast concurrent reconfiguration of hexagonal metamorphic robots, IEEE Trans. Robotics **21**(4), 621–631 (2005)

53.177 S. Murata, H. Kurokawa, S. Kokaji: Self-assembling machine, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 441–448

53.178 E. Yoshida, S. Murata, K. Tomita, H. Kurokawa, S. Kokaji: Distributed formation control for a modular mechanical system, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 1090–1097

53.179 H. Kurokawa, S. Murata, E. Yoshida, K. Tomita, S. Kokaji: A 3-D self-reconfigurable structure and experiments, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998) pp. 860–865

53.180 E. Yoshida, S. Murata, S. Kokaji, A. Kamimura, K. Tomita, H. Kurokawa: Get back in shape! A hardware prototype self-reconfigurable modular microrobot that uses shape memory alloy, IEEE Robotics Autom. Mag. **9**(4), 54–60 (2002)

53.181 J. Gargus, B. Kim, I. Llamas, J. Rossignac, C. Shaw: Finger Sculpting with Digital Clay, Tech. Rep. GIT-GVU-02-22 (2002)

53.182 K. Kotay, D. Rus, M. Vona, C. McGray: The self-reconfiguring robotic molecule, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 424–431

53.183 K. Kotay, D. Rus: Motion synthesis for the self-reconfiguring robotic molecule, IEEE Int. Conf. Intell. Robots Syst. (1998) pp. 843–851

53.184 K. Kotay, D. Rus: Algorithms for self-reconfiguring molecule motion planning, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000)

53.185 Z.J. Butler, K. Kotay, D. Rus, K. Tomita: Generic decentralized control for lattice-based self-reconfigurable robots, Int. J. Robotics Res. **23**(9), 919–937 (2004)

53.186 D. Rus, M. Vona: A basis for self-reconfiguring robots using crystal modules, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2000) pp. 2194–2202

53.187 D. Rus, M. Vona: Crystalline robots: Self-reconfiguration with compressible unit modules, Int. J. Robotics Res. **22**(9), 699–715 (2003)

53.188 J.W. Suh, S.B. Homans, M. Yim: Telecubes: Mechanical design of a module for self-reconfigurable robotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 4095–4101

53.189 C.-J. Chiang, G.S. Chirikjian: Modular robot motion planning using similarity metrics, Auton. Robots **10**, 91–106 (2001)

53.190 M. Koseki, K. Minami, N. Inou: Cellular robots forming a mechanical structure (evaluation of structural formation and hardware design of CHOBIE II), Proc. 7th Int. Symp. Distrib. Auton. Robotic Syst. (DARS) (2004) pp. 131–140

53.191 B.K. An: Em-cube: Cube-shaped, self-reconfigurable robots sliding on structure surfaces, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 3149–3155

53.192 C. Ünsal, P.K. Khosla: Mechatronic design of a modular self-reconfiguring robotic system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 1742–1747

53.193 K.C. Prevas, C. Ünsal, M.Ö. Efe, P.K. Khosla: A hierarchical motion planning strategy for a uniform self-reconfigurable modular robotic system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 787–792

53.194 K. Hosokawa, T. Tsujimori, T. Fujii, H. Kaetsu, H. Asama, Y. Kuroda, I. Endo: Self-organizing collective robots with morphogenesis in a vertical plane, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 2858–2863

53.195 S. Goldstein, J. Campbell, T. Mowry: Programmable Matter, IEEE Computer **38**(6), 99–101 (2005)

53.196 S.C. Goldstein, J.D. Campbell, T.C. Mowry: Programmable matter, Computer **38**(6), 99–101 (2005)

53.197 M.E. Karagozler, S.C. Goldstein, J.R. Reid: Stress-driven MEMS assembly + electrostatic forces = 1 mm diameter robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 2763–2769

53.198 P. Pillai, J. Campbell, G. Kedia, S. Moudgal, K. Sheth: A 3-D fax machine based on claytronics, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 4728–4735

53.199 P. White, K. Kopanski, H. Lipson: Stochastic self-reconfigurable cellular robotics, IEEE Conf. Robotics Autom. (2004) pp. 2888–2893

53.200 P. White, V. Zykov, J. Bongard, H. Lipson: Three dimensional stochastic reconfiguration of modular robots, Robotics Sci. Syst. (2005)

53.201 M. Tolley, J. Hiller, H. Lipson: Evolutionary design and assembly planning for stochastic modular robots, Proc. IEEE/RSJ. Int. Conf. Intell. Robotics Syst. (IROS) (2009) pp. 73–78

53.202 M. Tolley, H. Lipson: Fluidic manipulation for scalable stochastic 3-D assembly of modular robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 2473–2478

53.203 M.T. Tolley, H. Lipson: Programmable 3-D stochastic fluidic assembly of cm-scale modules, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 4366–4371

53.204 K. Gilpin, K. Kotay, D. Rus, I. Vasilescu: Miche: Modular shape formation by self-disassembly, Int. J. Robotics Res. **27**, 345–372 (2008)

53.205 R. Oung, F. Bourgault, M. Donovan, R. D'Andrea: The distributed flight array, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

53.206 G.J. Hamlin, A.C. Sanderson: Tetrobot: A modular system for hyper-redundant parallel robotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 154–159

53.207 A. Lyder, R.F.M. Garcia, K. Stoy: Mechanical design of Odin, an extendable heterogeneous deformable modular robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 883–888

53.208 A. Lyder, H.G. Peterson, K. Stoy: Representation and shape estimation of Odin, a parallel under-actuated modular robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 5275–5280

53.209 C.-H. Yu, K. Haller, D. Ingber, R. Nagpal: Morpho: A self-deformable modular robot inspired by cellar structure, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 3571–3578

53.210 M. Shimizu, A. Ishiguro, T. Kawakatsu: A modular robot that exploits a spontaneous connectivity control mechanism, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005) pp. 1899–1904

53.211 M. Shimizu, T. Mori, A. Ishiguro: A Development of a modular robot that enables adaptive reconfiguration, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 174–179

53.212 S. Funiak, P. Pillai, M.P. Ashley-Rollman, J.D. Campbell, S.C. Goldstein: Distributed localization of modular robot ensembles, Int. J. Robotics Res. **28**(8), 946–961 (2009)

53.213 M. Rubenstein, W.-M. Shen: Scalable self-assembly and self-repair in a collective of robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 1484–1489

53.214 M. Rubenstein, W.-M. Shen: Automatic scalable size selection for the shape of a distributed robotic collective, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 508–513

53.215 M. Rubenstein, C. Ahler, R. Nagpal: Kilobot: A low cost scalable robot system for collective behaviors, IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

53.216 E. Hawkes, B. An, N.M. Benbernou, H. Tanaka, S. Kim, E.D. Demaine, D. Rus, R.J. Wood: Programmable matter by folding, Proc. Natl. Acad. Sci. USA **107**(28), 12441–12445 (2010)

53.217 B. An, D. Rus: Programming and controlling self-folding sheets, IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

53.218 G. Whitesides, B. Grzybowski: Self-assembly at all scales, Sci. USA **295**, 2418–2421 (2002)

53.219 G.M. Whitesides, M. Boncheva: Beyond molecules: Self-assembly of mesoscope and macroscopic components, Proc. Natl. Acad. Sciences **99**(8), 4769–4774 (2002)

53.220 D.H. Garcias, J. Tien, T.L. Breen, C. Hsu, G.M. Whitesides: Forming electrical networks in three dimensions by self-assembly, Science **289**(5482), 1170–1172 (2000)

53.221 S. Miyashita, M. Kessler, M. Lungarella: How morphology affects self-assembly in a stochastic modular robot, IEEE Int. Conf. Robotics Autom. (2008) pp. 3533–3538

53.222 K. Hosokawa, I. Shimoyama, H. Miura: Dynamics of self-assembling systems: Analogy with chemical kinematics, Artif. Life **1**(4), 413–427 (1994)

53.223 M. Shimizu, K. Suzuki: A Self-repairing structure for modules and its control by vibrating actuation mechanisms, IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 4281–4286

53.224 P.W.K. Rothemund, E. Winfree: The program-size complexity of self-assembled squares, 32rd Annu. ACM Symp. Theory Comput. (2000) pp. 459–468

53.225 L. Adleman, Q. Cheng, A. Goel, M.-D. Huang: Running time and program size for self-assembled squares, 33rd Annu. ACM Symp. Theory Comput. (2001) pp. 740–748

53.226 G. Aggarwal, M.H. Goldwasser, M.-Y. Kao, R.T. Schweller: Complexities for generalized models of self-assembly, 15th Annu. ACM-SIAM Symp. Discrete Algorithms (2004) pp. 880–889

53.227 J. Bishop, S. Burden, E. Klavins, R. Kreisberg, W. Malone, N. Napp, T. Nguyen: Programmable parts: A demonstration of the grammatical approach to self-organization, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005) pp. 3684–3691

53.228 S. Griffith, D. Goldwater, J.M. Jacobson: Robotics: Self-replication from random parts, Nature **437**, 636 (2005)

53.229 C. Jones, M.J. Matarić: From local to global behavior in intelligent self-assembly, IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 721–726

53.230 J. Kelly, H. Zhang: Combinatorial optimization of sensing for rule-based planar distributed assembly, IEEE Int. Conf. Intell. Robots Syst. (2006) pp. 3728–3734

53.231 J. Werfel: Anthills Built to Order: Automating Construction with Artificial Swarms, Ph.D. Thesis (MIT, Cambridge 2006)

53.232 B. Donald, C.G. Levey, C.D. McGray, I. Paprotny, D. Rus: An untethered, electrostatic, globally controllable MEMS micro-robot, J. Microelectromech. Syst. **15**(1), 1–15 (2006)

53.233 B.R. Donald, C.G. Levey, I. Paprotny: Planar microassembly by parallel actuator of MEMS microrobots, J. Microelectromech. Syst. **17**(4), 789–808 (2008)

53.234 C. Pawashe, S. Floyd, M. Sitti: Assembly and disassembly of magnetic mobile micro-robots towards 2-D reconfigurable micro-systems, Int. Symp. Robotics Res. (2009)

53.235 N. Napp, S. Burden, E. Klavins: The statistical dynamics of programmed self-assembly, IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1469–1476

53.236 K. Gilpin, D. Rus: A distributed algorithm for 2-D shape duplication with smart pebble robots, IEEE Int. Conf. Robotics Autom. (ICRA) (2012)

53.237 K. Gilpin, A. Knaian, D. Rus: Robot pebbles: One centimeter robotic modules for programmable matter through self-disassembly, IEEE Int. Conf. Robotics Autom. (ICRA) (2010)

53.238 L.E. Parker: The effect of heterogeneity in teams of 100+ mobile robots. In: *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata*, ed. by A. Schultz, L.E. Parker, F. Schneider (Kluwer, Dordrecht 2003)

53.239 T. Balch: Hierarchic social entropy: An information theoretic measure of robot team diversity, Auton. Robots **8**(3), 209–238 (2000)

53.240 D. Jung, A. Zelinsky: Grounded symbolic communication between heterogeneous cooperating robots, Auton. Robots **8**(3), 269–292 (2000)

53.241 R.R. Murphy: Marsupial robots for urban search and rescue, IEEE Intell. Syst. **15**(2), 14–19 (2000)

53.242 G. Sukhatme, J.F. Montgomery, R.T. Vaughan: Experiments with cooperative aerial-ground robots. In: *Robot Teams: From Diversity to Polymorphism*, ed. by T. Balch, L.E. Parker (A K Peters, Natick 2002)

53.243 L.E. Parker, B. Kannan, F. Tang, M. Bailey: Tightly-coupled navigation assistance in heterogeneous multi-robot teams, Proc. IEEE Int. Conf. Intell. Robots Syst. (2004)

53.244 A. Howard, L.E. Parker, G.S. Sukhatme: Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment, and detection, Int. J. Robotics Res. **25**, 431–447 (2006)

53.245 L. Chaimowicz, B. Grocholsky, J.F. Keller, V. Kumar, C.J. Taylor: Experiments in multirobot air-ground coordination, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

53.246 L.E. Parker, F. Tang: Building multi-robot coalitions through automated task solution synthesis, Proc. IEEE **94**(7), 1289–1305 (2006)

53.247 R. Zlot, A. Stentz: Market-based multirobot coordination for complex tasks, Int. J. Robotics Res. **25**(1), 73–101 (2006)

53.248 J. Jennings, C. Kirkwood-Watts: Distributed mobile robotics by the method of dynamic teams. In: *Distributed Autonomous Robotic Systems 3*, ed. by T. Lueth, R. Dillmann, P. Dario, H. Wörn (Springer, Berlin, Heidelberg 1998)

53.249 E. Pagello, A. D'Angelo, E. Menegatti: Cooperation issues and distributed sensing for multirobot systems, Proc. IEEE **94**, 1370–1383 (2006)

53.250 B. Gerkey, M.J. Matarić: A formal analysis and taxonomy of task allocation in multi-robot systems, Int. J. Robotics Res. **23**(9), 939–954 (2004)

53.251 D. Gale: *The Theory of Linear Economic Models* (McGraw-Hill, New York 1960)

53.252 E. Balas, M.W. Padberg: On the set-covering problem, Oper. Res. **20**(6), 1152–1161 (1972)

53.253 B.B. Werger, M.J. Matarić: Broadcast of local eligibility for multi-target observation. In: *Distributed Autonomous Robotic Systems 4*, ed. by L.E. Parker, G. Bekey, J. Barhen (Springer, Tokyo 2000)

53.254 R.A. Brooks: A robust layered control system for a mobile robot, IEEE J. Robotics Autom. **RA-2**(1), 14–23 (1986)

53.255 S. Botelho, R. Alami: M+: A scheme for multirobot cooperation through negotiated task allocation and achievement, Proc. IEEE Int. Conf. Robotics Autom. (1999) pp. 1234–1239

53.256 R.G. Smith: The contract net protocol: High-level communication and control in a distributed problem solver, IEEE Trans. Comput. **C-29**(12), 1104–1113 (1980)

53.257 B. Dias, R. Zlot, N. Kalra, A. Stentz: Market-based multirobot coordination: A survey and analysis, Proc. IEEE **94**(7), 1257–1270 (2006)

53.258 B.P. Gerkey, M.J. Matarić: Sold! Auction methods for multi-robot coordination, IEEE Trans. Robotics Autom. **18**(5), 758–768 (2002)

53.259 H. Kose, U. Tatlidede, C. Mericli, K. Kaplan, H.L. Akin: Q-learning based market-driven multi-agent collaboration in robot soccer, Proc. Turk. Symp. Artif. Intell. Neural Netw. (2004) pp. 219–228

53.260 D. Vail, M. Veloso: Multi-robot dynamic role assignment and coordination through shared potential fields, multi-robot systems: From swarms to intelligent automata, Proc. Int. Workshop Multi-Robot Syst. (2003) pp. 87–98

53.261 M. Lagoudakis, E. Markakis, D. Kempe, P. Keshinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, S. Jain: *Auction-based multi-robot routing*, *Robotics: Science and Systems I* (MIT Press, Cambridge 2005)

53.262 G. Rabideau, T. Estlin, S. Schien, A. Barrett: A comparison of coordinated planning methods for cooperating rovers, Proc. AIAA Space Technol. Conf. (1999)

53.263 R. Zlot, A. Stentz, M.B. Dias, S. Thayer: Multi-robot exploration controlled by a market economy, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 3016–3023

53.264 J. Guerrero, G. Oliver: Multi-robot task allocation strategies using auction-like mechanisms, Proc. 6th Congr. Catalan Assoc. Artif. Intell. (2003) pp. 111–122

53.265 N. Kalra, D. Ferguson, A. Stentz: Hoplites: A market-based framework for planned tight coordination in multirobot teams, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005)

53.266 L. Lin, Z. Zheng: Combinatorial bids based multi-robot task allocation method, Proc. IEEE Int. Conf. Robotics. Autom. (ICRA) (2005) pp. 1145–1150

53.267 C.-H. Fua, S.S. Ge: COBOS: Cooperative backoff adaptive scheme for multirobot task allocation, IEEE Trans. Robotics **21**(6), 1168–1178 (2005)

53.268 E.G. Jones, B. Browning, M.B. Dias, B. Argall, M. Veloso, A. Stentz: Dynamically formed heterogeneous robot teams performing tightly-coupled tasks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 570–575

53.269 Y. Zhang, L.E. Parker: IQ-ASyMTRe: Forming executable coalitions for tightly coupled multirobot tasks, IEEE Trans. Robotics **29**(2), 400–416 (2012)

53.270 L. Vig, J.A. Adams: Multi-robot coalition formation, IEEE Trans. Robotics **22**(4), 637–649 (2006)

53.271 M. Bowling, M. Veloso: Simultaneous adversarial multi-robot learning, Proc. Int. Joint Conf. Artif. Intell. (2003)

53.272 F. Fernandez, L.E. Parker: A reinforcement learning algorithm in cooperative multi-robot domains, J. Intell. Robots Syst. **43**, 161–174 (2005)

53.273 C.F. Touzet: Robot awareness in cooperative mobile robot learning, Auton. Robots **2**, 1–13 (2000)

53.274 R. Steeb, S. Cammarata, F. Hayes-Roth, P. Thorndyke, R. Wesson: *Distributed Intelligence for Air Fleet Control*, Rand Corp. Tech. Rep. R-2728-AFPA (1981)

53.275 M. Benda, V. Jagannathan, R. Dodhiawalla: On Optimal Cooperation of Knowledge Sources, Boeing AI Center Tech. Rep. BCS-G2010-28 (1985)

53.276 T. Haynes, S. Sen: Evolving behavioral strategies in predators and prey. In: *Adaptation and Learning in Multi-Agent Systems*, ed. by G. Weiss, S. Sen (Springer, Berlin, Heidelberg 1986) pp. 113–126

53.277 S. Mahadevan, J. Connell: Automatic programming of behavior-based robots using reinforcement learning, Proc. AAAI (1991) pp. 8–14

53.278 S. Marsella, J. Adibi, Y. Al-Onaizan, G. Kaminka, I. Muslea, M. Tambe: On being a teammate: Experiences acquired in the design of RoboCup teams, Proc. 3rd Annu. Conf. Auton. Agents (1999) pp. 221–227

53.279 J. Pugh, A. Martinoli: Multi-robot learning with particle swarm optimization, Proc. 5th Int. Jt.

Conf. Auton. Agents Multiagent Syst. (2006) pp. 441–448

53.280 P. Stone, M. Veloso: Multiagent systems: A survey from a machine learning perspective, Auton. Robots **8**(3), 345–383 (2000)

53.281 M. Asada, E. Uchibe, S. Noda, S. Tawaratsumida, K. Hosoda: Coordination of multiple behaviors acquired by a vision-based reinforcement learning, Proc. IEEE/RSJ/GI Int. Conf. Intell. Robots Syst. (1994) pp. 917–924

53.282 M. Kubo, Y. Kakazu: Learning coordinated motions in a competition for food between ant colonies, Proc. 3rd Int. Conf. Simul. Adapt. Behav. (1994) pp. 487–492

53.283 R. Alami, S. Fleury, M. Herrb, F. Ingrand, F. Robert: Multi-robot cooperation in the MARTHA project, IEEE Robotics Autom. Mag. **5**(1), 36–47 (1998)

53.284 A. Stroupe, A. Okon, M. Robinson, T. Huntsberger, H. Aghazarian, E. Baumgartner: Sustainable cooperative robotic technologies for human and robotic outpost infrastructure construction and maintenance, Auton. Robots **20**(2), 113–123 (2006)

53.285 H.R. Everett, R.T. Laird, D.M. Carroll, G.A. Gilbreath, T.A. Heath-Pastore, R.S. Inderieden, T. Tran, K.J. Grant, D.M. Jaffee: *Multiple resource host architecture (MRHA) for the mobile detection assessment response system* (MDARS), SPAWAR Systems Technical Documen 3026, Revision A (2000)

53.286 Y. Guo, L.E. Parker, R. Madhavan: Towards collaborative robots for infrastructure security applications, Proc. Int. Symp. Collab. Technol. Syst. (2004) pp. 235–240

53.287 C. Hazard, P.R. Wurman, R. D'Andrea: Alphabet Soup: A testbed for studying resource allocation in multi-vehicle systems, Proc. AAAI Workshop Auction Mech. Robot Coord. (2006) pp. 23–30

53.288 D. Nardi, A. Farinelli, L. Iocchi: Multirobot systems: A classification focused on coordination, IEEE Trans. Syst. Man Cybern. B **34**(5), 2015–2028 (2004)

53.289 K. Passino: Biomimicry of bacterial foraging for distributed optimization and control, IEEE Control Syst. Mag. **22**(3), 52–67 (2002)

53.290 M. Schneider-Fontan, M. Matarić: Territorial multi-robot task division, IEEE Trans. Robotics Autom. **15**(5), 815–822 (1998)

53.291 I. Wagner, M. Lindenbaum, A.M. Bruckstein: Mac vs. PC – Determinism and randomness as complementary approaches to robotic exploration of continuous unknown domains, Int. J. Robotics Res. **19**(1), 12–31 (2000)

53.292 K. Sugawara, M. Sano: Cooperative behavior of interacting simple robots in a clockface arranged foraging field. In: *Distributed Autonomous Robotic Systems*, ed. by H. Asama, T. Arai, T. Fukuda, T. Hasegawa (Springer, Berlin, Heidelberg 2002)

53.293 P. Rybski, S. Stoeter, C. Wyman, M. Gini: A cooperative multi-robot approach to the mapping and exploration of Mars, Proc. AAAI/IAAI (1997)

53.294 S. Sun, D. Lee, K. Sim: Artificial immune-based swarm behaviors of distributed autonomous robotic systems, Proc. IEEE Int. Conf. Robotics Autom. (2001) pp. 3993–3998

53.295 A.I. Mourikis, S.I. Roumeliotis: Optimal sensor scheduling for resource-constrained localization of mobile robot formations, IEEE Trans. Robotics **22**(5), 917–931 (2006)

53.296 S. Kloder, S. Hutchinson: Path planning for permutation-invariant multirobot formations, IEEE Trans. Robotics **22**(4), 650–665 (2006)

53.297 C.W. Reynolds: Flocks, herds and schools: A distributed behavioral model, ACM SIGGRAPH Comput. Gr. **21**, 25–34 (1987)

53.298 T. Balch, R. Arkin: Behavior-based formation control for multi-robot teams, IEEE Trans. Robotics Autom. **14**(6), 926–939 (1998)

53.299 A. Jadbabaie, J. Lin, A.S. Morse: Coordination of groups of mobile autonomous agents using nearest neighbor rules, IEEE Trans. Autom. Control **48**(6), 988–1001 (2002)

53.300 C. Belta, V. Kumar: Abstraction and control for groups of robots, IEEE Trans. Robotics **20**(5), 865–875 (2004)

53.301 C.M. Topaz, A.L. Bertozzi: Swarming patterns in two-dimensional kinematic model for biological groups, SIAM J. Appl. Math. **65**(1), 152–174 (2004)

53.302 J.A. Fax, R.M. Murray: Information flow and cooperative control of vehicle formations, IEEE Trans. Autom. Control **49**(9), 1465–1476 (2004)

53.303 J.A. Marshall, M.E. Broucke, B.R. Francis: Formations of vehicles in cyclic pursuit, IEEE Trans. Autom. Control **49**(11), 1963–1974 (2004)

53.304 S.S. Ge, C.-H. Fua: Queues and artificial potential trenches for multirobot formations, IEEE Trans. Robotics **21**(4), 646–656 (2005)

53.305 P. Tabuada, G. Pappas, P. Lima: Motion feasibility of multi-agent formations, IEEE Trans. Robotics **21**(3), 387–392 (2005)

53.306 G. Antonelli, S. Chiaverini: Kinematic control of platoons of autonomous vehicles, IEEE Trans. Robotics **22**(6), 1285–1292 (2006)

53.307 J. Fredslund, M.J. Matarić: A general algorithm for robot formations using local sensing and minimal communication, IEEE Trans. Robotics Autom. **18**(5), 837–846 (2002)

53.308 Y. Cao, W. Yu, W. Ren, G. Chen: An overview of recent progress in the study of distributed multi-agent coordination, IEEE Trans. Ind. Inf. **9**(1), 427–438 (2012)

53.309 R.M. Murray: Recent research in cooperative control of multivehicle systems, J. Dyn. Syst. Meas. Control **129**(5), 571–583 (2007)

53.310 P.J. Johnson, J.S. Bay: Distributed control of simulated autonomous mobile robot collectives in payload transportation, Auton. Robots **2**(1), 43–63 (1995)

53.311 Z. Wang, E. Nakano, T. Matsukawa: Realizing cooperative object manipulation using multiple behaviour-based robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1996) pp. 310–317

53.312 K. Kosuge, T. Oosumi: Decentralized control of multiple robots handling an object, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1996) pp. 318–323

53.313 N. Miyata, J. Ota, Y. Aiyama, J. Sasaki, T. Arai: Cooperative transport system with regrasping car-like mobile robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1997) pp. 1754–1761

53.314 C.R. Kube, E. Bonabeau: Cooperative transport by ants and robots, Robotics Auton. Syst. **30**(1), 85–101 (2000)

53.315 R. Groß, M. Dorigo: Towards group transport by swarms of robots, Int. J. Bio-Inspired Comput. **1**(1), 1–13 (2009)

53.316 L.E. Parker: ALLIANCE: An architecture for fault tolerant, cooperative control of heterogeneous mobile robots, Proc. IEEE/RSJ/GI Int. Conf. Intell. Robots Syst. (1994) pp. 776–783

53.317 B. Donald, J. Jennings, D. Rus: Analyzing teams of cooperating mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (1994) pp. 1896–1903

53.318 Y. Mohan, S.G. Ponnambalam: An extensive review of research in swarm robotics, World Congr. Nat. Biol. Insp. Comput. (2009) pp. 140–145

53.319 A.J. Ijspeert, A. Martinoli, A. Billard, L.M. Gambardella: Collaboration through the exploitation of local interactions in autonomous collective robotics: The stick pulling experiment, Auton. Robots **11**(2), 149–171 (2001)

53.320 A. Martinoli, K. Easton, W. Agassounon: Modeling swarm robotic systems: A case study in collaborative distributed manipulation, Int. J. Robotics Res. **23**(4–5), 415–436 (2004)

53.321 S. Berman, Q. Lindsey, M.S. Sakar, V. Kumar, S.C. Pratt: Experimental study and modeling of group retrieval in ants as an approach to collective transport in swarm robotic systems, Proc. IEEE **99**(9), 1470–1481 (2011)

53.322 J.M. Esposito: Distributed grasp synthesis for swarm manipulation with applications to autonomous tugboats, IEEE Int. Conf. Robotics Autom. (2008) pp. 1489–1494

53.323 Z. Wang, V. Kumar: Object closure and manipulation by multiple cooperating mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 394–399

53.324 Z. Wang, Y. Hirata, K. Kosuge: Control a rigid caging formation for cooperative object transportation by multiple mobile robots, Proc. of IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 1580–1585

53.325 J. Fink, N. Michael, V. Kumar: Composition of vector fields for multi-robot manipulation via caging, Robotics Sci. Syst. Conf. (2007)

53.326 F. Arrichiello, H.K. Heidarsson, S. Chiaverini, G.S. Sukhatme: Cooperative caging and transport using autonomous aquatic surface vehicles, Intell. Serv. Robotics **5**(1), 73–87 (2012)

53.327 J. Werfel: Building patterned structures with robot swarms, Proc. 19th Int. Jt. Conf. Artif. Intell. (IJCAI) (2005) pp. 1495–1502

53.328 J. Werfel, Y. Bar-Yam, D. Rus, R. Nagpal: Distributed construction by mobile robots with enhanced building blocks, IEEE Int. Conf. Robotics Autom. (2006) pp. 2787–2794

53.329 J. Werfel, R. Nagpal: Extended stigmergy in collective construction, IEEE Intell. Syst. **21**(2), 20–28 (2006)

53.330 J. Werfel, R. Nagpal: Three-dimensional construction with mobile robots and modular blocks, Int. J. Robotics Res. **27**(3/4), 463–479 (2008)

53.331 Y. Terada, S. Murata: Automatic modular assembly system and its distributed control, Int. J. Robotics Res. **27**, 445–462 (2008)

53.332 J. Wawerla, G.S. Sukhatme, M.J. Mataric: Collective construction with multiple robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002) pp. 2696–2701

53.333 R.L. Stewart, R.A. Russell: Building a loose wall structure with a robotic swarm using a spatio-temporal varying template, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 712–716

53.334 R.L. Stewart, R.A. Russell: A distributed feedback mechanism to regulate wall construction by a robotic swarm, Adapt. Behav. **14**(1), 21–51 (2006)

53.335 T. Huntsberger, G. Rodriguez, P. Schenker: Robotics challenges for robotic and human Mars exploration, Proc. Robotics 2000 (2000) pp. 340–346

53.336 R.A. Brooks, P. Maes, M.J. Mataric, G. More: Lunar based construction robots, Proc. IEEE Int. Workshop Intell. Robots Syst. (IROS) (1990) pp. 389–392

53.337 C.A.C. Parker, H. Zhang, C.R. Kube: Blind bulldozing: Multiple robot nest construction, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2003) pp. 2010–2015

53.338 S. Sen, M. Sekaran, J. Hale: Learning to coordinate without sharing information, Proc. AAAI (1994) pp. 426–431

53.339 B. Tung, L. Kleinrock: Distributed control methods, Proc. 2nd Int. Symp. High Perform. Distrib. Comput. (1993) pp. 206–215

53.340 Z.-D. Wang, E. Nakano, T. Matsukawa: Cooperating multiple behavior-based robots for object manipulation, Proc. IEEE/RSJ/GI Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1524–1531

53.341 D. Rus, B. Donald, J. Jennings: Moving furniture with teams of autonomous robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (1995) pp. 235–242

53.342 F. Hara, Y. Yasui, T. Aritake: A kinematic analysis of locomotive cooperation for two mobile robots along a general wavy road, Proc. IEEE Int. Conf. Robotics Autom. (1995) pp. 1197–1204

53.343 J. Sasaki, J. Ota, E. Yoshida, D. Kurabayashi, T. Arai: Cooperating grasping of a large object by multiple mobile robots, Proc. IEEE Int. Conf. Robotics Autom. (1995) pp. 1205–1210

53.344 C. Jones, M.J. Matarić: Automatic synthesis of communication-based coordinated multi-robot systems, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 381–387

53.345 L.E. Parker: Cooperative robotics for multi-target observation, Intell. Autom. Soft Comput. **5**(1), 5–19 (1999)

53.346 A.W. Stroupe, M.C. Martin, T. Balch: Distributed sensor fusion for object position estimation by multi-robot systems, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 1092–1098

53.347 S. Luke, K. Sullivan, L. Panait, G. Balan: Tunably decentralized algorithms for cooperative target observation, Proc. 4th Int. Jt. Conf. Auton. Agents Multiagent Syst. (2005) pp. 911–917

53.348 S.M. LaValle, H.H. Gonzalez-Banos, C. Becker, J.-C. Latombe: Motion strategies for maintaining visibility of a moving target, Proc. IEEE Int. Conf. Robotics Autom. (1997) pp. 731–736

53.349 T.H. Chung, J.W. Burdick, R.M. Murray: A decentralized motion coordination strategy for dynamic target tracking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2416–2422

53.350 A. Kolling, S. Carpin: Multirobot cooperation for surveillance of multiple moving targets – A new behavioral approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1311–1316

53.351 B. Jung, G. Sukhatme: Tracking targets using multiple mobile robots: The effect of environment occlusion, Auton. Robots **13**(3), 191–205 (2002)

53.352 Z. Tang, U. Ozguner: Motion planning for multitarget surveillance with mobile sensor agents, IEEE Trans. Robotics **21**(5), 898–908 (2005)

53.353 M.A. Vieira, R. Govindan, G.S. Sukhatme: Scalable and practical pursuit-evasion with networked robots, Intell. Serv. Robotics **2**(4), 247–263 (2009)

53.354 L.E. Parker: Path planning and motion coordination in multiple mobile robot teams. In: *Encyclopedia of Complexity and System Science*, ed. by R.A. Meyers (Springer, Berlin, Heidelberg 2009)

53.355 D. Grossman: Traffic control of multiple robot vehicles, IEEE J. Robotics Autom. **4**, 491–497 (1988)

53.356 P. Caloud, W. Choi, J.-C. Latombe, C. Le Pape, M. Yim: Indoor automation with many mobile robots, IEEE Int. Workshop Intell. Robots Syst. (IROS) (1990) pp. 67–72

53.357 H. Asama, K. Ozaki, H. Itakura, A. Matsumoto, Y. Ishida, I. Endo: Collision avoidance among multiple mobile robots based on rules and communication, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1991) pp. 1215–1220

53.358 S. Yuta, S. Premvuti: Coordinating autonomous and centralized decision making to achieve cooperative behaviors between multiple mobile robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (1992) pp. 1566–1574

53.359 J. Wang: Fully distributed traffic control strategies for many-AGV systems, IEEE Int. Workshop Intell. Robots Syst. (1991) pp. 1199–1204

53.360 J. Wang, G. Beni: Distributed computing problems in cellular robotic systems, Proc. IEEE Int. Workshop Intell. Robots Syst. (1990) pp. 819–826

53.361 P. Svestka, M. Overmars: Coordinated path planning for multiple robots, Robotics Auton. Syst. **23**, 125–152 (1998)

53.362 M. Peasgood, C. Clark, J. McPhee: A complete and scalable strategy for coordinating multiple robots within roadmaps, IEEE Trans. Robotics **24**(2), 283–292 (2008)

53.363 M. Erdmann, T. Lozano-Perez: On multiple moving objects, Algorithmica **2**, 477–521 (1987)

53.364 C. Ferrari, E. Pagello, J. Ota, T. Arai: Multirobot motion coordination in space and time, Robotics Auton. Syst. **25**, 219–229 (1998)

53.365 M. Bennewitz, W. Burgard, S. Thrun: Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobiel robots, Robotics Auton. Syst. **41**(2), 89–99 (2002)

53.366 H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, H. Matasubara: RoboCup: A challenge problem of AI, AI Magazine **18**(1), 73–86 (1997)

53.367 H. Kitano, S. Tadokoro: RoboCup rescue: A grand challenge for multiagent and intelligent systems, AI Magazine **22**(1), 39–52 (2001)

53.368 B. Browning, J. Bruce, M. Bowling, M. Veloso: STP: Skills, tactics and plays for multi-robot control in adversarial environments, IEEE J. Control Syst. Eng. **219**, 33–52 (2005)

53.369 M. Veloso, P. Stone, K. Han: The CMUnited-97 robotic soccer team: Perception and multiagent control, Robotics Auton. Syst. **29**(2/3), 133–143 (1999)

53.370 T. Weigel, J.-S. Gutmann, M. Dietl, A. Kleiner, B. Nebel: CS Freiburg: Coordinating robots for successful soccer playing, IEEE Trans. Robotics Autom. **5**(18), 685–699 (2002)

53.371 P. Stone, M. Veloso: Task decomposition, dynamic role assignemnt, and low-bandwidth communicaiton for real-time strategic teamwork, Artif. Intell. **110**(2), 241–273 (1999)

53.372 C. Candea, H.S. Hu, L. Iocchi, D. Nardi, M. Piaggio: Coordination in multi-agent Robocup teams, Robotics Auton. Syst. **36**(2), 67–86 (2001)

53.373 L.E. Parker: Current state of the art in distributed autonomous mobile robotics, Distrib. Auton. Robotic Syst. **4**, 3–12 (2000)

53.374 K.R. Baghaei, A. Agah: Task allocation and communication methodologies for multi-robot systems, Intell. Autom. Soft Comput. **9**, 217–226 (2003)

53.375 T. Balch, L.E. Parker: Guest editorial, special issue on heterogeneous multi-robot systems, Auton. Robots **8**(3), 207–208 (2000)

53.376 M. Dorigo, E. Sahin: Guest editorial, special issue on swarm robotics, Auton. Robots **17**(2/3), 111–113 (2004)

53.377 M. Veloso, D. Nardi: Special issue on multirobot systems, Proc. IEEE **94**, 1253–1256 (2006)

53.378 T. Balch: Taxonomies of multi-robot task and reward. In: *Robot Teams: From Diversity to Polymorphism*, ed. by T. Balch, L.E. Parker (A K Peters, Natick 2002)

53.379 L.E. Parker, G. Bekey, J. Barhen (Eds.): *Distributed Autonomous Robotic Systems 4* (Springer, Berlin, Heidelberg 2000)

53.380 H. Asama, T. Arai, T. Fukuda, T. Hasegawa (Eds.): *Distributed Autonomous Robotic Systems 5* (Springer, Berlin, Heidelberg 2002)

53.381 R. Alami, R. Chatila, H. Asama (Eds.): *Distributed Autonomous Robotic Systems 6* (Springer, Berlin, Heidelberg 2006)

53.382 M. Gini, R. Voyles (Eds.): *Distributed Autonomous Robotic Systems 7* (Springer, Berlin, Heidelberg 2006)

53.383 H. Asama, H. Kurokawa, J. Ota, K. Sekiyama: *Distributed Autonomous Robotic Systems 8* (Springer, Berlin, Heidelberg 2009)

53.384 A. Martinoli, F. Mondada, N. Correll, G. Mermoud, M. Egerstedt, M.A. Hsieh, L.E. Parker, K. Stoy (Eds.): *Distributed Autonomous Robotic Systems*, Springer Tracts in Advanced Robotics, Vol. 83 (Springer, Berlin, Heidelberg 2013)

53.385 A. Schultz, L.E. Parker (Eds.): *Multi-Robot Systems: From Swarms to Intelligent Automata* (Kluwer, Dordrecht 2002)

53.386 A. Schultz, L.E. Parker, F. Schneider (Eds.): *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata* (Kluwer, Dordrecht 2003)

53.387 E. Sahin, W.M. Spears (Eds.): *Swarm Robotics: SAB 2004 Int. Workshop* (Springer, Berlin, Heidelberg 2004)

53.388 L.E. Parker, F. Schneider, A. Schultz (Eds.): *Multi-Robot Systems Volume III: From Swarms to Intelligent Automata* (Kluwer, Dordrecht 2005)

53.389 T. Balch, L.E. Parker (Eds.): *Robot Teams: From Polymorphism to Diversity* (A K Peters, Natick 2002)

53.390 F. Bullo, J. Cortés, S. Martinez: *Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms* (Princeton University Press, Princeton 2009)

53.391 S. Kernbach (Ed.): *Handbook of Collective Robotics: Fundamentals and Challenges* (CRC, Boca Raton 2013)

Multimedia Contents

Part F Robots

Part F Robots at Work

Ed. by Alex Zelinsky

54 Industrial Robotics
Martin Hägele, Stuttgart, Germany
Klas Nilsson, Lund, Sweden
J. Norberto Pires, Coimbra, Portugal
Rainer Bischoff, Augsburg, Germany

55 Space Robotics
Kazuya Yoshida, Sendai, Japan
Brian Wilcox, Pasadena, USA
Gerd Hirzinger, Wessling, Germany
Roberto Lampariello, Wessling, Germany

56 Robotics in Agriculture and Forestry
Marcel Bergerman, Pittsburgh, USA
John Billingsley, Toowoomba, Australia
John Reid, Moline, USA
Eldert van Henten, Wageningen, The Netherlands

57 Robotics in Construction
Kamel S. Saidi, Gaitherbsurg, USA
Thomas Bock, Munich, Germany
Christos Georgoulas, Munich, Germany

58 Robotics in Hazardous Applications
James Trevelyan, Crawley, Australia
William R. Hamel, Knoxville, USA
Sung-Chul Kang, Seoul, Korea

59 Robotics in Mining
Joshua A. Marshall, Kingston, Canada
Adrian Bonchis, Pullenvale, Australia
Eduardo Nebot, Sydney, Australia
Steven Scheding, Sydney, Australia

60 Disaster Robotics
Robin R. Murphy, College Station, USA
Satoshi Tadokoro, Sendai, Japan
Alexander Kleiner, Linköping, Sweden

61 Robot Surveillance and Security
Wendell H. Chun, Denver, USA
Nikolaos Papanikolopoulos, Minneapolis, USA

62 Intelligent Vehicles
Alberto Broggi, Parma, Italy
Alex Zelinsky, Canberra, Australia
Ümit Özgüner, Columbus, USA
Christian Laugier, Saint Ismier, France

63 Medical Robotics and Computer-Integrated Surgery
Russell H. Taylor, Baltimore, USA
Arianna Menciassi, Pisa, Italy
Gabor Fichtinger, Kingston, Canada
Paolo Fiorini, Verona, Italy
Paolo Dario, Pisa, Italy

64 Rehabilitation and Health Care Robotics
H.F. Machiel Van der Loos, Vancouver, Canada
David J. Reinkensmeyer, Irvine, USA
Eugenio Guglielmelli, Rome, Italy

65 Domestic Robotics
Erwin Prassler, Sankt Augustin, Germany
Mario E. Munich, Pasadena, USA
Paolo Pirjanian, Bedford, USA
Kazuhiro Kosuge, Sendai, Japan

66 Robotics Competitions and Challenges
Daniele Nardi, Rome, Italy
Jonathan Roberts, Brisbane, Australia
Manuela Veloso, Pittsburgh, USA
Luke Fletcher, Brisbane, Australia

Part F, Robots at Work, covers the advances in technology that are concerned with the growing area of robot applications, ranging from factory robotics, through a diverse array of industry applications such mining, agriculture, construction to health care and domestic robotics. The future vision for robotics is for the pervasive application of robots. The robots of the future will perform all the dangerous, dirty, and dreary (DDD) tasks. Joe Engelberger, the pioneer of the robotics industry, wrote in his 1989 book *Robotics in Service* that the inspiration to write the book came as a reaction to a forecast study of robot applications, which predicted that in 1995 applications of robotics outside factories (the traditional domain of industrial robots) would account for less than 1% of total sales. Engelberger believed that this forecast was wrong, and he instead predicted that the non-industrial class of robot applications would become the largest class of robot applications. Engelberger's prediction has yet to come to pass. However, he did correctly foresee the growth in non-traditional applications of robots. Previous parts of this Handbook show the great strides that robotics technology has made in the past 50 years. The technology has reached a level of maturity such that robots are now marching from the factories into field and service applications.

The topics in Part F cover the essentials of what is required to create robots that can operate in all environments and perform meaningful work. Part F describes fit-for-purpose robots and includes hardware design, control (of locomotion, manipulation, and interaction), perception, and user interfaces. The economic/social drivers for the particular applications are also discussed. Part F builds on all of the previous parts of the Handbook. Robotics Foundations (Part A) and Design (Part B) are essential for providing the basic mechanisms and control structures for any robot that is targeted for applications work. Sensing and Perception (Part C) and Manipulation and Interfaces (Part D) are critical capabilities for robots that need to interact with changing environments and perform manipulation tasks under human supervision. To fully utilize the potential of a working robot requires mobility and the ability to cooperate with other robotics, the basic technologies described in Moving in the Environment (Part E) are essential.

Chapter 54, Industrial Robotics, gives a history and description of typical industrial robot applications. Most robots today can trace their origin to early industrial robot designs. All the important foundations for robot control were initially developed with industrial applications in mind. The chapter describes how robot with different mechanisms are designed to fit different applications.

Chapter 55, Space Robots and Systems, deals with robotics technology for space applications. It covers two classes of robotics: orbital systems and operation of planetary surfaces. Any unmanned spacecraft is a robotic spacecraft. However, space robots are more capable devices that can facilitate manipulation, assembly, assisting astronauts, and extending exploration to remote planets as surrogates for human explorers. This chapter covers the key issues in space robots and systems: manipulation, mobility, sensing and perception, tele-operation and autonomy, and dealing with extreme environments. A historical overview and a review of the most recent technical advances for space robots is provided. An extensive description of the mathematics for the control of robotic devices in microgravity environments is also provided.

Chapter 56, Robotics in Agriculture and Forestry, describes how robotics has made a impact in the agricultural and forestry industries. Agriculture is an industry that is driven by productivity. Growing human population drives ever-increasing demands for greater food production. Robotics is a productivity technology that can be game changer in agriculture. The chapter focuses on case studies where robotics has made a marked difference to agriculture and forestry. The case studies cover the use of robotics in field crops, weeding, seeding, irrigation, orchard crops – fruit and vegetables, forestry, and livestock applications – breeding, harvesting, slaughter and processing. The case studies follow the progression of robotics technology from sensing through mobility to manipulation. The chapter emphasizes the agricultural field as a fruitful source for robotics applications, that are sufficiently demanding to require the development of new techniques, methods and technologies.

Chapter 57, Robotics in Construction, describes the construction automation concepts that have been developed and presents examples of construction robots that are in use and are in various stages of development. The chapter gives an overview of the construction industry and discusses the concept of automation versus robotics. The chapter discusses robotics in applications that are on construction sites. The chapter deals with the growing number robotics and automation application of offsite construction tasks such as prefabrication. Unsolved technical challenges including interoperability, connection systems, tolerances, power, and communications are discussed.

Chapter 58, Robotics in Hazardous Applications, discusses robot technologies for dealing with difficult and dangerous environments. The technology solutions that are adopted depend on the nature and magnitude of

the hazards. When the magnitudes of hazards reach the point that human exposure represents a direct threat to life or has long-term health consequences, e.g., nuclear radiation, some form of remote operation must be used. This chapter describes robotics technologies for dealing with eradicating landmines, and hazardous materials handling, such as explosives disposal, and handling of dangerous biological and nuclear material. Enabling technologies to support demining and hazardous material handling are discussed, including mobile platforms, manipulator design, teleoperation and control, and the supply of energy and communication signals through reliable and robust tethers. The chapter ends with a discussion of the advances that are required to progress the field. Progress is needed in mechatronics design, sensing, machine intelligence, and fully understanding the application and its associated economic factors.

Chapter 59, Mining Robotics, discusses the application of robotics to the mining industry. Mining remains physically demanding and hazardous work. The scope for the application of robotics is great since mining requires the handling of enormous quantities of material in a cost-effective and safe manner. High operational costs, the need for greater productivity, and improved health and safety outcomes are powerful drivers for robotics. Robotics and automation is providing the next step change in productivity and safety. This chapter reviews the modern mining practice and the technology drivers. It describes the enabling robot technologies associated with surface mining and underground mining. The chapter covers the technologies associated with digging, drilling, explosives handling, haulage, truck fleet management, autonomous mapping of mines and robotic rescue for recover in mining accidents. The chapter considers the challenges associated with mining in extreme environments such as undersea extraction of resources and the long-term possibilities of interplanetary exploration and mining.

Chapter 60, Disaster Robotics, describes the state of the art of an emerging applications field. In recent times robots have served as extensions of responders to a disaster, providing real-time video and other sensory data about the situation. The technology is still emerging and is beginning to be adopted by the international emergency response community. Robots are being used by fire departments and for bomb disposal. This chapter describes the basic nature of disasters, the impact on robot design, the types of robots actually used in disasters such as the Fukushima Nuclear disaster, promising designs, future concepts and benchmarks for evaluation of disaster robotics. The chapter discusses the fundamental challenges facing disaster robotics, including the key challenge of evolution from a concept to the adoption as a solution technology.

Chapter 61, Robot Surveillance and Security introduces surveillance and security robots for military and civil applications. The chapter describes the applications environment that covers ground, aerial and maritime domains for a wide range of surveillance and security robots. A systems overview is provided of the mobility component, sensor payload, communication system, and operator control interface, together with a description of the enabling technologies. The chapter concludes with a review of current research topics and discusses the future direction of robotic surveillance and security.

Chapter 62, Intelligent Vehicles, describes the emerging robotics application field of intelligent vehicles: motor vehicles that have autonomous functions and capabilities. The chapter describes why the development of intelligent vehicles is important and gives a brief history of the field and the potential benefits of the technology. It describes the enabling technologies for intelligent vehicles to sense vehicle, environment, and driver state, work with digital maps and satellite navigation, and communicating with intelligent transportation infrastructure. The chapter describes the challenges and solutions associated with road scene understanding. Advanced driver assistance systems which use robotics technologies to create new safety and convenience systems for motor vehicles, such as collision avoidance, lane keeping, parking assistance, as well as driver monitoring to mitigate against driver fatigue, inattention, and impairment are described.

Chapter 63, Medical Robotics and Computer-Integrated Surgery describes the development of medical robotics. In the last 20 years starting with brain surgery, orthopaedics, endoscopic surgery, and microsurgery, the field has expanded to include commercially marketed, clinically deployed systems and an active research community. This chapter provides a historical review of the field and discusses the major thrusts using examples from current and past research. Medical robots are described within the context of larger computer-integrated systems including presurgical planning, intraoperative execution, and postoperative assessment and follow-up. The basic concepts of computer-integrated surgery including the critical factors affecting the deployment and acceptance of medical robots are described. An overview of medical robot systems is provided, including remote telesurgery and robotic surgical simulators. The chapter concludes with a discussion on future research directions.

Chapter 64, Rehabilitation and Health Care Robotics, describes robotic systems that assist persons who have a disability or provide rehabilitation therapy for persons. The chapter provides a historical review of the field. Physical therapy and training robots as well as robotic aids for people with disabilities are described. Advances in smart prostheses and orthoses for rehabilitation robotics are also described. An overview of work in diagnosis and monitoring for rehabilitation is provided. The chapter provides a solid understanding of the future challenges for rehabilitation and heath care robotics.

Chapter 65, Domestic Robots, describes the technology that everyone would one day like in their home, which will vacuum the house, clean the kitchen, load the dishwasher, or polish the shoes. In spite of hundreds of millions of potential users surprisingly not many domestic robots exist, with one notable exception – the Roomba! This chapter examines what have been the success factors for commercializing domestic robots using cleaning robots as a representative example. The chapter reviews domestic floor cleaning robots, robotic pool cleaners, window cleaning robots, and robotic lawn mowing. The chapter also examines smart appliances such as sports robotics, and telepresence robots. A description of research projects for smart environments and smart homes is presented. The chapter describes the important underpinning technologies for domestic robots covering sensing, obstacle avoidance, localization, mapping and coverage.

Chapter 66, Robotics Competitions and Challenges, explores the use of competitions to accelerate robotics research and to promote education. Robot competitions have been shown to drive innovation in the robotics field. The chapter covers the two broad types of robot competition: human-inspired competitions and task based challenges. Human-inspired robot competitions, which are mostly sports contests, develop platforms that support problem solving. Task based challenges attract participants by presenting a tough new challenge for robotics technology. Three case studies of robot competitions are presented Robot Soccer, the Unmanned Aerial Vehicles (UAV) Challenge and the DARPA (Defense Advanced Research Projects Agency) Grand Challenges. The case studies describe the organizational challenges for participants, the benefits and limitations of competitions and what it takes to have a *good* robot competition.

Multimedia Contents

54. Industrial Robotics

Martin Hägele, Klas Nilsson, J. Norberto Pires, Rainer Bischoff

Much of the technology that makes robots reliable, human friendly, and adaptable for numerous applications has emerged from manufacturers of industrial robots. With an estimated installation base in 2014 of about 1.5 million units, some 171 000 new installations in that year and an annual turnover of the robotics industry estimated to be US$ 32 billion, industrial robots are by far the largest commercial application of robotics technology today.

The foundations for robot motion planning and control were initially developed with industrial applications in mind. These applications deserve special attention in order to understand the origin of robotics science and to appreciate the many unsolved problems that still prevent the wider use of robots in today's agile manufacturing environments. In this chapter, we present a brief history and descriptions of typical industrial robotics applications and at the same time we address current critical state-of-the-art technological developments. We show how robots with different mechanisms fit different applications and how applications are further enabled by latest technologies, often adopted from technological fields outside manufacturing automation.

We will first present a brief historical introduction to industrial robotics with a selection of contemporary application examples which at the same time refer to a critical key technology. Then, the basic principles that are used in industrial robotics and a review of programming methods will be outlined. We will also introduce the topic of system integration particularly from a data integration point of view. The chapter will be closed with an outlook based on a presentation of some unsolved problems that currently inhibit wider use of industrial robots.

54.1 **Industrial Robotics: The Main Driver for Robotics Research and Application** . 1386
54.2 **A Short History of Industrial Robots** 1386
54.3 **Industrial Robot Kinematics** 1392
54.4 **Typical Industrial Robot Applications** 1393
54.4.1 Handling 1393
54.4.2 Welding..... 1396
54.4.3 Assembly 1398
54.4.4 Painting 1402
54.4.5 Processing 1402
54.5 **Safe Human–Robot Collaboration** 1405
54.5.1 Overview of Basic Robot Safety Standards 1405
54.5.2 Types and Requirements of Human–Robot Collaboration .. 1406
54.5.3 Examples of Human–Robot Collaboration 1408
54.6 **Task Descriptions – Teaching and Programming** 1409
54.7 **System Integration**..... 1414
54.8 **Outlook and Long-Term Challenges** 1416
Video-References..... 1418
References..... 1418

54.1 Industrial Robotics: The Main Driver for Robotics Research and Application

Even though robots are considered a cornerstone of today's competitive manufacturing particularly in automobile and related component assembly, there are still challenges to solve for manufacturing to efficiently respond to changing consumer behavior and global shifts in competitiveness. Furthermore, high-growth industries (in electronics, food, logistics, and life-sciences) and emerging manufacturing processes (gluing, coating, laser-based processes, precision assembly, fiber material processing) as well as fulfilling sustainability regulations will increasingly depend on advanced robot technology [54.1]. Additionally, the range of feasible applications could significantly increase if robots were easier to install, to integrate with other manufacturing processes, and to program, particularly with adaptive sensing and automatic error recovery. Further challenges result from the integration of various types of controls (programmable logic controller (PLC), computer numerical control (CNC) sensors) with the robot controller, from close human–robot collaboration and fenceless production with both lightweight and heavy duty robots, and from an increasing need to save energy.

Design and production of industrial robots on the one hand, and the planning, integration, and operation of robot work cells on the other hand are largely independent engineering tasks. In order to be produced in sufficiently large quantities, a robot design should meet the requirements for the widest set of potential applications. As this is difficult to achieve in practice, various classes of robot designs regarding payload capacity, number of robot axes, and workspace volume have emerged for application categories such as assembly, palletizing, painting, welding, machining, and general handling tasks.

Generally, a robot workcell consists of one or more robots with controllers and so-called robot peripherals, e.g., grippers or tools, safety devices, sensors, and material transfer components for moving and presenting parts. Typically, the cost of a complete robot workcell is four to five times the cost of the robots alone; however, efforts are underway to drastically reduce these costs through use of increased robot functionality and artificial intelligence [54.2]. A robot workcell is usually the result of customized planning, integration, programming, and configuration, requiring significant engineering expertise. Standardized engineering methods, tools, and best-practice examples for specifying and designing robot workcells have become available to provide predictable performance and to secure investments [54.3].

Today's industrial robots are mainly rooted in the requirements of capital-intensive large-volume manufacturing, typically defined by the automotive, electronics, and electrical goods industries which make up 80% of all robot installations. Future industrial robots will not be a mere extrapolation of today's designs, but will rather follow new design principles addressing a much wider range of application areas and industries. At the same time, new technologies, particularly from the information technology (IT) or the consumer domain will have an increasing impact on the design, performance, use and cost of future industrial robots.

International and national standards now help to quantify robot performance and define safety precautions, geometry, and media interfaces [54.4]. Most robots operate behind secure barriers to keep people at a safe distance. Recently, improved safety standards have allowed direct human–robot collaboration, permitting robots and human factory workers to share the same workspace [54.5, 6].

54.2 A Short History of Industrial Robots

The invention of the industrial robot dates back to 1954 when inventor George Devol filed a patent on a *programmed article transfer* (Fig. 54.1). After teaming up with young engineer and entrepreneur Joseph Engelberger, the first robot company, Unimation, was founded. It put the first robot into service at a General Motors plant in 1961 for extracting parts from a die-casting machine. Most of the hydraulically actuated Unimates were sold through the following years for workpiece handling and for spot-welding of car bodies [54.7]. Soon, many other companies started to develop and manufacture industrial robots in many industrial nations; an innovation-driven industry was born [54.8]. The first International Symposium on Industrial Robotics (now ISR) took place in Chicago in 1970 and proved that robotics had become the field of activity of a vibrant research community.

The breakthrough Stanford Arm was designed as a research prototype in 1969 by Victor Scheinman (Chap. 4). The six-degree-of-freedom (6-DOF) all-

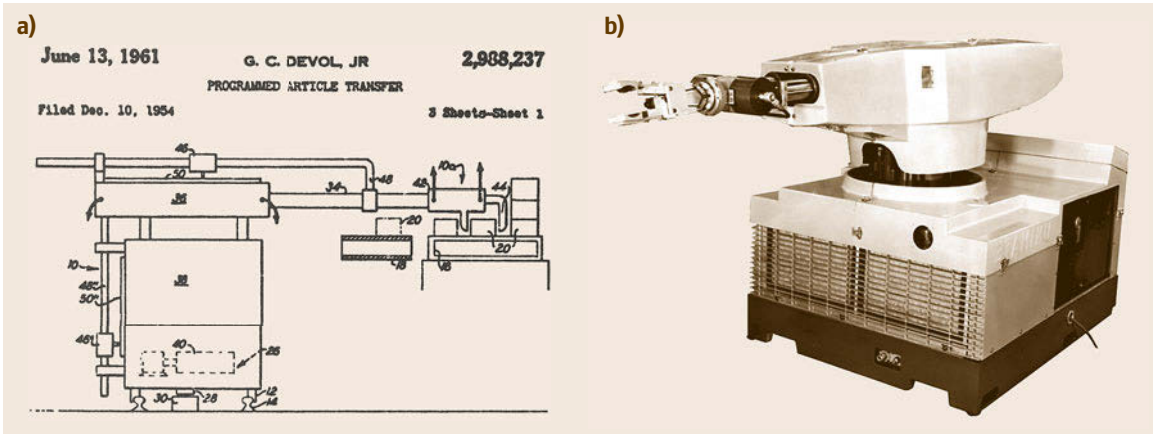


Fig.54.1a,b The invention of the industrial robot. **(a)** This patent was the start of a joint effort between G. Devol and J. Engelberger to form the first robot company, Unimation, a fusion of the terms *universal* and *automation*. The company was acquired by Westinghouse in the late 1980s and subsequently taken up by Stäubli in 1988. **(b)** The first Unimation performed a rather simple handling task in 1961 at a General Motors plant; other car manufacturers followed (courtesy of Smithsonian Institution Archives, Washington DC)

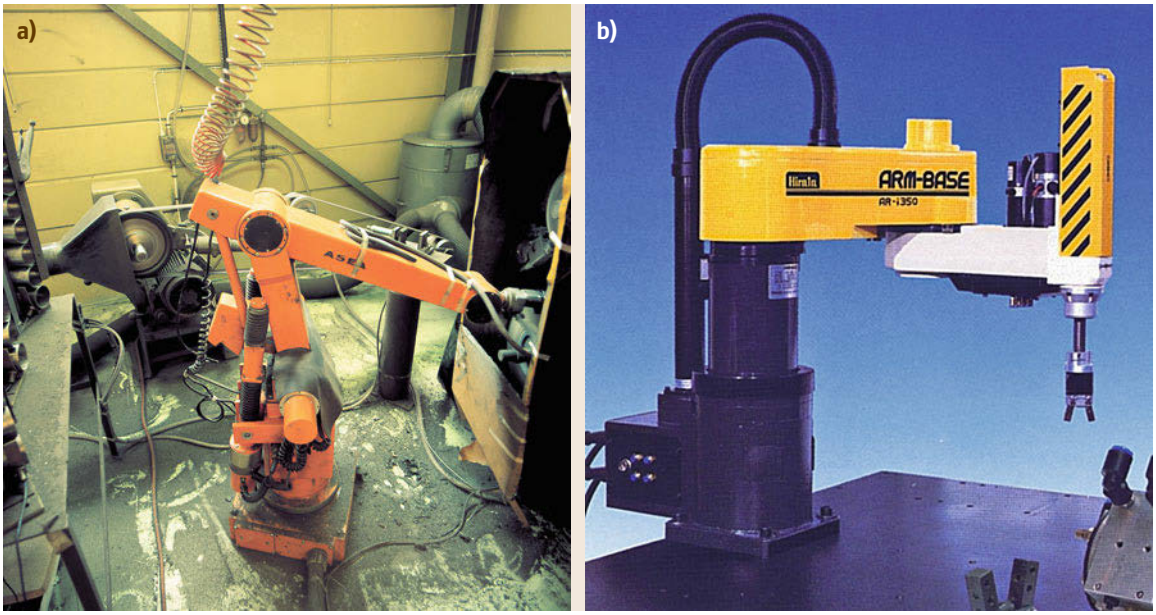


Fig.54.2a,b The all-electric **(a)** IRB-6 and **(b)** a SCARA-type kinematic. **(a)** First introduced in 1973, the IRB-6 has been a breakthrough development as it was the first serially produced robot product, which combined all-electric-drives technology and a microcomputer for motion control and programming. The robot proved very robust, and life-times of more than 25 years in harsh productions were reported (courtesy of ABB Automation). **(b)** The selective compliance assembly robot arm (SCARA) is particularly suited for assembly tasks as it combines rigidity in the vertical axis and compliance in the horizontal axis. In 1978, the first Hirata AR-300 was put together. Depicted is the successor design, the AR-i350 (courtesy of HIRATA Robotics, Mainz)

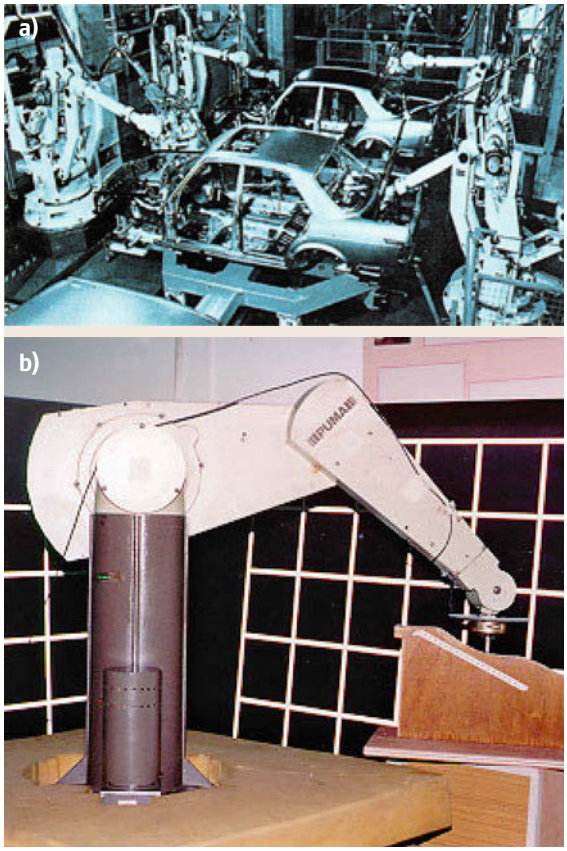


Fig.54.3a,b The KUKA IR 601/60 **(a)** and the Unimation PUMA (programmable universal machine for assembly) 560 **(b)**. **(a)** In 1978, the novel 6 DOF KUKA robot featured a parallel linkage for its second and third axes. At almost two tons of own weight, it could handle payloads of some 60 kg at maximum operating speed. The robot quickly became a workhorse for the automotive industry. **(b)** The six axis PUMA was inspired by the dexterity of a human arm. After its launch in 1979 by Unimation, it became one of the most popular arms and was used, due to its versatility and ease of use, as a reference in robotics research for many years

electric manipulator was controlled by a state-of-the-art computer of the time, a DEC PDP-6. The nonanthropomorphic kinematic configuration with one prismatic and five rotational joints was configured such that the equations for solving the robot kinematics were simple enough to speed up computations. Drives consisted of direct-current (DC) electric motors, harmonic drive and spur gear reducers, potentiometers and tachometers for position and velocity feedback [54.9]. Subsequent robot designs were strongly influenced by Scheinman's concepts (Figs. 54.2 and 54.3).

In 1973, the company ASEA (now ABB) introduced the first microcomputer-controlled all-electric industrial robot, the IRB-6, which allowed continuous path motion (CP), a precondition for many applications such as arc welding or material removal (Fig. 54.2). In the 1970s, intense diffusion of robots into car manufacturing set in mostly for (spot-)welding and handling applications (Fig. 54.3) [54.10].

In 1978, the selective compliance assembly robot arm (SCARA) was invented by *Makino* of Yamanashi University, Japan [54.11]. The ground-breaking four-axis low-cost design was perfectly suited for small parts assembly as the kinematic configuration allows fast and compliant arm motion (Fig. 54.4). Flexible assembly systems based on the SCARA robot in conjunction with compatible product designs (DFA) have contributed significantly to creating a boom in high-volume electronics production and consumer products [54.12]. Further optimization of robot dynamics and accuracy led to the first direct-drive SCARA robot, the AdeptOne in 1984 [54.13].

Requirements regarding a robot's speed, accuracy and weight have led to novel kinematic and transmission designs. An approach toward lightweight and stiff structures has been pursued since the 1980s by developing parallel kinematic machines (PKM) which connect the machine's basis with its end-effector by three to six parallel struts, see also Fig. 54.5. These so-called parallel robots (Chap. 4 and 18) are particularly suited to achieve short cycle times (e.g., for picking), precision (e.g., for material removal), or handling high workloads (Fig. 54.5) and have found their niches in advanced manufacturing [54.14]. However, workspace volumes tend to be significantly smaller than those of serial or open kinematic chain robots which are comparable in size.

Efforts of reducing mass and inertia of serial robot structures have been a primary research target, where the human arm with a weight-to-load ratio better than 1 : 1 was considered the ultimate benchmark. In 2006, robot manufacturer KUKA introduced their LBR lightweight prototype robot, a compact 7-DOF robot arm with advanced torque-control capabilities which has recently been introduced in high-performance industrial applications [54.15]. An obvious next step in approaching human dexterity is the recent introduction of two-armed robot designs with some recent developments being depicted in Fig. 54.7 [54.16]. In conjunction with a robot's capability to support safe human–robot collaboration, new manufacturing concepts can be implemented which expand capabilities, productivity, and ergonomic quality to manual workplaces [54.17].

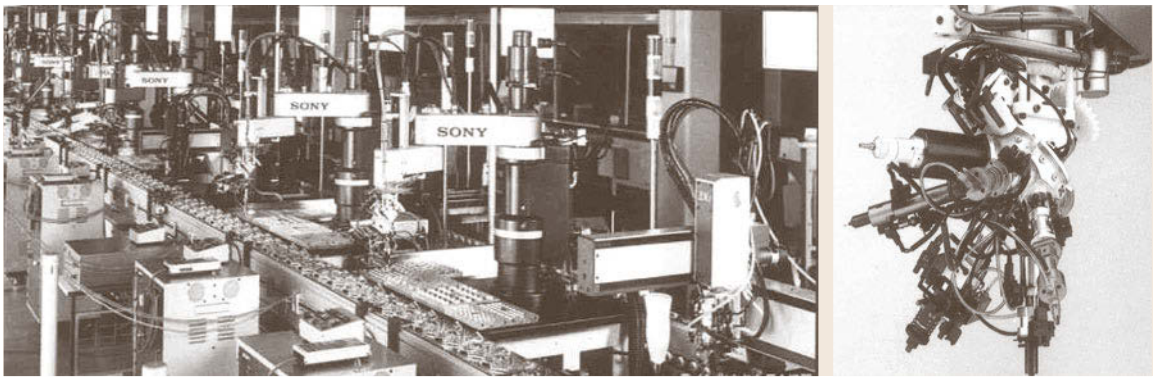


Fig. 54.4 An automated video cassette recorder (VCR) assembly line (about 1989) with SCARAs carrying a turret with multigripper tools. Typically five parts are added by one robot before the VCR is moved to the next station of the automated assembly line

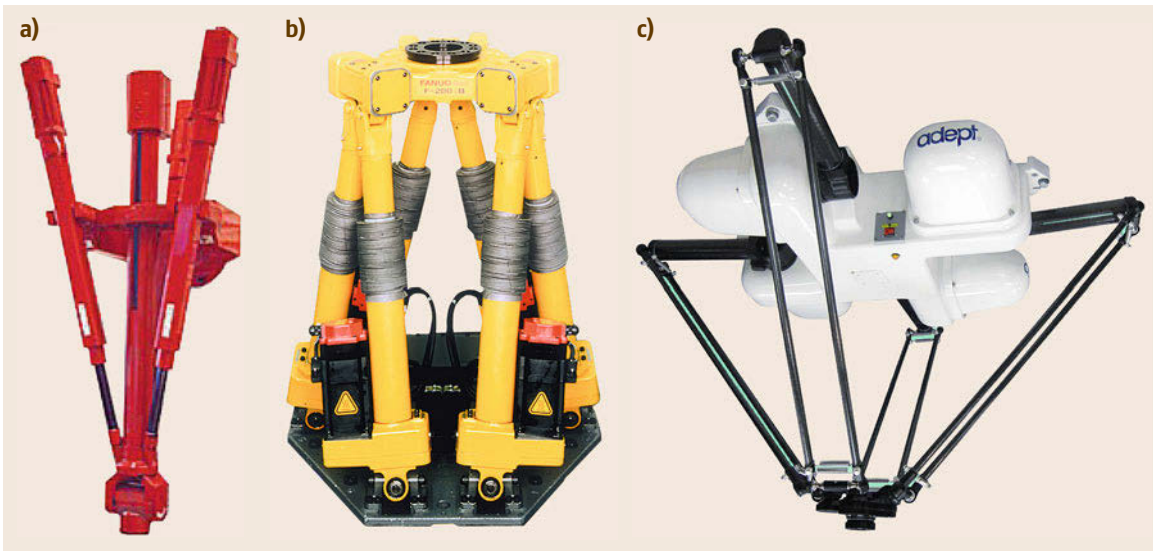


Fig.54.5a–c Parallel robots are slowly diffusing into various fields of industrial application: **(a)** the Neos Tricept 600, **(b)** Fanuc F-200iB. **(c)** Adept Quattro. **(a)** In 1992, Neos Robotics represented with their Tricept robot range a concept to combine the stiffness of machine tools with the dexterity of a robot for heavy-duty applications such as in friction stir welding (FSW) or machining of aluminum for the aircraft industry. **(b)** The Fanuc F-200iB introduced in 2002 is a 6-DOF parallel robot particularly designed for welding gun handling, deflashing, or for assembly tasks (100 kg payload, ± 0.1 mm accuracy) in automotive assembly processes; **(c)** the Adept Quattro (introduced in 2007, following the ABB FlexPicker in 1998) is suited for high-speed applications in packaging, manufacturing, assembly, and material handling. The quad dual-link arm design forms an over-determined kinematic linkage, which in the wrist is converted to the forth axis of end-effector rotation by means of an internal transmission in the wrist

In parallel to industrial robots, automated guided vehicles (AGV) have emerged. These mobile robots are used for moving workpieces or loading equipment following a predetermined or virtual path in industrial environments. Within the concept of automated flexible manufacturing systems (FMS) AGVs have become an important part of their routing flexibility. Initially, AGVs relied on prepared floors such as embedded wires, magnets, or other tags for motion guidance. Meanwhile, freely navigating AGVs along virtual trajectories are entering large-scale manufacturing and logistics. Usually, their navigation is based on laser scanners that provide an accurate two- or even three-dimensional map of the actual environment for self-localization and obstacle avoidance. Early on, combinations of AGVs and robot arms were sug-

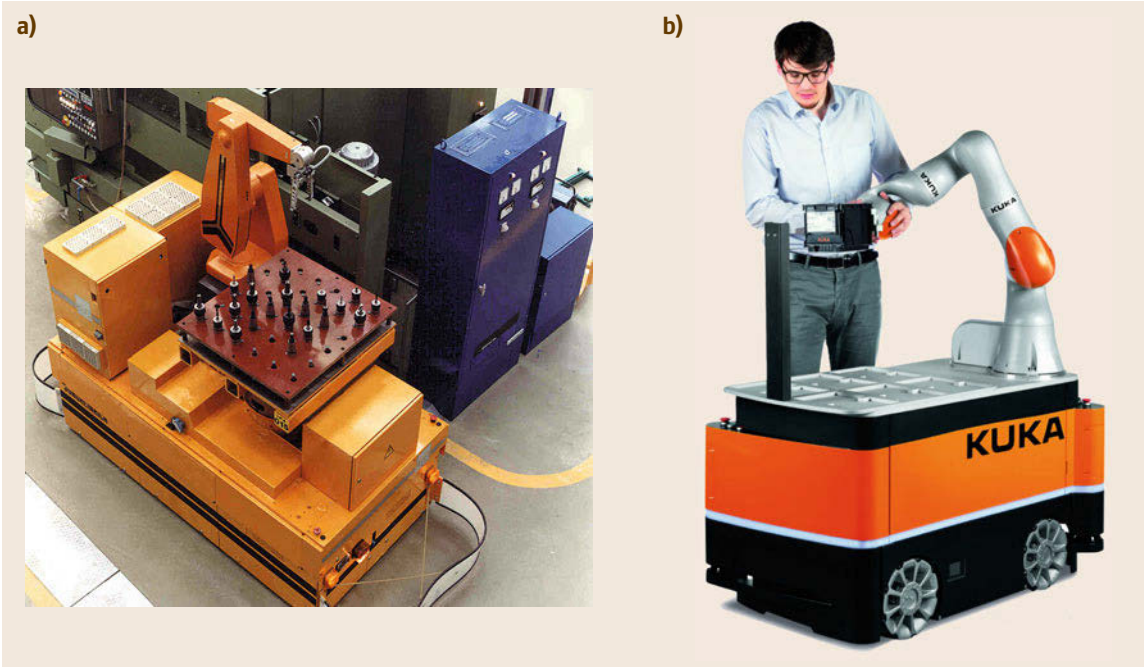


Fig.54.6a,b Mobile robots were introduced in the early 1980s for increased flexibility and reliability in factory logistics. **(a)** The MORO (1984) developed at Fraunhofer IPA was one of the first prototypes to combine a robot arm on a wire-bound mobile platform which follows a wire buried in the floor. **(b)** The KUKA omniRob features an omnidirectional platform and the LBR iiwa lightweight arm which form a highly kinematically redundant robot system (courtesy of KUKA)

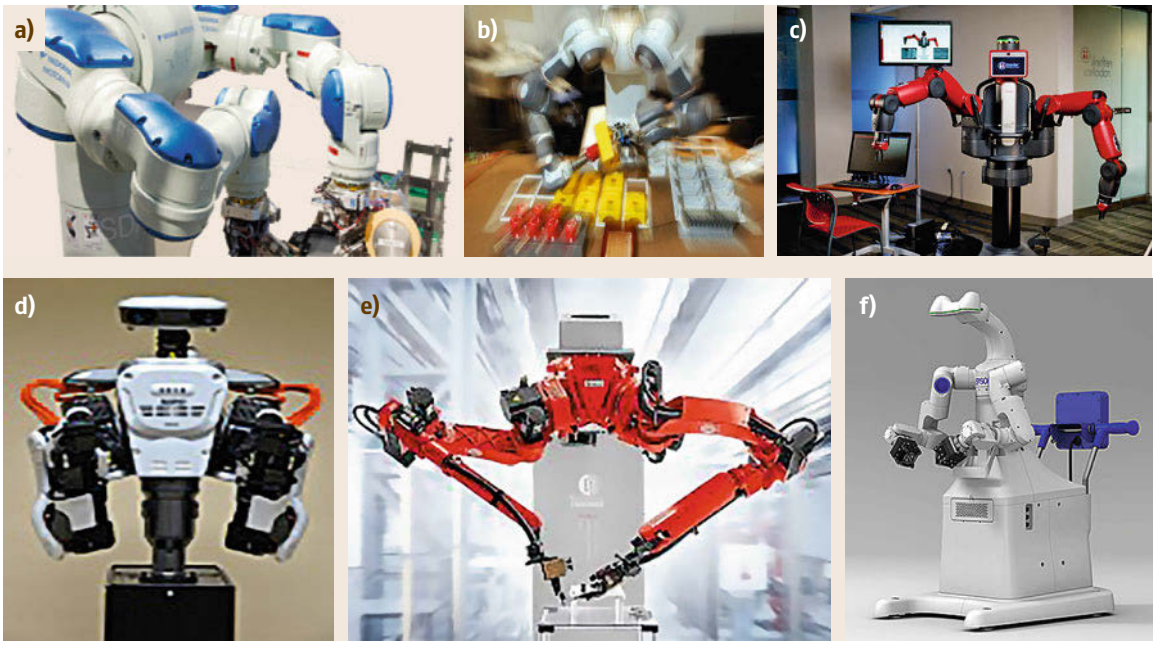


Fig.54.7a–f Examples of different designs of dual-arm robots (courtesy of **(a)** Motoman, **(b)** ABB, **(c)** Rethink Robotics, **(d)** Kawada Industries, **(e)** COMAU, **(f)** Seiko Epson)

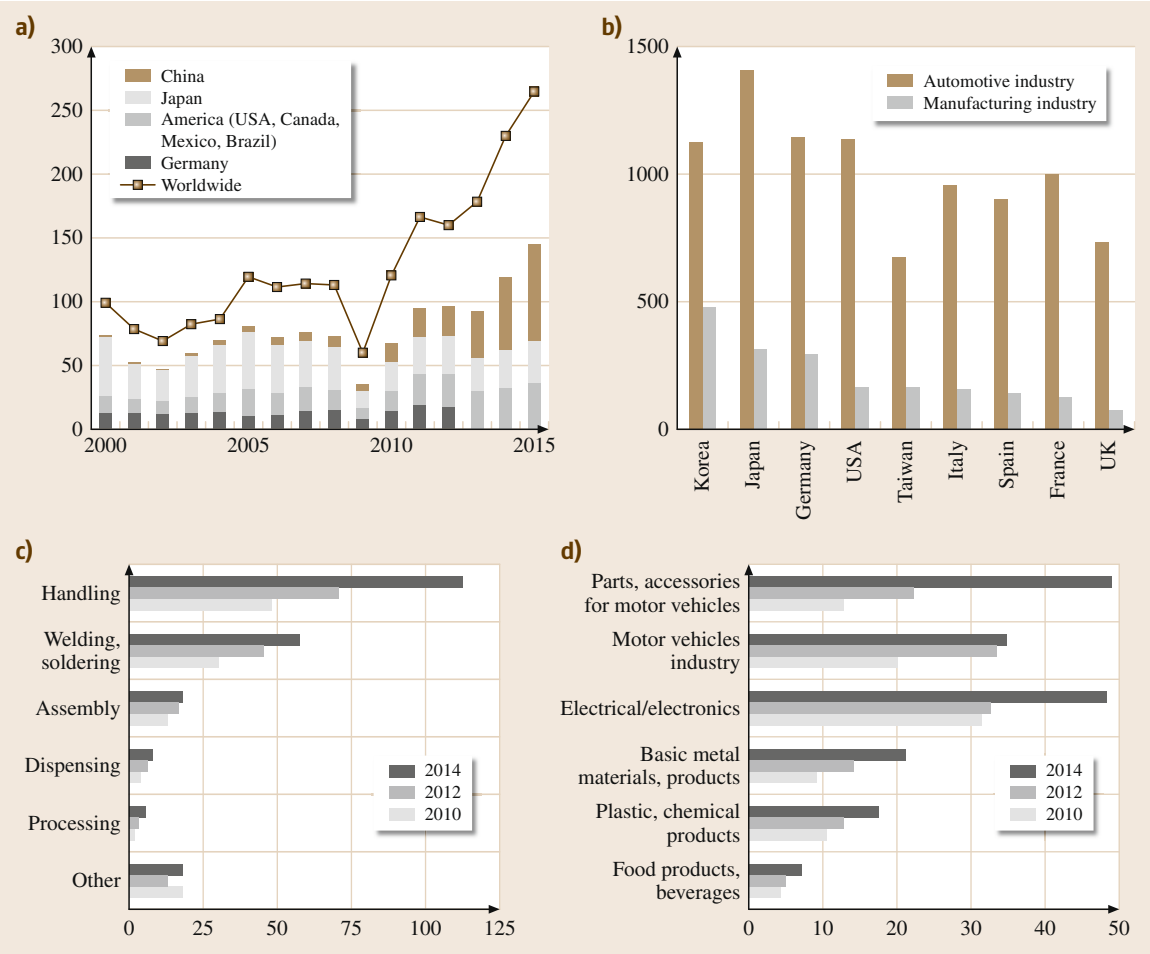


Fig.54.8a–d Statistics of worldwide industrial robotics use (after [54.1]). **(a)** Estimated annual robot installations in selected countries (1000 units, estimate for 2015), **(b)** Number of multipurpose industrial robots (all types) per 10 000 employees in the automotive and in manufacturing industries 2014. **(c)** Estimated worldwide annual shipments of industrial robots in main application areas. **(d)** Estimated worldwide annual shipments of industrial robots in main industrial branches

gested to automatically load and unload machine tools (Fig. 54.6). Safety and power supply have been an obstacle to these system's diffusion in industrial practice. Currently, first solutions for mobile manipulation appear [54.18].

The ability to use human and robot workers either interchangeably or in workspace sharing/collaboration scenarios in human workplaces motivated the design of anthropomorphic dual-arm robots (Fig. 54.7. Even though industrial acceptance initially has been low, advances in programming comfort, securing safe human–robot coexistence/collaboration and system cost have led to significant interest in using dual arms in agile manufacturing concepts, particularly in assembly and handling applications [54.19]. The dual-arm systems suggest a new way of using powerful and lean type of robot which is easy to install by the manufacturing end-user with little adaptation of manual workplaces.

Today, industrial robotics is seen as a central pillar to future manufacturing competitiveness and economic growth:

- The International Federation of Robotics (IFR) estimates that between 2000 and 2008 the robotics industry had created 8–10 million highly qualified jobs, either directly or indirectly. The prediction is that between 2012–2020 another 4 million jobs will be created in the *robot ecosystem* [54.20]. The extent of job creation by robotics has been dis-

cussed controversially. It is undisputed, however, that a wider use of robots in manufacturing is able to significantly strengthen a competitive position of a company or an industrial sector [54.21]. Economically, manufacturing productivity gains are particularly effective for economic growth. There is no sustainable product innovation without manufacturing competence which includes knowledge and practice of planning, designing, and operating advanced robotic systems [54.22] (Fig. 54.8).

- The average price for a robot in 2014 was in the order of US$ 46 800, which is about one-third of its equivalent price in 1990. At the same time, robot performance parameters such as speed, load capacity, and mean time between failures (MTBF) have dramatically improved. This means that automation has become more affordable, providing a faster return on investment [54.1].
- Traditionally, robot automation has not played a significant role in the implementation of lean manufacturing strategies. However, efforts are underway to introduce industrial robots to lean, agile manufacturing. Characteristics are robot solutions that can be flexibly added to manufacturing systems *on demand*, that are significantly less expensive on a life-cycle-costing (LCC) basis than today's systems due to reduced peripherals and systems integration (*system out of the box*) [54.23]. With robots becoming commodities in manufacturing they might be used as intuitively and naturally as a handheld power tool today. This would imply intuitive and safe human–robot collaboration and versatility due to advanced sensing, control, and embedding the robot set-up and operation in an IT infrastructure.
- Factories of the future will represent a network of self-organizing cyber physical systems (CPS). As part of this industrial internet CPSs embed computation, networking, and physical processes and can either represent manufacturing equipment such as machine tools, fixtures, trays, conveyors, tools, etc. or the workpiece which controls and memorizes its production. Robots are considered the centerpiece of future smart factories which combine manufacturing agility, profitability, human ergonomics, and minimized resource consumption [54.24].
- Robots in assembly have not reached their predicted installation potential mainly due to cheap labor cost and the lean assembly work systems which support highest flexibility and productivity at minimal waste. A reason is partly seen in the slow advances in dexterous manipulation for assembly tasks with industrial robustness. Here, torque-controlled lightweight robots [54.25] and two-armed robot systems have been proposed to imitate human ergonomics and task execution [54.26].
- New financing models such as leasing, *pay by service* will allow end-users to use robots on demand or to have manufacturing service providers to operate manufacturing lines on a *pay-on-production* basis [54.27].

Figure 54.8 depicts some key figures on the recent extent of industrial robot diffusion into manufacturing.

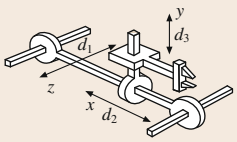
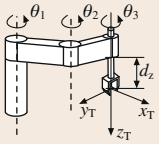
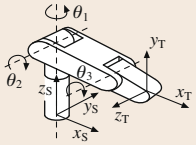
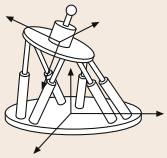
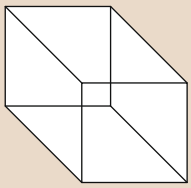
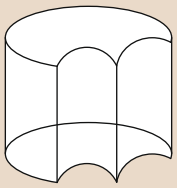
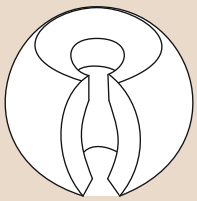
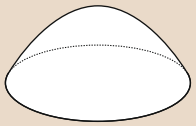
54.3 Industrial Robot Kinematics

By definition, an industrial robot is an automatically controlled, reprogrammable, multipurpose manipulator, programmable in three or more axes, which can be either fixed in place or mobile for use in industrial automation applications [54.28]. Robots can be categorized according to their number of independent kinematic axes, and their mechanical structure which affect most of the robots' kinematic properties, the computation methods used to determine joint motions, and the form and size of the robot workspace. Robot mechanical structures are composed of links that are rigid bodies connecting neighboring (prismatic, rotary, cylindrical or spherical) joints. The diagrams in Table 54.1 show several common types of robot mechanical structures. Of course, the workspace of industrial robots can be significantly expanded by placing the robot arm on an additional linear axis, sometimes reaching a length of more than 50 m, or even on mobile platforms. Furthermore, robot mechanical structures can be composed by joint modules which are connected by links to form task-specific designs.

With advances in the state of the art in motion control and computer hardware processing capabilities, computation is much less a constraint on mechanism choice than it was for early robot designers. The choice of mechanical structure of the robot depends mostly on fundamental mechanical requirements such as payload and workspace size. Considering a given level of cost, there is usually a trade-off between workspace size and stiffness. To enable the robot to reach inside or around obstacles it is clearly advantageous to use an articulated mechanical design.

Considering also the stiffness and accuracy (in a practical sense considering what is reasonable to build), the picture is more complex. Each of the first three types in Table 54.1 we refer to as serial kinematic

Table 54.1 Main categories of mechanical structures of industrial robots: Gantry is what a Cartesian coordinate robot is typically called with three prismatic joints, whose axes are coincident with a Cartesian coordinate system. The SCARA or selective compliance assembly robot arm has two parallel rotary joints to provide compliance in a selected plane. The articulated robot has three or more, typically six, rotary joints placed in series with their interconnecting links. The parallel link robot is characterized by links that form closed loop structures shown with prismatic joints, but can also have revolute joints such as the Delta robot (Fig. 54.5) (pictures courtesy of Güdel, ADEPT, ABB, PI Physik Instrumente)

Category	Gantry (or Cartesian)	SCARA	Articulate	Parallel
Robot main axes structure	 3 prismatic joints	 1 prismatic and 2(3) revolute joints	 3 revolute joints	 Typically with 3, 4 or 6 prismatic axis
Workspace shape				
Technical example				

machines (SKMs), while the last is a parallel kinematic machine (PKM). To obtain maximum stiffness, again for a certain minimum level of cost, the end-effector is better supported from different directions, and here the PKM has significant advantages. On the other hand, if high stiffness (but not low weight and high dexterity) is the main concern, a typical computerized numerical control (CNC) machine (e.g., for milling) is identical in principle to the gantry mechanism. There are also modular systems with servo-controlled actuators that can be used to build both robots with purpose-designed mechanisms.

54.4 Typical Industrial Robot Applications

Out of the many possible uses of industrial robots selected case studies on high-potential robot applications will be briefly described. Typical associated enabling technologies will be depicted.

54.4.1 Handling

Handling in robotics comprises numerous processes such as grasping, transporting, packaging, palletizing, and picking. As seen in Fig. 54.8 handling is the largest robot application field which is found in all branches of manufacturing and logistics. A central feature and major challenge in the engineering of robotic handling systems is the design of the gripper and associated grasping strategies given the physical properties of the workpiece, throughputs, and uncertainties regarding object geometry and location. Current high-potential application of robot handling systems are: tending of CNC machines for workerless shifts [54.29], palletizing, and lifting of objects for ergonomic reasons or

Fig.54.9a–d Units of sausage are cut from strings, then placed into the thermo-formed cavities before applying lidding. The coordination of the robots and the optimization of the picking frequency require a selection of the best path for each robot. Missed sausages are fed back on the conveyor for another try. The shown 4-DOF parallel robot reaches cycle times of 1–3 Hz and can move payloads of up to 8 kg (courtesy of robomotion, Germany)

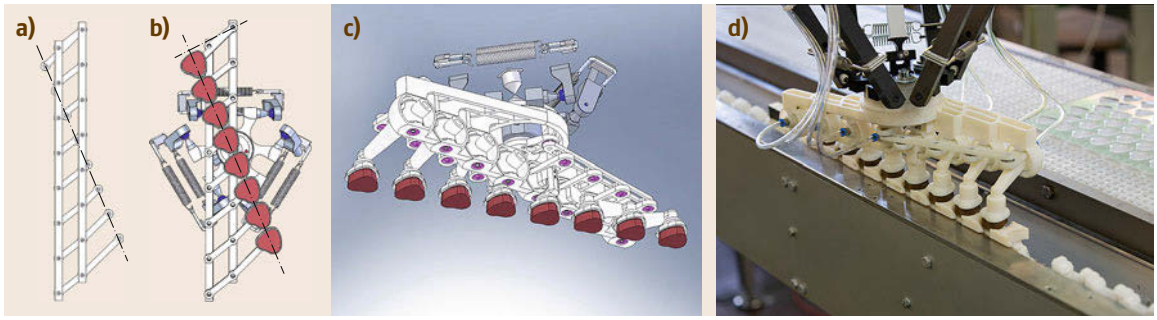


Fig.54.10a–d Lightweight customized vacuum gripper (0.75 kg mass) through additive manufacturing. Cookies are delivered continuously on a belt, grasped from the belt, and batches of eight are put on a blister matrix before final packaging. The gripper's spacing is pneumatically actuated and its rotation through the parallel robot's central rotational axis (courtesy of robomotion, Germany)

when limitations specified in load handling regulations are exceeded [54.30], for reasons of cleanliness as is typical in the food, pharmaceutical, and semiconductor industries [54.31–33], avoiding monotonous work and psychological strain, and ensure logistics quality through workpiece or object tracking [54.34]. In the following, two use-cases of material handling will be highlighted, each in a different industrial domain, and based on specific industrial robot type and enabling technologies.

Food Handling

The food sector is claimed to have significant potential for the application of robots as fundamental change to productivity, product quality, and worker ergonomics can be achieved [54.35]. In food automation, *untouched by human hand* entails critical requirements for robot automation such as the need for hygienic design, operational speed, ease of programming, and cost. In the past, these requirements had been difficult to achieve due to high throughputs, therefore requiring rapid grasps and fast robot motion, robust sensing for detecting object locations on conveyor belts. High speed at high flexibility apparently is a key in the individual handling of food objects. Therefore, fast SCARA and parallel robots have found wide acceptance in this field.

An example of a packaging line in food production is depicted in Fig. 54.9 where cut mini salamis are delivered in four streams per conveyor belt in random sequence. The positions of the sausages on the translucent belt are determined by a computer vision system. The robot picks the sausage from the measured position sequentially until the gripper holds three sausages which are then placed into cavities. With four parallel robots a maximum pick rate of 600 sausages per minute can be processed. Key of the application is its high-speed 2-D (two-dimensional) computer vision system which feeds the robot's path planning for collision-free picks at a minimum loss rate of unpicked salami [54.37].

Recent efforts have led to customized designs of gripper systems through 3-D (three-dimensional) printing (additive manufacturing) which for instance includes actuation through pneumatically driven bellows and low-wear metal joints. An example of a highly actuated gripper based on 3-D-printing for use in food handling is depicted in Fig. 54.10. Additive manufac-

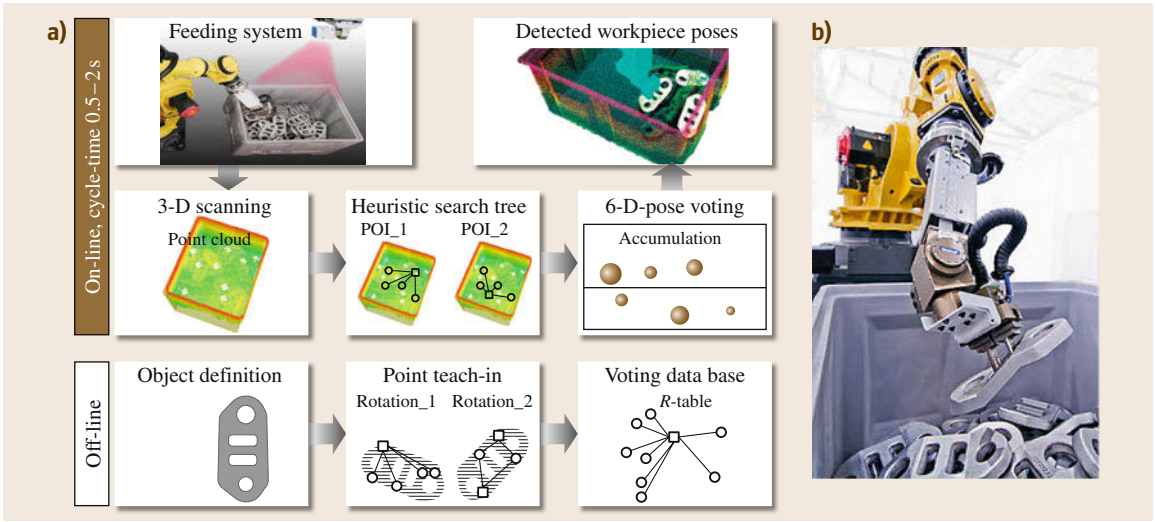


Fig.54.11a,b Procedure of a bin-picking method [54.36] **(a)** and gripper with additional degree of freedom for reaching deep into bins **(b)**. Depicted is a 2-D laser scanner on a swiveling unit for acquiring the point cloud in parallel to the robot's motion. The object detection itself consumes 0.5–2 s and is less time critical than the robot's motion and grasps

turing processes seem to be perfectly suited to achieve higher flexibility in manufacturing automation [54.38]. Numerous materials have become available for different additive manufacturing processes so that even specific manufacturing requirements can be matched. Initial doubts about gripper durability have been dispelled: lifetimes of more than 10 million load cycles for robot grippers manufactured on the basis of laser sintered polyamide have been reported.

Bin-Picking

Generally, industrial practice in robot workcell planning aims at finding a compromise between reducing the variation of the workpiece location and the cost of sensor systems to compensate for residual variation or uncertainties. Today, nearly all parts arrive at robot workcells in a repeatable manner, either being stored in special carriers or magazines, or by being transported and oriented by vibrating devices that allow the parts to settle into a predictable orientation for proper robot grasping. However, cost and flexibility requirements in manufacturing automation will result in reducing customized parts magazines to more universal carriers, containers, or conveyor belts. If randomly oriented on a conveyor belt or in a carrier, parts have to be properly identified and located so that the robot can produce an collision-free grasp.

The challenge of grasping partly or randomly ordered parts by robot has been referred to as bin-picking and has been investigated by numerous researchers since the mid-1980s [54.39]. Even though an abundance of approaches has been presented, only recently bin-picking installations have found their way into daily manufacturing in significant numbers. Bin-picking algorithms follow a typical sequence of steps: initial point cloud data acquisition, object detection, pose estimation, collision-free path and grasp planning, object grasping, and object placement. Most methods in bin-picking assume known geometrical representations (a computer-aided design CAD-model) of the workpiece in question including the specification of admissible grasps for applying template-matching methods [54.40, 41]. Figure 54.11 depicts a variant of a fast template-matching method, which encompasses the following steps for detecting object poses:

- For detecting the scene (e.g., a carrier or box filled randomly with workpieces) typically laser-based sensor systems are used for acquiring a sufficiently dense point cloud. The object pose detection then is considered as a combinatorial optimization problem for which a construction heuristic is applied. For this heuristic tree search, a finite set of possible workpiece poses is initially derived from the search space.
- In order to use a decision tree, the elements of the search set are split into two components: The first component describes a point of interest (POI) in the search space which is part of the workpiece

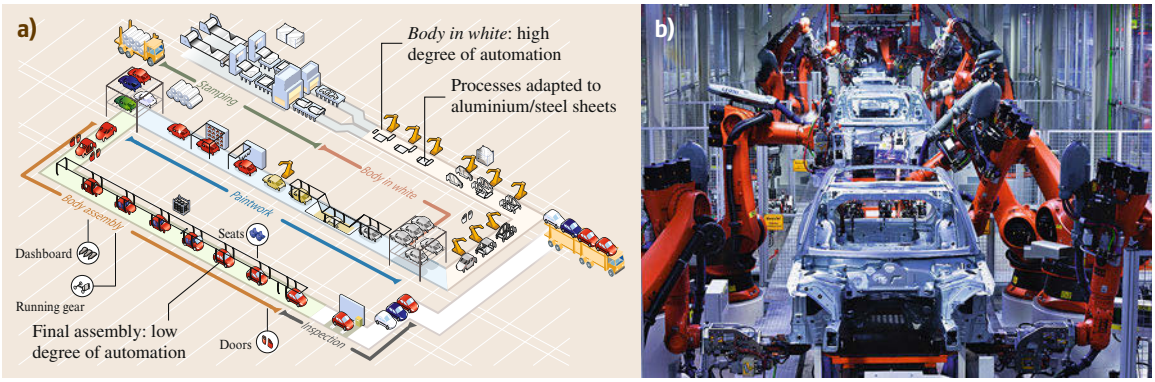


Fig.54.12a,b Car production **(a)** usually follows the illustrated steps along the assembly line: Stamping of the metal sheet into plates, fixing and alignment of the plates on trays, spot welding, painting the car body, and final assembly of the car body (doors, dashboard, windscreens, power-train seats, and tires). Car factories can host over 1000 industrial robots working typically three shifts per day (courtesy of PSA Peugeot Citroën, Paris and Art Movies, Paris). The Audi plant in Ingolstadt Germany **(b)** is highly automated. The picture shows spot welding robots along the *body-in-white* transfer line. Trays carrying car bodies pass through the *robot garden*

surface. The second component describes possible workpiece poses relative to a POI. The partial search quantities obtained thereby have a significant lower complexity compared to the original search set, since the points of interest can provide a constraint on relative workpiece poses, thus restricting their assumed freedom of movement.

- Typical tree search strategies such as best-first search can be used. In that case best-first search explores the search tree by expanding the most promising nodes first. These nodes are chosen according to a heuristic evaluation score, representing the estimated distance from the node to a solution.
- Final evaluation of the workpiece poses is provided by a six-dimensional (6-D) Hough voting procedure, i. e., a generalized Hough transform. The features used for Hough voting are sensor measurements located relative to a POI. For all possible constellations of a sensor measurement relative to a point of interest, a probabilistic statement about possible workpiece poses can be made. Through the superposition of all probability statements, solution candidates can be formed, which are subjected to a statistical test based on a quality rating. The obtained quality rating along with a given level of significance is used in order to decide about the acceptance of a workpiece pose.

The method is able to locate three to four workpieces on average within 0.5–2 s, using a standard desktop computer. Moreover, robot grippers are equipped with a seventh axis to allow grasping parts from corners of the bin. Furthermore, grippers should be formed in such a way that they may reach deep into the bin so that they offer only minimal collision volumes.

54.4.2 Welding

Welding is a manufacturing process that joins materials by applying heat, sometimes with pressure. Usually, workpiece material is melted at the process location often with additional filler material. Typical robot-based welding processes are spot welding, particularly in car body assembly, and gas-shielded metal arc welding (GMAW). With increasing compactness of laser sources and robot motion accuracy, laser welding is in the rise.

Manual welding requires skilled workers, as small imperfections in the weld can lead to severe consequences. Furthermore, welders are exposed to hazardous working conditions (fumes, problematic ergonomic working positions, heat, and noise) so that the use of robots has become beneficial in GMAW processes even for the smallest lot-sizes. Commonly, the automatic arc-welding process is based on a consumable wire electrode and a shielding gas that is fed through a welding gun. Modern welding robots are particularly suited through the following characteristics:

- Computer control allows efficient programming of task sequences, robot motions, external actuators, sensors, and communication with external devices such as welding sources.

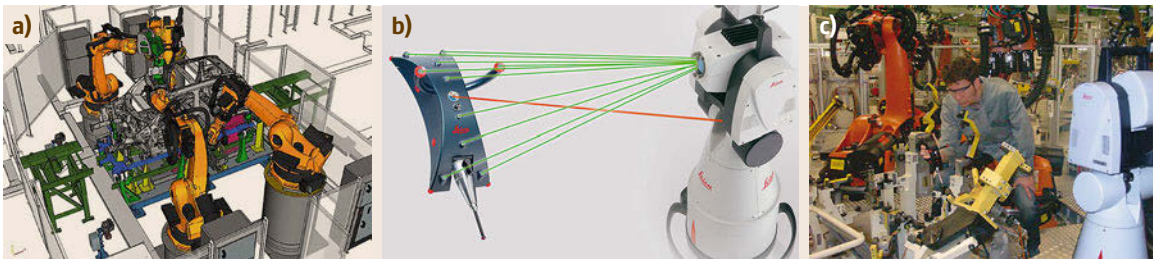


Fig.54.13a–c Offline programming of a spot welding workcell. **(a)** The robot workcell and the task execution are modeled on the basis of realistic robot models (geometry, kinematics, kinetics). **(b)** The shown laser tracker is a portable measurement system that relies on laser beams to accurately measure in a radial volume (accuracy of $\pm 10\,\text{ppm} = 10\,\mu\text{m/m}$, up to 80 m in diameter, measuring rate up to 3000 points/s). If measured objects cannot be equipped with reflecting targets or reached by the tracker, handheld probes are tracked instead. **(c)** A tracker in use for interactively measuring the geometry of the robot workcell (courtesy of Leica, now Hexagon MI)

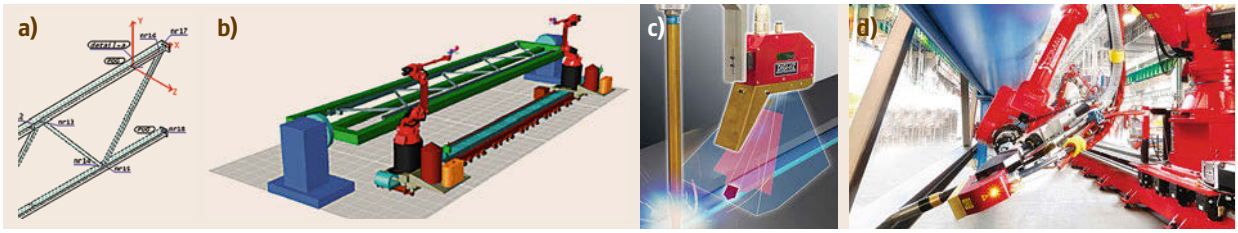


Fig.54.14a–d GMAW welding of building trusses in lot-size one by robot. Illustration **(a)** shows the CAD drawing of a steel truss with relevant information for welding process, **(b)** one-half of the welding workcell with two welding robots working simultaneously on the truss when the neighboring truss on the other half is loaded or unloaded, **(c)** the laser-based seam finding and tracking sensors and **(d)** the welding robot (courtesy of Servo Robot, Canada; Goldbeck, Germany)

- Free definition and parameterization of robot positions or orientations, reference frames, and paths.
- High repeatability and positioning accuracy of paths. Typically repeatability is some ± 0.05 mm and positioning accuracy is better than ± 1.0 mm. These values can be significantly improved through modern robot calibration methods [54.42].
- High speeds of the end-effector of up to 8 m/s for quick approach and depart motions.
- Typically, articulated robots have six DOF so that commanded orientations and positions in their workspace can be reached, which in the welding case means there is one DOF free for rotation around a rotational-symmetric welding tool. Additionally, workspace extensions by mounting the robot on a linear axis (seventh DOF) or even on mobile bases are common, especially for welding of large structures.
- Typical payloads range 6–150 kg. Higher load capacities are required for spot-welding guns (typically > 50 kg) and their cable package.
- Programmable logic controller (PLC) capabilities such that fast input/output control and synchronizing actions within the robot workcell are accomplished.
- Interfacing to high-level factory control through factory communication networks.

Electric current sources, torches, and peripheral devices for automatic cleaning and maintaining the GMAW torch (anti-splatter, wire-cutting, tool changer, etc.) are offered by specialized companies. Often sensors are used to track welding gaps and measure weld seams either before or synchronously with the welding process, thus adapting the robot's trajectory in the presence of workpiece variation and distortion. Also, collaborating robots have been introduced where one robot fixes and moves the workpiece in synchronization with another robot carrying a welding tool so that the weld can be performed with the pool of molten metal horizontal.

Spot Welding in the Automotive Industry with Offline Programming

Car manufacturing has been one of the key drivers in the technical development of industrial robots as the precision handling of spot-welding guns was one of the first breakthrough use cases (Fig. 54.12) *body-in-white* (i. e., unpainted car body) assembly is mostly done by robots, very much in contrast to the final assembly which is

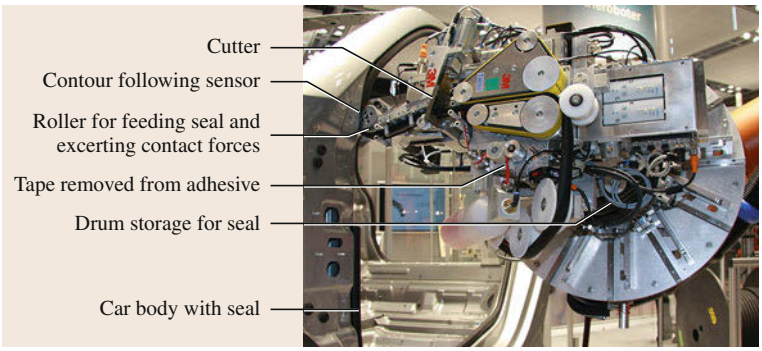


Fig. 54.15 A robot-guided tool for handling and processing limb material, in this case a self-adhesive seal for car bodies

dominated by manual work. Demands for faster cycle times have led to a concurrent and coordinated motion of the spot-welding gun and robot: the robot continues to move while the weld gun is simultaneously rotated about the electrode axis during welding [54.43].

Most of a spot welding robot's programming is done using offline programming (OLP) packages (Fig. 54.13. A library of robots, devices, and advanced CAD capabilities helps plan, program, visualize, and optimize layouts and complete production cycles under assumed manufacturing conditions. Robot programs can be generated and downloaded to robots workcells. A critical step is the calibration of the robot workcell with respect to the simulations [54.44].

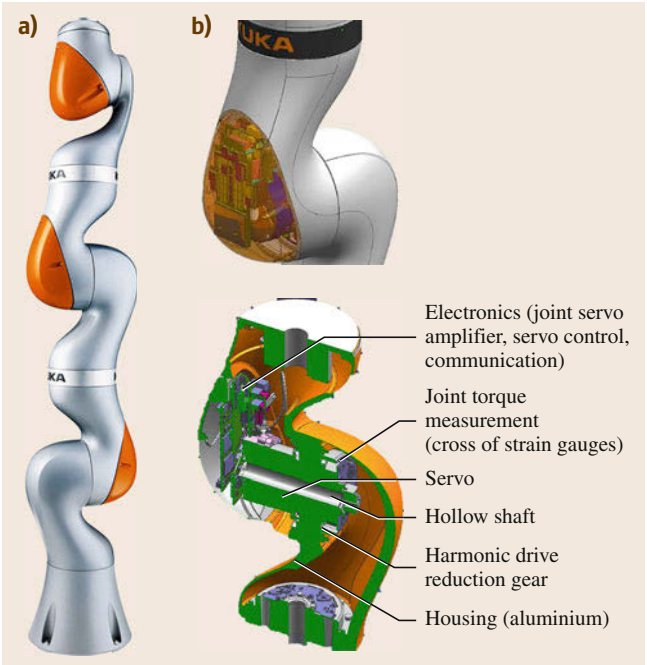


Fig.54.16a,b Design of the KUKA iiwa: **(a)** Shape and **(b)** integration of joint mechatronics

Arc Welding in Metal Construction

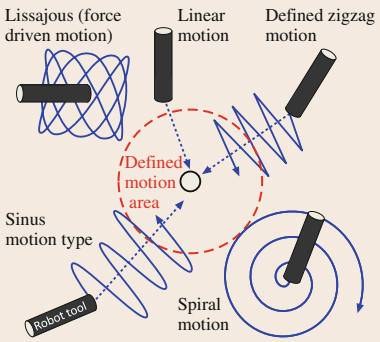
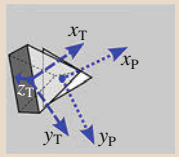
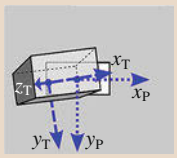
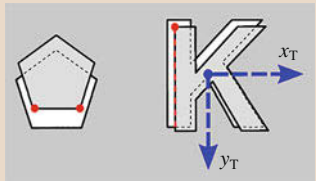
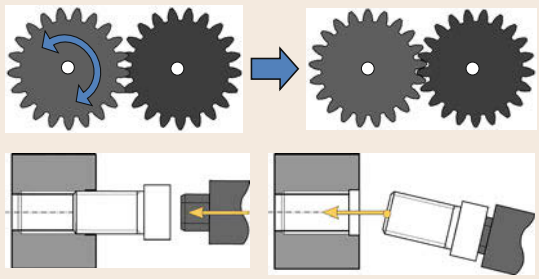
Normally steel constructions are designed using CAD programs that offer functions for GMAW-task definitions such as welding parameters, multipass seams, weld beads sequencing, etc. This information may be used for automatically generating welding robot programs, even in the case of lot-size one jobs.

As an example, the generation principle of a welding robot program is depicted in Fig. 54.14. Large-scale trusses of up to 15 m for large halls are welded-to-measure. The robot program is generated from the CAD drawing with process relevant information. Workpiece tolerances for example, induced by placing the steel components into the fixtures, by bending of the material under its own weight are compensated through active measurement. The robot-mounted sensor locates the weld seam by laser-based vision for shifting the generated programs in such a way that they match the actual weld seams. This calibration is automatically performed if expected and actual bead locations are within a range of ± 2.5 cm.

54.4.3 Assembly

Assembly in manufacturing describes the combination of subsystems or components to systems of a higher complexity through joining. Assembly in manufacturing comprises four process groups: joining, handling, controlling, and auxiliary processes (cleaning, adjustment, marking, etc.) [54.45]. The composition of these four functions may vary depending on batch size, product, and throughput: from assembly workcells to high-throughput assembly lines. Assembly processes form up to 80% of a product's manufacturing cost and this is where the greatest competitive advantage can be gained [54.46]. Therefore, optimization in assembly includes tightly interweaved aspects: Design

Table 54.2 Assembly subprocesses or *modules* and their implementation on the KUKA LBR iiwa

Assembly subprocess	Characterization	Principle
search: search module which supports several search motions or search strategies	Search motion type examples: ● Linear ● Zigzag ● Spiral ● Sinus ● Lissajous Commanding the search motion generation: ● Position-based trajectory ● Force-based trajectory ● Combination of position and force-based trajectory	
peg in hole: execution of typical part joining motions	Reduction of arbitrary joining part types to three abstract planar types: ● Round (arbitrary axial orientation) ● Triangular (defined axial orientation) ● Rectangular (additionally defined workpiece coordination system) Strategy of triangular types is common for numerous workpiece contours: ● Orientation is given, tipping is executed in one step; object pivots around one corner	  Parts T and P of joining motion with local coordinate systems  • *Red dot*: Corner which enters first • *Red dotted line*: edge which follows in tipping
gear, *search*: toothed gear joining motion and *screw-in* motion	● Meshing in toothed gears: Torque oscillations about gear axis and linear forward motion ● Screw-in motion: fixed torque for torque-based screwing ● Angle-controlled tightening required in most screwing process applications: command fixed torque, then turn by a defined angular increment.	

for Assembly (DFA), workcell and assembly line design as well as logistics and manufacturing organization [54.12]. Early on, industrial robots were used in assembly automation, particularly in high-throughput manufacturing lines (Fig. 54.4). However, robots are increasingly used in highly flexible workcells and will enter agile lean manufacturing workplaces as versatile tools at the hands of the human worker. In the following, selective use cases of robots in assembly will be described by detailing on specific enabling technologies.

Assembly of Limb Material

Numerous assembly processes include handling of limb materials such as rubber hoses, wire harnesses, etc. that have to be fixed in position in order to be joined (Fig. 54.15). Obviously stabilizing the material and securing process quality often result in ingenious grip-

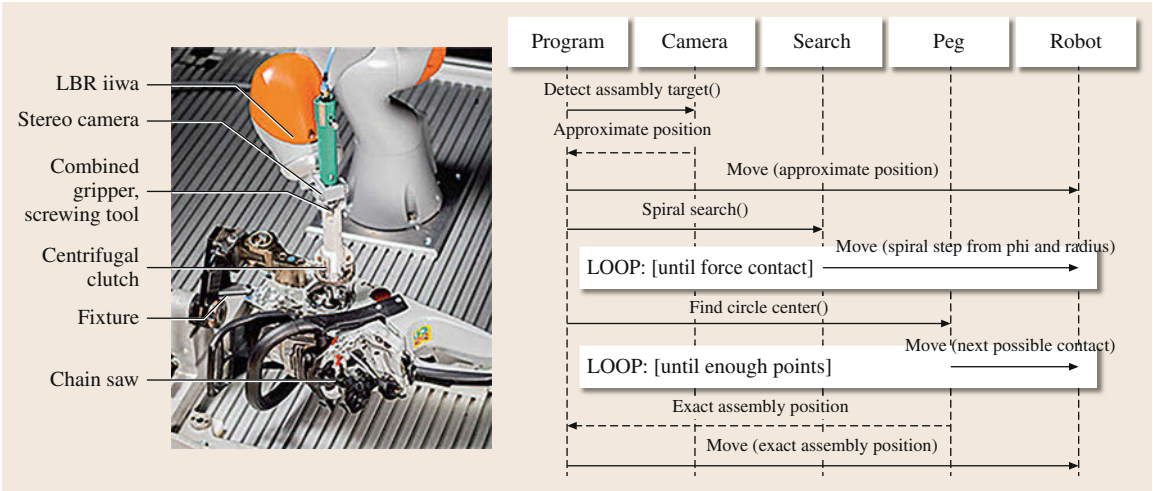


Fig. 54.17 Set up and implementation of a centrifugal clutch assembly for a chain saw with a sequence diagram depicting the consecutive steps until tightening the clutch. Through the robot's torque sensors in each of its links and an appropriate kinematic and dynamic model, the resulting forces at the tool tip are controlled

per designs involving additional actuation and sensing functionalities.

An example is the automatic application of self-adhesive seals as they easily lose their shape and can be stretched or compressed. Since manual application of adhesive seals to vehicle bodies or doors is sensitive and ergonomically problematic a robot-guided tool has to secure bonding of the material's surface to the car body. The seal material is fed from a roll under correct tension and the tape, which covers the adhesive, is removed and stored in a small tank. At the tip of the tool a laser sensor follows the car body or door contour and an actuated roller produces a continuous normal force on the seal. Both the laser sensor and roller's motion are translated into a tension free motion of the robot. In addition, a magazine on a flange ensures that the seal is correctly tensioned and a material reserve for one car door is provided [54.47].

Here, the robot acts as slave to guide a tool which acts as both measuring unit and precision actuator with own master controller. Further efforts aim at embedding rich sensor and control modalities in the robot to account for complex process control based on tactile and geometric information.

Advanced Robot-Based Assembly Process Control

Automation of advanced assembly processes depends on physical contact between the joined workpieces. For this contact formation to be controlled a robot should offer compliant motion control which is a control method that modulates robot position and velocity based on measured or estimated joint torques or contact forces [54.48]. Subject of intense research efforts for a long time application packages for compliant force control in industrial robots which fulfill requirements regarding versatility, robustness, and ease of use in programming have become available during the last ten years [54.49]. The solutions are commonly based on a 6-DOF force–torque sensor which is attached to the robot flange.

The fully torque-controlled DLR (Deutsches Zentrum für Luft- und Raumfahrt) lightweight robot broke new grounds as its 7-DOF redundant kinematic structure, torque sensing in every joint and a variety of compliance modes allowed difficult assembly tasks with complex contact formation during the joining process [54.15]. DLR and KUKA managed to successfully go the strenuous road from the original LBR invention, an idea made manifest in 1991, to a product, first applied in research and predevelopment of new industrial manufacturing concepts in a series of development steps: KUKA LBR3 (2006), LBR4 (2008), and LBR iiwa (2012) [54.25]. Figure 54.16 depicts the integrated mechatronic design of a joint with its unique joint-torque measurement.

To simplify the programming of complex joining processes several assembly subprocess modules were developed of which three are depicted in Table 54.2: the search for contact formation and the execution of two typical joining motions (peg-in-hole, toothed gear joining).

Figure 54.17 depicts a scenario of a force-based centrifugal clutch assembly for a chain saw. The robot

detects and locates the clutch workpieces on a tray by vision sensors. Now the grasped workpiece has to be joined and tightly screwed onto the shaft. In this case the robot's tactile capabilities and compliant behavior are used to achieve a robust and quick assembly. Once a rough shaft center position estimate is acquired by the robot's vision sensor, the robot approaches to this position and starts performing a search pattern until contact detection. The peg-in-hole process is followed by a screw-in motion for tightening the clutch. Alternative designs may be based on 6-DOF industrial robots with a wrist-mounted force–torque sensor to measure process-induced forces on the robot, or simply estimate the forces from the motor torque control [54.50].

The sequence diagram shows a simplified execution trace of a spiral search of a circular shape. Listing 54.1 lists part of the Java code in the KUKA Sunrise control system which identifies the three parts of the compliant assembly: main function, spiral search until collision, and border walk.

Listing 54.1 Extract from the code controlling the sequence of the centrifugal clutch assembly

Main Program:

```
Frame rough = camera.detectAssemblyTarget(); // Retrieve rough position from stereo vision
// Move robot at 30 mm/s. Linear motion. Stop if force > 20 Newton
ForceCondition forceCond = new ForceCondition(robot.getDefaultMotionFrame(), 20);
robot.move(lin(rough).setCartVelocity(30).breakWhen(forceCond)); // First approximation
Frame target = spiralSearch(rough,forceCond); // Obtain the assembly target
assemble(Target); // Assembles the shaft at the given position
```

Search Module. Returns the assembly target:

```
Frame spiralSearch(Frame rough, ForceCondition forceCond){ // Spiral search until collision
 for (double phi = 0; phi < 10.0 * pi; phi +=pi / 90.0){ // 5 loops, 2° step
  Frame spiral = rough.copy();
  Double radius = max_radius * phi / (10.0 * pi);
  spiral.setX(rough.getX + Math.cos(phi) * radius);
  spiral.setY(rough.getY + Math.sin(phi) * radius);
  robot.move(lin(spiral).setCartVelocity(30).breakWhen(forceCond));
  if (motion.getFiredBreakConditionInfo())return findCircleCenter(); // It collides
 }
 return null; // Target not found}
```

Border walk to find shaft center:

```
Frame findCircleCenter()\textbraceleft // Calculates the center of a circular shape
 ArrayList contact = new ArrayList<Frame> (); // Stores frames where collisions take place
 do{
   Frame next = getNextFrame(robot.getForce());
   // Force feedback combined with search pattern such as a grid produces next frame

   robot.move(lin(next).breakWhen(forceCond)); // Move to next position, if possible
   if (motion.getFiredBreakConditionInfo()) contact.add(robot.getDefaultMotionFrame());
  } while (!areEnoughContacts(contact)); // Until enough contact points to identify center
  return getCircleCenter(contact); // Circle center can be determined with three points
}
```

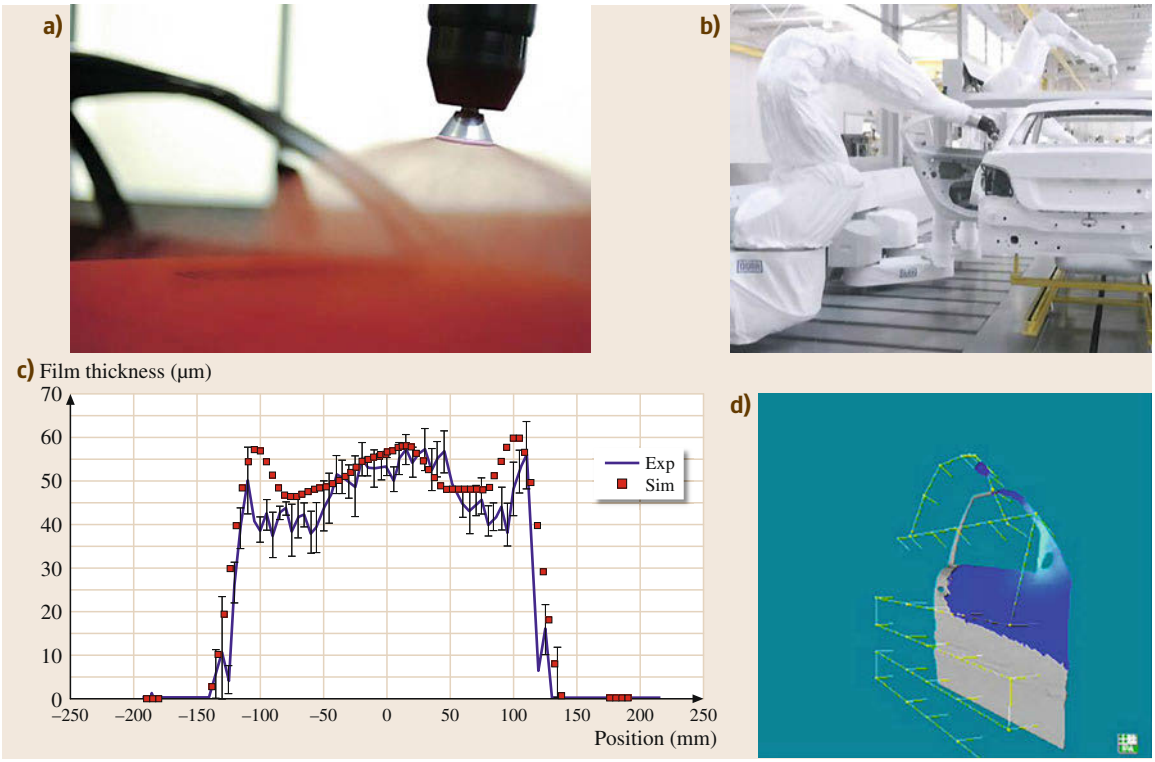


Fig.54.18a–d High-speed rotating atomizer and a multirobot workcell for car body painting. **(a)** A Dürr EcoBell2 paint gun which atomizes the paint material at the edge of the rotating bell disk by centrifugal forces. All current paint materials such as solvent-based or water-borne paints can be used in car production. **(b)** Multiple robots work in parallel for optimal throughput and accessibility of the car body. Most of the programming which includes the synchronization of the robots is performed offline, also with respect to optimal coverage of the painted surface. **(c)** Simulation of the painting process is critical for achieving highest yields (e.g., minimal overspray, uniform paint deposition, etc.) **(d)** Results from simulations for optimized program generation are shown (courtesy of Dürr, Fraunhofer IPA, Stuttgart)

54.4.4 Painting

Hazardous working conditions for human operators motivated Trallfa, a Norwegian company, to develop simple spray-painting robots in 1969, particularly for spraying bumpers and other plastic parts in the automotive industry. Initially pneumatically driven for anti-explosion reasons, today's robot designs are fully electric. They also have hooks and grippers to open hoods and doors during the painting task. Hollow wrists that house gas and paint hoses allow fast and agile motions. Spray guns for robots have evolved dramatically for delivering uniform quality using as little paint and solvent as possible and also for switching between different paint colors. Originally spray-painting robots replicated movements copied from human workers. Most of the programming for robot painting today is done offline as state-of-the-art programming systems offer integrated process simulations to optimize paint deposition, thickness, and coverage (Fig. 54.18). The simulation of the process is quite complex as different effects are taken into account such as turbulent flow field between atomizer and target, static electrical field between rotating bell disk and target, charging of the paint droplets at the bell, space charge effects due to the flow of charged paint droplets, and Coulomb forces acting on the droplets [54.52].

54.4.5 Processing

Material removal processes such as grinding, deburring, milling, and drilling are increasingly carried out by industrial robots with serial kinematics as they combine dexterity, versatility, and cost-effectiveness. The employed process tools are often combined with passive compliance or active force control as the workpiece geometry commonly exhibits tolerances in geometrical or material properties [54.53]. However, robot accuracy (± 0.5 mm range) compares poorly to values in the ± 0.01 mm range of typical machine

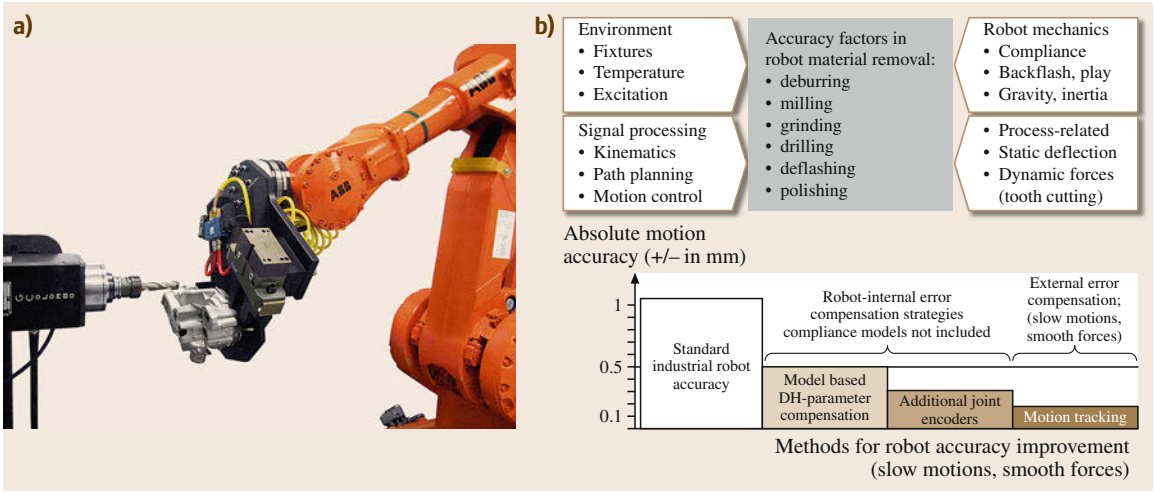


Fig.54.19a–d For small-to-medium-sized parts, a preferred configuration is to have the part grasped and guided by the robot relative to the fixed spindle. **(a)** The robot's gripper is mounted on a force–torque sensor to measure and limit process forces. For machining the edges of the workpiece (deflashing) the machining software package provides a machine-learning technique for optimising the motions [54.51]. **(b)** Influences on accuracy in robotic machining and methods for robot accuracy improvement. Still typical machine tool motion accuracies are in the order of ± 0.01 mm or better (courtesy of ABB, Fraunhofer IPA)

tools [54.54]. The lower eigenfrequencies and damping coefficients of mechanical structures should be as high as possible for precision: The lower eigenfrequencies of milling machines are in the range of 50–100 Hz as compared to 20–30 Hz for typical industrial robots [54.55, 56]. Figure 54.19 depicts the factors that affect the robot's accuracy in typical material removal applications. Figure 54.20 shows first eigenfrequencies, damping, and stiffness in Cartesian space. These characteristics are essential for the design of machining processes and resulting workpiece quality. More robot-guided processes such as laser welding and laser cutting depend on achieving similar accuracies.

Robot position accuracy results from geometric error sources (deviations between actual robot structure and assumed Denavit–Hartenberg (DH) parameters) and nongeometric parameters (compliance of the me-

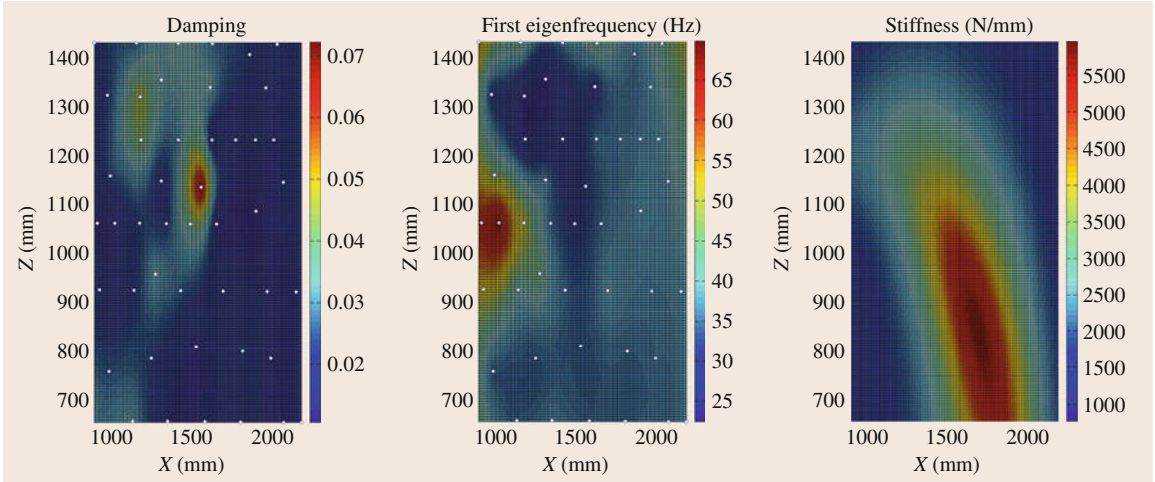


Fig.54.20a–c Robot dynamics in Cartesian space: **(a)** Damping and **(b)** first eigenfrequencies for a KUKA KR60 and **(c)** stiffness for a KR125 in a typical XZ-process plane (measured from the robot's first axis) of the robot's workspace (courtesy of ISG Stuttgart, Fraunhofer IPA)

Table 54.3 Characteristics of computerized numerical control (CNC) and robot controller (RC)

Category	CNC	RC	Interpretation
Targeted application	Machining, material removal	Handling, assembly	A CNC machine tool is single purpose; generally robots are universal machines
Motions	Path-based, complex contour-oriented	Point- or path-based and motion time-oriented	Extended look-ahead of CNC controllers allows detailed path description and adaption on $\gg 100$ via points (< 10 via points in most robot controllers)
Programming	On-site programming based on standardized programming language (G-code), ISO (International Organization for Standardization) 6983, use of computer aided manufacturing (CAM) tools	On-site teaching (teach pendant, editor) based on supplier specific languages, use of typical robot simulation environments	Whereas robots are traditionally programmed manually on site, CNC controllers use CAM technology to generate complex paths automatically based on CAD data
Command reading	Online interpreter, continuous loading of instruction	Initial loading of programs, which are usually interpreted, sometimes compiled)	Program size in robot controllers limited by memory, CNC interprets programs online, may execute an unlimited number of commands

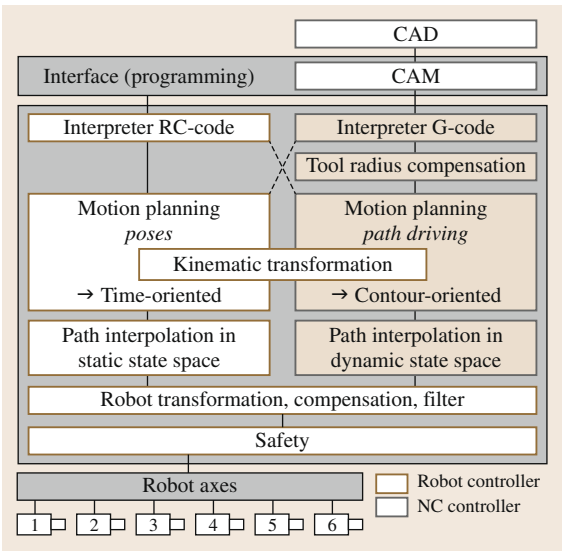


Fig. 54.21 Structure of an NC-kernel integrated into a robot controller (courtesy of ISG Stuttgart)

chanical structure and gear play). In order to reduce the impact of the nongeometric parameters several stepwise approaches can be implemented:

- In combination with force prediction or online measurements robot compliance can be compensated by means of joint stiffness models [54.57]. In drilling applications additional encoders are sometimes mounted on the arm side of robotic joints in order to measure gear-induced joint compliance and backlash, or those effects can be estimated based on determined joint properties [54.58], so that a compensation can be achieved.
- In combination with the geometric error calibration accuracies of better than ± 0.2 mm for typical robots have been realized in larger robot workspaces of $3 \times 3 \times 2\,\mathrm{m}^3$ [54.59].
- For higher accuracies further sensor and actuators systems have been introduced. Using optical tracking deviations of ± 0.2 mm have been demonstrated in steel [54.60]. With additional actuation deviations could be reduced to ± 0.1 mm [54.61].

Robot and CNC machine tool controls may have similar origins, but have taken different paths in developments over the years as depicted in the following Table 54.3.

Increasingly CNC controllers are used for robots in material removal applications for taking advantage of the well-established off-line programming tools in the CNC world and for improving motion accuracy of the robot for complex 3-D contours (Figs. 54.21 and 54.22. Modern robot controllers integrate so-called numerical control (NC) kernels which share components of the robot controller such as user interfaces, kinematic transformations, and safety functions.

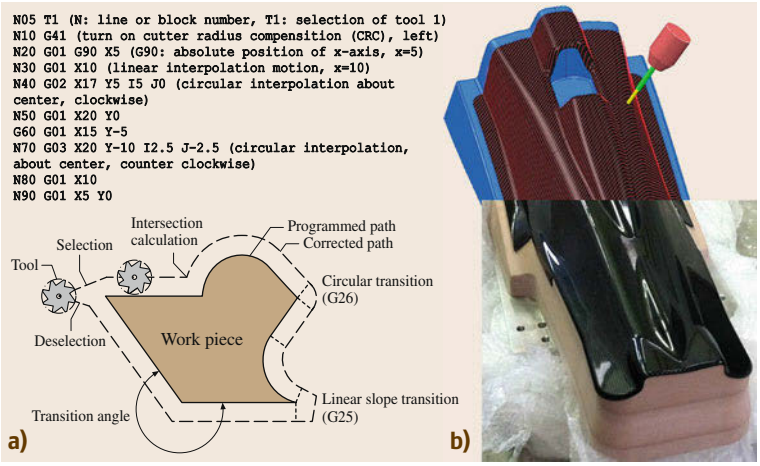


Fig. 54.22 (a) A simple CNC program (G-code, ISO 6983). A CNC program normally is machine independent and orients itself on the workpiece contour after processing. (b) Most CNC programs are generated automatically through CAM tools that transfer geometric information into executable G-code. The example shows generated tool trajectory generation for precision milling of a carbon fiber (CF) part and the finished workpiece (courtesy of Delcam, Birmingham; ISG, Stuttgart)

54.5 Safe Human–Robot Collaboration

Human–robot collaboration allows the combination of typical strengths of robots with some of the numerous strengths of humans. Typical strengths of industrial robots are high stamina, high payload capacity, precision, and repeatability. Strengths of human workers that are unmatched by any machinery comprise flexibility for new production tasks, creative problem-solving skills, and the ability to react to unforeseen situations.

However, industrial robots have a significant potential to harm humans. Therefore, standards for designing and operating industrial robot automation systems have been introduced and found international acceptance. Since 1999, efforts have been made to define measures, rules, and examples specifically for robots in collaborative modes.

54.5.1 Overview of Basic Robot Safety Standards

Machine safety standards provide guidelines for designing and operating any type of machinery. Their consideration is optional, but exhibits the straightforward way of verifying the fulfillment of the fundamental health and safety prerequisites of the Machinery Directive [54.63, 64]. Generally, safety standards are classified into three categories (Fig. 54.23):

- Type A standards: basic standards that define fundamental principles to achieve machine safety.
- Type B standards: generic standards that regulate either a specific safety aspect, e.g., separating dis-

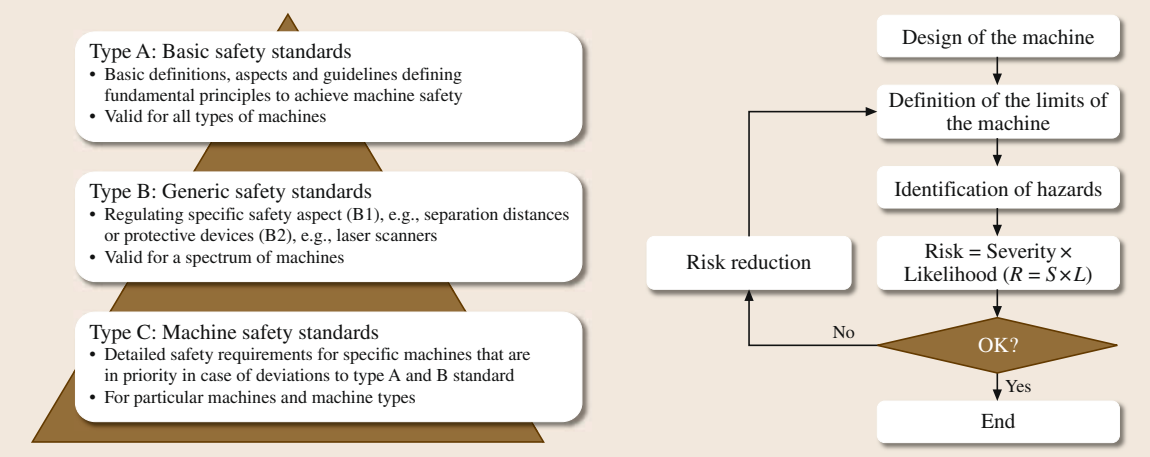


Fig. 54.23 Different types of standards and a simplified iterative risk assessment procedure. See ISO 12100 [54.62, Fig. 1 and 2] for a more detailed schematic representation of risk reduction process

tances, or specific protective devices, e.g., optoelectronic protective equipment like laser scanners or emergency stop (ISO 13850), and are valid for a spectrum of machines.
- Type C standards: machine standards that list detailed safety requirements for a specific type of machine, e.g., industrial robots.

For robotic safety the following standards are of highest importance:

- ISO 12100 [54.62]: Type A standard listing general principles of machine safety, in particular the risk assessment process.
- ISO 13849 [54.65] and IEC 62061 [54.66, 67]: Type B safety standards governing the design of control systems with safety functions.
- ISO 13855 and ISO 13857 [54.68, 69]: Safety distances for separating and nonseparating safety equipment, e.g., fences, light curtains, and laser scanners.
- ISO 10218-1/-2 [54.5, 6]: Type C standards specifically covering safety of industrial robots and robot system integration.

For the setup of any robot installation, an iterative risk assessment process as defined by ISO 12100 has to be conducted. Its workflow as depicted in Fig. 54.23 starts with a functional and geometric design of the machine followed by the definition of the machine limits including spatial boundaries and boundaries regarding usage, e.g., typical tasks, operator qualification, and environmental conditions. The process continues with an identification of the tasks which are consecutively assessed on its risks. Methods for risk estimation are, for example, risk trees as presented in Fig. 54.24. Thereby, any individual hazard is rated on its risk level by quantitatively considering the potential injury severity, the exposition to the hazard, and the possibility to avoid it. The risk estimation leads to a decision if all hazards have been adequately addressed in the machine design. If this is not the case, the machine design is modified to reduce the specific risk and the risk assessment process is repeated.

According to ISO 12100, measures to reduce risks need to be carried out according to the following priorities:

- Risk reduction through inherently safe design
- Risk reduction through safeguards and protective devices
- Risk reduction through information for use (e.g., work instructions, instructions to wear protective equipment).

The central safety standard family for robot safety is ISO 10218. Part 1 addresses the safety requirements of the robots, complemented by part 2, which focuses on the robot system, i. e., the complete integrated machine performing a production task. The second revision of part 1 and the first revision of part 2 were released in 2011 and replace any former robot safety standard, such as EN 775. These standards for the first time define *human–robot-collaboration* as a specific form of a robotic application in an industrial setting and provide guidelines for setting up such collaborative robot systems.

54.5.2 Types and Requirements of Human–Robot Collaboration

ISO 10218-1:2011 contains specific requirements on human–robot-collaboration. It defines four types of collaborative operations in which a human can collaborate with a robot in automatic mode, as depicted in Fig. 54.25. All controller functions for safe human–

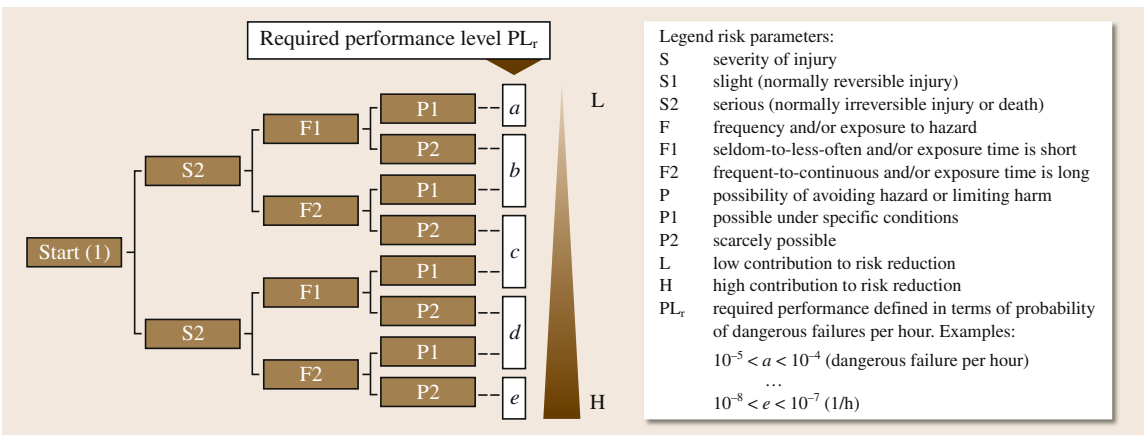


Fig. 54.24 Risk graph for determining required PL_r for safety function according to EN ISO 13849-1 (after [54.65])

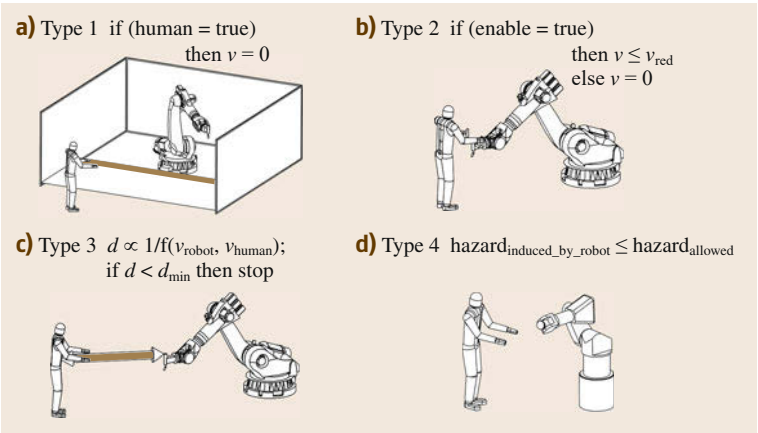


Fig.54.25a–d Modes of human–robot collaboration according to ISO 10218. **(a)** Stop on access with automatic task resumption, **(b)** hand-guiding **(c)** separation and speed reduction, **(d)** monitoring and power and force limiting

robot collaboration have to comply with performance level $PL_{\hat{d}}$ with category 3 structures (ISO 13849-1) or SIL 2 (IEC 62061) if the safety assessment does not yield a differing requirement:

- Safety-rated monitored stop (Fig. 54.25 Type 1): The robot is stopped upon access of the human to the collaborative workspace with the robot drives still in control. A so-called safety controller, now offered by most robot manufacturers, assures the standstill of the robot. The robot task can be automatically resumed as soon as the human operator has left the collaborative workspace. Human and robot share the same workspace, but the robot does not move while the human is present.
- Hand-guiding (Fig. 54.25 Type 2): This type of operation implies a direct physical interaction between human and robot with full control of the human over the robot movement. The human guides the robot through a direct input device (e.g., a handle) at or near the end-effector while activating a three-position enabling device (three-position hold-to run device). Thereby the position of the worker within the collaborative workspace is defined. A safety controller for delimiting the robot speed to a specified threshold is required. Hand-guiding in combination with graphic support through icons or 3-D simulation is in particular suitable for intuitive programming of industrial robots [54.70] during automatic mode of the robot.
- Speed and separation monitoring (Type 3): The relative speed between robot and human as well as their distance are actively monitored. If the human is present, the robot has to maintain a safe combination of speed and distance to the human to be able to stop any hazardous motion before a contact with the human may occur. Again safety controllers in conjunction with safe surveillance sensors (including safe sensor data processing) are needed to supervise speed and position of the robot for human

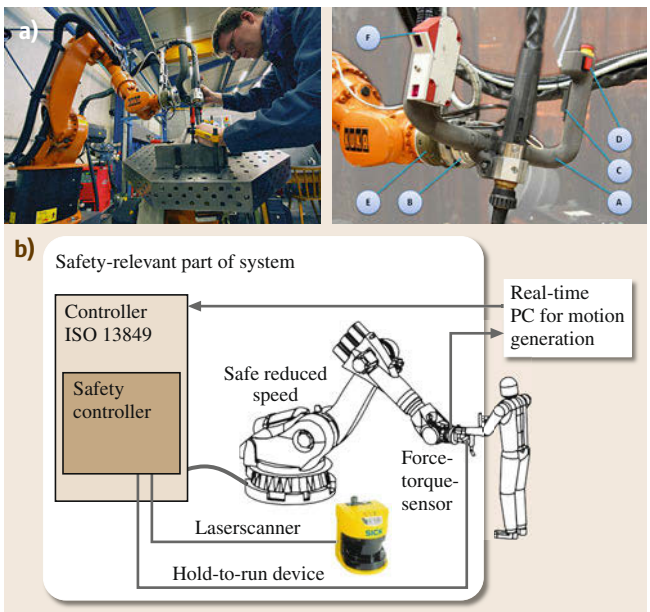


Fig.54.26a,b Intuitive instruction of a welding robot by lead-through programming: **(a)** the worker guides the robot by a flange-mounted (B) handle (A) while pressing safety switches (three-position hold-to run button) (C) on either side. The force exerted on the handle is measured by a force–torque sensor (E) and translated into robot motion. The seam tracking sensor (F) simultaneously measures the workpiece contour for precisely localizing and extracting weld seams. The recorded robot motion is visualized and edited in a simulation environment before task execution. **(b)** The safety relevant part of the system contains a three-position hold-to-run device, a laser scanner for safety monitoring during automatic program execution and a speed monitoring function during hand-guiding is an operational feature which does not have to be realized with a specific safety integrity level (courtesy of Fraunhofer IPA)

motion capturing. Currently, relatively few sensors in industrial automation offer this capability with the required safety integrity level. However, such systems will be available in the future, then being able to feature novel safety strategies for dynamic distance control [54.71].

- Power and force limiting (Type 4): The mechanical hazard potential of the robot is sufficiently reduced to allow a direct, physical interaction of human and robot without an additional safety controller. This is achieved by appropriately limiting collision forces through the design of the robot system such that in the event of a contact between the human and a robot biomechanical tolerance limits are not exceeded.

The ISO 10218 standards in their latest revision as of 2011 explicitly demand that speeds, distances, powers and forces are to be sufficiently limited, but does not give precise threshold values for these limitations. These threshold values need to be determined through risk assessment for a specific application that is foreseen with the robot system. Currently, the technical committee ISO/TC184/SC2 *Robots and robotic devices* develops a technical specification ISO/TS 15066 (in *committee draft* (CD) status of December 2015). It has been drafted to offer more guidance on the risk assessment for collaborative robots and will be released in the near future [54.72]. The procedure to achieve safety outlined in TS 15066 is new for machine safety (not only for robot safety) and is expected to make its way into further standards, especially into the second revision of ISO 10218-2.

For the first time, ISO/TS 15066 introduces tolerance values for the physical strain of the human body. These strain thresholds include maximum forces and maximum pressures for different body parts that can be sustained without suffering from pain or even injury. The risk assessment process involves determining body regions with risk of contact depending on the specific applications and workflows. Based on this information, it can be proved experimentally or in simulation that the given severity threshold values are not exceeded due to limited mechanical or robot control parameters [54.73, 74]. The technical specification aims at transforming highly complex biomechanical injury thresholds into controllable robot performance limits. One of the principles used for dynamic collision analysis is based on the fundamental robot dynamics theory which enables the representation of the reflected inertia of a robot system at any point along the robot structure. Drop-test results from medical injury assessment have been made available for designing robot controllers [54.75].

Part 2 of the ISO 10218 standard provides guidelines regarding the aspects that need to be taken into account during robot workcell setup and system integration. It states the need not only to assess the robot itself, but to take into account the complete application, particularly the end-effector, process, environment, and typical work tasks. This is of special importance for collaborative human–robot operation. As the application always needs to be taken into account, it is not possible to design a *safe robot*, but only to provide robots that are equipped with safety features for setting up a safe collaborative operation.

54.5.3 Examples of Human–Robot Collaboration

In the following, two examples for human–robot collaboration are given.

Intuitive Instruction of a Welding Robot by Lead-Through Programming

Figure 54.26 presents a robot welding workcell which is designed for small batch size production. An intuitive teaching process implementing *hand-guiding* according to ISO 10218 (Type 2 in Fig. 54.25) significantly speeds up robot programming time. The presence of the human during the collaborative operation is monitored via a laser-scanner activating safety zones and safe speed reduction of the robot safety controller during the hand-guiding operation. The recorded motion can be executed automatically after switchover as soon as the human is out of the collaborative workspace.

Collaborative Assembly of Battery-Cases

In this case, sensitive tasks are carried out by the human, while strenuous tasks are executed automatically by a small payload robot (Fig. 54.27 for the cell design). Thereby, the safety concept comprises the requirements for *speed and separation monitoring* (Type 1 in Fig. 54.25) from ISO 10218 and involves two light curtains with a signal processing on a safety PLC and a robot with a safety controller. Switchover between the safety zones is activated upon detection of human presence in a collaborative workplace area. Each safety zone statically monitors the minimum possible distance to the robot's active workspace thereof deriving the maximum possible robot speed. Such a system reduces space requirements for a robot installation while enhancing the flexibility of the assembly process due to the collaborative nature of this assembly process.

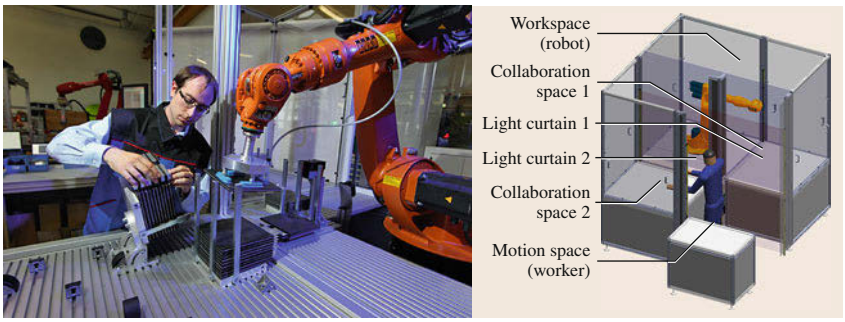


Fig. 54.27 The robot's workcell is divided into three segments by two light curtains which trigger in which area the robot works at lower velocity when collaborating with the worker in the collaboration spaces

54.6 Task Descriptions – Teaching and Programming

Although desirable, a robot cannot be instructed in the same way that one would instruct a skilled human worker how to carry out a task. With skilled human workers knowing the applications, devices, processes, and the general requirements on the product to be manufactured, we would only need to summarize what needs to be done. Humans have, with or without awareness thereof, extensive knowledge about motions, physical effects, cause–effect relationships, learned procedures, etc., and they maintain such knowledge for their own good. A human is skilled because physical capabilities are combined with reuse of that learned/maintained knowledge, and the purpose of the manufacturing operation can be explained and understood.

Robots are not capable of performing such knowledge-based behaviors in a productive manner. Instead, instructions have to be quite explicit and motion oriented, and even motion planning is rarely used since it is difficult to encode much of the required background knowledge. We could aim for programming principles that resemble instructing a (totally) unskilled worker, telling precisely how every aspect of the task is to be performed. Such an explicit way of instructing the robot then should be human friendly, but the performance requirements motivated by productivity needs imply that methods for defining the task need to reflect machine/robot properties considering product data and production processes. Existing approaches include:

- Manually guiding the robot to the positions of interest, or even along the desired paths or trajectories if human accuracy is enough (Fig. 54.26).
- Having simple ways to make use of CAD data whenever available (Fig. 54.13).
- Using different complementary modalities (paths of communication between the human and the robot), such as pointing devices and 3-D graphics.
- Choreographing the task movements, for instance loops and conditions, without requiring extensive programming competencies.
- Means of describing acceptable variation, e.g., acceptable deviations from the nominal path or POI.
- Specification of how external sensing should be used for path adaptations or for handling unknown variations.

Most of these items still mean that the robot is rather dumb, and the knowledge of solutions is maintained among human experts. Consequently, robots are time-consuming and hard to instruct for productive industrial tasks, and the created solutions are not reusable since the knowledge is not explicitly represented in a way that is useful for the robot. To embody knowledge in the robot system is the way to make task definition more efficient, which can be better understood if we review the origins and the progress so far. Initially, mainly during the 1970s and 1980s, there were some painting robots that could be programmed by manual guidance. This was possible due to the following abilities:

- Applications such as painting permitted the use of a lightweight arm, including the end-effector, possible balanced with respect to gravity if needed.
- The accuracy requirements were (compared to today) modest so it was possible to use back-drivable drive trains and actuators, and the definition of the motions could then be done manually by the operator moving the end-effector along the path. The recorded poses, including the timing/speed information, defined the programmed motion.
- Since no inverse kinematics was needed during programming or real-time operation, it was not a problem from a computing point of view to use arm kinematics without singularities in the workspace.

- The requirements of optimality of painting motions were also modest, compared to recent years when environmental conditions (for nature and workers) called for minimal use of paint.

It is often referred to as an inevitable problem that there are singularities to be handled within the workspace. However, to simplify the kinematics and its inverse from a software point of view (e.g., during the 1980s considering the power of the microprocessors and algorithms available at that time), robots were actually designed to have simple (to compute) inverse kinematics. For instance, the wrist orientation was decoupled from the translation by the arm by using wrist axes that intersected with the arm axes. The resulting singularities within the workspace could be managed by restrictions on wrist orientations, but an unfortunate implication was that robot arms were no longer back-drivable (close to the singularities) when designs were (due to engineering and repeatability requirements) adopted to standard industrial controllers. Then with the development of microprocessor-based industrial controllers and the definition of motions based on jogging (manual moves by for instance using a joystick) and CAD data, the means of robot programming became closer to computer programming (extended with motion primitives).

Robot programming languages and environments have traditionally been separated into online programming (using the actual robot in situ) and OLP (using software tools without occupying the robot). With the increasing power of OLP tools, their emerging ability to connect to the physical robot, and the increasing level of software functions that are embedded into the robot control system, online programming is now unusual, except to verify and manually adjust programs generated offline. Of course, it is economically important to minimize downtime for robot programming, and advanced sensor-based applications may be too hard to develop without access to the true dynamics of the physical workcell for fine-tuning. Nevertheless, robot languages and software tools must provide for both methods of programming.

Even though robot languages from different manufacturers look similar, there are semantic differences that have to do with both the meaning when programs are running (the robot performing its operations) and the way the robot is instructed. The need to ensure that existing robot programs can operate with replacement robots and controllers, and also to make use of existing knowledge in robot programmers and incorporated into OLP software, requires manufacturers to continue supporting their original proprietary languages. Features such as backward execution (at least of motion statements) and interactive editing during interpretation by the robot (in combination with restrictions to make the programming simpler) also make robot languages different from conventional computer programming languages.

During the last decade and in current developments, the trend has been to focus on the tool (robot end-effector) and on the process knowledge needed for the manufacturing process, and to let the operator express the robot task in such terms. This development results in a need for an increased level of abstraction to simplify the programming, reflecting the fact that the so-called robot programmer knows the production process very well, but has quite limited programming skills. To understand why such a high level of abstraction has not come into widespread usage, we may compare this with the early days of industrial robotics when there was no kinematics software built into the controllers, and hence the robots were programmed via joint-space motions. (Kinematics here deals with the relation between robot motors/joints and the end-effector motions.)

Built-in inverse kinematics permits tool motion to be specified in Cartesian coordinates, which is clearly a great simplification in many applications. That is, the robot user could focus more on the work to be carried out by the robot and less on the robot itself. However, robot properties such as joint limits cannot be neglected. Until the beginning of the 1990s, robots did not follow programmed trajectories very well at high speeds or accelerations, and full accuracy could only be achieved at low speed. Modern robot systems with high-performance model-based motion control perform their tasks with much greater accuracy at high speed due to model-based control features (Chap. 6).

There are increasing opportunities to raise the level of abstraction to simplify the use of robots even more by encoding more knowledge about the robot, tools, the process, and workcells into control systems. This is a gradual development with possibilities depending on the application; when knowledge is not encoded the robot programmer needs to be more aware of robot properties and how to handle them. Reasons for not including such knowledge typically lie in the fact that it is simply too difficult to do so or because the robot programmer does neither have the competence nor the tools nor reasons to do so. With reference to Table 54.4, the following example explains that type of trade-off and what it means in terms of centricity.

A machine part consisting of steel plates and pipes is to be manufactured by means of welding, and there is robot and process equipment available for production of this (and other) types of workpieces:

- A product-centric system would generate configurations and robot programs, and instruct the oper-

Table 54.4 Robot-related centricity, ranging from a high-level view of the work to be accomplished in terms of the product to manufacture, to a low-level robot motion view that, in practice, constraints what manufacturing operations can be performed

User application ↑ / Robot constraint ↓	Product	Description of the final shape and assemblies of the workpieces, in terms of that product; the robot system plans the operations
	Process	Based on known sequences of specific manufacturing operations, each of these is specified in terms of their processes parameters
	Tool	The motion of the robot-held tool is specified in terms of programs or manual guidance; the user knows the process it accomplishes
	Arm	The robot arm and its end-plate for tool mounting is programmed how to move in Cartesian space
	Joint	For each specified location, the joint angles are specified, so straight-line motions are difficult; the robot provides coordinated servo control

ator for whatever manual assistance that might be needed (e.g. clamping and fixturing). The system would determine the welding data such as the type of welding and how many passes for each seam.

- A process-centric system would instead accept input in terms of the welding parameters to use, including the order of the welds. The system would select input signals to the process equipment (such as what output voltage the robot controller should set such that the welding will be done with a certain current), and robot programs specifying the motions of the welding torch would be generated.
- A tool-centric way of programming would require the operator to set up the process manually by configuring the equipment in terms of their native settings, and appropriate tool data would be configured such that the robot controller can accomplish the programmed motions, which are specified by giving coordinates and motions data referring to the end-effector.
- An arm-centric system is similar to the tool-centric approach in that Cartesian and straight-line motions can be (and need to be) explicitly specified/programmed by the programmer, but extra work is needed since the robot does not support a general tool frame.
- A joint-centric style would be needed if one of the very early robot systems is being used, requiring a straight line to be programmed by lots of joint-space poses close to each other.

Thus, the question is, are you programming robot joint servos to make the robot provide a service for you, are you programming/commanding an arm how to move a tool, is it the tool that is made programmable by means of the robot, is it manufacturing services that are ordered by specifying the desired process parameters, or is it an intelligent system that simply can produce your product? Still today in industry, there are reasons to think and program considering all five levels, although the tool-centric alternative is kind of a baseline. Hence tool-data needs to be maintained in the robot program, as in the next example below.

As a final goal related to the product-centric view, so-called task-level programming is desirable. This has been a goal since the 1980s, and implicitly also since the very beginning of robotics. It would mean that the user simply tells the robot what should be done in common terms and the robot would know how to do it, but it would require extensive knowledge about the environment and so-called *machine intelligence*. The need for extensive modeling of the environment of the robot is well known. Sensing of the environment is costly, but in an industrial environment it should only be needed occasionally. Modeling has to encompass full component dynamics and the limitations of the manufacturing process. With these difficulties, task-level programming is not yet achievable in practice. If we limit our domain, however, such as to machining where motion planning is already used for so-called machine tools, generating the robot programs from product descriptions is feasible since the process (with tolerances due to robot compliance) is modeled.

As indicated in the application examples, it is now common practice to generate robot programs from geometric data in CAD files. That is, the CAD application could be the environment used for specifying how the robot should perform the required operations on the specified parts, which is the CAM part of CAD/CAM (Computer Aided Manufacturing, often in term of a back-end to the CAD system, and hence usually not explicitly referred to). CAM is not quite task-level programming since human operators do the overall planning. CAD software packages are powerful 3-D tools that are very common among manufacturing companies. Consequently, using these tools for robot programming is desirable since the operator may start the OLP of the necessary manufacturing operations us-

ing the 3-D model of the product, even before selecting a robot.

There is an ongoing competition along the so-called vertical integration from product data down to machine operations, including at least the following three approaches:

- CAD providers increasingly offer advanced robot-aware functions, thereby expanding their field of service and expertise from a higher (CAD) to a lower (robot) level. An example of a robot-aware CAM system is depicted in Fig. 54.22.
- Robot providers have developed programming tools with CAD data import and various motions planning modules, and thus, expanded their offerings to higher levels. One such software is shown in Fig. 54.29.
- A further development is to include a CNC kernel into the robot controller, as suggested in Fig. 54.21, making the robot behave as a CNC machine.

These approaches as such are more complementary than competing, with the competing stakeholders presently exploring what benefits can be gained for customers and business in each type of application.

Listing 54.2 Example pick-and-place operation in the ABB robot programming language Rapid

```
PROC PickInPallet()
 MoveJ Offs(pPickInPallet,-500,0,500),v2000,z100,tGripper\WObj:=wobjPalletStatic;
 MoveL RelTool(pPickInPallet,0,0,-100),v2000,z100,tGripper\WObj:=wobjPallet;
 MoveL RelTool(pPickInPallet,0,0,0),v100,fine,tGripper\WObj:=wobjPallet;
 Grip;
 MoveL RelTool(pPickInPallet,0,0,-100),v500,z10,tGripper\WObj:=wobjPallet;
 MoveJDO Offs(pPickInPallet,-500,0,500),v2000,z100,tGripper\WObj:=wobjPalletStatic,
 doNewObject,1;
ENDPROC

PROC PlaceAtOutFeeder()
 MoveJ Offs(pDropOutFeeder,0,0,200),v2000,z10,tGripper\WObj:=wobjOutFeeder;
 WaitDI diOutFeederReady,1;
 MoveL Offs(pDropOutFeeder,0,0,0),v200,fine,tGripper\WObj:=wobjOutFeeder;
 Release;
 MoveLDO Offs(pDropOutFeeder,0,0,200),v500,z10,
 tGripper\WObj:=wobjOutFeeder,doStartOutFeeder,1;
 MoveL pFromOutFeeder,v2000,z10,tGripper\WObj:=wobjMachine;
ENDPROC

PROC Grip()
 SetDO doGrip,1;
 WaitDI diGripped,1;
ENDPROC
PROC Release()
 SetDO doGrip,0;
 WaitDI diGripped,0;
ENDPROC
```

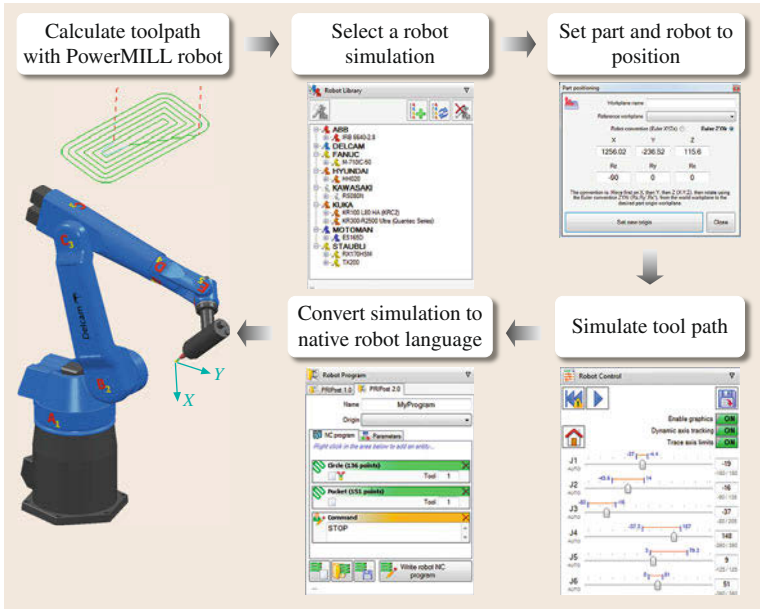


Fig. 54.28 Workflow according to the PowerMill software from Delcam with its robot module. Note that step 4 includes also simulation of robot properties such as closeness to singularities

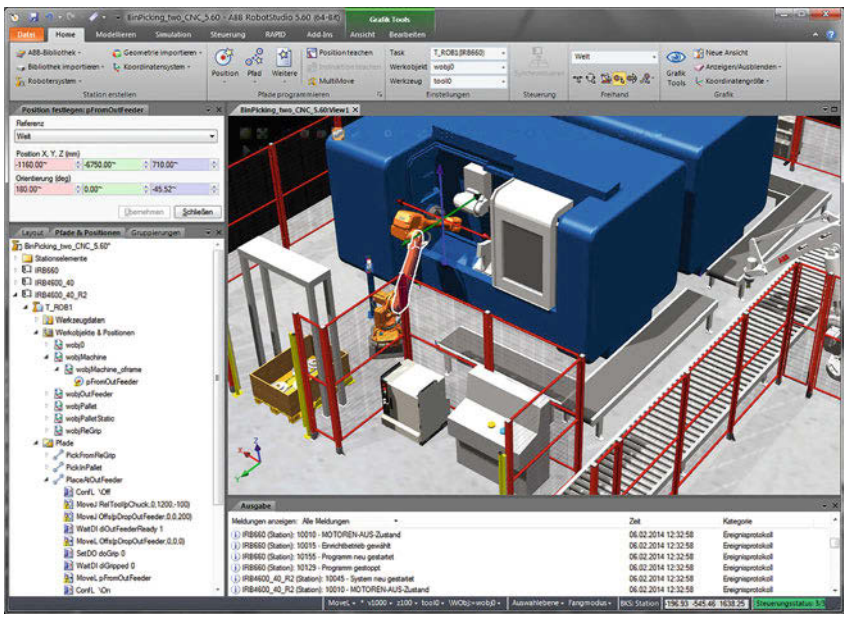


Fig. 54.29 Robot programming environment, with a virtual controller (console to the lower right) for each robot including the model-based embedded control software compiled for the personal computer (PC). To the left there are the instances of all objects depicted as a tree structure, ranging from complete machines down to the bits of the I/O signals

At step 5 in Fig. 54.28, the motion specification is generated and expressed in the native language of the robot. On the other hand, the tool path could be expressed in machine-independent, standardized G-code. A further step would be to integrate a machine tool interface and control into the robot controller which is depicted in Fig. 54.22. What approach to take is an area of current research and development (R&D) and business decisions, but most robots are programmed in their own language only, since the needed system functions as a consequence of Table 54.4.

To exemplify task definitions based on a native robot language, consider the application depicted in Fig. 54.29, which includes bin-picking, handling, machine tending, conveyer tracking, and palletizing. Listing 54.2 lists a sample program with some of the functions from that application programmed in ABB's proprietary robot language Rapid. The program being

tool-centric shows in the tool argument of the Move statements, the *tGripper* that is a reference to the tool data including the tool frame. At the same time, it is arm-centric in the way that the execution of motions is performed per arm (with support for synchronized execution for multiarm robots), as expressed by the *MoveL* statements. Correspondingly, the *MoveJ* performs a joint-space motion.

Note that Grip and Release are in separate procedures where execution waits until a hardware acknowledged signal is obtained. For Move statements L denote Linear, J means Jointspace, functions Offs and RelTool compute target poses based on other determined poses (i. e., frames). The v-constants are velocities in mm s^{-1}, and z-constants (or fine) specify how close to the target pose the motion should be. DI refers to digital input and DO to digital output. The WObj is the work object, i. e., the specification of the base frame of the motions.

When programming motion behaviors like a search strategy, which could be considered being a motion primitive from an application point of view, use of the end-user language of that robot might not be the best option. Instead a computer programming language can be more appropriate as shown in Listing 54.1. In the ABB case, the strategies are implemented as part of an application package as for assembly, and the code is internally written in C (cannot be viewed or changed by the user, but has optimized real-time performance) while the functionality is exposed to the robot user via special statements like FCRefSpiral [54.51] and more to accomplish the principles of Table 54.2. A variety of solutions are available for different robot brands [54.76].

Considering a complete setup with multiple robots, either with a few robots and some peripherals like in Fig. 54.9 or a complete manufacturing line as in Fig. 54.12, there are several robot programs that need to be programmed to work together and hence there is a coordination and complexity problem. A useful approach is to use a service-oriented approach, considering robots as servers providing services according to a set of programs that are exposed on the factory or production-cell network which means programming is done by:

- Building services (using several technologies) which are available, discoverable or not, remotely accessible by the application programmer
- Building applications that coordinate and use those services.

For such system to be programmable, however, the equipment (typically from different suppliers) has to be configured and interconnected according to the sometimes overlooked art of robot system integration.

54.7 System Integration

It is interesting to note that connecting different workcell devices with each other, and integrating them into a working system, is hardly mentioned in the robotics literature. Nevertheless, in actual nontrivial installations, this part typically represents, apart from the cost for peripheral device, roughly half of the overall installation costs. The automation scenario includes all the problems of integrating computers and their peripherals, plus additional issues that have to do with the variety of (electrically and mechanically incompatible) devices and their interaction with the physical environment (including their inaccuracies, tolerances, and unmodeled physical effects such as backlash and friction). The number of variations is enormous so it is often not possible to create reusable solutions. In total, this results in a need for extensive engineering to put a robot to work (Table 54.5). This engineering is what we call system integration.

Carrying out system integration according to current practice is not a scientific problem as such (although how to improve the situation is), but the obstacles it comprises form a barrier to applying advanced sensor-based control for improved flexibility, as

Table 54.5 Stages of system integration, typically carried out in the order listed

Physical	Selecting equipment based on dimensioning for mechanical size, load, and stress
	Mechanical interfacing (locations, adapter plates, etc.)
	Electrical power supply (voltages and currents for robots, effectors, feeders, etc.)
	Connections for analog signals (shielding, scaling, currents, binary levels, etc.)
	Safety design and risk assessment
Communication	Interconnections for single-bit digital I/O
	Byte-wise data communication, including latencies and bit rates
	Transfer of byte sequences
Configuration	Configuration of messages between interacting devices
	Establishment of services
	Tuning for performance and resource utilization
Application	Definition of application-level functions/services
Task	Application programming, using the application-level services

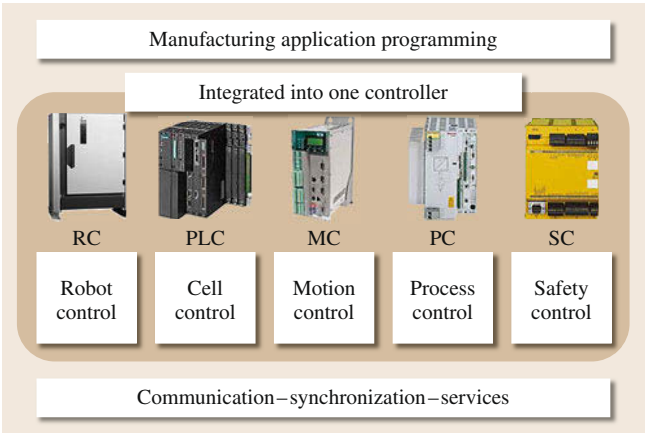


Fig. 54.30 Manufacturing cell with a multicontroller architecture with dedicated controllers (robot motions, production sequences, positioning systems, processes and worker safety). In modern robot controllers, such as the KUKA KR C4, these controllers are replaced by software tasks of a single multifunctional robot controller, thus reducing investment costs on controllers and communication, simplifying engineering, programming and diagnosis, and enhancing process quality and shortening cycle time ►

is needed in small series production. In particular, in future types of applications using external sensing and high-performance feedback control within the workcell, system integration will be an even bigger problem since it includes tuning of the feedback too.

For large series production (e.g. in the automotive industry), the engineering cost of system integration is less of a problem since its cost per manufactured part is small. On the other hand, the trend toward smaller series

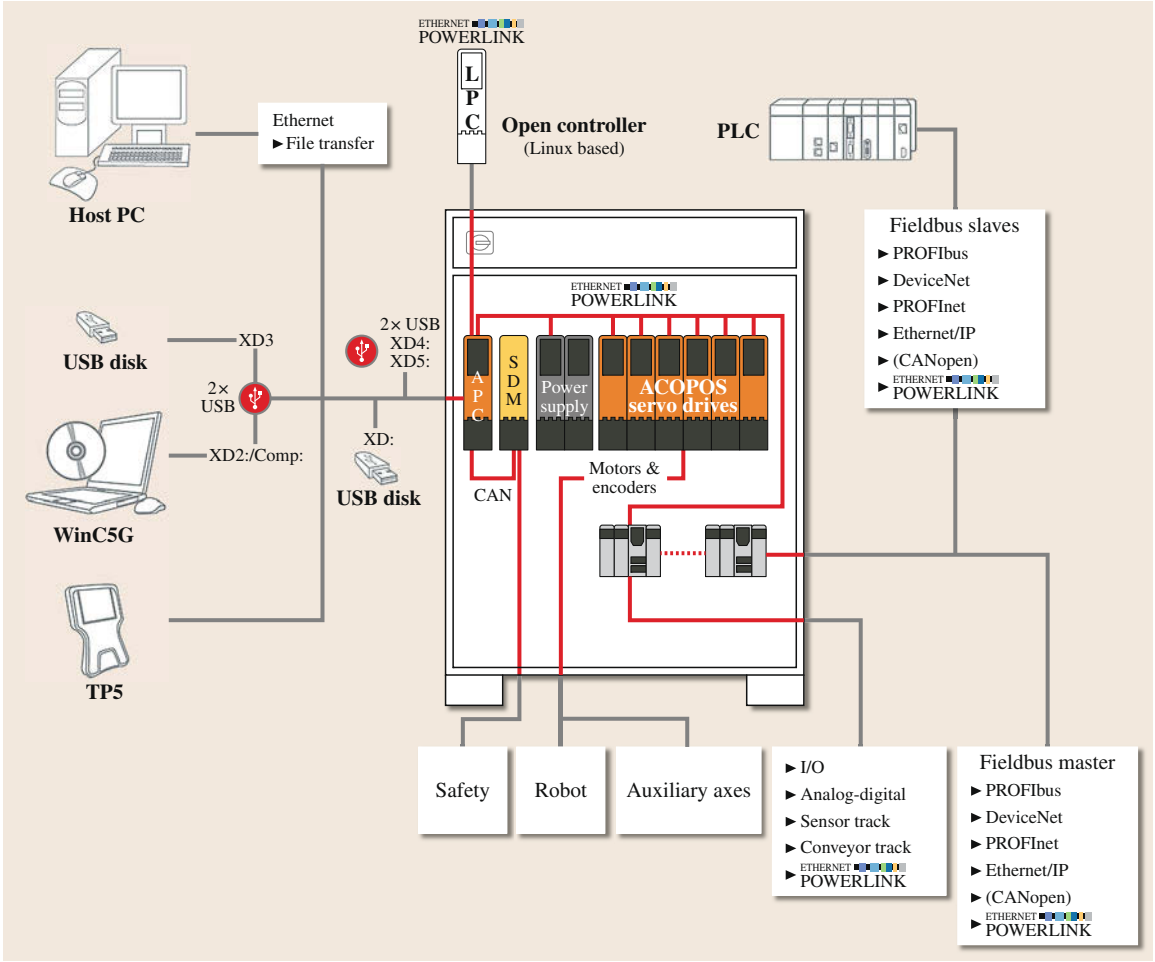


Fig. 54.31 Control modules and interfaces in a typical (but open) robot controller. Several of the named network technologies are based on Ethernet, ranging from normal LAN to different real-time protocols (courtesy of COMAU)

of customized products, or products with many variants that are not kept in stock, calls for high flexibility and short changeover times. Flexibility in this context refers to variable product variants, batch sizes, and process parameters. In particular, this is a problem in small and medium-sized productions, but the trend is similar for larger enterprises as flexibility requirements are continuously on the rise. One may think that simply by using standards for input/output (I/O) and well-defined interfaces, integration should be just a matter of connecting things together and run the system.

Of course, the object-oriented software solution would be to have a gripper class with operations grasp and release. That would be appropriate for a robot simulator in pure software, but for integration of real systems such encapsulation of data (abstract data types) would introduce practical difficulties because:

- Values are explicit and accomplished by external (in this case) hardware, and for testing and debugging we need to access and measure them.
- Online operator interfaces permit direct manipulation of values, including reading of output values.

The use of functions of object methods (so-called set:ers and get:ers) would only complicate the picture; maintaining consistency with the external devices would be no simpler.

Already in small installations, the complexity of system integration becomes apparent. So-called vertical integration means integration of low-level devices with a high-level factory control system, whereas integration of peripherals on (for instance) workcell level is called horizontal integration (Fig. 54.30).

Lack of self-descriptive and self-contained data descriptions that are also useful at the real-time level further increases the integration effort since data interfaces/conversions typically have to be manually written. In some cases, as illustrated in Fig. 54.31, there are powerful software tools available for the integration of the robot user level and the engineering level [54.77]. The fully (on all levels) integrated and nonproprietary system such as the recently discussed Industry4.0 or industrial Internet initiatives is still is a challenge, particularly for small and medium-sized productions [54.24].

54.8 Outlook and Long-Term Challenges

The widespread use of industrial robots in standard, large-scale production such as the automotive industry, where robots perform repetitive tasks in well-known environments, resulted in the common opinion that industrial robotics is a solved problem. This opinion was underpinned by the robot systems' impressive automatic performance, based on advanced semiautomatic programming and resulting in an unbeatable product quality when compared to manual labor. However, large-scale production comprises only a minor part of the work needed on an industrial scale in any wealthy society, especially considering the number of companies and the variety of applications and processes that could and should be automated for productivity, health and sustainability reasons.

Global prosperity and wealth requires resource-efficient and human-assistive robots. The challenges today are to recognize and overcome the barriers that are currently preventing robots from being more widely used, especially in small and medium-sized manufacturing.

Taking a closer look at the scientific and technological barriers, we find the following challenges:

- *Human-friendly task specification*, including intuitive ways of expressing permitted/normal/expected variations and errors. There are numerous upcoming and promising techniques for user-friendly human–robot interaction (such as speech, gestures, manual guidance, and so on), but the focus is still on specification of the nominal task rather than the complete task which is capable of handling foreseen deviations from the nominal case (Part G, Robots and Humans). Taking care of these expected deviations can account for up to 80% of the total programming time.
- *Intuitive human–robot interaction* The foreseen variations, and the unforeseen variations experienced during robot work, are difficult to manage. When instructing a human, he/she has an extensive and typically implicit knowledge about the work and the involved processes. To teach a robot, it is an issue both how to realize what the robot does not know, and how to convey the missing information efficiently. Furthermore, a person should feel comfortable and familiar with the robot's functionality and operation during all operational modes including maintenance and error recovery. Research and development toward industrial robotics usability and ergonomics for increasing acceptance and operator efficiency has been of surprisingly low activity in the past.
- *Efficient mobile manipulation*. Successful implementations and systems are available for both mobility and for manipulation, but accomplished

in different systems and using different types of (typically incompatible) platforms. A first step would be to accomplish mobile manipulation at all, combining all degrees of freedom of both the mobile base and arm(s), intelligently exploiting system redundancies to achieve a given task. A second step providing truly dependable autonomous navigation (with adaptive, but predictable understanding of constraints, such as frequent environmental changes and typical shop-floor dynamics, and appropriate sensor-based reactions) dexterous manipulation with multifingered grippers, and robust force/torque interaction with environments (that have unknown stiffness). As a concluding step, all this needs to be done with decent performance using reasonably priced hardware, and with human–robot interfaces according to the previous items.

- *Low-cost components including low-cost actuation*. Actuation of high-performance robots represent about two-third of the overall robot cost, and improved modularity often results in a higher total hardware cost (due to less opportunities for mechatronic optimization). On the other hand, cost-optimized (with respect to certain applications) systems result in more-specialized components and smaller volumes, with higher costs for small series production of those components. Since future robotics and automation solutions might provide the needed cost efficiency for small series customized components, we can interpret this as a boot-strapping problem, involving both technical and business aspects. The starting point is probably new core components that can fit into many types of systems and applications, calling for more mechatronics research and synergies with other products.
- *Composition of subsystems*. In most successful fields of engineering, the principle of superposition holds, meaning that problems can be divided into subproblems and that the solutions can then be superimposed (added/combined) onto each other such that the total solution comprises a solution to the overall problem. These principles are of key importance in physics and mathematics, and within engineering some examples are solid-state mechanics, thermal dynamics, civil engineering, and electronics. However, there is no such thing for software, and therefore not for mechatronics (which includes software) or robotics (programmable mechatronics) either. Thus, composition of un-encapsulated subsystems is costly in terms of engineering effort. Even worse, the same applies to encapsulated software modules and subsystems. For efficiency, system interconnections should go directly to known (and hopefully standardized) interfaces, to avoid the indirections and extra load (weight, maintenance, etc.) of intermediate adapters (applying to both mechanics for end-effector mounting and to software). Interfaces can be agreed upon, but the development of new versions typically maintains backward compatibility (newer devices can be connected to old controller), while including the reuse of devices calls for mechanisms for forward compatibility (automatic upgrade based on meta information of new interfaces) to cover the case that a device is connected to a robot that is not equipped with all the legacy or vendor-specific code.
- *Embodiment of engineering and research results*. Use or deployment of new technical solutions today still starts from scratch, including analysis, understanding, implementation, testing, and so on. This is the same as for many other technical areas, but the exceptional wide variety of technologies involved with robotics and the need for flexibility and upgrading makes it especially important in this field. Embodiment into components is one approach, but knowledge can be applicable to engineering, deployment, and operation, so the representation and the principle of usage are two important issues. Improved methods are less useful if they are overly domain specific or if engineering is experienced to be significantly more complicated. Software is imperative, as well as platform and context dependent, while know-how is more declarative and symbolic. Thus, there is still a long way to go for efficient robotics engineering and reuse of know-how.
- *Open dependable systems*. Systems need to be open to permit extensions by third parties, since there is no way for system providers to foresee all upcoming needs in a variety of new application areas. On the other hand, systems need to be closed such that the correctness of certain functions can be ensured. Extensive modularization in terms of hardware and supervisory software make systems more expensive and less flexible (contrary to the needs of openness). Highly restrictive frameworks and means of programming will not be accepted for widespread use within short-time-to-market development. Most software modules do not come with formal specification, and there is less understanding of such needs. Thus, systems engineering is a key problem.
- *Sustainable manufacturing*. Manufacturing is about transformation of resources into products, and productivity (low cost and high performance) is a must. One aspect of sustainability deals with energy-efficiency of the devices that are used for production. Saving energy can constrain the path planning of an

individual and a whole fleet of robots, and even shift the operation of energy-consuming tasks in periods of the day where energy cost are low. A second aspect of sustainability deals with recycling of scarce resources in terms of materials and noble earths. In most cases, this can be achieved by crushing the product and sorting the materials, but in some cases disassembly and automatic sorting of specific parts are needed. There is, therefore, a need for robots in recycling and de-manufacturing. Based on future solutions to the above items, this is then a robot application challenge.

An overall issue is how both industry and academia can combine their efforts such that sound business can be combined with scientific research so that future developments overcome the barriers that are formed by the above challenges.

Video-References

VIDEO 260 SMErobotics project video
available from http://handbookofrobotics.org/view-chapter/54/videodetails/260

VIDEO 261 SMErobot video coffee break (English)
available from http://handbookofrobotics.org/view-chapter/54/videodetails/261

VIDEO 262 SMErobot final project video
available from http://handbookofrobotics.org/view-chapter/54/videodetails/262

VIDEO 265 SMErobot – New parallel kinematic with unique concepts for demanding handling and process applications
available from http://handbookofrobotics.org/view-chapter/54/videodetails/265

VIDEO 266 SMErobot D4 *The woodworking assistant*
available from http://handbookofrobotics.org/view-chapter/54/videodetails/266

VIDEO 380 SMErobotics demonstrator D1 assembly with dual-arm industrial manipulators
available from http://handbookofrobotics.org/view-chapter/54/videodetails/380

VIDEO 381 SMErobotics demonstrator D2 human–robot cooperation in wooden house production
available from http://handbookofrobotics.org/view-chapter/54/videodetails/381

VIDEO 382 SMErobotics demonstrator D3 assembly with sensitive compliant robot arms
available from http://handbookofrobotics.org/view-chapter/54/videodetails/382

VIDEO 383 SMErobotics demonstrator D4 welding robot assistant
available from http://handbookofrobotics.org/view-chapter/54/videodetails/383

References

54.1 The International Federation of Robotics (IFR): *World Robotics 2015. Statistics, Market Analysis, Forecasts, Case Studies and Profitability of Robot Investments* (International Federation of Robotics, Frankfurt 2015)

54.2 D. Bourne: My Boss the Robot, Scientific American **308**(5), 37–41 (2013)

54.3 M.P. Groover: *Automation, Production Systems, and Computer-Integrated Manufacturing*, 3rd edn. (Prentice-Hall, Upper Saddle River 2007)

54.4 International Organization for Standardization: Standards catalogue, 25.040.30: Industrial robots. Manipulators, http://www.iso.org/iso/home/store/catalogue_ics/catalogue_ics_browse.htm?ICS1=25&ICS2=40&ICS3=30 (2014)

54.5 International Organization for Standardization: ISO 10218-1:2011. Robots and robotic devices – Safety requirements for industrial robots – Part 1: Robots, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=51330 (2011)

54.6 International Organization for Standardization: ISO 10218-2:2011. Robots and robotic devices – Safety requirements for industrial robots – Part 2: Robot systems and integration, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=41571 (2011)

54.7 L. Westerlund: *The Extended Arm of Man. A History of the Industrial Robot* (Informationsförlaget, Stockholm 2000)

54.8 D. Hunt: *Industrial Robotics Handbook* (Industrial, New York 1983)

54.9 V.D. Scheinman: Design of a Computer Controlled Manipulator, Ph.D. Thesis (Stanford Univ., Stanford 1969)

54.10 International Federation of Robotics (IFR): History of Industrial Robots. From the first installation until today (International Federation of Robotics, Frankfurt 2012), http://www.ifr.org/uploads/media/History_of_Industrial_Robots_online_brochure_by_IFR_2012.pdf

54.11 H. Makino: Assembly robot, US4341502 A (1982)

54.12 G. Boothroyd, P. Dewhurst, W. Knight: *Product Design for Manufacture and Assembly*, 3rd edn. (CRC, Boca Raton 2011)

54.13 R. Curran, G. Mayer: The architecture of the Adept0ne® direct-drive robot, Am. Control Conf., Boston (1985)

54.14 X.-J. Liu, J. Wang: Performance evaluation of parallel mechanisms. In: *Parallel Kinematics*, Springer Tracts in Mechanical Engineering, (Springer, Berlin, Heidelberg 2014) pp. 185–238

54.15 S. Haddadin, S. Parusel, R. Belder, A. Albu-Schaeffer: It is (almost) all about human safety: A novel paradigm for robot design control and planning, Lect. Notes Comput. Sci. **8153**, 202–215 (2013)

54.16 R. Bloss: Innovations like two arms, cheaper prices, easier programming, autonomous and collaborative operation are driving automation deployment in manufacturing and elsewhere, Assem. Autom. **33**(4), 312–316 (2013)

54.17 S. Kock, T. Vitor, M. Björn, H. Jerregard, M. Källmann, I. Lundberg: Robot concept for scalable, flexible assembly automation, Proc. IEEE Int. Symp. Assem. Manuf. (ISAM), Tampere (2011)

54.18 S. Bogh, M. Hvilshoj, M. Kristiansen, O. Madsen: Identifying and evaluating suitable tasks for autonomous industrial mobile manipulators (AIMM), Int. J. Adv. Manuf. Technol. **61**(5-8), 713–726 (2012)

54.19 C.Y.K. Smith, L. Nalpantidis, X. Gratal, P. Qi, D. Dimarogonas, D. Kragic: Dual arm manipulation – a survey, Robot. and Auton. Syst. **60**(10), 1340–1353 (2012)

54.20 P. Gorle, A. Clive: *Positive Impact of Industrial Impacts on Employment* (International Federation of Robotics, Frankfurt 2013)

54.21 J. Manyika, M. Chui, J. Bughin, R. Dobbs, P. Bisson, A. Marrs: Disruptive technologies: Advances that will transform life, business, and the global economy, http://www.mckinsey.com/insights/business_technology/disruptive_technologies (2013)

54.22 E. Westkämper: *Towards the Re-Industrialization of Europe: A Concept for Manufacturing for 2030* (Springer, Berlin, Heidelberg, 2014)

54.23 M. Hedelind, M. Jackson: How to improve the use of industrial robots in lean manufacturing systems, J. Manuf. Technol. Manag. **22**(7), 891–905 (2011)

54.24 H. Kagermann, W. Wahlster, J. Helbig: *Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0*. http://www.acatech.de/fileadmin/user_upload/Baumstruktur_nach_Website/Acatech/root/de/Material_fuer_Sonderseiten/Industrie_4.0/Final_report__Industrie_4.0_accessible.pdf (Forschungsunion/acatech, Frankfurt 2013)

54.25 R. Bischoff, J. Kurth, G. Schreiber, R. Koeppe: The KUKA-DLR lightweight robot arm – a new reference platform for robotics research and manufacturing, 41st Int. Symp. Robot. 6th Ger. Conf. Robot. (ROBOTIK), Munich (2010)

54.26 M. Hedelind, S. Kock: Requirements on flexible robot systems for small parts assembly: A case study, Proc. 2011 IEEE Int. Symp. Assem. Manuf. (ISAM), Tampere (2011) pp. 1–7

54.27 S. Kinkel, E. Kirner, H. Armbruster, A. Jäger: Relevance and innovation of production-related services in manufacturing industry, Int. J. Technol. Manag. **55**(3/4), 263–273 (2011)

54.28 International Organization for Standardization: ISO 8373. Robots and robotic devices – Vocabulary, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=53830 (2012)

54.29 R. Bloss: Review of manufacturing cells as they achieve high levels of autonomy and flexibility, Assem. Autom. **33**(2), 112–116 (2013)

54.30 A. Kahn: *The Encyclopedia of Work-related Illnesses, Injuries, and Health Issues* (Facts On File, New York 2004)

54.31 M. Wilson: Developments in robot applications for food manufacturing, Ind. Robot Int. J. **37**(6), 498–502 (2010)

54.32 R. Moreno Masey, J. Gray, T. Dodd, D. Caldwell: Guidelines for the design of low-cost robots for the food industry, Ind. Robot Int. J. **37**(6), 509–517 (2010)

54.33 K. Mathia: *Principles and Applications in Cleanroom Automation* (Cambridge Univ. Press, Cambridge 2010)

54.34 D. Rossi, E. Bertolini, M. Fenaroli, F. Marciano, M. Alberti: A multi-criteria ergonomic and performance methodology for evaluating alternatives in "manuable" material handling, Int. J. Ind. Ergon. **43**(4), 314–327 (2013)

54.35 D. Caldwell: *Robotics and Automation in the Food Industry: Current and Future Technologies* (Woodhead, Cambridge 2012)

54.36 M. Palzkill, A. Verl: Object pose detection in industrial environment, Proc. 7th Ger. Conf. Robot. (ROBOTIK), Munich (2012)

54.37 R. Mattone, M. Divona, A. Wolf: Sorting of items on a moving conveyor belt. Part 2: Performance evaluation and optimization of pick-and-place operations, Robot. Comput.-Integr. Manuf. **16**(2/3), 81–90 (2000)

54.38 A. Grzesiak, R. Becker, A. Verl: The bionic handling assistant: A success story of additive manufacturing, Assem. Autom. **4**, 329–333 (2011)

54.39 K. Ikeuchi, B. Horn, S. Nagata, T. Callahan, O. Fein: Picking up an object from a pile of objects, A.I. Memo No. 726 (Artif. Intell. Lab./MIT, Cambridge 1983)

54.40 M.-Y. Liu, O. Tuzel, A. Veeraraghavan, T. Marks, R. Chellappa: Fast object localization and pose estimation in heavy clutter for robotic bin picking, Int. J. Robotics Res. **31**(8), 951–973 (2012)

54.41 J.J. Rodrigues, J.-S. Kim, M. Furukawa, J. Xavier, P. Aguiar, T. Kanade: 6D pose estimation of textureless shiny objects using random ferns for bin picking, IEEE/RSJ Int. Conf. Intell. Robots Syst., Vilamoura (2012)

54.42 Z. Pan, J. Polden, N. Larkin, S. Van Duin, J. Norrish: Recent progress on programming methods for industrial robots, Robot. Comput.-Integr. Manuf. **28**(2), 87–94 (2012)

54.43 P. Klüger: Future body-in-white concepts. Increased sustainability with reduced investment and life-cycle-cost, Automot. Manuf. Solut., Shanghai (2013)

54.44 M. Shah, R.D. Eastman, T. Hong: An overview of robot-sensor calibration methods for evaluation of perception systems, Workshop Perform. Metr. Intell. Syst., New York (2012)

54.45 Deutsches Institut für Normung: DIN8593-0:2003-09: Manufacturing processes joining – Part 0: General; classification, subdivision, terms and definitions, http://www.beuth.de/de/norm/din-8593-0/65031206 (2003)

54.46 N. Lohse, H. Hirani, S. Ratchev, M. Turitto: An ontology for the definition and validation of assembly processes for evolvable assembly systems, Proc. 6th IEEE Int. Symp. Assem. Task Plan. (2005) pp. 242–247

54.47 K. Malecki: Vehicle body mounted door seals on a roll, Adhesion Adhesives Sealants **6**(1), 23–25 (2009)

54.48 S. Kock, T. Vittor, B. Matthias: Robot concept for scalable, flexible assembly automation, Proc. IEEE Int. Symp. Assem. Manuf. (ISAM), Tampere (2011)

54.49 H. Chen, J. Wang, G. Zhang, T. Fuhlbrigge, S. Kock: High-precision assembly automation based on robot compliance, Int. J. Adv. Manuf. Technol. **45**(9/10), 999–1006 (2009)

54.50 A. Stolt, M. Linderoth, A. Robertsson, R. Johansson: Force controlled robotic assembly without a force sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Saint Paul (2012)

54.51 ABB Robotics: Application manual – force control for machining, http://new.abb.com/products/robotics/application-equipment-and-accessories/flexfinishing/function-package-for-force-control (2013)

54.52 J. Domnick, Z. Yang, Q. Ye: Simulation of the film formation at a high-speed rotary bell atomizer used in automotive spray painting processes, Proc. 22nd Eur. Conf. Liq. At. Spray Syst. ILASS Eur., Como (2008)

54.53 M. Jonsson, A. Stolt, A. Robertsson, S. von Gegerfelt, K. Nilsson: On force control for assembly and deburring of castings, Prod. Eng. **7**(4), 351–360 (2013)

54.54 Z. Pan, H. Zhanh, Z. Zhu, J. Wang: Chatter analysis of robotic machining process, J. Mater. Process. Technol. **173**(3), 301–309 (2006)

54.55 M. Weck, C. Brecher: *Werkzeugmaschinen 2 – Konstruktion und Berechnung* (Springer, Berlin, Heidelberg 2006)

54.56 U. Schneider, M. Ansaloni, M. Drust, F. Leali, A. Verl: Experimental investigation of sources of error in robot machining, Int. Workshop Robot. Smart Manuf., Porto (2013)

54.57 J. Wang, H. Zhang, T. Fuhlbrigge: Improving machining accuracy with robot deformation compensation, Int. Conf. Intell. Robots Syst., St. Louis (2009)

54.58 C. Lehmann, B. Olofsson, K. Nilsson, M. Halbauer, M. Haage, A. Robertsson, O. Sornmo, U. Berger: Robot joint modeling and parameter identification using the clamping method, IFAC Conf. Manuf. Modell. Manag. Control, Saint Petersburg (2013)

54.59 B. Saund, R. DeVlieg: High accuracy articulated robots with CNC control system, SAE Int. J. Aerosp. **6**(2), 780–784 (2013)

54.60 U. Schneider, J.R. Diaz Posada, M. Drust, A. Verl: Position control of an industrial robot using an optical measurement system for machining purposes, Proc. 11th Int. Conf. Manuf. Res. (2013) pp. 307–312

54.61 U. Schneider, B. Olofsson, O. Sörnmo, M. Drust, A. Robertsson, M. Hägele, R. Johansson: Integrated approach to robotic machining with macro/micro actuation, Robotics Comput.-Integr. Manuf. **30**(6), 636–647 (2014)

54.62 International Organization for Standardization: ISO 12100:2010. Safety of machinery – General principles for design – Risk assessment and risk reduction, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=51528 (2010)

54.63 European Parliament: Directive 2006/42/EC of the European Parliament and the Council of 17 May 2006 on machinery, and amending Directive 95/16/EC (recast), http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2006:157:0024:0086:EN:PDF (2006)

54.64 Siemens: *Easy Implementation of the European Machinery Directive, Functional Safety of Machines and Systems* (Siemens, Fürth 2012)

54.65 International Organization for Standardization: ISO 13849-1:2006. Safety of machinery – Safety-related parts of control systems – Part 1: General principles for design, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=3491 (2006)

54.66 International Electrotechnical Commission: IEC 62061 ed1.0. Safety of machinery – Functional safety of safety-related electrical, electronic and programmable electronic control systems: http://webstore.iec.ch/webstore/webstore.nsf/artnum/033604!opendocument (2005)

54.67 International Electrotechnical Commission: IEC 61508 ed2.0. Functional safety of electrical/electronic/programmable electronic safety-related systems, http://www.iec.ch/functionalsafety/standards (2010)

54.68 International Organization for Standardization: ISO 13855:2010. Safety of machinery – Positioning of safeguards with respect to the approach speeds of parts of the human body, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=42845 (2010)

54.69 International Organization for Standardization: ISO 13857:2008. Safety of machinery – Safety distances to prevent hazard zones being reached by upper and lower limbs, http://www.iso.org/iso/home/store/catalogue_detail.html?csnumber=39255 (2008)

54.70 R. Hollmann, A. Rost, M. Hägele, A. Verl: A HMM-based approach to learning probability models of programming strategies for industrial robots, 2010 IEEE Int. Conf. Robotics Autom., Anchorage (2010)

54.71 B. Lacevic, P. Rocco, A. Zanchettin: Safety assessment and control of robotic manipulators using danger field, IEEE Trans. Robotics **29**(5), 1257–1270 (2013)

54.72 International Organization for Standardization: ISO/TC 184/SC 2. Robots and robotic devices,

http://www.iso.org/iso/home/store/catalogue_tc/catalogue_tc_browse.htm?commid=54138&development=on (2014)

54.73 S. Oberer-Treitz, A. Puzik, A. Verl: Measuring the collision potential of industrial robots, 41st Int. Symp. Robot. 6th Germ. Conf. Robot. (2010)

54.74 R. Behrens, N. Elkmann: Study on meaningful and verified thresholds for minimizing the consequences of human-robot collisions, IEEE Int. Conf. Robotics Autom. (ICRA) (2014) pp. 3378–3383

54.75 S. Haddadin, S. Haddadin, A. Khoury, T. Rokahr, S. Parusel, R. Burgkart, A. Bicchi, A. Albu-Schaeffer: On making robots understand safety: Embedding injury knowledge into control, Int. J. Robotics Res. **31**(13), 1578–1602 (2012)

54.76 J. Marvel, J. Falco: *Best practices and performance metrics using force control for robot assembly* (Nat. Inst. Stand. Technol., Gaithersburg 2012)

54.77 R. Zurawski: *Integration Technologies for Industrial Automated Systems* (CRC, Boca Raton 2006)

Multimedia Contents

55. Space Robotics

Kazuya Yoshida, Brian Wilcox, Gerd Hirzinger, Roberto Lampariello

In the space community, any unmanned spacecraft can be called a robotic spacecraft. However, *Space Robots* are considered to be more capable devices that can facilitate manipulation, assembling, or servicing functions in orbit as assistants to astronauts, or to extend the areas and abilities of exploration on remote planets as surrogates for human explorers.

In this chapter, a concise digest of the historical overview and technical advances of two distinct types of space robotic systems, orbital robots and surface robots, is provided. In particular, Sect. 55.1 describes orbital robots, and Sect. 55.2 describes surface robots. In Sect. 55.3, the mathematical modeling of the dynamics and control using reference equations are discussed. Finally, advanced topics for future space exploration missions are addressed in Sect. 55.4.

55.1 **Historical Developments and Advances of Orbital Robotic Systems** 1424
55.1.1 Robotic Arms for Assistance of Human Space Flight 1424
55.1.2 Future-Oriented Space Robot Experiments 1426

55.2 **Historical Developments and Advances of Surface Robotic Systems** 1430
55.2.1 Teleoperated Rovers 1430
55.2.2 Autonomous Rovers 1432
55.2.3 Research Systems 1435
55.2.4 Sensing and Perception 1436
55.2.5 Estimation 1437
55.2.6 Manipulators for In-Situ Science 1437

55.3 **Mathematical Modeling** 1437
55.3.1 Space Robot as an Articulated Body System 1437
55.3.2 Equations for Free-Floating Manipulator Systems 1438
55.3.3 Generalized Jacobian and Inertia Matrices 1439
55.3.4 Linear and Angular Momenta 1440
55.3.5 Virtual Manipulator 1440
55.3.6 Dynamic Singularity 1441
55.3.7 Reaction Null-Space (RNS) 1441
55.3.8 Motion Planning Issues 1441
55.3.9 Equations for Flexible-Based Manipulator Systems 1443
55.3.10 Advanced Control for Flexible Structure Based Manipulators 1444
55.3.11 Contact Dynamics and Impedance Control 1446
55.3.12 Dynamics of Mobile Robots 1447
55.3.13 Wheel Traction Mechanics 1448

55.4 **Future Directions of Orbital and Surface Robotic Systems** 1452
55.4.1 Robotic Maintenance and Service Missions 1452
55.4.2 Robonaut and JUSTIN 1453
55.4.3 Aerial Platforms 1454
55.4.4 Mobility Concepts and Subsurface Platforms 1456

55.5 **Conclusions and Further Reading** 1457

Video-References 1457

References 1458

55.1 Historical Developments and Advances of Orbital Robotic Systems

Key issues in space robots and systems are characterized as follows:

Manipulation – Although manipulation is a basic technology in robotics, microgravity in the orbital environment requires special attention to the motion dynamics of manipulator arms and objects being handled. Reaction dynamics that affect the base body, impact dynamics when the robotic hand contacts an object to be handled, and vibration dynamics due to structural flexibility are included in this issue.

Mobility – The ability for locomotion is particularly important in exploration robots (rovers) that travel on the surface of a remote planet. These surfaces are natural and rough, and thus challenging to traverse. Sensing and perception, traction mechanics, and vehicle dynamics, control and navigation; all of these mobile robotics technologies must be demonstrated in a natural untouched environment.

Teleoperation and Autonomy – There is a significant time delay between a robotic system at a work site and a human operator in an operation room on the earth. In earlier orbital robotics demonstrations, the latency was typically 5 s, but can be several tens of minutes, or even hours for planetary missions. Telerobotics technology is therefore an indispensable ingredient in space robotics, and the introduction of autonomy is a reasonable consequence.

Extreme Environments – In addition to the microgravity environment that affects the manipulator dynamics or the natural and rough terrain that affects surface mobility, there are a number of issues related to extreme space environments that are challenging and must be solved in order to enable practical engineering applications. Such issues include extremely high or low temperatures, high vacuum or high pressure, corrosive atmospheres, ionizing radiation, and very fine dust.

The first robotic manipulator arm used in the orbital environment is the shuttle remote manipulator system (SRMS). It was successfully demonstrated in the STS-2 mission in 1981 and was operational until the end of the shuttle era. This success opened a new era of orbital robotics and inspired a number of mission concepts to the research community. One ultimate goal that has been discussed intensively after the early 1980s is the application to the rescue and servicing of malfunctioning spacecraft by a robotic free-flyer or free-flying space robot (e.g., ARAMIS report [55.1], Fig. 55.1). In later years, manned service missions were conducted for the capture-repair-deploy procedure of malfunctioning satellites (Intelsat 603 by STS-49, for example) and for the maintenance of the Hubble space telescope (STS-61, 82, 103, and 109). For all of the examples, the Space Shuttle, a manned spacecraft with dedicated maneuverability, was used. However, unmanned servicing missions have not yet become operational. Although there were several demonstration flights, such as ETS-VII and Orbital Express (to be elaborated later), the practical technologies for unmanned satellite servicing missions await solutions to future challenges.

55.1.1 Robotic Arms for Assistance of Human Space Flight

Space Shuttle Remote Manipulator System

Onboard the space shuttle, the SRMS, or Canadarm, is a mechanical arm that maneuvers a payload from the payload bay of the space shuttle orbiter to its deployment position and then releases it [55.2]. It can also grapple a free-flying payload, maneuver it to the payload bay of the orbiter, and berth it back into the orbiter. The SRMS was first used on the second Space Shuttle mission STS-2, launched in 1981. Since then, it was used more than 100 times during space shuttle flight missions, performing such payload deployment or berthing as well as assisting human extra vehicular activities (EVAs). Servicing and maintenance missions to the Hubble space telescope and construction tasks of the International Space Station (ISS) have also been

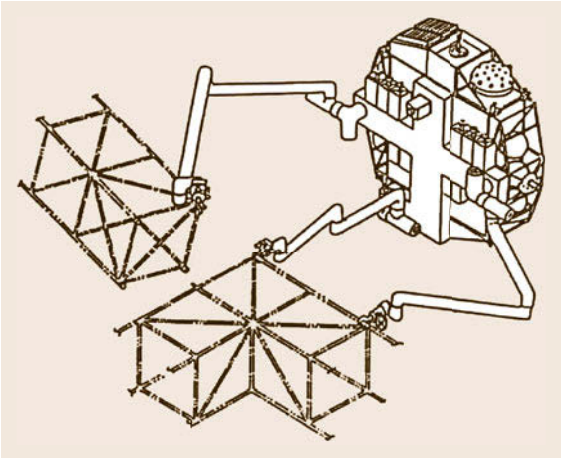


Fig. 55.1 A conceptual design of telerobotic servicer (after [55.1])

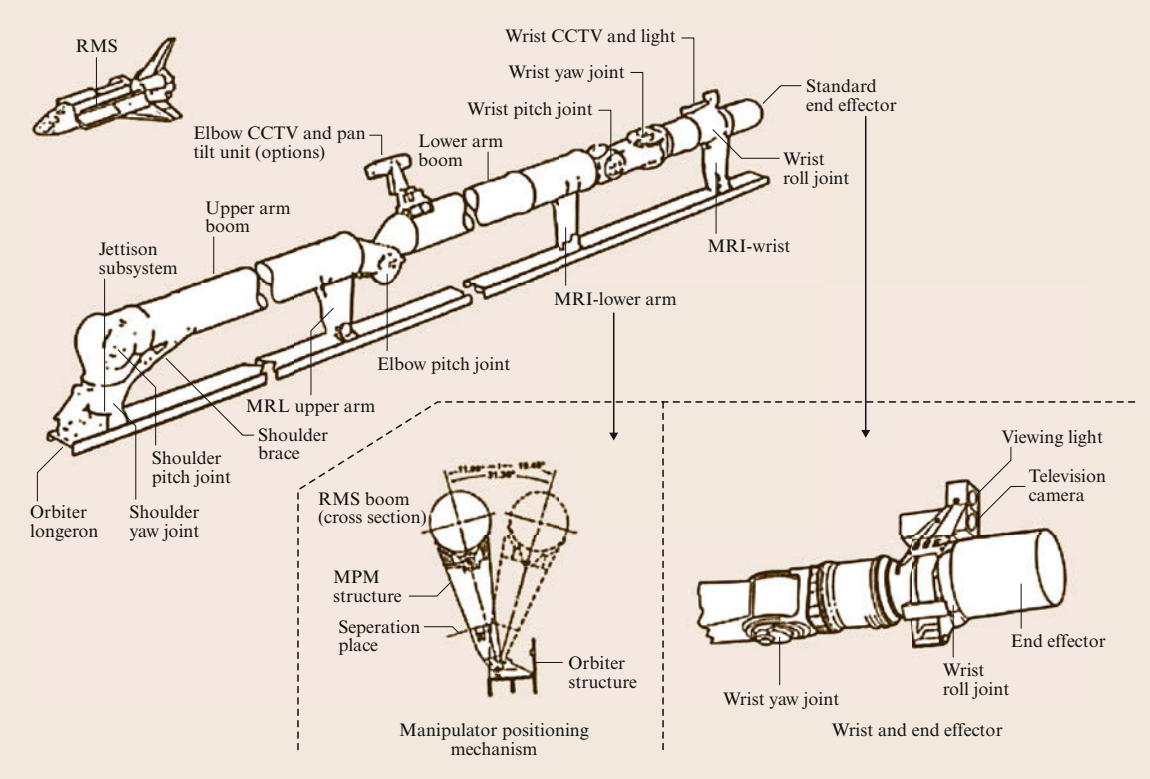


Fig. 55.2 Space Shuttle remote manipulator system (SRMS) (after [55.2])

successfully carried out by the cooperative use of the SRMS with human EVAs.

As depicted in Fig. 55.2, the SRMS arm was 15 m long and had 6-degrees of freedom (DOF), comprising shoulder yaw and pitch joints, an elbow pitch joint, and wrist pitch, yaw, and roll joints. Attached to the end of the arm was a special gripper system called the standard end effector (SEE), which was designed to grapple a pole-like fixture (GF) attached to the payload.

By attaching a foothold at the end point, the arm could serve as a mobile platform for an astronaut's EVAs (Fig. 55.3).

After the Space Shuttle COLUMBIA accident during STS-107, NASA outfitted the SRMS with the orbiter boom sensor system – a boom containing instruments to inspect the exterior of the shuttle for damage to the thermal protection system [55.3].

ISS Mounted Manipulator Systems

The ISS is the largest international technology project, with 15 countries making significant cooperative contributions. The ISS is an *outpost* of human presence in space, as well as a *flying laboratory* with substantial facilities for science and engineering research. In order

Fig. 55.3 Space shuttle remote manipulator system (SRMS) used as a platform for an astronaut's extra vehicular activity in the shuttle cargo bay

to facilitate various activities on the station, there are several robotic systems, some of which are already operational, while others are ready for launch.

The space station remote manipulator system (SSRMS), or Canadarm 2 (Fig. 55.4), was the next generation of SRMS for use on the ISS [55.4]. Launched in 2001 during STS-100 (ISS assembly flight 6A), the SSRMS has played a key role in the construction and maintenance of the ISS both by assisting astronauts during EVAs and using the SRMS on the Shuttle to hand over a payload from a Shuttle to the SSRMS. The arm is 17.6 m long when fully extended and has 7-DOF. Latching end effectors, through which power, data, and video can be transmitted to and from the arm, are attached to both ends. The SSRMS is self-relocatable using an inch-worm like movement with alternate grappling of power data grapple fixtures (PDGFs), which are installed over the station's exterior surfaces to provide the power, data, and video, as well as a foothold.

As another mobility aid for the SSRMS to cover wider areas of ISS, mobile base system (MBS) was added in 2002 by STS-111 (ISS assembly flight UF-2). The MBS provides lateral mobility as it traverses the rails on the main trusses [55.5].

The special purpose dexterous manipulator (SPDM), or Dextre, which is attached at the end of the SSRMS, is a capable mini-arm system to facilitate the delicate assembly tasks currently handled by astronauts during EVAs. The SPDM is a dual arm manipulator system, where each manipulator (with 3 m length) has 7-DOF and is mounted on a one degree-of-freedom body joint. Each arm has a special tool mechanism dedicated to the handling of standardized orbital replacement units (ORUs). The arms are teleoperated from a Robotic WorkStation (RWS) inside the space station [55.6].

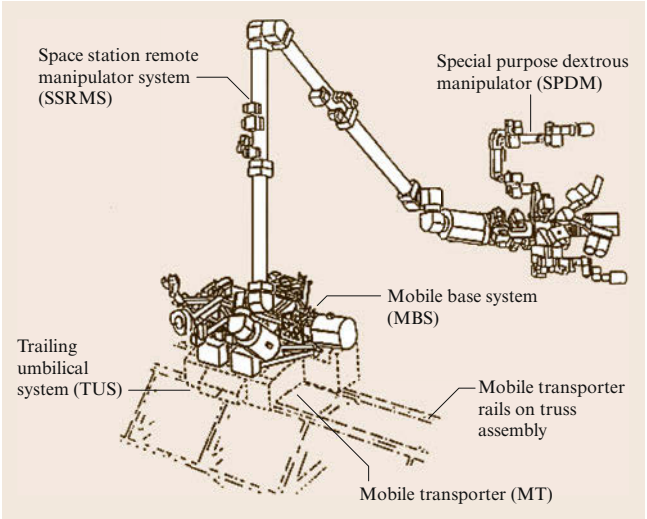


Fig. 55.4 Space station remote manipulator system (SSRMS) (after [55.4])

The European Space Agency (ESA) will also provide a robotic manipulator system for the ISS, the European robotic arm (ERA), and will be used mainly to work on the Russian segments of the station [55.7]. The arm is 11.3 m long and has 7-DOF. The basic configuration and functionality are similar to SSRMS [55.8].

In Japan, the Japanese experiment module remote manipulator system (JEMRMS), as shown in Fig. 55.5, was developed by the Japan Space Exploration Agency (JAXA) [55.9–11]. The arm was launched with the STS-124 Mission in 2008 and is now operative on the Japanese module of the ISS. JEMRMS comprises two components: the main arm, a 9.9 m long, six-degree-of-freedom arm, and the small fine arm, a 1.9 m long, six-degree-of-freedom arm.

Unlike the SSRMS or the ERA, the main arm does not have self-relocation capability, but is fitted with a small fine arm, with which JEMRMS can form a serial 12-degree-of-freedom macro-micro manipulator system. After installation, the arm is used to handle and relocate the components for the experiments and observations on the exposed facility.

55.1.2 Future-Oriented Space Robot Experiments

ROTEX

The Robot Technology Experiment (ROTEX) developed by the German Aerospace Agency (DLR), is one of the important milestones of robotics technology in space [55.12], as it demonstrated the first (remote) ground control of a space robot with of up to 6 s round-trip signal delay using geostationary relay satellites. A multisensory robotic arm was flown on Space Shuttle COLUMBIA (STS-55) in 1993. Although the robot worked inside a work cell on the shuttle, several key technologies, such as a multisensory gripper, teleoperation from the ground and by the astronauts, shared autonomy, and time-delay compensation by use of a predictive graphic display were successfully tested (Fig. 55.6 and VIDEO 330).

Presumably the most spectacular experiment in ROTEX was the fully automatic grasping of a small free-floating cube with flattened edges by the ground computers who evaluated the stereo images from the robot gripper, estimated the motion, predicted for the above-mentioned 6 s and sent up the commands for grasping (VIDEO 331).

ETS-VII

Engineering Test Satellite VII (ETS-VII), shown in Fig. 55.7, is another milestone in the development of robotics technology in space, particularly in the area of satellite servicing. ETS-VII was an unmanned spacecraft developed and launched by the National Space Development Agency of Japan (NASDA, currently JAXA) in November 1997. A number of experiments were successfully conducted using a 2 m long, six-degree-of-freedom manipulator arm mounted on its carrier satellite.

The mission objective of ETS-VII was to test free-flying robotics technology and to demonstrate its utility in unmanned orbital operation and servicing tasks. The mission consisted of two subtasks: autonomous rendezvous/docking (RVD) and a number of robot experiments (RBT). The robot experiments included: (1) teleoperation from the ground with a large time delay; (2) robotic servicing task demonstrations such as ORU exchange and deployment of a space structure; (3) dynamically coordinated control between the manipulator reaction and the satellite attitude response; and (4) capture and berthing of a cooperative target satellite [55.13].

The communication time delay due to radio propagation (speed-of-light) is relatively small, for example 0.25 s for a round trip to geostationary Earth orbit (GEO). However, to have a global coverage of communication in low Earth orbit (LEO) operations, the signals are transmitted via multiple nodes including data relay satellites located at GEO and ground stations. This makes the transmission distance longer, and even more additional delays are added at each node. As a result, the cumulative delay becomes some seconds, actually 5–7 s in case of ETS-VII mission, more or less the same as in the ROTEX experiment. However, it is important to state that these delays have been unnecessarily long as the optimal communication infrastructure was (and is still) missing. Figure 55.8 shows that for a low orbit robot satellite (with typically 1.5 h orbital period) half of the orbit (i. e., around 45 min) would have communication from the ground station via one relay satellite yielding approximately 600 ms round trip delay (including computational delays). For a robot in geostationary orbit having permanent communication to the ground station, the round trip delay would be only 300 ms.

In ETS-VII, opportunities for academic experiments were also opened to Japanese universities and European institutions (e.g., DLR and ESA), and important flight data were obtained that validate the concepts and theories for free-flying space robots [55.14, 15]. As an example, DLR performed experiments aiming at demonstrating robot control methods for performing ORU exchange tasks (see for example VIDEO 332) and dedicated satellite attitude control via *swimming* or *waiving* motions of the robot arm.

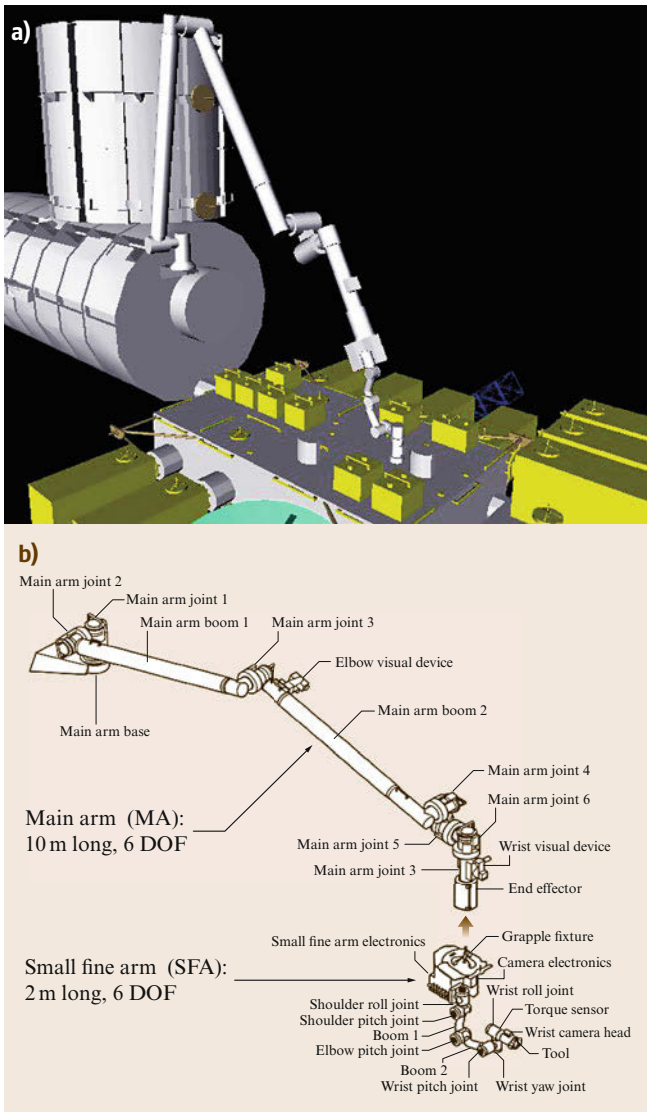


Fig. 55.5 (a) Japan Experiment Module (JEM) on the ISS and (b) the JEMRMS manipulator system

Ranger

Ranger is a teleoperated space robot being developed at the University of Maryland, Space Systems Laboratory [55.16]. Ranger consists of two seven-degree-of-freedom manipulators with interchangeable end effectors to perform such tasks as changeout of orbital replacement units (ORUs) in orbit. Also discussed was the changeout of the electronics controller unit (ECU) of the Hubble space telescope, which previously required human EVA. A number of tests and demon-

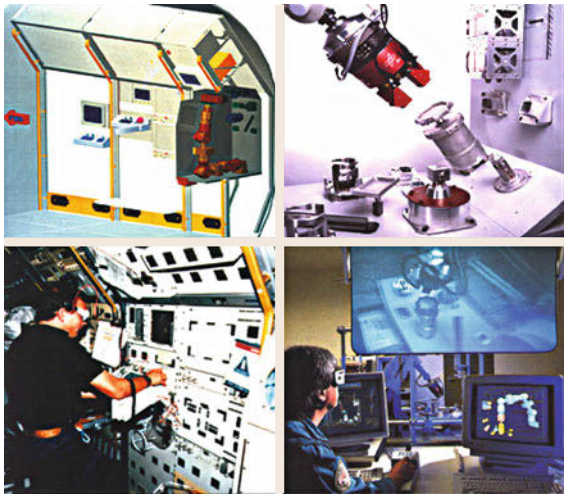


Fig. 55.6 ROTEX manipulator arm onboard the Spacelab D2 mission, the first remotely controlled space robot

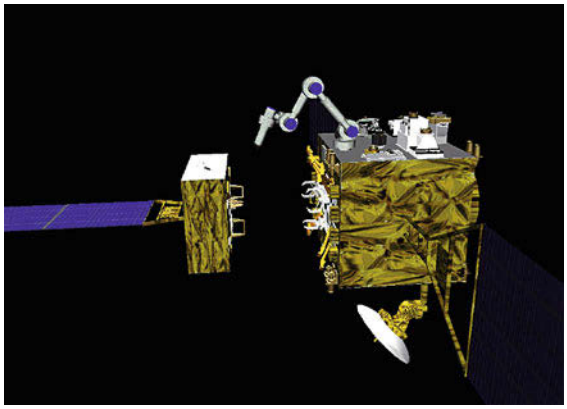


Fig. 55.7 Japanese Engineering Test Satellite ETS-VII

strations for servicing missions have been conducted at the University of Maryland Neutral Buoyancy Facility (Fig. 55.9). Originally designed for a free-flying flight experiment, Ranger had been redesigned for a shuttle flight experiment, but ultimately was not manifested on a shuttle flight.

Orbital Express

The Orbital Express Space Operations Architecture program is a DARPA program developed to validate the technical feasibility of robotic on-orbit refueling and reconfiguration of satellites, as well as autonomous rendezvous, docking, and manipulator berthing [55.17]. The system consists of the autonomous space transport robotic operations (ASTRO) vehicle, developed by Boeing Integrated Defense Systems, and a prototype modular next-generation serviceable satellite, NextSat, developed by Ball Aerospace. The ASTRO vehicle is

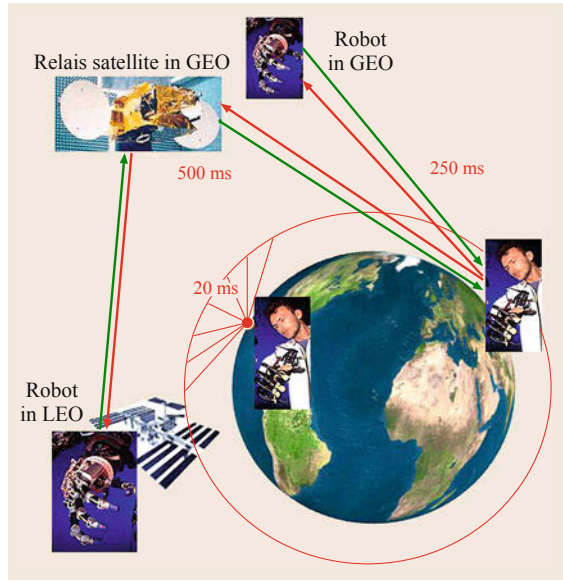


Fig. 55.8 Minimal *Roundtrip* signal propagation delay to robots in LEO is around 0.6 s, to robots in GEO around 0.3 s

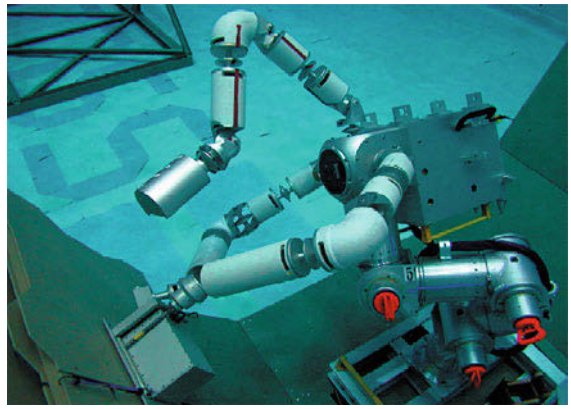


Fig. 55.9 Neutral buoyancy test of the Ranger telerobotic shuttle experiment

equipped with a robotic arm to perform satellite capture and ORU exchange operations (Fig. 55.10).

After its launch in March 2007, various mission scenarios were conducted. These scenarios include (1) visual inspection, fuel transfer, and ORU exchange on NextSat using ASTRO's manipulator arm when both spacecrafts are connected, (2) separation of NextSat from ASTRO, orbital maneuvers by ASTRO, and fly-around, rendezvous, and docking with NextSat, and (3) capture of NextSat using ASTRO's manipulator arm.

These scenarios were successfully completed by July 2007, with ASTRO's onboard autonomy using onboard cameras and advanced video guidance system.

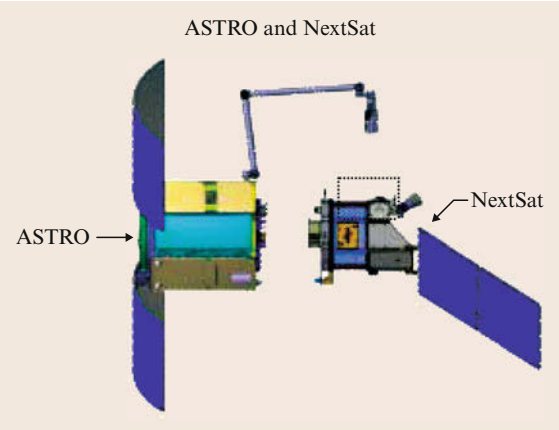


Fig. 55.10 Orbital express flight mission configuration

ROKVISS and Delayed Teleoperation

As a precursor demonstration of the planned German Orbital Servicing demonstration mission DEOS (see below), the German Aerospace Agency (DLR) has developed and flown a 0.5 m long, two-degree-of-freedom manipulator arm with a dedicated test bench, called robotic components verification on the ISS (ROKVISS) (Fig. 55.11) [55.18].

Using the 7–8 min contact window of the ISS overflight through a dedicated real-time space link, resulting in a roundtrip delay as small as 20 ms, this was the first high-fidelity telepresence system in the history of space flight that allowed force feedback teleoperation. A large amount of data documenting the evolution of electrical and mechanical properties of the robot (such as sensor accuracy, friction, and motor parameters) has been collected during the six year mission.

ROKVISS was launched by an unmanned Russian progress transport vehicle in 2004 and installed on the outer platform of the Russian segment of the station in early 2005 (VIDEO 333).

After six years, the system was still fully functional, even though it had undergone only low-cost qualification, i. e., most of the electronic components were off-the-shelf and the system as a whole had been qualified by radiation, vibration, and temperature tests. The long-time verification of these joints for outer space has been a second main goal of ROKVISS (VIDEO 334). Toward the end of the mission, ROKVISS was teleoperated from the private home of the project leader using standard internet communication to DLR's professional ground station. Round-trip delay went then up to around 400 ms still allowing telepresence with force-reflection. At the end of 2011, ROKVISS was brought back to Earth for completion of verification and validation tests (VIDEO 336).

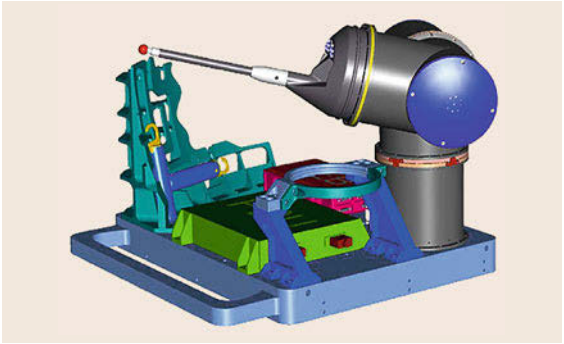


Fig. 55.11 ROKVISS – 6 years on ISS

Although the numbers of joints was small, challenging experiments of telepresence were conducted, in which human operators from ground teleoperated the arm using a force feedback joystick and stereo vision images. The ROKVISS arm, on the other hand, was equipped with joint torque sensors, allowing full impedance control and advanced bilateral control techniques. The secondary goal of the experiment was the space qualification of the joint drives, which are the key components of DLR's torque-controlled lightweight robots [55.19].

No electrical or mechanical damages were visible, and the main observation was that joint friction had nearly doubled in vacuum on ISS, but went back to its original value when the arm was back on earth again. In the framework of ROKVISS and the accompanying investigations, important basic clarifications of the general telepresence potentials and prerequisites could be achieved.

Already in the early days of space telerobotics as pushed forward by JPL, there have been estimates that humans might be able to master signal delays of up to 1 s in the visual system (i. e., looking at delayed images) and up to 500 ms with the haptic system from the proprioceptive point of view.

The interesting, fairly recent result is here [55.20, 21], where feasibility of force-reflecting teleoperation with communication delays as high as 650 ms has been proven. By using the so-called time domain passivity control approach, the mechanical energy of the system is observed and controlled in real time, such that the system is passive for any given communication channel characteristics, including varying time delays and packet loss. These results have been verified with a unique teleperesence experiment, where two torque-controlled light weight arms located in Oberpfaffenhofen (Germany) were connected through a real relayed communication link that used the geostationary satellite ARTEMIS, ground station communication an-

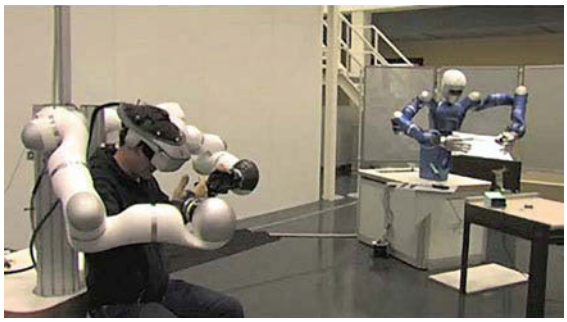


Fig. 55.12 Force-reflecting telepresence demos at DLR

tennas located in Garching (Germany) and a data mirror in Redu (Belgium) [55.22].

Recently, telepresence experiments with a fairly complex multidegree-of-freedom (DOF) system have been performed [55.23]. The system consisted of the DLR's *Space JUSTIN* humanoid robot and a human–machine interface based on two torque-controlled light weight arms as a force reflecting hand controller (haptic device) (Fig. 55.12 and VIDEO 337). Complex tasks that required with high levels of dexterity, such (un)screwing with a screwdriver and soldering, were performed with time delays up to 500 ms (VIDEO 338).

It is therefore concluded that teleoperation with force feedback is possible over the whole earth orbit.

The DEOS Demonstration Project

The German space robotics-demonstration mission DEOS, based on the ROKVISS-joint torque-controlled arm technology, aims at the demonstration of the maturity and availability of key technologies as needed for an orbit servicing. DEOS is comprised of two satellites, servicer and client (Fig. 55.13). Its main goal is to find, approach, and capture an uncontrolled and uncooperative satellite in LEO (VIDEO 339). After successful capturing some typical repair and maintenance tasks are planned to be demonstrated before the servicing satellite enters the atmosphere for a controlled descent together with the captured satellite. Project developments are currently being pursued to advance the technological readiness level, in view of future missions. Developments also include adaptations to Active Debris Removal missions in study, such as e.Deorbit (ESA), for the deorbiting of the ENVISAT satellite.

Fig. 55.13 DEOS – orbital servicing experiment with client (*left*) and servicer (*right*)

Robonaut and JUSTIN

Humanoid (human-like) robots have been developed to conduct human-compatible dexterous tasks. NASA's Robonaut and DLR's JUSTIN are such representative examples. These robots are elaborated in Sect. 55.4.2.

55.2 Historical Developments and Advances of Surface Robotic Systems

The research on surface exploration rovers began in the mid-1960s, with an initiative (that never flew) for an unmanned rover for the Surveyor lunar landers and a manned rover (Moon buggy) for the human landers in the United States. In the same period, research and development began for a teleoperated rover named Lunokhod in the Soviet Union. Both the Apollo-manned rover and the Lunokhod-unmanned rover were successfully demonstrated in the early 1970s on Moon [55.24]. In the 1990s, the exploration target had expanded to Mars, and in 1997, the Mars Pathfinder Mission successfully deployed a microrover named Sojourner that safely traversed the rocky field adjacent to the landing site by autonomously avoiding obstacles [55.25, 26]. Following this success, today, autonomous robotic vehicles are considered indispensable technology for planetary exploration. The twin Mars Exploration Rovers, Sprit and Opportunity, were launched in 2003, and have had remarkable success in terms of remaining operational in the harsh environment of Mars for over three years. Each has traveled more than 5000 m and has made significant scientific discoveries using on-board instruments [55.27, 28].

55.2.1 Teleoperated Rovers

The first remotely operated robotic space surface vehicle was Lunokhod (Fig. 55.14) [55.24]. Lunokhod 1 landed on the Moon on November 17, 1970 as a payload on the lander Luna-17, and Lunokhod 2 landed on the

Fig. 55.14 Lunokhod

Moon on January 16, 1973. Both were 8-wheeled skid-steered vehicles having a mass of about 840 kg, where almost all the components were in a pressurized *bathtub* thermal enclosure with a lid that closed over the tub to allow it to survive the deep cold ($\approx$ 100 K) of the long lunar nights using only the heat emitted by small pellets of radioisotope. On the inside of the lid were solar arrays which recharged batteries during the day as required to maintain operation of the vehicle. Lunokhod 1 operated for 322 Earth days, traversing over 10.5 km during that period, and returned over 20 000 TV images, 200 high-resolution panoramas, and the results of more than 500 soil penetrometer tests and 25 soil analyses using its x-ray fluorescence spectrometer. Lunokhod 2 operated for about 4 months, having traversed more than 37 km, with the mission officially terminated on June 4, 1973. It has been reported that Lunokhod 2 was lost prematurely when it began sliding down a crater slope and hasty commands were sent in response which ultimately caused end-of-mission.

Each of the eight wheels on the Lunokhod vehicles were 0.51 m in diameter and 0.2 m wide, giving an effective ground pressure of less than 5 kPa based on an assumed sinkage of 3 cm. Each wheel had a brush-type DC motor, a planetary gear reduction, a brake, and a dis-engagement mechanism allowing it to free-wheel in the event of some problem with the motor or gears. The mobility commands to the vehicle included two speeds forward or backward, braking, and turning to the right or the left either while moving or in place.

The vehicles had both gyroscope and accelerometer-based tilt sensors which could automatically stop the vehicle in the event of excessive tilt of the chassis. Typical mobility commands specified a time duration over which the motors would run, and then stop. Precision turning commands specified the angle through which the vehicle should turn. These commands were terminated when the specified turn angle had been reached according to the heading gyroscope. Odometry was determined by a ninth *small wheel* which was unpowered and lightly loaded and used only to determine over-the-ground distance. There was an on-board current overload system, and motor currents, pitch and roll measurements, distance traversed, and many component temperatures were telemetered to the ground operators.

The Lunakod crew consisted of a driver, a navigator, a lead engineer, an operator for steering the *pencil-beam* communication antenna, and a crew commander. The driver viewed a monoscopic television image from the vehicle, and gave the appropriate commands (*turn*, *proceed*, *stop*, or *back up*) along with their associated parametric value in terms of duration or angle. The navigator viewed displays of telemetry from the vehicles' *course gyroscope*, *gyrovertical sensor*, and odometer, and was responsible for calculating the trajectory of the vehicle and laying out the route to be followed. Thus, the driver was responsible for vehicle stability about its center of mass and the navigator was responsible for the trajectory of that center of mass. The lead engineer (assisted by many specialists as required), was responsible for assessing the health of the on-board systems. The lead engineer provided both routine updates on energy supply, thermal conditions, etc., as well as possible emergency alerts such as extreme motor currents or chassis tilt. The pencil-beam antenna operator oversaw the functioning of an independent ground-based closed-loop control system that servoed the antenna to always point at Earth, independent of the vehicle motion. The crew commander supervised the implementation and execution of the overall plan, gave any detailed commands for making actual contact with the surface (e.g., by the penetrometer) and also could override any command to the vehicle as he viewed the same information as the driver.

The entire driving system was tested extensively prior to the Lunokhod 1 mission at a *lunodrome* having simulated lunar terrain which proved to be more challenging than that actually encountered during the Lunokhod 1 mission. Despite this, the operators of Lunokhod 1 said they *encountered a dangerous situ-*

ation (unforeseen entrance into a crater, rolling onto a rock, etc.) slightly more often than once per kilometer. This was attributed to inadequate driving experience, the modest quality of the television images, and the poor illumination conditions on the Moon. The driving direction was often selected primarily to give the best images; even so, the operators reported *fictitious dangers* caused by varying illumination conditions. In the first three months (lunar days) of operation, the vehicle traversed 5224 m in 49 h of driving using 1695 driving commands, including about 500 turns. Sixteen signals were sent for protection against excessive tilt during that time; approximately 140 craters were traversed at maximum slope angles of 30°.

With the approximately 2.6 s speed-of-light delay, the operators stated that *control experience confirmed the desirability of movement in a starting-stopping regime with mandatory stopping each few meters.* The soil properties were found to *differ substantially even in terrain sectors not very distant from one another.* The soil penetrometer determined that the upper layer of regolith varied from a stiffest where the penetrometer required about 16 kg (Earth weight) of force to penetrate about 26 mm, to a weakest measurement where only 3 kg of Earth weight caused a penetration of about 39 mm. The cone penetrometer had a base diameter of 50 mm and a cone height of 44 mm. Thus the upper layer of regolith had a rate of increase of loadbearing strength ranging from about 400 kPa m^{-1} for the weakest soil to about 3 MPa m^{-1} for the stiffest soil. Crater walls and the immediate ejecta blanket around craters generally exhibited the weakest soil. Below 5–10 cm of penetration depth, the regolith generally became rapidly stiffer. The mean value of wheel slippage for the first three lunar days was about 10%. On horizontal terrain, the slippage ranged from zero to 15% depending on the surface irregularities and ground inhomogeneity. On crater slopes, the slip increased to 20–30%. The specific resistance of the Lunokhod wheels was generally in the range of 0.05–0.25, while the specific free traction (the ratio of traction to weight) was in the range of 0.2–0.41. The crater distribution in the area explored by Lunokhod 1 was found to be closely approximated by the formula $N(D) = AD^{-\delta}$, where $N(D)$ is the number of craters larger than diameter D meters per hectare of lunar surface, A is a scale factor found to be about 250, and δ is the distribution exponent, found to be about 1.4 [55.24].

55.2.2 Autonomous Rovers

In the mid-1960s, research began on a lunar rover at the US Jet Propulsion Laboratory (JPL) in Pasadena, California, when it was proposed to put a small rover on the Surveyor lunar landers. These landers were led by JPL (based on a system-level contract to Hughes), and were designed to land softly on the Moon to establish the safety of such landing prior to the Apollo landers with humans aboard. At the time it was speculated (notably by T. Gold) that the Moon might be covered in a thick layer of soft dust that would *swallow* any lander. In 1963, JPL issued a contract to build a small rover concept prototype in support of the Surveyor program to the General Motors Defense Research Laboratories in Goleta, California. That GM facility had recently hired *Bekker*, who was considered *the father of off-road locomotion*, having written several seminal textbooks on the subject, and having introduced many of the key concepts relating soil properties to off-road vehicle performance that are still used today [55.29, 30] (Sect. 55.3.12).

Bekker and his team proposed an articulated 6-wheeled vehicle based on a novel 3 cab configuration with an axial spring-steel suspension. This vehicle exhibited remarkable mobility, being able to climb vertical steps up to 3 wheel radii high, and crossing crevasses 3 wheel radii wide. Notable people working with Bekker were Farenc Pavlics, who went on to lead the development of the mobility system for the Apollo lunar rover (under contract from Boeing), and Fred Jindra, who developed the underlying equations describing the mobility of the 6-wheeled articulated vehicle that were later used by Don Bickler in conceiving of the *rocker bogie* chassis used on Sojourner and the Mars Exploration Rovers. Bekker and his team proposed the 6 wheeled articulated vehicle after experimenting with many types of vehicles, including multitracked vehicles, screw-type vehicles (for fine powdered terrain), and others. The 6-wheeled vehicle demonstrated in scale-model testing superior performance in both soft and rocky terrain.

They built and delivered two vehicles that were about 2 m long with approximately 0.5 m wheel diameters. Those vehicles were used in testing throughout the 1960s and early 1970s to conduct simulated operations to determine how such vehicles could actually be used on the Moon. One key issue was that the speed-of-light round trip from the Moon (about 3 s) precluded direct *driving* of the vehicle. Perhaps most annoying was the fact that, during vehicle motion, the highly directional radio antenna used to communicate with Earth would lose its pointing, and so communications would briefly be lost. This meant that operators driving the rover would be confronted with a series of still images, instead of a stream of moving images. It was quickly realized that much of an operator's *situation awareness* and depth perception needed to drive a vehicle with a monocular camera comes from motion. It

was very difficult to drive from frozen monocular images. A crude form of stereo was incorporated where the camera mast was raised and lowered slightly and the operator could switch back and forth between the two views.

Following the successful landing of several of the Surveyor spacecraft, and the discovery that all landing sites seemed to have relatively firm soil, it was concluded that the Surveyor lunar rover was not needed. As a result, the prototype was used for research into the early 1970s, and subsequently restored for use again in research in the 1980s, becoming the first vehicle to be outfitted with *waypoint navigation* of the sort later used on the Sojourner and MER missions.

About the time Viking was conceived and developed, JPL began the *1984* Mars rover effort. (1984 was an energetically favorable launch opportunity from Earth to Mars, and the next likely major mission opportunity after Viking.) Two testbed vehicles were developed, a *software* prototype and a *hardware* prototype. The software prototype had a *Stanford* arm, designed by Vic Scheinman (who went on to design the Unimation PUMA arm and many other famous early robotic devices). This was the only *1.5-scale* Stanford arm ever built. *Lewis* and *Bejczy* became well-known in robotics for solving the kinematics of this arm, one of is not the first full kinematics ever done in robotics up to that time, e.g., [55.31]. A stereo pan-tilt head was implemented and equipped with the first solid-state cameras to become available. A number of very important works were published in the 1977 International Joint Conference on artificial intelligence, e.g., [55.32, 33]. The first *hand-eye-locomotion* coordination was done with this vehicle, where a rock was designated in a stereo image, and the vehicle maneuvered autonomously to a point where the arm could reach out and pick up the rock. One of the first demonstrations of *pin in hole* insertion and other *dexterous* manipulations were also done with this system in the 1970s.

The *hardware* prototype was built using elastic *loop wheels* made by Lockheed [55.34]. The vehicle was battery powered and controlled via a handheld RC unit of the type used by hobbyists.

In late 1982, JPL had a contract with the US Army to study the use of robotic vehicles in support of the US Army. During this study, Brian Wilcox at JPL proposed a technique to reduce the need for a real-time video link or high-bandwidth communication channel between the vehicle and the operator. This technique (which became known as computer-aided remote driving, or CARD) [55.35] required the transmission of a single stereo image from the vehicle to the operator, so the operator could designate waypoints in that image using a 3-D cursor. By use of a single stereo image instead of a continuous stream of monocular images, the amount of information that needed to be transmitted by the vehicle was reduced by orders of magnitude. JPL first demonstrated CARD on the resurrected surveyor lunar rover vehicle (SLRV, which had been painted baby-blue and so became known as the *Blue Rover*, Fig. 55.15a), and later on a modified Humvee. During field tests in the Mojave desert in 1988, CARD was demonstrated on the Humvee with path designations of 100 m per stereo image, and with time-to-designate each path of only a few seconds.

As the CARD work was ongoing, an internally funded effort at JPL demonstrated a concept called *semiautonomous navigation* (SAN). This concept involved humans on Earth designating *global* paths using maps of the sort that could be developed from orbiter imagery, and then having the vehicle autonomously refine and execute a *local* path that avoids hazards. The moderate success of that effort led to a NASA funded effort, leading to the development of a new vehicle, called Robby (Fig. 55.15b). Robby was a larger vehicle that could support the on-board computing and power

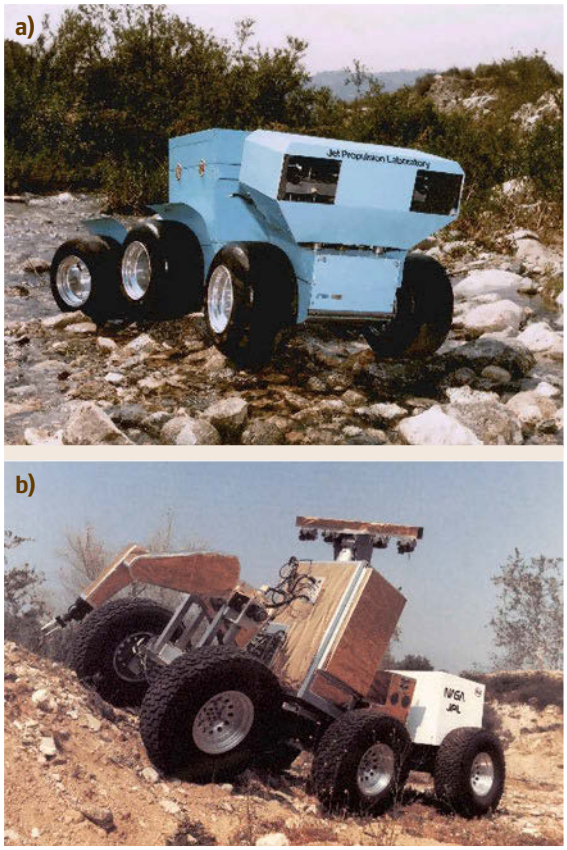


Fig.55.15a,b Six-wheel articulated body rovers developed by JPL **(a)** SLRV and **(b)** Robby

needed for untethered operation. (The SLRV had been tethered to a VAX 11/750 minicomputer over a 1500 foot tether during arroyo field testing of CARD.) For the first time (in 1990), an autonomous vehicle had made a traverse through an obstacle field that was faster than a rover could have done on Mars using human path designation done on Earth.

However, Robby had a severe *public relations* problem – it was perceived as too large. Of course, none of the computers or power systems had been miniaturized or lightweighted – it was composed entirely of the lowest-cost components that could do the job. However, because it was the size of a large automobile, observers and NASA management got the impression that future rovers would be car-sized or even truck-sized vehicles. This was compounded by the Mars rover sample return (MRSR) study done by JPL in the late-1980s which suggested a mass for the rover of 882 kg. An independent study of the MRSR study by Science Applications International, Inc. (SAIC) estimated the overall cost at $13B. When word of this *outrageous* price tag filtered around NASA Headquarters and into the Congressional Staff, MRSR was summarily killed. Robby died along with it. At about the same time, NASA funded Carnegie Mellon University to develop *Ambler* (Fig. 55.16), a large walking robot that was able to autonomously choose safe footfall locations, also as a testbed Mars Rover [55.36, 37]. Ambler had a similar public-relations problem, being about the same mass as Robby, that the NASA management community was very skeptical that such large systems could affordably be flown to Mars. Both Robby and Ambler had all-onboard power and computing systems, which at that time were not sufficiently miniaturized to make autonomous rovers credible for actual flight missions. *Moore's law* was not only causing the computing technology to become miniaturized at a high rate, but also the energy required per computing instruction was dropping rapidly. This meant that early systems devoted most of their power to computing rather than to motive power. Later systems, such as the Mars exploration rovers, have a more nearly equal balance between power for mobility and power for computation. Future systems will presumably devote the majority of their power to mobility as opposed to computation.

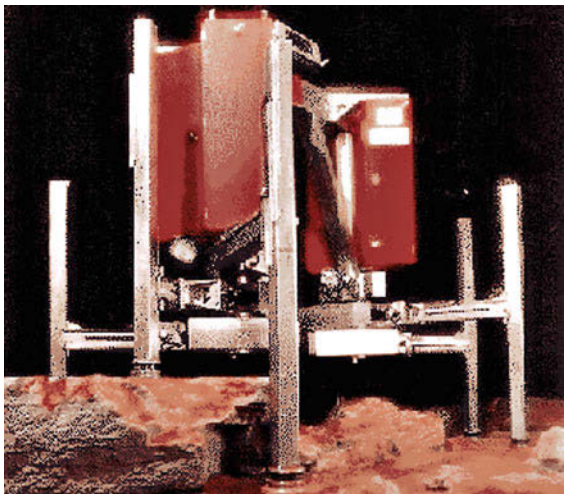


Fig. 55.16 Ambler

Soon thereafter, the Mars Environmental Survey (MESUR) mission set was proposed, as a lower cost alternative to a sample return mission. The MESUR Pathfinder mission was proposed as a first test of what was envisioned as a network of 16–20 surface stations to provide global coverage of Mars. A small rover was proposed to the Mars Science Working group [55.38, 39]. A very short-term development effort culminated in a demonstration in July 1992 of a ≈ 4 kg rover that could move to directed points on the surface nearby a lander using stereo designation of waypoints in a 3-D display of frozen images taken from a lander mast camera pair (Fig. 55.17). This demonstration was sufficiently successful that a similar rover was manifested for the Mars Pathfinder mission. The Pathfinder rover (Fig. 55.18) was later named Sojourner, and became the first autonomous vehicle to traverse the surface of another planet, using a hazard detection and avoidance system to move safely between waypoints through a rockfield [55.25, 26]. The hazard detection system avoided obstacles, and also was used to position the vehicle accurately in front of rocks. Sojourner operated successfully for 83 Mars days (until the failure of the lander, which was acting as a communications relay between the rover and Earth). Sojourner examined approximately a dozen rock and soil samples with its Alpha-Proton-x-ray spectrometer, which gives the elemental composition of the rocks and soil. The success of Sojourner led directly to the decision to build the twin Mars exploration rovers launched in 2003. Both Sojourner and the subsequent Mars exploration rovers *Spirit* and *Opportunity*, and the Mars Science Laboratory *Curiosity*, use waypoint designation in stereo images by the human operator together with autonomous hazard detection and avoidance to keep the rovers safe if they should wander off the designated path.

During the 1992–1993 summer season in Antarctica, the Dante I robot, built by Carnegie-Mellon University and funded by NASA, attempted to rappel into the caldera of the active volcano Mt. Erebus. Dante was a walking robot, and was the first serious attempt to make a robot rappel down a grade that was too steep to traverse using purely frictional contact. Unfortunately, the extreme cold (even in the summer) compounded

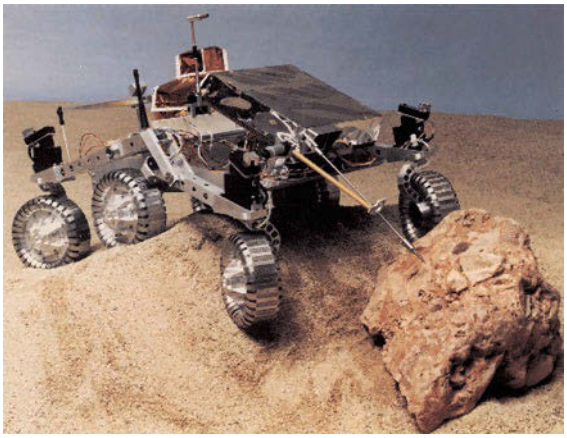


Fig. 55.17 Rocky 4

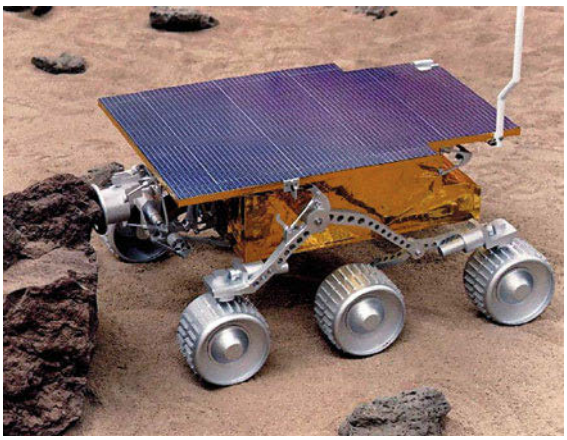


Fig. 55.18 The Pathfinder rover, Sojourner

by human error caused a kink in the fiber-optic umbilical to snag going through an eyelet, breaking the high-bandwidth fiber communications on which the system depended. The fiber could not be repaired in the field, and so the mission was aborted. Undaunted, in the summer of 1994, Dante II (Fig. 55.19) made a successful rappel into the caldera of Mt. Spur in Alaska, exploring the active vents on the crater floor in a way that would be unsafe in the extreme if done with human explorers. The Dante robot series demonstrated that rappelling, especially when combined with legged locomotion, allows robots to conduct exploration to extremely hazardous sites in ways that humans cannot.

In 1984, NASA started the Telerobotics Research program [55.40, 41]. This program demonstrated various aspects of on-orbit assembly, maintenance, and servicing. Some highlights of this activity were the automated tracking and grappling of a free-spinning satellite (suspended with a counterweight and gimbal

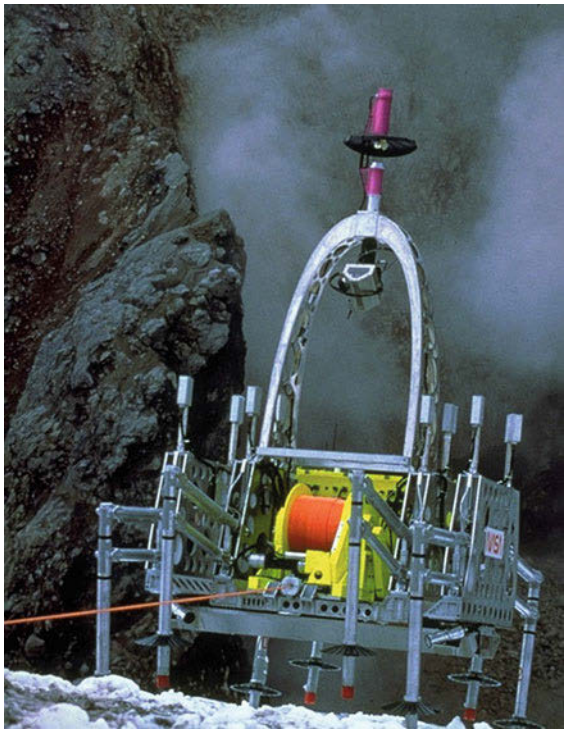


Fig. 55.19 Dante II at Mt. Spur in Alaska

for realistic reactions under external forces), connection of a flight-like fluid coupler, and many *busy box* functions such as door opening, threaded fastener mating and demating, use of power tools, dual-arm manipulation of a simulated hatch cover and flexible thermal blanket, etc., by various control approaches ranging from force-reflecting teleoperation to fully autonomous sequences. This activity ended in about 1990.

55.2.3 Research Systems

There have been many mobile robots built by government, university, and industrial groups whose objective was to develop new technologies for planetary surface exploration, or to excite students or young engineers about the possibilities in that area. Carnegie Mellon University developed the Ambler, Dante, Nomad, Hyperion, Zoe, and Icebreaker robot series. The Jet Propulsion Laboratory (JPL), Draper Labs, MIT, Sandia National Lab, and Martin-Marietta (later Lockheed-Martin) each built more than one planetary surface robot testbed. The Marsokhod chassis built by VNII Transmach of St. Petersburg, Russia was used by research groups there and also in Toulouse (LAAS and CNRS) [55.42, 43] as well as the NASA Ames Research Center and McDonnell Douglas Corporation (later part of the Boeing Company) in the US.

These research platforms have been used for two basic avenues of research. One avenue is to perfect safe driving techniques on planetary surfaces, despite the speed-of-light latencies inherent in robotic exploration of the planets. This includes the waypoint navigation technology developed at JPL in the 1980s, where frozen stereo images are used to plan a possibly lengthy series of waypoints or activity sites, and then executed with various sorts of reflexive hazard avoidance or safing techniques, such as used on the Sojourner rover on Mars in 1997. The other is to develop higher level autonomy for improved science data return or mission robustness. Technologies in this latter category include mission planners that attempt to optimize routes and activity sequences based on time, limits to peak power, total energy, expected temperature, illumination angles, availability of communications, and others.

Fig. 55.20 The Mars exploration rovers, Spirit and Opportunity, with a manipulator arm in front

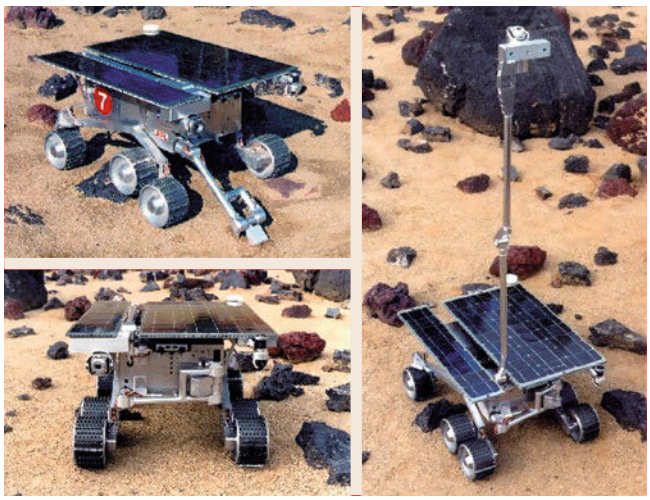


Fig. 55.21 Rocky 7

Automated classification of possible science targets based on clustering of spectral data, figure-ground segmentation of rocks, and other approaches have been attempted with some success. At the time of this writing, some of these technologies have been uploaded to the twin Mars exploration rovers *Spirit* and *Opportunity* (Fig. 55.20) [55.27], including automated detection of temporary events of scientific interest such as dust-devils and clouds [55.44]. The Spirit rover was lost when it broke through a surface crust into a small crater filled with fine dust in late 2009, with attempts to free it continuing until mid 2010. The Mars Science Laboratory *Curiosity* was landed in August 2012 and seeks to explore Mt. Sharpe in the center of Gale Crater.

The ESA with a number of contractors, e.g., ASTRIUM and DLR has been working for several years on a 6-wheeled Mars rover ExoMars that is supposed to be flown after 2018 as well as on a *next lunar lander* NLL Rover, but it is not finally clear, when and in which configuration the mission might be realized.

55.2.4 Sensing and Perception

In the 1980s, most planetary rover sensing research was based on laser ranging or stereo vision. Stereo vision was too computationally intensive for early low-power, radiation-hard processors, so the Sojourner Mars rover used a simple form of laser ranging to determine which areas were safe to traverse. Between the launch of Sojourner (1996) and the Mars exploration rovers (2003) sufficient progress had been made in radiation-hardened flight processors that stereo vision was used for hazard detection on MER [55.45], mostly in experiments conducted with the Rocky-7 rover (Fig. 55.21). This allowed much larger numbers of range points to be incorporated into the hazard-detection algorithm (thousands of points, instead of the 20 discrete range points used by Sojourner). Perception of hazards on Sojourner was based on simple computations of average slope and roughness over the 4×5 array of range measurements, as well as the maximum height differences.

The two MER rovers and the MSL rover use a more sophisticated evaluation of the safety of the rovers along a large number of candidate arcs from its current location. Many other algorithms for perception of terrain hazards have been used with reasonable success by various organizations. Today it is probably fair to say that the *unsolved problems* lie not in the area of geometric hazards (e.g., hazards that can be evaluated completely based on accurate knowledge of the shape of the terrain) but rather in the area of nongeometric hazards (e.g., hazards where uncertainties in

the load-bearing or frictional properties of the terrain determine the safety of a proposed traverse). Accurate estimation of load-bearing or friction properties of terrain by remote sensing is a very challenging task that will not be completely *solved* anytime soon, if ever.

55.2.5 Estimation

Most estimation for planetary surface exploration relates to the internal state of the robot, or its position, pose and kinematic configuration with respect to the environment. Internal state sensors such as encoders on any active or passive articulations in the vehicle are used, along with a kinematic model and inertial sensors such as accelerometers and gyroscopes, to estimate the pose of the vehicle in inertial space. Perceptual algorithms such as surface reconstruction from clouds of range-points as developed by stereo vision put terrain geometry estimates into this same representation. Heading in inertial space is generally the most difficult to reliably estimate, due to the lack of navigation aids such as the global positioning system, or any easily measured heading reference such as a global magnetic field. Integration of rate-gyro data is used to maintain local attitude during motion while accurate estimation of the rotation axis for Mars is possible by long integrations of 3 axis rate-gyro data while the vehicle is stopped. Similar approaches are probably not feasible for the Moon, because of its slow rotation rate. Imaging of the solar disk or constellations of stars at precisely known times can be combined with stored models of the rotation of the planet to allow accurate estimation of the complete pose of the vehicle in inertial space. Kalman filtering or related techniques are generally employed to reduce the effects of measurement noise.

55.2.6 Manipulators for In-Situ Science

The Mars exploration rovers were the first planetary exploration vehicles to have general-purpose manipulators. (Lunokhod had a single-purpose soil penetrometer, and Sojourner had a single degree-of-freedom device to place an Alpha-Proton X-Ray spectrometer in direct contact with the terrain.) The MER arms each have 5°-of-freedom and a reach of over 1 m. Accurate gravity-sag models of the lightweight arm allow the precise position to be predicted in advance of any command to deploy an instrument, and contact sensors allow the arm to stop before any excessive forces build up in the relatively flexible arm. Future arms for planetary surface operations, especially any proposed assembly, maintenance, or servicing tasks as part of the proposed lunar outpost, will require force sensing to protect the stronger but much more rigid arms from damage, as well as to allow controlled forces to be applied to the terrain or workpieces. Of course, there is a huge body of knowledge associated with industrial robot arms, and undersea robotics (e.g., for the offshore oil industry), but such arms are generally very heavy, fast, and stiff compared with credible systems for planetary surface use. Delicate force control has rarely been applied to industrial settings. Space hardware is necessarily very lightweight, and so both the arms and the workpieces will need to have well-resolved force sensing and control to prevent damage to one or both. Because of severe limits on both mass and power, and to avoid unnecessary risk, space manipulation tends to be slow. Historically, this means the gear ratio between each motor and the corresponding output shaft is very large, making the use of motor current as an estimator for output torque very problematic. Other low-mass and robust means for accurate sensing of applied forces in the space environment are needed.

55.3 Mathematical Modeling

Broadly speaking, both on-orbit manipulators and surface mobile robots are considered to be common articulated body systems with a moving base. One point that clearly distinguishes them from other ground-based robots, such as industrial manipulators, is the existence of a moving base.

55.3.1 Space Robot as an Articulated Body System

The robotic systems discussed in this chapter comprise one or multiple articulated limbs mounted on a base body that has a dynamic coupling with these limbs. Typical styles of such moving base systems are categorized into the following groups [55.46].

Free-Floating Manipulator Systems

A space free-flyer that has one or more manipulator arms, as illustrated schematically in Fig. 55.22a, is a typical example of this group. When operating the manipulator arm(s), the position and orientation of the base spacecraft fluctuates due to the manipulator reaction. The kinetic momentum of the system is conserved if no external forces or moments are applied, and the conservation law for this system governs the reaction dynamics. Coordination or isolation, between the base

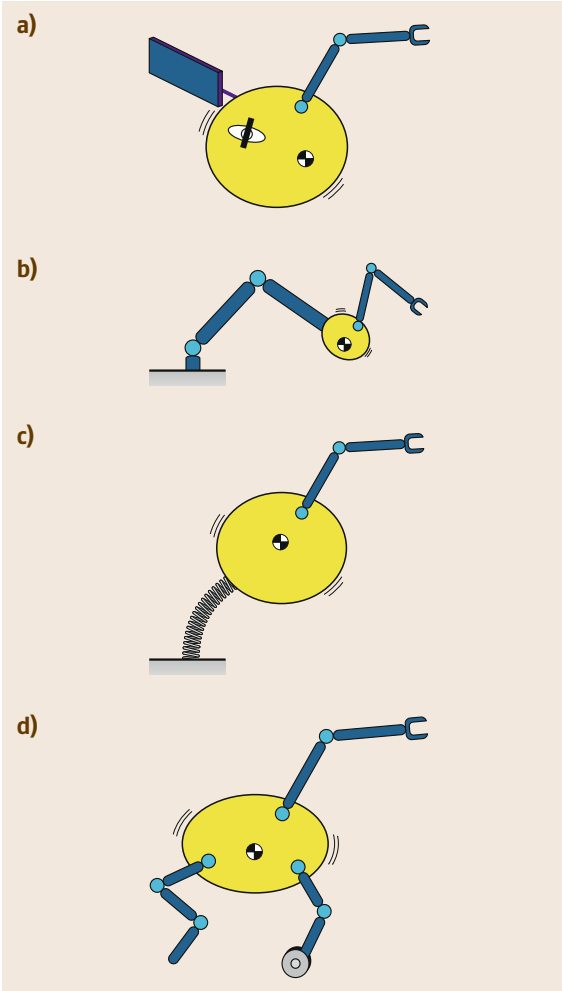


Fig.55.22a–d Four basic types for moving base robots: **(a)** Free-floating manipulator system, **(b)** macro–micro manipulator system, **(c)** flexible structure-based manipulator system, and **(d)** surface locomotive robot system

and manipulator dynamics is key to advanced motion control.

Macro–Micro Manipulator Systems

A robotic system that comprises a relatively small arm (microarm) for fine manipulation mounted on a relatively large arm (macro-arm) for coarse positioning, is called a macro–micro manipulator system. The SSRMS (Canadarm2) and the SPDM (Dextre) system, as well as the JEMRMS on the Japanese module of the ISS, are good examples. Here, the connecting interface at the end point of the macro arm or the root of the micro arm is modeled as the base body (Fig. 55.22b). A free-flying space robot may be treated in this group when its base body is actively controlled by actuators that produce external forces and moments, such as gas-jet thrusters. In this case, these actuators can be modeled as a virtual macro-arm [55.47].

Flexible-Based Manipulator Systems

If the macro arm behaves as a passive flexible (elastic) structure in a macro–micro manipulator system, the system is considered to be a flexible-based manipulator (Fig. 55.22c). Such a situation can be observed in the operation of the ISS, when the SSRMS is servo or brake locked after its coarse operation. Here, the issue is that the base body (the root of the micro arm, or the end of the macro arm, according to the definition above) is subject to vibrations that will be excited by the reaction of the micro arm.

Mobile Robots with Articulated Limbs

Mobile robots for surface locomotion have the same structure in terms of the dynamics equation as the above groups. This group includes wheeled vehicles, walking (articulated limb) robots, and their hybrids. In a wheeled vehicle, suspension mechanisms, if any, are also modeled as articulated limb systems. The forces and moments yielded by contact with ground, or planetary surface, govern the motion of the system (Fig 55.22d).

55.3.2 Equations for Free-Floating Manipulator Systems

Let us first consider a free-floating system with a single or multiple manipulator arm(s) mounted on a base spacecraft. The pioneering work in the mathematical models of this type of space manipulator systems were conducted in late 1980s and early 1990s and collected in the book [55.48], published in 1993. In this section, the models that are widely accepted today are introduced.

The base body, termed link 0, is floating in inertial space without any external forces or moments. At the end point of the arm(s), external forces/moments may apply. For such a system, the equation of motion is expressed as follows

$$\begin{pmatrix} \mathbf{H}_\mathrm{b} & \mathbf{H}_\mathrm{bm} \\ \mathbf{H}_\mathrm{bm}^\mathrm{T} & \mathbf{H}_\mathrm{m} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{x}}_\mathrm{b} \\ \ddot{\boldsymbol{\phi}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_\mathrm{b} \\ \boldsymbol{c}_\mathrm{m} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\tau} \end{pmatrix} + \begin{pmatrix} \mathbf{J}_\mathrm{b}^\mathrm{T} \\ \mathbf{J}_\mathrm{m}^\mathrm{T} \end{pmatrix} \boldsymbol{F}_\mathrm{h} \,. \tag{55.1}$$

The kinematic relationship among $\boldsymbol{x}_\mathrm{h}$, $\boldsymbol{x}_\mathrm{b}$, and $\boldsymbol{\phi}$ is expressed using Jacobian matrices as

$$\dot{\boldsymbol{x}}_\mathrm{h} = \mathbf{J}_\mathrm{m}\dot{\boldsymbol{\phi}} + \mathbf{J}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,, \tag{55.2}$$

$$\ddot{\boldsymbol{x}}_\mathrm{h} = \mathbf{J}_\mathrm{m}\ddot{\boldsymbol{\phi}} + \dot{\mathbf{J}}_\mathrm{m}\dot{\boldsymbol{\phi}} + \mathbf{J}_\mathrm{b}\ddot{\boldsymbol{x}}_\mathrm{b} + \dot{\mathbf{J}}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,. \tag{55.3}$$

Where the notations are listed as follows:

$\boldsymbol{x}_\mathrm{b} \in \mathbb{R}^6$: position/orientation of the base
$\boldsymbol{\phi} \in \mathbb{R}^n$: joint angle of the arm
$\boldsymbol{x}_\mathrm{h} \in \mathbb{R}^{6k}$: position/orientation of the end point(s)
$\boldsymbol{\tau} \in \mathbb{R}^n$: joint torque of the arm
$\boldsymbol{F}_\mathrm{h} \in \mathbb{R}^{6k}$: external forces/moments on the end point(s)
n: number of total joints
k: number of manipulator arms,

and $\mathbf{H}_\mathrm{b}$, $\mathbf{H}_\mathrm{m}$, and $\mathbf{H}_\mathrm{bm}$ are inertia matrices for the base body, manipulator arm, and the coupling between the base and the arm, respectively, and $\boldsymbol{c}_\mathrm{p}$ and $\boldsymbol{c}_q$ are nonlinear Coriolis and centrifugal forces, respectively.

For a free-floating manipulator in orbit, the gravity forces exerted on the system can be neglected, and so the nonlinear term becomes

$$\boldsymbol{c}_\mathrm{b} = \dot{\mathbf{H}}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} + \dot{\mathbf{H}}_\mathrm{bm}\dot{\boldsymbol{\phi}}_\mathrm{b} \,.$$

Integrating the upper set of equation in (55.1) with respect to time, we obtain the total momentum of the system as

$$\mathcal{L} = \int \mathbf{J}_\mathrm{b}^\mathrm{T}\boldsymbol{F}_\mathrm{h}\,\mathrm{d}t = \mathbf{H}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} + \mathbf{H}_\mathrm{bm}\dot{\boldsymbol{\phi}} \,. \tag{55.4}$$

For the case in which reaction wheels are mounted on the base body, they are included as additional manipulator arms.

55.3.3 Generalized Jacobian and Inertia Matrices

From (55.2) and (55.4), the coordinates of the manipulator base $\dot{\boldsymbol{x}}_\mathrm{b}$, which are passive and unactuated coordinates, can be eliminated, as follows

$$\dot{\boldsymbol{x}}_\mathrm{h} = \hat{\mathbf{J}}\dot{\boldsymbol{\phi}} + \dot{\boldsymbol{x}}_\mathrm{h0} \,, \tag{55.5}$$

where

$$\hat{\mathbf{J}} = \mathbf{J}_\mathrm{m} - \mathbf{J}_\mathrm{b}\mathbf{H}_\mathrm{b}^{-1}\mathbf{H}_\mathrm{bm} \,, \tag{55.6}$$

and

$$\dot{\boldsymbol{x}}_\mathrm{h0} = \mathbf{J}_\mathrm{b}\mathbf{H}_\mathrm{b}^{-1}\mathcal{L} \,. \tag{55.7}$$

Since $\mathbf{H}_\mathrm{b}$ is the inertia tensor of a single rigid body (the manipulator base), it is always positive definite, and so its inverse exists.

The matrix $\hat{\mathbf{J}}$ was first introduced in [55.49, 50] and is referred to as the *generalized jacobian matrix*. In its original definition, it was assumed that no external forces/moments acted on the system. If $\boldsymbol{F}_\mathrm{h} = \mathbf{0}$, then the term $\dot{\boldsymbol{x}}_\mathrm{h0}$ becomes constant, and, in particular, if the system has zero initial momentum $\dot{\boldsymbol{x}}_\mathrm{h0} = \mathbf{0}$, then (55.5) becomes simple. However, note that in the derivation of (55.6), zero or constant momentum is not a necessary condition.

Using this matrix, the manipulator hand can be operated under a resolved motion-rate control or resolved acceleration control in inertial space. Thanks to the generalized Jacobian, although the reactive base motion occurs during the operation, the hand is not disturbed by the motion.

From the upper and lower sets of equations in (55.1), $\ddot{\boldsymbol{x}}_\mathrm{b}$ can be eliminated to obtain the following expression

$$\hat{\mathbf{H}}\ddot{\boldsymbol{\phi}} + \hat{\boldsymbol{c}} = \boldsymbol{\tau} + \hat{\mathbf{J}}\boldsymbol{F}_\mathrm{h} \,, \tag{55.8}$$

where

$$\hat{\mathbf{H}} = \mathbf{H}_\mathrm{m} - \mathbf{H}_\mathrm{bm}^\mathrm{T}\mathbf{H}_\mathrm{b}^{-1}\mathbf{H}_\mathrm{bm} \,. \tag{55.9}$$

The matrix $\hat{\mathbf{H}}$ is known as the *generalized inertia matrix* for space manipulators [55.48]. This matrix represents the inertia property of the system in the joint space and can be mapped onto the Cartesian space using the generalized Jacobian matrix

$$\hat{\mathbf{G}} = \hat{\mathbf{J}}\hat{\mathbf{H}}^{-1}\hat{\mathbf{J}}^\mathrm{T} \,. \tag{55.10}$$

The matrix $\hat{\mathbf{G}}$ is referred to as the *inversed inertia tensor* for space manipulators and is useful for the discussion of impact dynamics when a space manipulator collides with or captures a floating target in orbit [55.51].

The generalized Jacobian matrix (GJM) is a useful concept, with which the manipulator end point can perform positioning or trajectory tracking control by a simple control algorithm regardless of the attitude deviation during the operation.

A simplified laboratory demonstration was carried out using a two-dimensional free-floating test bed called *EFFORTS* [55.52]. To simulate the motion in a microgravity environment, a robot model was floated on a thin film of pressurized air on a horizontal plate, so that frictionless motion with momentum conservation was achieved.

Figure 55.23 depicts the test bed and a typical experimental result. For Fig. 55.23b, the control command was given to the floating robot by

$$\dot{\boldsymbol{\phi}} = \hat{\mathbf{J}}^{-1}\dot{\boldsymbol{x}}_\mathrm{d} \,, \tag{55.11}$$

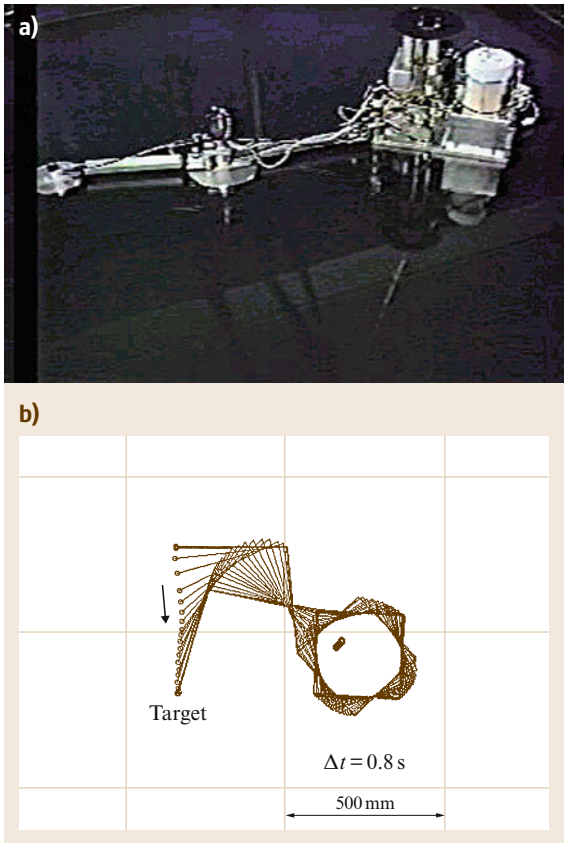


Fig.55.23a,b Laboratory test bed for a free-floating space robot: **(a)** The EFFORTS test bed, **(b)** a target capture result

where $\dot{\boldsymbol{x}}_{\mathrm{d}}$ is the desired velocity of the manipulator end point, the value of which was given and updated by on-line measurement of the end point position $\boldsymbol{x}_{\mathrm{h}}$ and the target position $\boldsymbol{x}_{\mathrm{t}}$, as follows

$$\dot{\boldsymbol{x}}_{\mathrm{d}} = \frac{\boldsymbol{x}_{\mathrm{t}} - \boldsymbol{x}_{\mathrm{h}}}{\Delta t} \,, \tag{55.12}$$

where Δt is the time interval of the on-line control loop. The desired end point velocity was simply resolved into the joint velocity by (55.11).

The result clearly shows that the manipulator end point properly reached the target in an optimum manner, although the robot base rotated considerably due to the manipulator reaction. Note that, since the target was stationary in this example, the resulting motion trace was a straight line. However, thanks to the on-line control, the manipulator was also able to track and reach a moving target with the same control law.

The validity and effectiveness of the GJM-based manipulator control were also demonstrated in orbit by Japanese ETS-VII mission [55.14].

55.3.4 Linear and Angular Momenta

The integral of the upper set of equation in (55.1) gives a momentum equation, as shown in (55.4), which is composed of the linear and angular momenta. The linear part is expressed as

$$\breve{\mathbf{H}}_{\mathrm{b}} \boldsymbol{v}_{\mathrm{b}} + \breve{\mathbf{H}}_{\mathrm{bm}} \dot{\boldsymbol{\phi}} = \boldsymbol{P} \,, \tag{55.13}$$

where $\boldsymbol{v}_{\mathrm{b}}$ is the linear velocity of the base, $\boldsymbol{P}$ is the initial linear momentum, and the inertia matrices with the mark of $\breve{(\cdot)}$ are the corresponding components for the linear momentum [55.48]. When the linear momentum is further integrated, the result verifies the principle that the mass centroid of the entire system either remains stationary or translates with a constant velocity.

The angular momentum equation, however, does not have a second integral, and therefore provides a first-order nonholonomic constraint [55.53]. The equation can be expressed as

$$\tilde{\mathbf{H}}_{\mathrm{b}} \boldsymbol{\omega}_{\mathrm{b}} + \tilde{\mathbf{H}}_{\mathrm{bm}} \dot{\boldsymbol{\phi}} = \boldsymbol{L} \,, \tag{55.14}$$

where $\boldsymbol{\omega}_{\mathrm{b}}$ is the angular velocity of the base, $\boldsymbol{L}$ is the initial angular momentum, and the inertia matrices with the mark of $\tilde{(\cdot)}$ are the corresponding components for the angular momentum [55.48]; $\tilde{\mathbf{H}}_{\mathrm{bm}} \dot{\boldsymbol{\phi}}$ represents the angular momentum generated by the manipulator motion.

Equation (55.14) can be solved for $\boldsymbol{\omega}_{\mathrm{b}}$ with zero initial angular momentum

$$\boldsymbol{\omega}_{\mathrm{b}} = -\tilde{\mathbf{H}}_{\mathrm{b}}^{-1} \tilde{\mathbf{H}}_{\mathrm{bm}} \dot{\boldsymbol{\phi}} \,. \tag{55.15}$$

This expression describes the resulting disturbance motion of the base when there is joint motion $\dot{\boldsymbol{\phi}}$ in the manipulator arm.

There are a number of points worth discussion when analyzing this equation. The magnitudes and directions of the maximum and minimum disturbances can be obtained from the singular value decomposition of the matrix $\left(-\tilde{\mathbf{H}}_{\mathrm{b}}^{-1} \tilde{\mathbf{H}}_{\mathrm{bm}}\right)$ and displayed on the map. Such a map is called a *disturbance map* [55.54, 55]. Equation (55.15) is also used for the feed-forward compensation in the coordinated manipulator/base control model [55.56, 57].

55.3.5 Virtual Manipulator

The concept of the *virtual manipulator* (VM) is an augmented kinematic representation that considers the base motion due to reaction forces or moments. The model is based on the fact that the mass centroid of the entire system does not move in the free-floating system without any external forces [55.58]. The mobility

of the end point of the arm is decreased by the base motion. In the VM representation, such mobility degradation is expressed by virtually shrinking the length of the real arm according to the mass property. Note that the VM considers only linear momentum conservation. If the differential expression of VM is obtained using a Jacobian matrix, the Jacobian is not a conventional kinematic Jacobian, but rather a version of the generalized Jacobian defined by the combination of the kinematic equation (55.2), and the linear momentum equation (55.13).

55.3.6 Dynamic Singularity

Dynamic singularities are singular configurations in which the manipulator end point loses mobility in some inertial direction [55.59]. Dynamic singularities are not found in earth-based manipulators, but rather occur in free-floating space manipulator systems due to the coupling dynamics between the arm and the base. Dynamic singularities coincide with the singularities of the generalized Jacobian matrix determined by (55.6). The singular value decomposition (SVD) of a manipulator Jacobian matrix provides *manipulability* analysis. Likewise, the SVD of the generalized Jacobian matrix yields the manipulability analysis of a free-floating space manipulator [55.60]. Figure 55.24 shows the comparison of the manipulability distribution between a 2-DOF ground-based manipulator and a 2-DOF floating manipulator, from which the degradation of the manipulability is observed in the space arm due to the dynamic coupling.

55.3.7 Reaction Null-Space (RNS)

From a practical point of view, any change in the base attitude is undesirable. As such, manipulator motion planning methods that minimize the base attitude disturbance have been investigated extensively. Analysis of the angular momentum equation reveals that the ultimate goal of achieving zero disturbance is possible.

The following is the angular momentum equation with zero initial angular momentum $\boldsymbol{L} = \boldsymbol{0}$ and the zero attitude disturbance $\boldsymbol{\omega}_{\mathrm{b}} = \boldsymbol{0}$ given in (55.14)

$$\tilde{\mathbf{H}}_{\mathrm{bm}}\dot{\boldsymbol{\phi}} = \boldsymbol{0}\,. \tag{55.16}$$

This equation yields the following null-space solution

$$\dot{\boldsymbol{\phi}} = (\mathbf{I} - \tilde{\mathbf{H}}_{\mathrm{bm}}^{+}\tilde{\mathbf{H}}_{\mathrm{bm}})\dot{\boldsymbol{\zeta}}\,. \tag{55.17}$$

The joint motion given by (55.17) is guaranteed not to disturb the base attitude. Here, the vector $\dot{\boldsymbol{\zeta}} \in \mathbb{R}^n$ is arbitrary and the null-space of the inertia matrix $\tilde{\mathbf{H}}_{\mathrm{bm}} \in \mathbb{R}^{3\times n}$ is called the *reaction null-space (RNS)* [55.61].

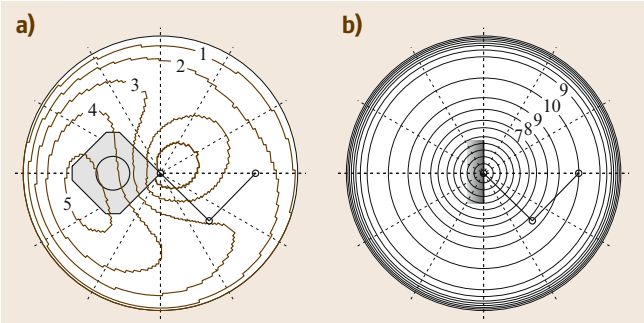


Fig.55.24a,b Manipulability distribution in the work space (after [55.60])

The DOF for $\dot{\boldsymbol{\zeta}}$ is $n-3$. For example, if a manipulator arm mounted on a free-floating space robot has 6-DOF, i. e., $n = 6$, then 3-DOF remain in the reaction null-space. These DOFs can be specified by introducing additional motion criteria, such as end point positioning of the arm. Such manipulator operation that produces no reaction in the base is called the *reactionless manipulation* [55.62].

The validity and effectiveness of the RNS-based reactionless manipulation were demonstrated in orbit by Japanese ETS-VII mission [55.14].

55.3.8 Motion Planning Issues

This subsection addresses the generation of feasible trajectories for a free-flying robot for executing typical point-to-point or grasping tasks. The subject falls under the problem domain of motion planning, with the aim of satisfying motion constraints which generally cannot be satisfied with use of local methods alone (feedback control and model predictive control). A trajectory resulting from the motion planning is fed to a tracking controller, which accounts for any modeling errors, to accomplish the task in question. This approach also aims at providing autonomous skills for supporting a human ground operator.

A typical task of interest here is that of a point-to-point maneuver of a robot manipulator mounted on a servicer satellite, to bring its end-effector in some desired inertial position and orientation. This task may require actuation of the servicer, or may, if possible, be preferably executed in free-floating mode, to avoid issues related to the use of the on-board thrusters, such as fuel expenditure. In this context, to minimize the attitude change of the servicer resulting from the manipulator motion, a noticeable fundamental result for a point-to-point maneuver, derived from nonlinear optimization theory, is the V-maneuver [55.63]. Intuitively,

the attitude change is minimized by making the robot first move radially inward, toward the system center of mass, before turning and then radially outward, to reach the new desired final position.

A second task that has received much attention is the grasping of a free tumbling target, like a defected satellite. Currently, there are a number of space programs worldwide which are addressing this task, like the preparation of the demonstration mission DEOS of the DLR (Sect. 55.1.2) for grasping a small satellite in LEO, the e.Deorbit study of the ESA for the deorbiting of the defected ENVISAT satellite and PHOENIX of DARPA for grasping a geostationary satellite in the graveyard orbit. Different approaches can be found in the literature to tackle this problem, to include [55.64, 65] for feedback control, [55.66] for model predictive control, [55.67] for optimal control in Cartesian space and [55.68] for nonlinear optimization, to mention some.

Due to the nonholonomic nature of the dynamics of a free-floating robot (Sect. 55.3.4), in order to satisfy all relevant motion constraints, the motion planning problems above can only be solved through numerical integration. Note in fact, that the final system configuration for a given final end-effector position in inertial space is function of the whole path taken by the robot throughout the motion.

Principally, the motion planning tasks above may be formulated as an optimal control problem of the type

$$\min_{t_\mathrm{f},\boldsymbol{\phi}(t)} \Gamma(\boldsymbol{\phi}(t), \boldsymbol{\tau}(t), t_\mathrm{f}) \,, \tag{55.18}$$

subject to

$$g(t_\mathrm{f}, \boldsymbol{\phi}(t)) = 0 \tag{55.19}$$

$$h(t_\mathrm{f}, \boldsymbol{\phi}(t)) \leq 0 \,, \tag{55.20}$$

for $0 \leq t \leq t_\mathrm{f}$ and where t_f is the final time; $\boldsymbol{\phi}(t)$ is the vector of joint positions; $\boldsymbol{\tau}(t)$ is the vector of joint torques; Γ is a predefined cost function; g are equality constraints to include, for example, the state transition equations; h are inequality constraints, for example the joint box constraints on position, velocity and torque, or collision avoidance constraints. Other motion constraints may include inequality constraints on the end-effector forces during contact, or other operational constraints.

This consists of an infinite dimensional problem, in the given time interval, which generally cannot be solved in closed form. The direct shooting optimization methods lend themselves well to solving these problems iteratively [55.69], where for example the independent DOF (in the case of a free-floating system, the robot joint angles) are parameterized in time as

$$\boldsymbol{\phi} = \boldsymbol{\phi}(t, \boldsymbol{p}) \tag{55.21}$$

with $\boldsymbol{p} \in \mathbb{R}^n$, for n optimization parameters, where $\boldsymbol{\phi}$ may be a polynomial function, a B-spline function, or any other. This way, the problem becomes that of finding a suitable value for the parameters $\boldsymbol{p}$ which satisfies all motion constraints (feasibility), as well as perhaps minimizes a cost function (optimality), like the time-to-collision [55.70], the end time [55.67], or the mechanical energy [55.68]. The problem is as such transformed into a finite dimensional problem and can be solved as a nonlinear programming problem (NLP) with classical numerical iterative methods such as sequential quadratic programming (SQP).

For example, for a Cartesian point-to-point problem, we have the following supplementary equality constrains at the end time

$$\boldsymbol{X}^\mathrm{e} = \int_0^{t_\mathrm{f}} \hat{\mathbf{J}}\dot{\boldsymbol{\phi}}(t, \boldsymbol{p})\, \mathrm{d}t = \boldsymbol{X}^\mathrm{e}_\mathrm{des} \,, \tag{55.22}$$

where $\boldsymbol{X}^\mathrm{e} \in \mathbb{R}^n$ denotes the end-effector pose (array matrix of dimension DOFe $\times$ 1). Note that the constraint itself has $\infty^{\mathrm{DOF+3-DOFe}}$ solutions for the system configuration (for DOF robot joints), however these all imply a specific base body attitude as well as a specific robot configuration. Solving the problem of reaching a specific system configuration is a hard nonholonomic control problem, which is avoided by solving the constraint thorough the integral above and an adequate parameter set $\boldsymbol{p}$.

Extensions of this are necessary to treat the grasping task. The latter can ideally be separated into three phases: approach, tracking and, stabilization. The first comprises a point-to-point maneuver, however with a nonzero end velocity. The second comprises a tracking maneuver, in which the Cartesian motion of the robot end-effector is dictated by the tumbling motion and the geometry of the target as it follows the grasping point and homes in onto it, to finally grasp it. Note that this phase aims at minimizing the impact between the end-effector and the target. The third phase involves a robot joint velocity decay maneuver, once the target is grasped. This formulation results in a multiple-phase problem, for which the boundaries between the phases introduce supplementary motion constraints in the motion planning problem.

From a methodological point of view, note that there is no simple measure to determine if and when the grasping point on the target will be reachable from the current robot configuration (Fig. 55.25) and whether

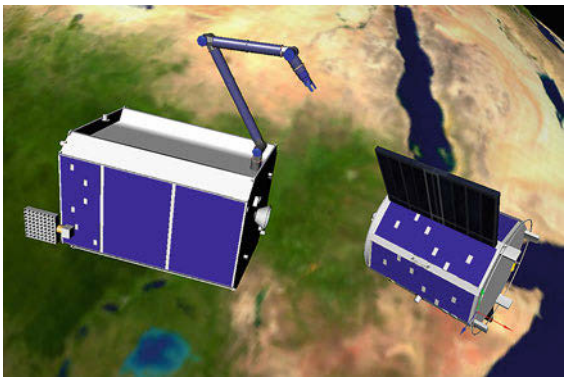


Fig. 55.25 Orbital scenario: servicer satellite with 7-DOF manipulator and target satellite with solar panels. Coordinate system of a predefined grasping point on target ring shown

the trajectory which derives from a local control law will be feasible at all times (accounting for the motion constraints listed above). It is also necessary to provide information on the time synchronization between the motion of the grasping point on the target and that of the robot. Furthermore, the nonlinear nature of the robot kinematics needs to be exploited to favor a successful grasp. These considerations speak in favor of the use of a reference trajectory, which is computed off-line by means of a global search, based on a motion prediction of the target and on a geometric model of the same target and of the space robot [55.68].

It is also well known that optimization methods suffer from convergence issues (arising from local minima), if a judicious initial guess is not available. It is for this reason that a look-up table approach is necessary [55.68] in order to provide a sufficiently high probability of convergence for a given grasping task.

The necessity of a look-up table also arises from the long computation times of the optimization process, which result from the aforementioned necessity to integrate the equations of motion for any given optimization iteration. This time is generally reduced if a solution close to the sought one is given as a starting point. An attempt to eliminate the necessity of integrating the equations of motion was also made in [55.71], where a differentially flat representation for a free-floating robot was sought. In such a formulation, there are as many differential equations as there are independent state variables, or flat variables, and as such, any parameterization of the flat variables is solution of the equations of motion. Such a representation was found for the case in which the robot has three joints and the center of mass of its load lies on the rotation axis of the last joint.

It is also of interest to make some considerations on practical technological issues which can influence one or the other future research direction. With regard to the minimization of the servicer attitude change during robotic operations, it is worth realizing that this is generally only an issue for maintaining the communication link from low Earth orbit with a geostationary satellite, for which a high pointing accuracy is required (note that for a communication link from low Earth orbit to ground instead, an omnidirectional antenna is sufficient). Therefore, rather than limiting the robot workspace to its reaction null-space to avoid any attitude change at all, in which case the resulting robot movements would generally be confined to the extent of becoming of little practical use, a simple technological solution is possible through the implementation of a gimbal joint for the communication antenna on the servicer.

Another simple technological solution to a theoretically involved problem is that of controlling the attitude of the servicer by means of adequate closed-loop maneuvers of the robot. As is well known, reaction wheels on a satellite achieve the same result, also with the same principle of momentum transfer, but however with a far simpler control law than that necessary for a robot. It is then important to realize that, although reaction wheels were until today far too small for any useful robotic application, larger reaction wheels are currently being developed, for example, in the context of the BIROS mission of the DLR, which is planned to fly in 2014. Although these will still not be able to fully compensate the typical robot dynamic coupling terms (if wanting to fully stabilize the servicer attitude), they will however be able to allow significant attitude slew maneuvers of the servicer quickly, without having to resort to the use of the robot or of the thrusters. Furthermore, they will be useful to reduce the robot singularities, due to the enhanced system's actuation.

55.3.9 Equations for Flexible-Based Manipulator Systems

Next, let us consider a flexible-based manipulator system in which a single or multiple manipulator arm(s) are supported by a flexible beam or a spring and damper (visco-elastic) system. For such a system, the equation of motion is obtained using the following variables:

$\boldsymbol{x}_{\mathrm{b}} \in \mathbb{R}^6$: position/orientation of the base
$\boldsymbol{\phi} \in \mathbb{R}^n$: joint angle of the arm
$\boldsymbol{x}_{\mathrm{h}} \in \mathbb{R}^{6k}$: position/orientation of the end point(s)
$\boldsymbol{F}_{\mathrm{b}} \in \mathbb{R}^6$: forces/moments to deflect the flexible base
$\boldsymbol{\tau} \in \mathbb{R}^n$: joint torque of the arm
$\boldsymbol{F}_{\mathrm{h}} \in \mathbb{R}^{6k}$: external force/torque on the end point(s).

$$\begin{pmatrix} \mathbf{H}_\mathrm{b} & \mathbf{H}_\mathrm{bm} \\ \mathbf{H}_\mathrm{bm}^\mathrm{T} & \mathbf{H}_\mathrm{m} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{x}}_\mathrm{b} \\ \ddot{\boldsymbol{\phi}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_\mathrm{b} \\ \boldsymbol{c}_\mathrm{m} \end{pmatrix} = \begin{pmatrix} \boldsymbol{F}_\mathrm{b} \\ \boldsymbol{\tau} \end{pmatrix} + \begin{pmatrix} \mathbf{J}_\mathrm{b}^\mathrm{T} \\ \mathbf{J}_\mathrm{m}^\mathrm{T} \end{pmatrix} \boldsymbol{F}_\mathrm{h} \,, \tag{55.23}$$

$$\dot{\boldsymbol{x}}_\mathrm{h} = \mathbf{J}_\mathrm{m}\dot{\boldsymbol{\phi}} + \mathbf{J}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,, \tag{55.24}$$

$$\ddot{\boldsymbol{x}}_\mathrm{h} = \mathbf{J}_\mathrm{m}\ddot{\boldsymbol{\phi}} + \dot{\mathbf{J}}_\mathrm{m}\dot{\boldsymbol{\phi}} + \mathbf{J}_\mathrm{b}\ddot{\boldsymbol{x}}_\mathrm{b} + \dot{\mathbf{J}}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,. \tag{55.25}$$

Here, with the gravitational force $\boldsymbol{g}$ in Cartesian space, the term $\boldsymbol{c}_\mathrm{b}$ is generally expressed as

$$\boldsymbol{c}_\mathrm{b} = \boldsymbol{f}(\mathbf{x}_\mathrm{b}, \boldsymbol{\phi}, \dot{\boldsymbol{x}}_\mathrm{b}, \dot{\boldsymbol{\phi}}) + \boldsymbol{g}(\boldsymbol{x}_\mathrm{b}, \boldsymbol{\phi}) \,. \tag{55.26}$$

The difference from the equation of a free-floating manipulator system (55.1) is the existence of the base constraint force $\boldsymbol{F}_\mathrm{b}$. Let $\mathbf{D}_\mathrm{b}$ and $\mathbf{K}_\mathrm{b}$ be the matrix representing the damping and spring factors, respectively, of the flexible-base. The constraint forces and moments $\boldsymbol{F}_\mathrm{b}$ are then expressed as

$$\boldsymbol{F}_\mathrm{b} = -\mathbf{D}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} - \mathbf{K}_\mathrm{b}\Delta\boldsymbol{x}_\mathrm{b} \,. \tag{55.27}$$

Since the base is constrained, the total momentum is not conserved, and it might be meaningless to check the system momentum. However, it is important to consider the partial momentum $\mathcal{L}_\mathrm{m}$ for the part of the manipulator arm

$$\mathcal{L}_\mathrm{m} = \mathbf{H}_\mathrm{bm}\dot{\boldsymbol{\phi}} \,, \tag{55.28}$$

which is termed the *coupling momentum* [55.72]. Its time derivative describes the forces and moments $\boldsymbol{F}_\mathrm{m}$, which are yielded by the dynamic reaction from the manipulator arm to the base

$$\boldsymbol{F}_\mathrm{m} = \mathbf{H}_\mathrm{bm}\ddot{\boldsymbol{\phi}} + \dot{\mathbf{H}}_\mathrm{bm}\dot{\boldsymbol{\phi}} \,. \tag{55.29}$$

Using $\boldsymbol{F}_\mathrm{b}$ and $\boldsymbol{F}_\mathrm{m}$, the upper set of equation in (55.23) is rearranged as

$$\mathbf{H}_\mathrm{b}\ddot{\boldsymbol{x}}_\mathrm{b} + \mathbf{D}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} + \mathbf{K}_\mathrm{b}\Delta\boldsymbol{x}_\mathrm{b} = -\boldsymbol{g} - \boldsymbol{F}_\mathrm{m} + \mathbf{J}_\mathrm{b}^\mathrm{T}\boldsymbol{F}_\mathrm{h} \,. \tag{55.30}$$

Equation (55.23) or (55.30) is a familiar expression for flexible-base manipulators [55.73, 74].

55.3.10 Advanced Control for Flexible Structure Based Manipulators

In this subsection, advanced control issues for a macro–micro space manipulator are discussed. The SSRMS attached to the SPDM (Fig. 55.4) and the JEMRMS (Fig. 55.5) are examples. Operation modes for this class of space manipulators include coarse motion by the macro (long-reach) component and fine manipulation by the micro component. In normal cases, these two control modes are executed exclusively. Namely, while one component is active the other component should be servo (or brake) locked. Thus, during the operation of the micro component, the macro component behaves just as a passive base.

Due to the flexible nature of the space long-reach arm, the macro part is subject to vibrations. These vibrations can be excited during coarse positioning and may remain for a long time after each operation. In fine manipulation, the macro arm behaves as a passive flexible structure, but then vibrations can be excited by reactions from the motion of the micro arm. These motions degrade the control accuracy and operational performance of the system. In practice, the booms are usually sufficiently stiff, but flexibility comes mainly from the low stiffness at the joints and gear trains. Moreover, lightweight and microgravity characteristics make the structure sensitive to yield vibrations, and the surrounding vacuum, or the lack of air viscosity, provides a reduced damping effect to the structure.

Conventionally, the vibration issue has been managed for SRMS and SSRMS by the operational skill of well-trained astronauts and by limiting the maximum operational velocity according to the inertia of the handling object. However, if an advanced controller is introduced on the ISS, the training time for astronauts will be reduced and the operational speed can be increased.

Here, the following two subtasks are considered in dealing with this issue:

- Suppression of the vibrations of the flexible base
- End point control in the presence of vibrations.

To suppress the vibrations of the macro arm (flexible base), the coupled dynamics is effectively used. Such control is called *coupled vibration suppression control* [55.75]. The coupled dynamics is a solution space of the micro arm motion with maximum coupling with the vibration dynamics of the macro arm. Note that this solution space is perpendicular to the reaction null-space introduced in Sect. 55.3.7. Since the spaces are orthogonal, the coupled vibration suppression control and reactionless manipulation can be superimposed without any mutual interference.

The motion command of the micro arm to suppress the vibrations is determined with a feedback of the linear and angular velocity of the end point of the macro arm $\dot{\boldsymbol{x}}_\mathrm{b}$

$$\ddot{\boldsymbol{\phi}} = \mathbf{H}_\mathrm{bm}^{+}\mathbf{H}_\mathrm{b}\mathbf{G}_\mathrm{b}\dot{\boldsymbol{x}}_\mathrm{b} \,, \tag{55.31}$$

where $(\cdot)^+$ denotes the right pseudo-inverse, and $\mathbf{G}_\mathrm{b}$ is a positive definite gain matrix.

If written in the form of a joint torque input, the vibration control law is expressed as follows

$$\boldsymbol{\tau} = \mathbf{H}_{\mathrm{m}}\mathbf{H}_{\mathrm{bm}}^{+}\mathbf{G}_{\mathrm{b}}\dot{\boldsymbol{x}}_{\mathrm{b}} \,. \tag{55.32}$$

In the presence of redundancy in the micro arm, (55.31) can be extended to control with a null-space component

$$\ddot{\boldsymbol{\phi}}_{\mathrm{m}} = \mathbf{H}_{\mathrm{bm}}^{+}(\mathbf{H}_{\mathrm{b}}\mathbf{G}_{\mathrm{b}}\dot{\boldsymbol{x}}_{\mathrm{b}} - \dot{\mathbf{H}}_{\mathrm{bm}}\dot{\boldsymbol{\phi}}_{\mathrm{m}}) + \boldsymbol{P}_{\mathrm{RNS}}\boldsymbol{\zeta} \,, \tag{55.33}$$

where $\boldsymbol{\zeta}$ is an n-DOF arbitrary vector and $\boldsymbol{P}_{\mathrm{RNS}} = (\mathbf{E} - \mathbf{H}_{\mathrm{bm}}^{+}\mathbf{H}_{\mathrm{bm}})$ is a projector onto a null space of the coupling inertia matrix $\mathbf{H}_{\mathrm{bm}}$. When the micro arm is operated using (55.33), the closed-loop system is expressed as

$$\mathbf{H}_{\mathrm{b}}\ddot{\boldsymbol{x}}_{\mathrm{b}} + \mathbf{H}_{\mathrm{b}}\mathbf{G}_{\mathrm{b}}\dot{\boldsymbol{x}}_{\mathrm{b}} + \mathbf{K}_{\mathrm{b}}\Delta\boldsymbol{x}_{\mathrm{b}} = \boldsymbol{F}_{\mathrm{b}} + \mathbf{J}_{\mathrm{b}}^{\mathrm{T}}\boldsymbol{F}_{\mathrm{h}} \,. \tag{55.34}$$

Equation (55.34) represents a second-order damped vibration system. With no force input, i. e., $\boldsymbol{F}_{\mathrm{b}} = \boldsymbol{F}_{\mathrm{h}} = 0$, the vibrations converge to zero with a proper choice of the gain matrix $\mathbf{G}_{\mathrm{b}}$.

For the determination of vector $\boldsymbol{\zeta}$, feedback control to reduce the positioning error of the micro arm end point is considered. The error vector is defined as

$$\tilde{\boldsymbol{x}}_{\mathrm{h}} = \boldsymbol{x}_{\mathrm{h}}^{d} - \boldsymbol{x}_{\mathrm{h}} \,. \tag{55.35}$$

After some derivation, the control law for the joint torque of the micro arm is obtained in the following form [55.75, 76]

$$\boldsymbol{\tau} = ((\mathbf{J}_{\mathrm{m}}^{\mathrm{T}})^{+}\mathbf{H}_{\mathrm{m}}\boldsymbol{P}_{\mathrm{RNS}})^{+}(\mathbf{K}_{\mathrm{p}}\tilde{\boldsymbol{x}}_{\mathrm{h}} - \mathbf{K}_{\mathrm{d}}\mathbf{J}_{\mathrm{m}}\dot{\boldsymbol{\phi}}) - \mathbf{G}_{\mathrm{m}}\dot{\boldsymbol{\phi}} \,, \tag{55.36}$$

where $\mathbf{K}_{\mathrm{p}}$, $\mathbf{K}_{\mathrm{d}}$, and $\mathbf{G}_{\mathrm{m}}$ are positive definite gain matrices.

Figure 55.26 shows a block diagram for the control system described by (55.32) and (55.36).

As a simplified demonstration, a planar system with a four-joint redundant manipulator arm atop a flexible beam is considered. Figure 55.27 shows the vibration amplitude of the flexible beam after an impulsive external force. The graph labeled *w/o vs* depicts the vibrations of the beam without any manipulator control but with natural damping. The graph labeled *with vs* depicts the case in which vibration suppression control given by (55.32) is applied, where the vibrations are damped quickly.

In addition, Fig. 55.28 shows the end point motion of the manipulator arm during the control. The graph labeled *w/o RNS* is the case of using (55.32), where the base vibrations were successfully suppressed but

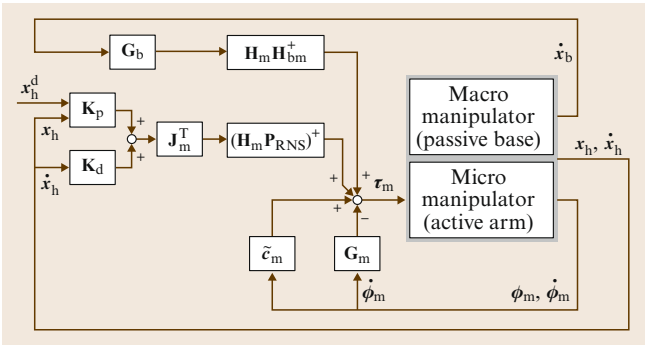


Fig. 55.26 Block diagram for simultaneous vibration suppression and manipulator end point control for a flexible-structure-mounted manipulator system

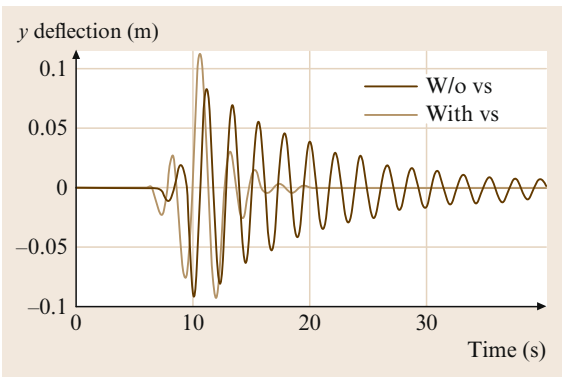


Fig. 55.27 Vibrations of the flexible base

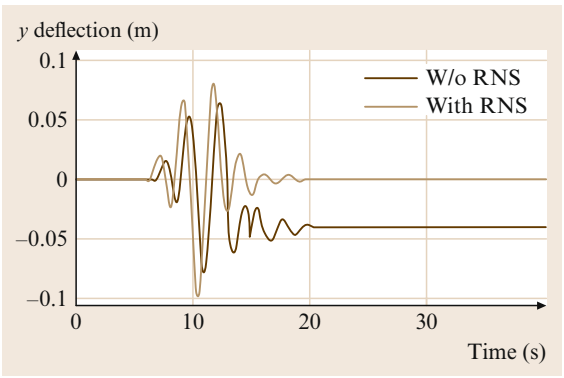


Fig. 55.28 Positioning error of the manipulator end tip

the position of the manipulator end point was deflected by this suppression behavior. The graph labeled *with RNS* depicts the case in which both vibration suppression control given by (55.32) and the end point control given by (55.36) are applied simultaneously. This last case shows that the vibrations were damped successfully and that the positioning error of the manipulator

end point converged to zero. This is a result of the redundancy of the arm.

Here, note that the proposed control method requires precise information on dynamic characteristics, such as the inertia parameters of the arms and the handled payload, if any. To achieve more practical applications, the proposed method must be extended to schemes for parameter identification [55.77] and adaptive control [55.78], with which the convergence of the control is guaranteed even with imprecise a priori knowledge of the dynamic parameters [55.76].

55.3.11 Contact Dynamics and Impedance Control

The capture and retrieval operation of a floating and tumbling *target*, such as a malfunctioning satellite, by a manipulator arm mounted on a servicing robot (called a *chaser*) can be decomposed into the following three phases:

1. Approaching phase (before contact with the target)
2. Contact/impact phase (at the moment of contact)
3. Post contact phase (after contact or grasping).

If the contact is impulsive, the first and the third phases are discontinued by the impulsive phenomena of the second phase. Understanding of this impulsive phenomena is indispensable when designing a comprehensive capture control scheme. In this subsection, the formulation of impact dynamics is first considered, and impedance control (which is useful to minimize the impact forces and prolong the contact duration) is then discussed.

Let us consider a chain of rigid links composed of $n+1$ bodies freely floating in inertial space. As discussed in Sect. 55.3.3, the equation of motion for this type of system becomes (55.8). Here, the impulsive contact force is assumed to be applied at the manipulator end tip and is expressed as $\boldsymbol{F}_\mathrm{h} = (\boldsymbol{f}_\mathrm{h}^\mathrm{T}, \boldsymbol{n}_\mathrm{h}^\mathrm{T})^\mathrm{T} \in \mathbb{R}^6$. This impulsive force also yields a change in the system momenta $(\boldsymbol{P}_\mathrm{g}^\mathrm{T}, \boldsymbol{L}_\mathrm{g}^\mathrm{T})^\mathrm{T} \in \mathbb{R}^6$, expressed as

$$\begin{aligned}\begin{pmatrix}\dot{\boldsymbol{P}}_\mathrm{g}\\ \dot{\boldsymbol{L}}_\mathrm{g}\end{pmatrix} &= \begin{pmatrix} w\mathbf{E} & 0\\ 0 & \mathbf{I}_\mathrm{g}\end{pmatrix}\begin{pmatrix}\dot{\boldsymbol{v}}_\mathrm{g}\\ \dot{\boldsymbol{\omega}}_\mathrm{g}\end{pmatrix} + \begin{pmatrix}0\\ \boldsymbol{\omega}_\mathrm{g}\times\mathbf{I}_\mathrm{g}\boldsymbol{\omega}_\mathrm{g}\end{pmatrix}\\ &= \begin{pmatrix}\mathbf{E} & 0\\ \tilde{\boldsymbol{r}}_\mathrm{gh} & \mathbf{E}\end{pmatrix}\begin{pmatrix}\boldsymbol{f}_\mathrm{h}\\ \boldsymbol{n}_\mathrm{h}\end{pmatrix},\end{aligned} \tag{55.37}$$

where $\mathbf{E}$ is a 3×3 identity matrix, and the symbols with the suffix g indicate the corresponding values observed around the mass centroid of the $n+1$ link system.

In the above equations, (55.8) describes an internal joint motion (termed *local motion*) of the system, whereas (55.37) describes the overall motion (termed *global motion*) about the centroid of the system. As a result of force input $\boldsymbol{F}_\mathrm{h}$, the floating chain induces both the local motion around its articulated joints and the global translation and rotation with respect to the centroid.

From (55.8) and (55.37), the acceleration of the manipulator end tip $\boldsymbol{\alpha}_\mathrm{h}$ can be expressed in the inertial frame as

$$\boldsymbol{\alpha}_\mathrm{h} = \mathbf{G}^*\boldsymbol{F}_\mathrm{h} + \boldsymbol{d}^*\,, \tag{55.38}$$

where

$$\mathbf{G}^* = \hat{\mathbf{J}}\hat{\mathbf{H}}^{-1}\hat{\mathbf{J}}^\mathrm{T} + \mathbf{R}_\mathrm{h}\mathbf{M}^{-1}\mathbf{R}_\mathrm{h}^\mathrm{T}\,, \tag{55.39}$$

$$\mathbf{R}_\mathrm{h} = \begin{pmatrix}\mathbf{E} & -\tilde{\boldsymbol{r}}_\mathrm{gh}\\ 0 & \mathbf{E}\end{pmatrix}, \quad \mathbf{M} = \begin{pmatrix} w\mathbf{E} & 0\\ 0 & \mathbf{I}_\mathrm{g}\end{pmatrix}, \tag{55.40}$$

and $\boldsymbol{d}^*$ is a velocity dependent term.

Equations (55.38)–(55.40) are expressions for the motion of the hand (the point at which collision occurs) induced by the impact force $\boldsymbol{F}_\mathrm{h}$, where the matrix $\mathbf{G}^*$, which is the augmented version of (55.10), represents the dynamic characteristics of this system.

Further augmentation for the inverted inertia matrix has been discussed for the case in which the contact duration is not considered to be infinitesimal [55.51].

Now let us assume the case in which two free-floating chains, A and B, with dynamic characteristics $\mathbf{G}_\mathrm{A}^*$ and $\mathbf{G}_\mathrm{B}^*$ collide with each other at their respective hands (end points) and an impact force $\boldsymbol{F}_\mathrm{h}$ is induced by this collision.

The equations of motion at the instance of collision are

$$\mathbf{G}_\mathrm{A}^*\boldsymbol{F}_\mathrm{h} = \begin{pmatrix}\dot{\boldsymbol{v}}_\mathrm{hA}\\ \dot{\boldsymbol{\omega}}_\mathrm{hA}\end{pmatrix} - \boldsymbol{d}_\mathrm{A}^* \tag{55.41}$$

for the chain A, and

$$\mathbf{G}_\mathrm{B}^*(-\boldsymbol{F}_\mathrm{h}) = \begin{pmatrix}\dot{\mathrm{v}}_\mathrm{hB}\\ \dot{\boldsymbol{\omega}}_\mathrm{hB}\end{pmatrix} - \boldsymbol{d}_\mathrm{B}^* \tag{55.42}$$

for the chain B, where the subscripts A and B indicate the label of the chain.

Assuming that $\mathbf{G}_\mathrm{A}^*$ and $\mathbf{G}_\mathrm{B}^*$ remain constant during the infinitesimal contact duration and the velocity-dependent terms $\boldsymbol{d}_\mathrm{A}^*$ and $\boldsymbol{d}_\mathrm{B}^*$ are small and negligible, integration of (55.41) and (55.42) yields

$$\mathbf{G}_\mathrm{A}^*\int_t^{t+\delta t}\boldsymbol{F}_\mathrm{h}\,\mathrm{d}t = \begin{pmatrix}\boldsymbol{v}'_\mathrm{hA}\\ \boldsymbol{\omega}'_\mathrm{hA}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}, \tag{55.43}$$

$$\mathbf{G}_\mathrm{B}^*\int_t^{t+\delta t}\boldsymbol{F}_\mathrm{h}\,\mathrm{d}t = \begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}'_\mathrm{hB}\\ \boldsymbol{\omega}'_\mathrm{hB}\end{pmatrix}, \tag{55.44}$$

where $\{'\}$ indicates the velocity after the collision. Integration of $\boldsymbol{F}_\mathrm{h}$

$$\overline{\boldsymbol{F}}_\mathrm{h} = \lim_{\delta t \to 0} \int_{t}^{t+\delta t} \boldsymbol{F}_\mathrm{h}\, \mathrm{d}t \tag{55.45}$$

represents the impulse (force-time product) acting on both chains. Providing that the total momenta of the two systems are strictly conserved before and after the collision, we obtain the following expression from (55.43) and (55.44)

$$\begin{aligned}\left(\mathbf{G}_\mathrm{A}^* + \mathbf{G}_\mathrm{B}^*\right)\overline{\boldsymbol{F}}_\mathrm{h} = &\left\{\begin{pmatrix}\boldsymbol{v}'_\mathrm{hA}\\ \boldsymbol{\omega}'_\mathrm{hA}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}'_\mathrm{hB}\\ \boldsymbol{\omega}'_\mathrm{hB}\end{pmatrix}\right\}\\ &+ \left\{\begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}\right\}.\end{aligned} \tag{55.46}$$

In general collision analysis, the coefficient of restitution (elasticity factor) associated with the relative velocities before and after the collision is often employed [55.79]. If we accept this restitution coefficient for 6-DOF linear and angular velocities, the relationship between relative velocities before and after contact becomes

$$\begin{pmatrix}\boldsymbol{v}'_\mathrm{hA}\\ \boldsymbol{\omega}'_\mathrm{hA}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}'_\mathrm{hB}\\ \boldsymbol{\omega}'_\mathrm{hB}\end{pmatrix} = \boldsymbol{\epsilon}\left\{\begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}\right\}, \tag{55.47}$$

where

$$\boldsymbol{\epsilon} \equiv \mathrm{diag}(e_1, \ldots, e_6)\,, \quad 0 \le e_i \le 1 \tag{55.48}$$

is the restitution coefficient matrix.

Substituting (55.47) into (55.46), the impulse induced by this collision can be expressed only by the relative velocity of two points before the contact

$$\overline{\boldsymbol{F}}_\mathrm{h} = (\mathbf{E}_6 + \boldsymbol{\epsilon})\mathbf{G}_\Sigma^{*-1}\mathcal{V}_\mathrm{hAB}\,, \tag{55.49}$$

where

$$\mathbf{G}_\Sigma^* = \mathbf{G}_\mathrm{A}^* + \mathbf{G}_\mathrm{B}^*\,, \tag{55.50}$$

$$\mathcal{V}_\mathrm{hAB} = \begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}. \tag{55.51}$$

Using the introduced notation, the magnitude of impulse is expressed as

$$\| \overline{\boldsymbol{F}}_\mathrm{h} \| = \sqrt{(\mathbf{E}_6 + \boldsymbol{\epsilon})^\mathrm{T}\mathcal{V}_\mathrm{hAB}^\mathrm{T}\mathbf{G}_\Sigma^{*-T}\mathbf{G}_\Sigma^{*-1}\mathcal{V}_\mathrm{hAB}(\mathbf{E}_6 + \boldsymbol{\epsilon})}\,, \tag{55.52}$$

and the velocity after collision becomes

$$\begin{aligned}\begin{pmatrix}\boldsymbol{v}'_\mathrm{hA}\\ \boldsymbol{\omega}'_\mathrm{hA}\end{pmatrix} = &\left(\mathbf{G}_\mathrm{A}^{*-1} + \mathbf{G}_\mathrm{B}^{*-1}\right)^{-1}\\ &\times\left[(\mathbf{E}_6 + \boldsymbol{\epsilon})\mathbf{G}_\mathrm{B}^{*-1}\begin{pmatrix}\boldsymbol{v}_\mathrm{hB}\\ \boldsymbol{\omega}_\mathrm{hB}\end{pmatrix}\right.\\ &\left.+(\mathbf{G}_\mathrm{A}^{*-1} - \boldsymbol{\epsilon}\mathbf{G}_\mathrm{B}^{*-1})\begin{pmatrix}\boldsymbol{v}_\mathrm{hA}\\ \boldsymbol{\omega}_\mathrm{hA}\end{pmatrix}\right],\end{aligned} \tag{55.53}$$

where suffixes A and B are interchangeable.

These expressions are considered to be an augmentation of the impact theory for a two-point-mass system into articulated body systems.

Impedance control is a concept by which we can control the manipulator end tip so as to obtain the desired mechanical impedance characteristics. Such control is useful to alter the dynamic characteristics of the arm during the contact phase. In a special case, the desired impedance of the manipulator end tip (hand) may be tuned to achieve *impedance matching* with the colliding target object so that the hand can easily maintain stable contact with the target [55.80].

Let $\mathbf{M}_\mathrm{d}$, $\mathbf{D}_\mathrm{d}$, and $\mathbf{K}_\mathrm{d}$ be the matrices for the desired impedance properties of inertia, viscosity, and stiffness, respectively, measured at the manipulator end point. The equation of motion for the desired system is then expressed as

$$\mathbf{M}_\mathrm{d}\ddot{\boldsymbol{x}}_\mathrm{h} + \mathbf{D}_\mathrm{d}\Delta\dot{\boldsymbol{x}}_\mathrm{h} + \mathbf{K}_\mathrm{d}\Delta\boldsymbol{x}_\mathrm{h} = \boldsymbol{F}_\mathrm{h}\,. \tag{55.54}$$

From (55.8) and (55.54), the impedance control law for a free-floating manipulator system is obtained as [55.81]

$$\begin{aligned}\boldsymbol{\tau}_\mathrm{h} = &\mathbf{H}^*\hat{\mathbf{J}}^{-1}\{\mathbf{M}_\mathrm{d}^{-1}(\mathbf{D}_\mathrm{d}\Delta\dot{\boldsymbol{x}}_\mathrm{h} + \mathbf{K}_\mathrm{d}\Delta\boldsymbol{x}_\mathrm{h} - \boldsymbol{F}_\mathrm{h})\\ &- \dot{\hat{\mathbf{J}}}\dot{\boldsymbol{\phi}} - \ddot{\boldsymbol{x}}_\mathrm{gh}\} - \hat{\mathbf{J}}^\mathrm{T}\boldsymbol{F}_\mathrm{h} + \boldsymbol{c}^*\,.\end{aligned} \tag{55.55}$$

The usefulness of the impedance control in free-flying space robots has been discussed in [55.81–83].

55.3.12 Dynamics of Mobile Robots

The equation of motion for a mobile robot that has multiple articulated limbs, such as that shown in Fig. 55.29,

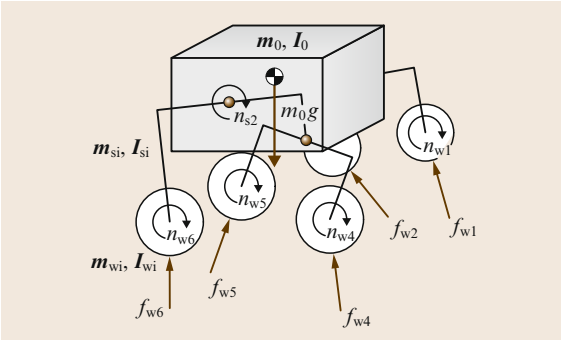


Fig. 55.29 Schematic model of a mobile robot

is given by the following equation

$$\begin{pmatrix} \mathbf{H}_{\mathrm{b}} & \mathbf{H}_{\mathrm{bm1}} & \cdots & \mathbf{H}_{\mathrm{bm}k} \\ \mathbf{H}_{\mathrm{bm1}}^{\mathrm{T}} & \mathbf{H}_{\mathrm{m11}} & \cdots & \mathbf{H}_{\mathrm{m1}k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{H}_{\mathrm{bm}k}^{\mathrm{T}} & \mathbf{H}_{\mathrm{m1}k}^{\mathrm{T}} & \cdots & \mathbf{H}_{\mathrm{mk}k} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{x}}_{\mathrm{b}} \\ \ddot{\boldsymbol{\phi}}_1 \\ \vdots \\ \ddot{\boldsymbol{\phi}}_k \end{pmatrix} + \begin{pmatrix} \boldsymbol{c}_{\mathrm{b}} \\ \boldsymbol{c}_{\mathrm{m1}} \\ \vdots \\ \boldsymbol{c}_{\mathrm{m}k} \end{pmatrix} = \begin{pmatrix} \boldsymbol{F}_{\mathrm{b}} \\ \boldsymbol{\tau}_1 \\ \vdots \\ \boldsymbol{\tau}_n \end{pmatrix} + \begin{pmatrix} \mathbf{J}_{\mathrm{b}}^{\mathrm{T}} \boldsymbol{F}_{\mathrm{ex}} \\ \mathbf{J}_{\mathrm{m1}}^{\mathrm{T}} \boldsymbol{F}_{\mathrm{ex1}} \\ \vdots \\ \mathbf{J}_{\mathrm{m}k}^{\mathrm{T}} \boldsymbol{F}_{\mathrm{ex}k} \end{pmatrix}, \tag{55.56}$$

where the symbols have the following meanings:

k number of limbs
$\boldsymbol{x}_{\mathrm{b}} \in \mathbb{R}^6$: position/orientation of the base
$\boldsymbol{\phi} = (\boldsymbol{\phi}_1^{\mathrm{T}}, \ldots, \boldsymbol{\phi}_k^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^n$: articulated joint angles
$\boldsymbol{x}_{\mathrm{ex}} = (\boldsymbol{x}_{\mathrm{ex1}}^{\mathrm{T}}, \ldots, \boldsymbol{x}_{\mathrm{ex}k}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{6k}$: position/orientation of the end points
$\boldsymbol{F}_{\mathrm{b}} \in \mathbb{R}^6$: forces/moments directly apply on the base
$\boldsymbol{\tau} = (\boldsymbol{\tau}_1^{\mathrm{T}}, \ldots, \boldsymbol{\tau}_k^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^n$: joint articulated torque
$\boldsymbol{F}_{\mathrm{ex}} = (\boldsymbol{F}_{\mathrm{ex1}}^{\mathrm{T}}, \cdots, \boldsymbol{F}_{\mathrm{ex}k}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{6k}$: external forces/moments on the end points.

Note that for Fig. 55.29, $\boldsymbol{F}_{\mathrm{ex}i} = [\boldsymbol{f}_{\mathrm{wi}}^{\mathrm{T}}, \boldsymbol{n}_{\mathrm{wi}}^{\mathrm{T}}]^{\mathrm{T}}$.

Comparing the above equations with (55.1), no difference is observed in the mathematical structure. The gravity force on the vehicle main body and the configuration-dependent gravity terms of the articulated bodies are included in c_{b} and $c_{\mathrm{m}i}$, respectively. In practice, however, one substantial difference is the existence of ground contact forces/moments at the end point of each limb. Unlike floating target capture discussed in Sect. 55.3.10, the contact is not considered impulsive, but instead continues for a nonnegligible period of time. In such cases, a well-accepted approach is to explicitly evaluate the contact forces/moments $\boldsymbol{F}_{\mathrm{ex}}$, according to the virtual penetration of the end point into the collided object or the ground surface [55.79].

In cases in which each limb has a wheel on its end terminal, rather than the point penetration model, a wheel traction model will be adopted to evaluate $\boldsymbol{F}_{\mathrm{ex}}$. For planetary exploration missions, rovers (mobile robots) are expected to travel over natural rough terrain. A number of studies have examined the modeling of tire traction forces on loose soil, called *regolith* (where there is no organic component) [55.84–98]. Particularly, these studies investigate the soil mechanics called *terramechanics* to understand the tractive forces generated by wheels.

In the following subsection, models for wheel traction mechanics are summarized.

55.3.13 Wheel Traction Mechanics

Terramechanics is the study of soil properties, specifically the interaction of wheeled, legged, or tracked vehicles and various surfaces. For the modeling of the wheel traction forces and the analysis of the vehicle mobility, textbooks written by *Bekker* [55.29, 30] and *Wong* [55.99] are good references. Although these books were written in the 1960s and 1970s, basic formulae from these books are frequently cited by researchers even today [55.88]. In this subsection, the models for a rigid wheel on loose soil are summarized.

Slip Ratio and Slip Angle

Slips are generally observed when a rover travels on loose soil. In particular, slips in the lateral direction are observed during steering or slope-traversing maneuvers.

The slip in the longitudinal direction is measured by slip ratio s, which is defined as a function of the longitudinal traveling velocity v_x of the wheel and the circumference velocity of the wheel $r\omega$ (r is the wheel radius and ω is the angular velocity of the wheel)

$$s = \begin{cases} \dfrac{r\omega - v_x}{r\omega} & \text{if } |r\omega| > |v_x| : \text{driving} \\ \dfrac{r\omega - v_x}{v_x} & \text{if} |r\omega| < |v_x| : \text{braking} . \end{cases} \tag{55.57}$$

The slip ratio takes a value from -1 to 1.

On the other hand, the slip in the lateral direction is measured by slip angle β, which is defined in terms of v_x and the lateral traveling velocity v_y as

$$\beta = \tan^{-1}\left(\frac{v_y}{v_x}\right) . \tag{55.58}$$

Note that the above definitions, (55.57) and (55.58) have been traditionally used in the vehicle community as standards. However, planetary rovers in a challenging terrain, such as Spirit and Opportunity on Mars, experienced the cases in which the rovers slip backward while attempting to drive up hill, or travel faster than the wheel's circumference velocity in downhill driving. In these cases, the slip ratio can exceed the range from -1 to 1. Also while traversing side slopes, the case may arise in which $v_y > 0$ but v_x is nearly 0, making the definition (55.58) nearly singular. Therefore, these definitions are needed to be further discussed for the rovers in very loose terrain.

Wheel–Soil Contact Angle

Figure 55.30 depicts a schematic model of a rigid wheel contacting loose soil. In the figure, the angle from the surface normal to the point at which the wheel initially makes contact with the soil ($\angle$ AOB) is defined as the

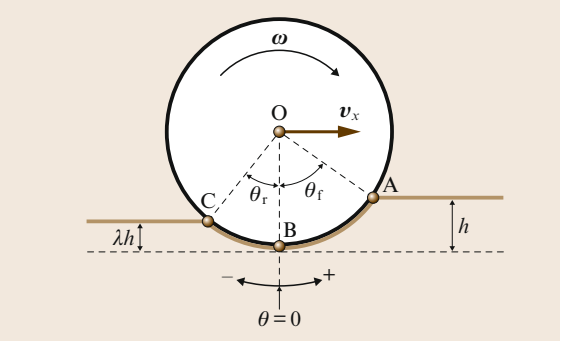


Fig. 55.30 Wheel contact angles

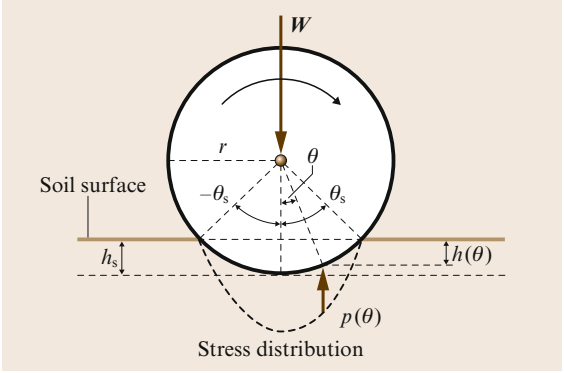


Fig. 55.31 Static sinkage

entry angle. The angle from the surface normal to the point at which the wheel departs from the soil (∠ BOC in Fig. 55.30) is the exit angle. The wheel contact region on loose soil is represented from the entry angle to the exit angle.

The entry angle θ_f is geometrically described in terms of wheel sinkage h as

$$\theta_f = \cos^{-1}\left(1 - \frac{h}{r}\right) . \tag{55.59}$$

The exit angle θ_r is described using the wheel sinkage ratio λ, which denotes the ratio between the forward and rear sinkage of the wheel

$$\theta_r = \cos^{-1}\left(1 - \frac{\lambda h}{r}\right) . \tag{55.60}$$

The value of λ depends on the soil characteristics, the wheel surface pattern, and the slip ratio. It becomes smaller than 1.0 when the soil compaction occurs, but can be greater than 1.0 when the soil is dug up by the wheel and transported to the rear region of the wheel.

Wheel Sinkage

The amount of wheel sinkage is constituted by static and dynamic components. The static sinkage depends on the vertical load on the wheel, while the dynamic sinkage is caused by the rotation of the wheel.

According to the equation formulated by *Bekker* [55.29], the static stress $p(h)$ generated under a flat plate, which has a sinkage h and a width b, is calculated as

$$p(h) = \left(\frac{k_c}{b} + k_\phi\right) h^n , \tag{55.61}$$

where k_c and k_ϕ are pressure-sinkage modules, and n is the sinkage exponent. Applying (55.61) to the wheel, as shown in Fig. 55.31, the static sinkage is evaluated as follows.

First, the wheel sinkage $h(\theta)$ at an arbitrary wheel angle θ is geometrically given by

$$h(\theta) = r(\cos\theta - \cos\theta_s) , \tag{55.62}$$

where θ_s is the static contact angle. Then, substituting (55.62) into (55.61) yields

$$p(\theta) = r^n \left(\frac{k_c}{b} + k_\phi\right) (\cos\theta - \cos\theta_s)^n . \tag{55.63}$$

The wheel eventually sinks into the soil until the stress from the soil balances the vertical load W on the wheel.

$$\begin{aligned} W &= \int_{-\theta_s}^{\theta_s} p(\theta) br \cos\theta \, d\theta \\ &= r^{n+1} (k_c + k_\phi b) \int_{-\theta_s}^{\theta_s} (\cos\theta - \cos\theta_s)^n \cos\theta \, d\theta . \end{aligned} \tag{55.64}$$

Using this equation, the static contact angle θ_s is evaluated for the given W. In practice, (55.64) does not yield a closed form solution for θ_s, although θ_s can be evaluated numerically.

Finally, the static sinkage h_s is obtained by substituting θ_s into the following equation

$$h_s = r(1 - \cos\theta_s) . \tag{55.65}$$

However, as illustrated in Fig. 55.32, the dynamic sinkage becomes a complicated function depending on the slip ratio of the wheel, the wheel surface pattern, and the soil characteristics. Although it is difficult to obtain an analytical form for the dynamic sinkage, it is again possible to evaluate the dynamic sinkage numerically, using the condition $W = F_z$, where F_z is the normal force given by (55.76), which will be presented

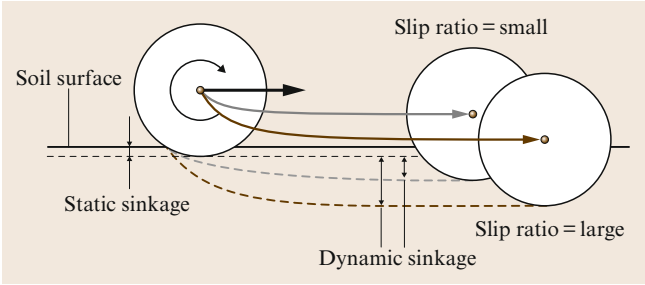


Fig. 55.32 Dynamic sinkage

later herein. The force F_z increases with the wheel sinkage because the area of the contact patch increases accordingly.

Stress Distribution Under the Wheel

Based on terramechanics models, the stress distribution under the rotating wheel can be modeled as shown in Fig. 55.33.

The normal stress $\sigma(\theta)$ is determined by the following equation [55.87, 88]

$$\sigma(\theta) = r^n \left(\frac{k_c}{b} + k_\phi\right)(\cos\theta - \cos\theta_f)^n \quad \text{for } \theta_m \le \theta < \theta_f ,$$

$$\sigma(\theta) = r^n \left(\frac{k_c}{b} + k_\phi\right)\left\{\cos\left[\theta_f - \frac{\theta - \theta_r}{\theta_m - \theta_r}(\theta_f - \theta_m)\right] - \cos\theta_f\right\}^n \quad \text{for } \theta_r < \theta \le \theta_m . \tag{55.66}$$

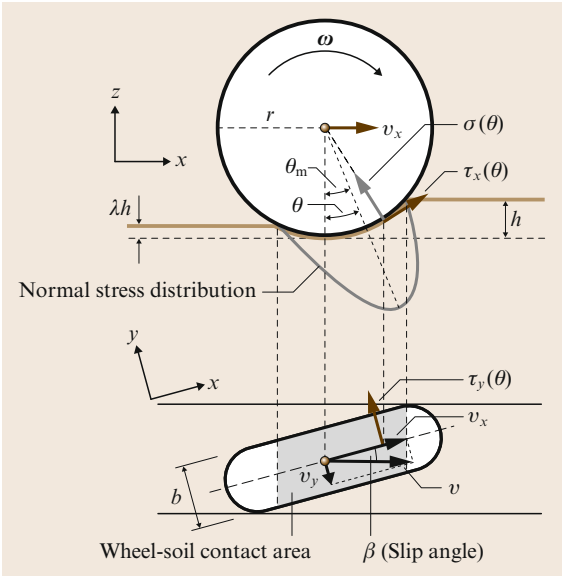


Fig. 55.33 Stress distribution model under a wheel

Note that the above equations are based on Bekker's formula, as given in (55.61), and they become equivalent to the *Wong–Reece* model for normal stress [55.100] when $n = 1$. Also note that by linearizing this distribution, *Iagnemma* et al. [55.84, 88] developed a Kalman-filter-based method to estimate the soil parameters.

The term θ_m is the specific wheel angle at which the normal stress is maximized

$$\theta_m = (a_0 + a_1 s)\theta_f , \tag{55.67}$$

where a_0 and a_1 are parameters that depend on the wheel–soil interaction. Their values are generally assumed to be $a_0 \approx 0.4$ and $0 \le a_1 \le 0.3$ [55.100].

The maximum terrain shear force is a function of the terrain cohesion c and internal friction angle ϕ, and can be computed from Coulomb's equation

$$\tau_{max}(\theta) = c + \sigma_{max}(\theta)\tan\phi . \tag{55.68}$$

Based on the above equation, the shear stresses under the rotating wheel, $\tau_x(\theta)$ and $\tau_y(\theta)$, are written as follows [55.101]

$$\tau_x(\theta) = [c + \sigma(\theta)\tan\phi]\left(1 - e^{-j_x(\theta)/k_x}\right) , \tag{55.69}$$

$$\tau_y(\theta) = [c + \sigma(\theta)\tan\phi]\left(1 - e^{-j_y(\theta)/k_y}\right) , \tag{55.70}$$

where k_x and k_y are the shear deformation moduli in each direction. In addition, j_x and j_y, which are the soil deformations in each direction, can be formulated as a function of the wheel angle θ with the slip ratio and the slip angle, respectively [55.89, 100]

$$j_x(\theta) = r[\theta_f - \theta - (1 - s)(\sin\theta_f - \sin\theta)] , \tag{55.71}$$

$$j_y(\theta) = r(1 - s)(\theta_f - \theta)\tan\beta . \tag{55.72}$$

Drawbar Pull: F_x

Using the normal stress $\sigma(\theta)$ and the shear stress in the x direction $\tau_x(\theta)$, the drawbar pull F_x, which is the net traction force exerted from the soil to the wheel, is calculated as the integral from the entry angle θ_f to the exit angle θ_r [55.100]

$$F_x = rb\int_{\theta_r}^{\theta_f} [\tau_x(\theta)\cos\theta - \sigma(\theta)\sin\theta]\, d\theta . \tag{55.73}$$

Side Force: F_y

The side force F_y appears in the lateral direction of the wheel when the vehicle makes steering maneuvers or traverses a side slope. The side force is decomposed into two components [55.89]

$$F_y = F_u + F_s ,$$

where F_u is the force produced by the shear stress in the y direction $\tau_y(\theta)$ underneath the wheel and F_s is the reaction force generated by the bulldozing phenomenon on a side face of the wheel. The above equation can be rewritten as

$$F_y = \int_{\theta_r}^{\theta_f} \left\{ \underbrace{rb\tau_y(\theta)}_{F_u} + \underbrace{R_b[r - h(\theta)\cos\theta]}_{F_s} \right\} d\theta \,. \tag{55.74}$$

Here, *Hegedus's bulldozing resistance estimation* [55.102] is employed to evaluate the side face force F_s. As shown in Fig. 55.34, a bulldozing resistance R_b is generated on a unit width blade when the blade moves toward the soil. According to Hegedus's theory, the bulldozed area is defined by a destructive phase that is modeled by a planar surface. In the case of a horizontally placed wheel, the angle of approach α' should be zero; R_b can then be calculated as a function of wheel sinkage $h(\theta)$

$$R_b(h) = D_1 \left[ch(\theta) + D_2 \frac{\rho_d h^2(\theta)}{2} \right] , \tag{55.75}$$

where

$$D_1(X_c, \phi) = \cot X_c + \tan(X_c + \phi) \,,$$

$$D_2(X_c, \phi) = \cot X_c + \frac{\cot^2 X_c}{\cot \phi} \,.$$

In the above equations, ρ_d denotes the soil density. Based on *Bekker*'s theory [55.29], the destructive angle X_c can be approximated as

$$X_c = 45^\circ - \frac{\phi}{2} \,.$$

Normal Force: F_z

The normal force F_z is obtained in the same manner as for the (55.73) [55.100]

$$F_z = rb \int_{\theta_r}^{\theta_f} [\tau_x(\theta)\sin\theta + \sigma(\theta)\cos\theta] \, d\theta \,, \tag{55.76}$$

which should balance the normal load of the wheel in a static condition.

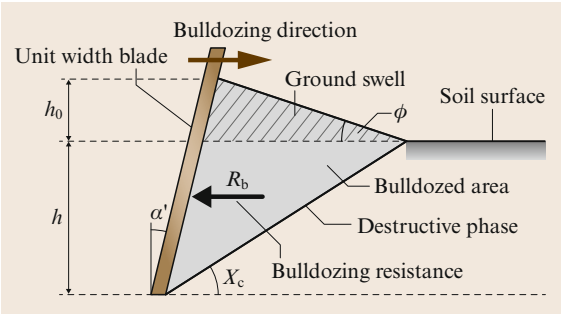


Fig. 55.34 Estimation model of the bulldozing resistance

Motion dynamics simulation for a vehicle traveling over loose soil can be performed by plugging the forces F_x, F_y, and F_z obtained from the above equations into the equation of motion (55.5).

A better understanding of the soil–wheel contact and traction mechanics is important in order to improve the navigation and control behavior of exploration rovers, in terms of minimization of wheel slippage, for example. Reducing the wheel slippage will increase the power efficiency of surface locomotion, decrease the errors in path tracking maneuvers, and decrease the risks of wheel spinning and sinking, which can cause immobilization of the vehicle.

One key in realizing such advanced control of slippage minimization is determining how to properly estimate the slip ratios and slip angles in real time using on-board sensors. The slip ratio is determined by the ratio between the wheel spinning velocity and the traveling velocity of the vehicle, but proper sensing of the velocity of the vehicle is usually difficult. One simple solution is to use a free wheel specialized for traveling velocity measurement. Another solution is to employ inertial sensors, which are however usually subject to noise and drift.

An alternative, but promising possibility is visual odometry, which is based on optical flow or feature tracking in the sequence of optical images. Actually, this technique has been applied to the Mars exploration rovers, Spirit, and Opportunity, in their long range navigation, and verified to be very useful. Particularly, the algorithm based on feature detection and tracking using stereo pair of cameras provides reliable results with good accuracy for the estimation of driving distance as well as the wheel slippage [55.28].

55.4 Future Directions of Orbital and Surface Robotic Systems

55.4.1 Robotic Maintenance and Service Missions

For many years, we have sent satellites and other systems into space without caring too much about what might happen at the end of their life cycle. There has been recently awareness about the dramatic increase of space debris and the danger of a fatal chain reaction increased in collisions. Generally speaking, space debris removal may become a prerequisite for future spaceflight. Space systems above approx. 600 km flight path altitude are not currently accessible to astronauts by means of present transport systems and therefore are excluded from any kind of human removal, repair, or maintenance.

In contrast, satellites equipped with robotic arms or humanoid robonauts may be remotely controlled or only supervised from Earth in any orbit including the geostationary one. In the future, they should be able to support astronauts during routine and maintenance work on space stations, capture uncontrollably tumbling satellites, prolong their life-time by repair or refuel, deorbit or relocate them if necessary. Efficient telerobotic and telepresence technologies allow us to select flexibly the appropriate level of robot autonomy within *shared autonomy* frameworks between ground operator and space robot.

Telepresence technologies as mentioned above make sure that by real-time feedback of stereo images and force/torque information, the operator on the ground gets the feeling as though he was actually working at the remote site. High-quality telepresence requires low round-trip communication time delays. The challenge here is twofold: (a) to provide the mentioned technically feasible communication infrastructure, and (b) to apply the mentioned optimized delay-compensating telepresence technologies which yield satisfactory haptic feedback up to 650 ms delay. For large robots mounted on a carrier satellite, their dynamic interactions, including the physical contacts when grasping a target, have to be mastered (Sect. 55.3). Autonomous skills are needed for supporting a human ground operator in performing the risky task of grasping and stabilizing noncooperative tumbling targets (satellite or space debris).

Due to the high cost of space validation missions, simulation capabilities are crucial for the development and verification of on-orbit servicing systems. This applies both for the required hardware simulation facilities including sensors and illumination effects as well as for dynamics modeling techniques and software tools. Various hardware simulators using industrial robots for simulation of chaser and target satellite motion and the dynamic interaction with the space-robot have been built up not only in DLR (Fig. 55.35).

Robotic maintenance and service missions for space infrastructure have been a long-term dream in the space robotics community since their conceptual designs were first published in ARAMIS report in the early 1980s (Fig. 55.1) [55.1].

ROTEX, ETS-VII, Ranger, and ASTRO that were introduced in the earlier section are technological developments toward this goal, but robotic maintenance and service missions have not become routinely operational yet (A good comparative study of orbital robotic missions is provided by [55.103]). The Hubble space telescope (HST) is a huge space telescope which has the capability to be serviced in orbit, but it has been visited by Space Shuttle and serviced (components exchanged and trouble fixed) only by human EVA. After the COLUMBIA accident in 2003, NASA seriously considered the possibility of robotic maintenance of the HST, investigating available technologies and selecting a prime contractor of the mission development. Figure 55.36 depicts one possible configuration for the robotic rescue mission. Ultimately, it was decided to perform this last servicing mission with human astronauts. Maintenance of the HST involves tasks that are too complicated to be done by a robot, because the HST itself was designed for human-based maintenance and not specifically designed for robots.

Robonaut, which is described in the following subsection, is therefore considered as an interesting option to be capable in conducting practical maintenance and service missions, due to its compatibility and similar level of dexterity with human astronauts.

Fig. 55.35 OOS-SIM, a hardware-in-the-loop simulator in DLR

Fig. 55.36 A conceptual drawing for robotic rescue of Hubble space telescope

55.4.2 Robonaut and JUSTIN

Robonaut is a humanoid robot designed by the Robot Systems Technology Branch at NASA Johnson Space Center in a collaborative effort with DARPA. The Robonaut project seeks to develop and demonstrate a robotic system that can function as an EVA astronaut equivalent. Robonaut jumps generations ahead by eliminating the robotic scars (e.g., special robotic grapples and targets), but it still keeps the human operator in the control loop through its telepresence control system. Robonaut is designed to be used for *EVA* tasks (extravehicular activities or space walks), i. e., those which were not specifically designed for robots.

A key challenge is to build machines that can help humans work and explore in space. Working side by side with humans, or going where the risks are too great for people, machines like Robonaut will expand capabilities for construction and discovery. Over the past five decades, space flight hardware has been designed for human servicing. Space walks are planned for most of the assembly missions for the ISS, and they are a key contingency for resolving on-orbit failures. To maintain compatibility with existing EVA tools and equipments, a humanoid shape and an assumed level of human performance (at least a human in a space suit) are required for this robotic surrogate.

The manipulator and dexterous hand have been developed with a substantial investment in mechatronics design. The arm structure has embedded avionics elements within each link, reducing cabling and noise interference. Robonaut has been designed based on a biologically inspired approach. For example, it uses a chordate neurological system in data management, bringing all feedback to a central nervous system, where even low-level servo control is performed. Such a biologically inspired approach is extended to left-right computational symmetry, sensor and power duality and kinematical redundancy, enabling learning and optimization in mechanical, electrical and software forms.

Robonaut has a broad mix of sensors including thermal, position, tactile, force and torque instrumentation, with over 150 sensors per arm. The control system for Robonaut includes an onboard, real-time CPU with miniature data acquisition and power management. Offboard guidance is delivered with human supervision using a telepresence control station with human tracking.

Robonaut 2 (Fig. 55.37), the latest generation of the Robonaut family, launched to the ISS aboard Space Shuttle Discovery on the STS-133 mission in February 2011. It is the first humanoid robot in space, and although its initial job is demonstrating its capabilities inside the space station, the goal is that through upgrades and advancements it will one day venture outside the station to help spacewalkers make repairs or additions to the station. Robonaut 2 is a dexterous, anthropomorphic robotic torso that has significant technical improvements over its predecessor to make it a far more valuable tool for astronauts. Upgrades include: increased force sensing, greater range of motion, higher bandwidth, and improved dexterity. Robonaut 2's integrated mechatronic design results in a more compact and robust distributed control system with a fraction of the wiring of the original Robonaut.

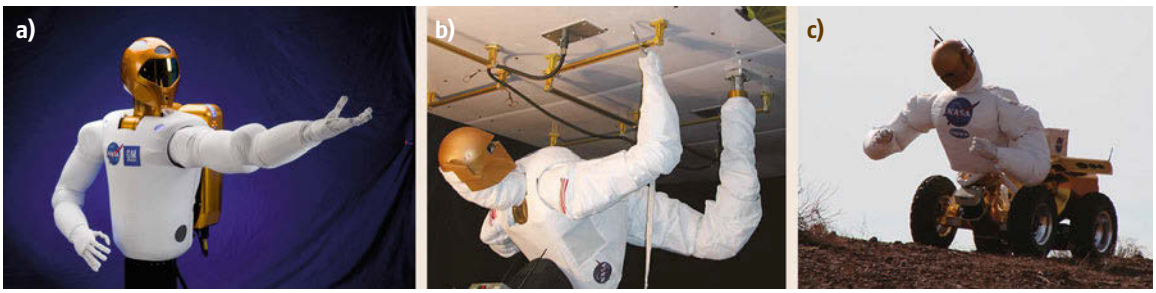


Fig.55.37a–c NASA's Robonaut family: **(a)** Robonaut 2, **(b)** Zero-G Leg for surface inspection of ISS, **(c)** Centaur with a surface mobility system

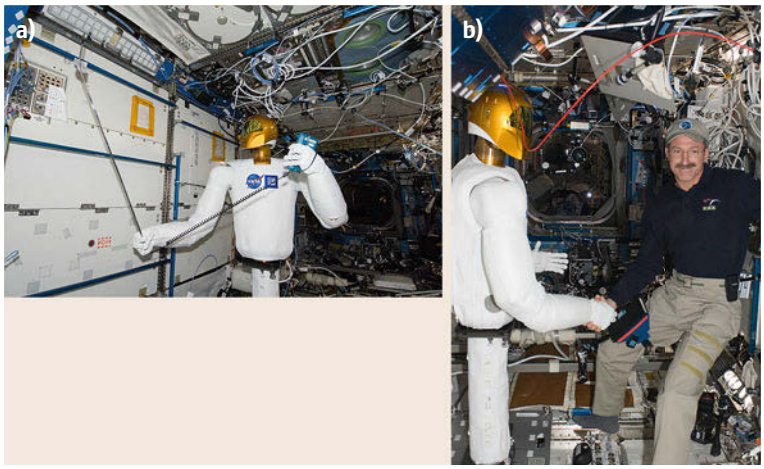


Fig.55.38a,b Robonaut 2 onboard ISS: **(a)** measuring airflow, **(b)** shaking hands with ISS Commander Dan Burbank

Robonaut 2, also called R2, has completed many firsts during its two years on the ISS. During its initial checkout, it used American sign language to say *Hello World.* R2 illustrated its unique control system design that permits it to work directly with astronauts by shaking hands with ISS Commander Dan Burbank (Fig. 55.38). More recently, it has been using standard crew tools to measure airflow and demonstrate its ability to perform autonomous inventory scans. As part of gaining experience that will be useful once R2 starts working on the outside of the Space Station, on-board crew have successfully demonstrated teleoperation. Using a variety of sensors that track human hand, arm and neck motion, astronaut Tom Marshburn, while also onboard the station, became the first person to remotely control R2 to have it catch a free flying object inside the ISS.

One potential application of Robonaut technology is a regular monitoring and contingent maintenance work of human habitant modules of the space station. Figure 55.37b depicts such an application where Robonaut crawls on the surface of the station module by using hand rails which were originally designed for human EVA.

The application of the humanoid robot is not limited to orbital tasks. Figure 55.37c depict an idea to combine the humanoid torso on a surface mobility system, which shall be useful for robotic planetary explorations.

DLR's anthropomorphic JUSTIN is based on high-fidelity joint-torque-controlled light weight-technology and adjustable whole-body compliance in Cartesian space. JUSTIN on the mobile platform (Fig. 55.39) has actuated joints and torque controlled sensors. With JUSTIN's upper body the new delay compensating technologies have been verified using copies of JUSTIN's light weight arms as force reflecting hand-controllers up to delays of slightly more than 700 ms.

The European space agency ESA too, is pushing forward robonaut-type concepts, e.g., via testing a dexterous 4-finger-hand DEXHAND as developed in contract by DLR (Fig. 55.40) In spring 2012, the DEXHAND successfully passed the acceptance test and is delivered to ESA.

55.4.3 Aerial Platforms

There are three planetary candidates for aerial robotic systems: Venus, Mars, and Titan (a moon of Saturn) [55.104, 105]. Venus has a very dense but hot atmosphere (460 °C and 65 kg m^{-3} at the surface), and so can easily float relatively heavy payloads. Mars has a very thin and cold atmosphere (somewhat variable but often −100 °C and 0.02 kg m^{-3}). Titan has an atmosphere even colder than Mars ($\approx$ 100 K) but about 50% denser than Earth's atmosphere. Thus very different vehicles have been envisioned for the three candidate mission targets. On Venus, buoyant devices are generally considered, especially those that can continuously or periodically rise high enough to reach moderate temperatures where conventional electronics can survive. One candidate is to use a phase-change fluid as part of the buoyant system, so that the fluid can condense in the cool upper atmosphere and be trapped in a pressure vessel, causing a loss of buoyancy and allowing the vehicle to descend, possibly all the way to the surface. After a brief stay, and before the heat flux to the interior of the device destroys all the sensitive equipment, a valve would be opened so that the phase change fluid can evaporate, increasing the buoyancy and allowing the craft to ascend to the cool upper atmosphere. After a suitable period of heat rejection into this cool zone, the process can be repeated, perhaps indefinitely. The density of the Venus atmosphere is sufficiently high that powered dirigibles can be used, so that the buoy-

ant vehicles can use propulsion and steering to reach particular locations in the atmosphere or on the surface [55.106].

In contrast, the Mars atmosphere is too thin for powered dirigibles to work (at least with the power-to-weight ratio of any current propulsion technology). Balloon aerobots could be deployed in the Mars atmosphere, and could ascend and descend, but probably could not be steered precisely to specific locations, at least not by use of a propulsion system. Polar balloons could circumnavigate either pole many times, or equatorial balloons could make one partial circuit around the planet, until they impact the *Tharsis Bulge*, a North–South string of high-altitude volcanoes that represents an essentially impenetrable barrier to any equatorial balloon having a reasonable payload. Because of the problems with *lighter-than-air* vehicles in the thin Mars atmosphere, there has been considerable study of airplanes for use in exploring Mars. Aircraft can be designed to have reasonable lift-to-drag ratios in the Mars atmosphere, so that their performance is not too different from airplanes on Earth. Most often considered are gliders that deploy directly from an aeroshell that comes in to the Mars atmosphere at hypersonic velocity, and then proceed to glide hundreds or a thousand kilometers before impact. One common mission concept is to fly down the great Valles Marineris canyon, taking high-resolution imagery and spectrometry of the walls of that canyon. Powered aircraft have also been considered, including those that land and regenerate their propellant (e.g., using solar power and atmospheric CO_2) so as to be able to make multiple flights.

On Titan, like Venus, buoyant devices are generally considered more attractive than surface vehicles (although helicopters have been proposed). Also like Venus, the atmosphere on Titan contains many obscuring particles and aerosols so that high-resolution imaging over a broad spectrum is only possible by getting close to the surface. This makes balloons or powered dirigibles very attractive. On Venus the extreme surface temperature makes it challenging to make a surface vehicle operate for any extended duration. On Titan, there is a significant risk that some sort of hydrocarbon *goo* exists on the surface that might foul any surface vehicle. Thus both Titan and Venus are considered especially attractive targets for the use of aerobots, especially in the form of powered dirigibles. Navigation of such aerobots presumably would be accomplished primarily by sensing the terrain and navigating relative to any landmarks that can be discerned. When these vehicles operate in the upper atmosphere, they can augment their position knowledge by means of sun or star tracking (as referenced to the local vertical). Deeper in the atmosphere, this may not be possible. One key issue is whether direct communications to Earth are envisioned, or relay via satellite. If there is a satellite in orbit, it can provide considerable radio-navigation assistance and relatively frequent communications when the aerobot is on the side away from the Earth (both Venus and Titan spin very slowly). But a satellite relay is expensive, so the least expensive options require that the dirigible have a large high-gain antenna (usually presumed to be inside the gas bag). Radio-based servo pointing at the Earth will provide precise navigation information (again along with precise measurements of local vertical). However, when the aerobot goes out-of-sight beyond the limb of the planet, it may spend days or weeks out of communications with the Earth. This is probably the situation calling for the highest degree of autonomy of any that have been envisioned in robotic planetary exploration of the solar system.

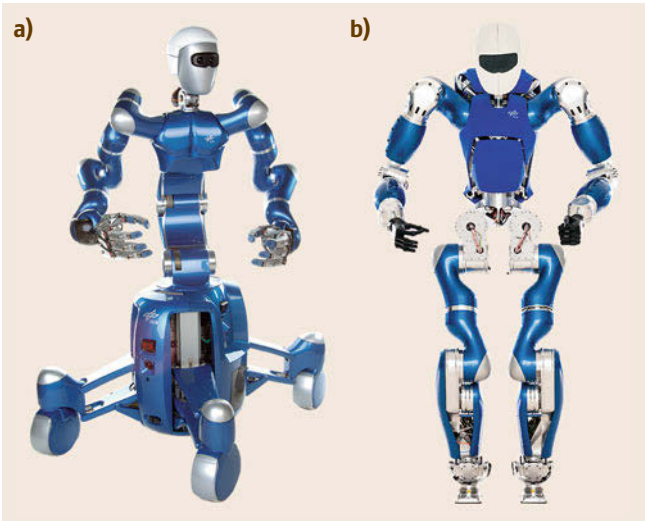


Fig.55.39a,b DLR' JUSTIN, wheeled version **(a)** and legged version **(b)**

Fig. 55.40 DEXHAND

55.4.4 Mobility Concepts and Subsurface Platforms

For high mobility on Moon, planets, and asteroids there is still not a final answer which technology would be optimal. Although multilegged crawlers (e.g., DLR's six-legged version as shown in Fig. 55.41) seem to be the best alternative for investigating steep craters, 4-wheeled rovers may climb up and down unbelievably steep slopes. May be wheel–leg combinations as realized in JPL's ATHLETE (Fig. 55.42) or in DLR's conceptual design (Fig. 55.43) will turn out to be the optimal solution.

Precise autonomous landing based on visual data is a prerequisite for exploration, closely related to the Future Space Systems program.

DLR's main interest however aims at fast locomotion by local autonomy thus (including collision avoidance and real-time path planning) circumventing the problem of long signal delays from 3 s (Moon) to 15–30 min (Mars). Stereo cameras with field-programmable gate array (FPGA) processor chips are capable of modeling the environment in 3-D real time, using e.g., the so-called semiglobal matching (SGM). Thus the goal of moving up to 10 km per hour seems realizable now.

Other modes of mobility may be superior when gravity is e.g., only 10 000 times smaller than that on earth as is the case on some asteroids. e.g., for a Japanese mission Hayabusa 2 a *jumping shoe-box* is developed by DLR using just a small excentric motor that causes moderate hopping motions over a few 100 m without reaching the fairly low escape velocity.

Subsurface exploration of planetary bodies holds great promise: it is believed that a liquid-water aquifer may exist at significant depths on Mars, and perhaps an under-ice ocean on Europa and Ganymede which probably represent the best possible locations within the solar system to look for extant (as opposed to extinct) extraterrestrial life. Also, in the lunar polar *dark craters* there is some evidence of the existence of water ice or other volatiles, and perhaps there exists a layered geologic record of impacts in the Earth–Moon system in these cold-traps. Even access to a depth of a few meters holds the promise of reaching pristine scientific samples that have not been exposed to thermal cycling or ionizing radiation [55.107].

The prevailing wisdom has been that traditional sorts of drilling rigs are required to access deep underground, involving drill towers, multisegmented drill strings, large robotic systems to serve the function of a terrestrial drilling crew, and large power systems. Also, terrestrial drilling is usually done using large amounts of fluids (water, air, or mud) to flush away cuttings and to cool and lubricate the cutter. The NASA Mars Technology Program has funded contractors that demonstrated reaching 10 m of depth in a realistic setting with segmented drill strings without the use of fluids. While this is much less than needed to reach the putative liquid water, it is much more than is reachable by previous techniques [55.108].

Fig. 55.41 DLR's six-legged crawler

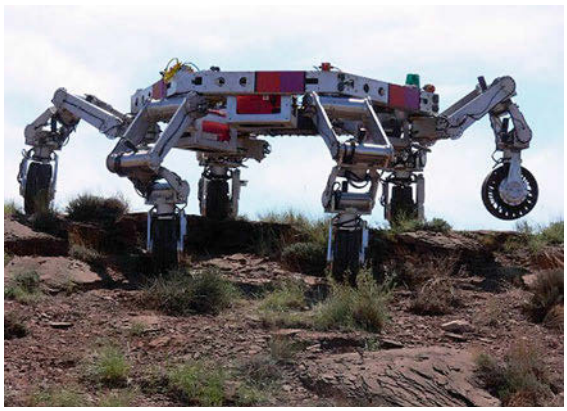


Fig. 55.42 JPL's ATHLETE

Fig. 55.43 Modular rover concept

Other approaches have been proposed such as Moles or Inchworms that could be relatively self-contained and yet might reach great depths without the mass and complexity of a large drill tower and segmented drill string. A key issue is that it appears that the needed energy cannot be stored on-board such self-contained drills, at least if it is stored as chemical energy. This is because drilling through terrain requires that some of the chemical bonds that hold the terrain together be broken, and so if the energy of chemical bonds is used to provide that power, then a given volume of chemical energy storage can only advance some fixed ratio of its length into the terrain, where the ratio is determined by the efficiency in taking bond energies of one sort to break bonds of a different sort. Based on these considerations, it appears unlikely that a completely self-contained subsurface vehicle could advance more than perhaps a hundred times its own length. Unless nuclear power sources are considered (and they have been), this requires some sort of tether to the surface to provide a nearly unlimited source of energy. Another problem for subsurface vehicles is that rock tends to expand when it is pulverized (in a process called *comminution*). Nonporous rock typically expands in volume by a few tens of percent when excavated, which means that fully self-contained subsurface vehicles have a severe *conservation of volume* problem. In principle the rock can be compressed back into its original volume, but this generally requires pressures much greater than the compressive strength of the original rock. The energy required to do this is much larger than the energy required to excavate the rock in the first place, and would become the dominant use of energy in an already energy-intensive effort.

As a result, it is generally assumed that any subsurface vehicle must keep some access tunnel open to the surface so that the excess volume of cuttings can be transported out. If this tunnel is available, then it seems that a means for getting power from the surface is also available, so that self-contained nuclear power is not needed. Subsurface vehicles with diameters as small as one or a few centimeters have been proposed that could potentially reach great depths within the mass and power constraints of feasible planetary robotic exploration missions.

55.5 Conclusions and Further Reading

Space robotics as a field is still in its infancy. The speed-of-light delays inherent in remote space operations makes problematic the master–slave teleoperation approach that has been very useful in the undersea and nuclear industries. Space robotics lacks the highly repetitive operations in a tightly structured environment that characterize industrial robotics. Hardware handled by space robots is very delicate and expensive. All three of these considerations have led to the fact that relatively few space robots have been flown, they have been very slow in operation, and only a small variety of tasks have been attempted. Nonetheless, the potential rewards of space robotics are great – exploring the solar system, creating vast space telescopes that may unlock the secrets of the universe, and enabling any viable space industries all seem to require major use of space robots. The scale of the solar system is not so great (a few light-hours) that human intelligence cannot always supplement even the most remote space robot that becomes confused or stuck. Indeed, for the Moon (with only a few seconds of time delay) it seems that hazard avoidance and reliable closure of force-feedback loops is all that is required to make a highly useful robotic system. For Mars (with tens of minutes of time delay), along with hazard avoidance and force loop closure, it seems that robust anomaly detection (with modest reflexive *safing* procedures) and perhaps scientific-novelty detection are probably all that is needed. High levels of autonomy are enhancing but not enabling for work in the inner solar system, and become more and more desirable for robots that are sent farther into the outer solar system.

For further reading, the following materials are suggested [55.109–114].

Video-References

VIDEO 330 DLR ROTEX: The first remotely controlled space robot
available from http://handbookofrobotics.org/view-chapter/55/videodetails/330

VIDEO 331 DLR predictive simulation compensating 6 seconds round-trip delay
available from http://handbookofrobotics.org/view-chapter/55/videodetails/331

VIDEO 332 DLR GETEX manipulation experiments on ETSVII
available from http://handbookofrobotics.org/view-chapter/55/videodetails/332
VIDEO 333 DLR ROKVISS animation
available from http://handbookofrobotics.org/view-chapter/55/videodetails/333
VIDEO 334 DLR ROKVISS camera images pulling spring
available from http://handbookofrobotics.org/view-chapter/55/videodetails/334
VIDEO 336 DLR ROKVISS disassembly
available from http://handbookofrobotics.org/view-chapter/55/videodetails/336
VIDEO 337 DLR telepresence demo remove cover
available from http://handbookofrobotics.org/view-chapter/55/videodetails/337
VIDEO 338 DLR telepresence demo with time delay
available from http://handbookofrobotics.org/view-chapter/55/videodetails/338
VIDEO 339 DLR DEOS demonstration mission simulation
available from http://handbookofrobotics.org/view-chapter/55/videodetails/339

References

55.1 D.L. Akin, M.L. Minsky, E.D. Thiel, C.R. Kurtzman: *Space Applications of Automation, Robotics and Machine Intelligence Systems (ARAMIS). Phase II* (NASA, Washington 1983)

55.2 C.G. Wagner-Bartak, J.A. Middleton, J.A. Hunter: Shuttle remote manipulator system hardware test facility, 11th Space Simulat. Conf. (1980) pp. 79–94, NASA CP-2150

55.3 S. Greaves, K. Boyle, N. Doshewnek: Orbiter boom sensor system and shuttle return to flight: Operations analyses, AIAA Guidance Navigation Contr. Conf. Exhibit. San Francisco (2005)

55.4 C. Crane, J. Duffy, T. Carnahan: A kinematic analysis of the space station remote manipulator system (SSRMS), J. Robotic Syst. **8**, 637–658 (1991)

55.5 M.F. Stieber, C.P. Trudel, D.G. Hunter: Robotic systems for the International Space Station, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 3068–3073

55.6 D. Bassett, A. Abramovici: Special purpose dexterous manipulator (SPDM) requirements verification, Proc. 5th Int. Symp. Artif. Intell. Robotics Autom. Space (1999) pp. 43–48, ESA SP-440

55.7 R. Boumans, C. Heemskerk: The European robotic arm for the International Space Station, Robotics Auton. Syst. **23**(1), 17–27 (1998)

55.8 P. Laryssa, E. Lindsay, O. Layi, O. Marius, K. Nara, L. Aris, T. Ed: International Space Station robotics: A comparative study of ERA, JEMRMS and MSS, Proc. 7th ESA Workshop Adv. Space Technol. Robotics Autom. (ASTRA) (2002)

55.9 T. Matsueda, K. Kuraoka, K. Goma, T. Sumi, R. Okamura: JEMRMS system design and development status, Proc. IEEE Natl. Telesyst. Conf. (NTC) (1991) pp. 391–395

55.10 S. Doi, Y. Wakabayashi, T. Matsuda, N. Satoh: JEM remote manipulator system, J. Aeronaut. Space Sci. Jpn. **50**(576), 7–14 (2002)

55.11 H. Morimoto, N. Satoh, Y. Wakabayashi, M. Hayashi, Y. Aiko: Performance of Japanese robotic arms of the International Space Station, 15th IFAC World Congr. (2002)

55.12 G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl: Sensor-based space robotics – ROTEX and its telerobotic features, IEEE Trans. Robotics Autom. **9**(5), 649–663 (1993)

55.13 M. Oda, K. Kibe, F. Yamagata: ETS-VII – Space robot in-orbit experiment satellite, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 739–744

55.14 K. Yoshida: Engineering test satellite VII flight experiments for space robot dynamics and control: Theories on laboratory test beds ten years ago, now in orbit, Int. J. Robotics Res. **22**(5), 321–335 (2003)

55.15 K. Landzettel, B. Brunner, G. Hirzinger, R. Lampariello, G. Schreiber, B.-M. Steinmetz: A unified ground control and programming methodology for space robotics applications – demonstrations on ETS-VII, Proc. Int. Symp. Robotics (ISR) (2000) pp. 422–427

55.16 J.C. Parrish, D.L. Akin, G.G. Gefke: The ranger telerobotic shuttle experiment: Implications for operational EVA/robotic cooperation, Proc. SAE Int. Conf. Environ. Syst. (2000)

55.17 J. Shoemaker, M. Wright: Orbital express space operations architecture program, Proc. SPIE **5088**, 1–9 (2003)

55.18 C. Preusche, D. Reintsema, K. Landzettel, G. Hirzinger: Robotics component verification on ISS ROKVISS – Preliminary results for telepresence, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 4595–4601

55.19 K. Landzettel, A. Albu-Schaffer, C. Preusche, D. Reintsema, B. Rebele, G. Hirzinger: Robotic on-orbit servicing – DLR's experience and perspective, Proc. IEEE/RSJ Int. Conf. Intell. Robots Systems (IROS) (2006) pp. 4587–4594

55.20 J.-H. Ryu, J. Artigas, C. Preusche: A passive bilateral control scheme for a teleoperator with time-varying communication delay, Mechatronics **20**(7), 812–823 (2010)

55.21 J. Artigas, J.-H. Ryu, C. Preusche, G. Hirzinger: Network representation and passivity of delayed teleoperation systems, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 177–183

55.22 E. Stoll, U. Walter, J. Artigas, C. Preusche, P. Kremer, G. Hirzinger, J. Letschnik, H. Pongrac: Ground

verification of the feasibility of telepresent on-orbit servicing, J. Field Robotics **26**(3), 287–307 (2009)
55.23 C. Ott, J. Artigas, C. Preusche, G. Hirzinger: Subspace-oriented energy distribution for the time domain passivity approach, Proc. IEEE/RSJ Int. Conf. Intell. Robots Systems (IROS'11) (2011) pp. 665–671
55.24 A.P. Vinogradov: *Lunokhod 1 Mobile Lunar Laboratory* (JPRS, Washington 1971), JPRS identification number 54525
55.25 A. Mishkin: *Sojourner: An Insider's View of the Mars Pathfinder Mission* (Berkley Books, New York 2004)
55.26 B. Wilcox, T. Nguyen: Sojourner on mars and lessons learned for future planetary rovers, 28th Int. Conf. Environ. Syst. (1998)
55.27 M. Maimone, A. Johnson, Y. Cheng, R. Willson, L. Matthies: Autonomous navigation results from the Mars exploration rover (MER) mission, Proc. 9th Int. Symp. Exp. Robotics (ISER) (2004)
55.28 M. Maimone, Y. Cheng, L. Matthies: Two years of visual odometry on the Mars exploration rovers, J. Field Robotics **24**(3), 169–186 (2007)
55.29 M.G. Bekker: *Off-The-Road Locomotion* (Univ. Michigan Press, Ann Arbor 1960)
55.30 M.G. Bekker: *Introduction to Terrain-Vehicle Systems* (Univ. Michigan Press, Ann Arbor 1969)
55.31 R.A. Lewis, A.K. Bejczy: Planning considerations for a roving robot with arm, Int. Jt. Conf. Artif. Intell. (IJCAI) (1973) pp. 308–316
55.32 A. Thompson: The navigation system of the JPL Robot, Int. Jt. Conf. Artif. Intell. (IJCAI) (1977) pp. 749–757
55.33 Y. Yakimovsky, R.T. Cunningham: A system for extracting three-dimensional measurements from a stereo pair of TV cameras, Comp. Graph. Image Process. **7**, 195–210 (1978)
55.34 D. Dooling: *Planetary rovers might roam better with an elastic loop mobility system, NASA Science News, http://science.nasa.gov/newhome/ headlines/msad28apr98_1b.htm* 1989)
55.35 B.H. Wilcox: *Computer-Aided Remote Driving*, NASA Tech Briefs 18; 3 (NASA, Washington 1994)
55.36 E. Krotkov, J. Bares, T. Kanade, T. Mitchell, R. Simmons, W. Whittaker: Ambler: a six-legged planetary rover, 5th Int. Conf. Adv. Robotics, Robots Unstructured Environ. (ICAR), Vol. 1 (1991) pp. 717–722
55.37 J. Bares, W. Whittaker: Walking robot with a circulating gait, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 2 (1990) pp. 809–816
55.38 B. Wilcox, D. Gennery: A Mars rover for the 1990s, J. Br. Interplanet. Soc. **40**, 484–488 (1987)
55.39 B. Wilcox, L. Matthies, D. Gennery: Robotic vehicles for planetary exploration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1992)
55.40 D. Gennery, T. Litwin, B. Wilcox, B. Bon: Sensing and perception research for space telerobotics at JPL, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 311–317
55.41 J. Balaram, S. Hayati: A supervisory telerobotics testbed for unstructured environments, J. Robotics Syst. **9**(2), 261–280 (1992)
55.42 G. Giralt, L. Boissier: The French planetary rover VAP: concept and current developments, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1992) pp. 1391–1398, LAAS Report No. 92227
55.43 R. Chatila, S. Lacroiz, G. Giralt: A case study in machine intelligence: Adaptive autonomous space rovers, Int. Conf. Field Serv. Robotics (FSR) (1997) pp. 101–108, LAAS Report No. 97463
55.44 A. Castano, A. Fukunaga, J. Biesiadecki, L. Neakrase, P. Whelley, R. Greeley, M. Lemmon, R. Castano, S. Chien: Autonomous detection of dust devils and clouds on Mars, Proc. IEEE Int. Conf. Image Process. (2006) pp. 2765–2768
55.45 L. Matthies: Stereo vision for planetary rovers: Stochastic modeling to near real-time implementation, Int. J. Comput. Vis. **8**(1), 71–91 (1992)
55.46 K. Yoshida, D.N. Nenchev, M. Uchiyama: Moving base robotics and reaction management control, 7th Int. Symp. Robotics Res., ed. by G. Giralt, G. Hirzinger (Springer, New York 1996) pp. 101–109
55.47 J. Russakow, S.M. Rock, O. Khatib: An operational space formulation for a free-flying multi-arm space robot, 4th Int. Symp. Exp. Robotics (1995) pp. 448–457
55.48 Y. Xu, T. Kanade (Eds.): *Space Robotics: Dynamics and Control* (Kluwer, Boston 1993)
55.49 Y. Umetani, K. Yoshida: Continuous path control of space manipulators mounted on OMV, Acta Astronaut. **15**(12), 981–986 (1987), presented at the 37th IAF Conf, Oct. 1986
55.50 Y. Umetani, K. Yoshida: Resolved motion rate control of space manipulators with generalized Jacobian matrix, IEEE Trans. Robotics Autom. **5**(3), 303–314 (1989)
55.51 K. Yoshida: Impact dynamics representation and control with extended inversed inertia tensor for space manipulators, 6th Int. Symp. Robotics Res. (1994) pp. 453–463
55.52 K. Yoshida: Experimental study on the dynamics and control of a space robot with the experimental free-floating robot satellite (EFFORTS) simulators, Adv. Robotics **9**(6), 583–602 (1995)
55.53 Y. Nakamura, R. Mukherjee: Nonholonomic path planning of space robots via a bidirectional approach, IEEE Trans. Robotics Autom. **7**(4), 500–514 (1991)
55.54 S. Dubowsky, M. Torres: Minimizing attitude control fuel in space manipulator systems, Proc. Int. Symp. Artif. Intell. Robotics Autom. (i-SAIRAS) (1990) pp. 259–262
55.55 S. Dubowsky, M. Torres: Path planning for space manipulators to minimize spacecraft attitude disturbances, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1991) pp. 2522–2528
55.56 K. Yoshida: Practical coordination control between satellite attitude and manipulator reaction dynamics based on computed momentum concept, Proc. 1994 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1578–1585

55.57 M. Oda: Coordinated control of spacecraft attitude and its manipulator, Proc. 1996 IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 732–738

55.58 Z. Vafa, S. Dubowsky: On the dynamics of manipulators in space using the virtual manipulator approach, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1987) pp. 579–585

55.59 E. Papadopoulos, S. Dubowsky: Dynamic singularities in the control of free-floating space manipulators, ASME J. Dyn. Syst. Meas. Control **115**(1), 44–52 (1993)

55.60 Y. Umetani, K. Yoshida: Workspace and manipulability analysis of space manipulator, Trans. Soc. Instrum. Contr. Eng. **E-1**(1), 116–123 (2001)

55.61 D.N. Nenchev, Y. Umetani, K. Yoshida: Analysis of a redundant free-flying spacecraft/manipulator system, IEEE Trans. Robotics Autom. **8**(1), 1–6 (1992)

55.62 D.N. Nenchev, K. Yoshida, P. Vichitkulsawat, M. Uchiyama: Reaction null-space control of flexible structure mounted manipulator systems, IEEE Trans. Robotics Autom. **15**(6), 1011–1023 (1999)

55.63 V.H. Schulz: Reduced SQP Methods for Large Scale Optimal Control Problems in DAE with Application to Path Planning Problems for Satellite Mounted Robots, Ph.D. Thesis (Univ. Heidelberg, Heidelberg 1995)

55.64 E. Papadopoulos, S. Moosavian: Dynamics and control of multi-arm space robots during chase and capture operations, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1994) pp. 1554–1561

55.65 K. Yoshida, D. Dimitrov, H. Nakanishi: On the capture of tumbling satellite by a space robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006)

55.66 C.G. Henshaw: A unification of artificial potential function guidance and optimal trajectory planning, 15th Space Flight Mech. Meet. (2005)

55.67 F. Aghili: A prediction and motion-planning scheme for visually guided robotic capturing of free-floating tumbling objects with uncertain dynamics, IEEE Trans. Robotics **28**(3), 634–649 (2012)

55.68 R. Lampariello, G. Hirzinger: Generating feasible trajectories for autonomous on-orbit grasping of spinning debris in a useful time, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013)

55.69 T. Binder, L. Blank, R. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J.P. Schloeder, O. von Stryk: Introduction to model based optimization of chemical processes on moving horizons. In: *Online Optimization of Large Scale Systems*, ed. by M. Grotschel, S.O. Krumke, J. Rambau (Springer, Berlin, Heidelberg 2001) pp. 295–339

55.70 S. Jacobsen, C. Lee, C. Zhu, S. Dubowsky: Planning of safe kinematic trajectories for free-flying robots approaching an uncontrolled spinning satellite, ASME Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf. (DETC) (2002)

55.71 S.K. Agrawal, K. Pathak, J. Franch, R. Lampariello, G. Hirzinger: A differentially flat open-chain space robot with arbitrarily oriented joint axes and two momentum wheels at the base, IEEE Trans. Autom. Contr. **54**(9), 2185–2191 (2009)

55.72 D.N. Nenchev, K. Yoshida, M. Uchiyama: Reaction null-space based control of flexible structure mounted manipulator systems, IEEE 35th Conf. Decis. Control (1996) pp. 4118–4123

55.73 W.J. Book, S.H. Lee: Vibration control of a large flexible manipulator by a small robotic arm, Proc. Am. Control Conf. (1989)

55.74 M.A. Torres: Modelling, Path-Planning and Control of Space Manipulators: The Coupling Map Concept, Ph.D. Thesis (MIT, Cambridge 1993)

55.75 S. Abiko, K. Yoshida: An effective control strategy of japanese experimental module remote manipulator system (JEMRMS) using coupled and uncoupled dynamics, Proc. 7th Int. Symp. Artif. Intell. Robotics Autom. Space (2003), Paper AS18 (CD-ROM)

55.76 S. Abiko: Dynamics and Control for a Macro-Micro Manipulator System Mounted on the International Space Station, Ph.D. Thesis (Tohoku Univ., Tokyo 2005)

55.77 S. Abiko, K. Yoshida: On-line parameter identification of a payload handled by flexible based manipulator, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 2930–2935

55.78 S. Abiko, K. Yoshida: An adaptive control of a space manipulator for vibration suppression, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)

55.79 G. Gilardi, I. Shraf: Literature survey of contact dynamics modeling, Mech. Mach. Theory **37**, 1213–1239 (2002)

55.80 K. Yoshida, H. Nakanishi, H. Ueno, N. Inaba, T. Nishimaki, M. Oda: Dynamics, control, and impedance matching for robotic capture of a non-cooperative satellite, Adv. Robotics **18**(2), 175–198 (2004)

55.81 H. Nakanishi, K. Yoshida: Impedance control of free-flying space robot for orbital servicing, J. Robotics Mechatron. **18**(5), 608–617 (2006)

55.82 P.M. Pathak, A. Mukherjee, A. DasGupta: Impedance control of space robots using passive degrees of freedom in controller domain, J. Dyn. Syst. Meas. Control **127**, 564–578 (2006)

55.83 S. Abiko, R. Lampariello, G. Hirzinger: Impedance control for a free-floating robot in the grasping of a tumbling target with parameter uncertainty, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006)

55.84 K. Iagnemma, H. Shibly, S. Dubowsky: On-line traction parameter estimation for planetary rovers, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

55.85 K. Yoshida, H. Hamano: Motion dynamics of a rover with slip-based traction model, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002)

55.86 A. Jain, J. Guineau, C. Lim, W. Lincoln, M. Pomerantz, G. Sohl, R. Steele: Roams: Planetary surface rover simulation environment, Proc. 7th Int. Symp. Artif. Intell. Robotics Autom. Space (i-SAIRAS) (2003)

55.87 K. Yoshida, T. Watanabe, N. Mizuno, G. Ishigami: Terramechanics-based analysis and traction con-

trol of a lunar/planetary rover, Proc. Int. Conf. Field Serv. Robotics (FSR) (2003)
55.88 K. Iagnemma, S. Dubowsky: *Mobile Robots in Rough Terrain: Estimation, Motion Planning, and Control With Application to Planetary Rovers*, Springer Tracts in Advanced Robotics, Vol. 12 (Springer, Berlin, Heidelberg 2004)
55.89 K. Yoshida, G. Ishigami: Steering characteristics of a rigid wheel for exploration on loose soil, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004)
55.90 H. Shibly, K. Iagnemma, S. Dubowsky: An equivalent soil mechanics formulation for rigid wheels in deformable terrain, with application to planetary exploration rovers, J. Terramech. **42**(1), 1–13 (2005)
55.91 R. Bauer, W. Leung, T. Barfoot: Experimental and simulation results of wheel-soil interaction for planetary rovers, Proc. 2005 IEEE Int. Conf. Intell. Robots Syst. (IROS) (2005)
55.92 A. Ellery, N. Patel, R. Bertrand, J. Dalcomo: Exomars rover chassis analysis and design, Proc. 8th Int. Symp. Artif. Intell. Robotics Autom. Space (i-SAIRAS) (2005)
55.93 A. Gibbesch, B. Schäfer: Multibody system modelling and simulation of planetary rover mobility on soft terrain, Proc. 8th Int. Symp. Artif. Intell. Robotics Autom. Space (i-SAIRAS) (2005)
55.94 G. Ishigami, K. Yoshida: Steering characteristics of an exploration rover on loose soil based on all-wheel dynamics model, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)
55.95 G. Ishigami, A. Miwa, K. Nagatani, K. Yoshida: Terramechanics-based model for steering maneuver of planetary exploration rovers on loose soil, J. Field Robotics **24**(3), 233–250 (2007)
55.96 V.K. Tiwari, K.P. Pandey, P.K. Pranav: A review on traction prediction equations, J. Terramech. **47**(3), 191–199 (2010)
55.97 L. Ding, H. Gao, Z. Deng, K. Nagatani, K. Yoshida: Experimental study and analysis on driving wheels' performance for planetary exploration rovers moving in deformable soil, J. Terramech. **48**(1), 27–45 (2011)
55.98 G. Meirion-Griffith, M. Spenko: A modified pressure-sinkage model for small, rigid wheels on deformable terrains, J. Terramech. **48**(2), 149–155 (2011)
55.99 J.Y. Wong: *Theory of Ground Vehicles* (Wiley, New York 1978)
55.100 J.Y. Wong, A.R. Reece: Prediction of rigid wheel preformance based on the analysis of soil-wheel stresses. Part I, Preformance of driven rigid wheels, J. Terramech. **4**, 81–98 (1967)
55.101 Z. Janosi, B. Hanamoto: The analytical determination of drawbar pull as a function of slip for tracked vehicle in deformable soils, Proc 1st Int. Conf. Terrain-Veh. Syst. (1961)
55.102 E. Hegedus: *A Simplified Method for the Determination of Bulldozing Resistance*, Land Locomotion Laboratory, Rep. No. 61U.S. (Army Tank Automotive Command, Warren 1960)
55.103 I. Rekleitis, E. Martin, G. Rouleau, R. L'Archeveque, K. Parsa, E. Dupuis: Autonomous capture of a tumbling satellite, J. Field Robotics **24**(4), 275–296 (2007)
55.104 J. Blamont: Balloons on other planets, Adv. Space Res. **1**, 63–69 (1981)
55.105 J.A. Cutts, K.T. Nock, J.A. Jones, G. Rodriguez, J. Balaram, G.E. Powell, S.P. Synott: Aerovehicles for planetary exploration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995)
55.106 M.K. Heun, J.A. Jones, J.L. Hall: Gondola design for Venus deep-atmosphere aerobot operations, AIAA 36th Aerosp. Sci. Meeting Exhibit. ASME Wind Energy Symp. (1998)
55.107 S. Miller, J. Essmiller, D. Beaty: Mars deep drill – A mission concept for the next decade, Space Conf. Exhibit. (2004) pp. 2004–6048
55.108 S. Mukherjee, P. Bartlett, B. Glass, J. Guerrero, S. Stanley: Technologies for exploring the martian subsurface, Proc. IEEE Aerosp. Conf. (2006), Paper No. 1349
55.109 C.F. Ruoff, S.B. Skaar (Eds.): *Teleoperation and Robotics in Space*, Progress in Astronautics and Aeronautics Series (AIAA, Notre Dame 1994)
55.110 J. Field Robot. Special Issue on Space Robotics, Part I–Part III, **24**(3–5), 167–434 (2007)
55.111 G.F. Bekey (Ed.): *International Assessment of Research and Development in Robotics* (WTEC, Baltimore 2006), http://www.wtec.org/robotics/
55.112 NSRC (Ed.): *Assessment of Options for Extending the Life of the Hubble Space Telescope*, National Research Council: Final Rep. (National Academies Press, Washington 2005)
55.113 A.M. Howard, E.W. Tunstel: *Intelligence for Space Robotics* (TSI, San Antonio 2006)
55.114 T.B. Sheridan: Space teleoperation through time delay: Review and prognosis, IEEE Trans. Robotics Autom. **9**(5), 592–606 (1993)

Multimedia Contents

56. Robotics in Agriculture and Forestry

Marcel Bergerman, John Billingsley, John Reid, Eldert van Henten

Robotics for agriculture and forestry (A&F) represents the ultimate application of one of our society's latest and most advanced innovations to its most ancient and important industries. Over the course of history, mechanization and automation increased crop output several orders of magnitude, enabling a geometric growth in population and an increase in quality of life across the globe. Rapid population growth and rising incomes in developing countries, however, require ever larger amounts of A&F output. This chapter addresses robotics for A&F in the form of case studies where robotics is being successfully applied to solve well-identified problems. With respect to plant crops, the focus is on the in-field or in-farm tasks necessary to guarantee a quality crop and, generally speaking, end at harvest time. In the livestock domain, the focus is on breeding and nurturing, exploiting, harvesting, and slaughtering and processing. The chapter is organized in four main sections. The first one explains the scope, in particular, what aspects of robotics for A&F are dealt with in the chapter. The second one discusses the challenges and opportunities associated with the application of robotics to A&F. The third section is the core of the chapter, presenting twenty case studies that showcase (mostly) mature applications of robotics in various agricultural and forestry domains. The case studies are not meant to be comprehensive but instead to give the reader a general overview of how robotics has been applied to A&F in the last 10 years. The fourth section concludes the chapter with a discussion on specific improvements to current technology and paths to commercialization.

56.1 **Section Scope** 1464
56.2 **Challenges and Opportunities** 1465
56.3 **Case Studies** 1467
56.3.1 Optimized Coverage for Arable Farming 1467
56.3.2 Weed Control 1468
56.3.3 High Precision Seeding 1470
56.3.4 Crop Yield Estimation 1471
56.3.5 Precision Irrigation 1471
56.3.6 Tree Fruit Production 1472
56.3.7 Vehicle Formation Control 1473
56.3.8 Date Palm Tree Spraying 1474
56.3.9 Plant Probing 1475
56.3.10 Cucumber Harvesting 1476
56.3.11 Cucumber Leaf Removal 1477
56.3.12 Rose Harvesting 1477
56.3.13 Strawberry Harvesting 1477
56.3.14 Pot Handling in Nurseries and Greenhouses 1478
56.3.15 Precision Forestry 1479
56.3.16 Semi-Automation of Forwarder Crane 1480
56.3.17 Livestock Breeding and Nurturing 1482
56.3.18 Livestock Exploitation 1483
56.3.19 Livestock Harvesting, Slaughtering, and Processing 1484
56.3.20 Aerial-Based Precision Agriculture 1485
56.4 **Conclusion** 1487
Video-References 1488
References 1489

Robotics for agriculture and forestry represents the ultimate application of one of our society's latest and most advanced innovations to its most ancient industries. Since the dawn of civilization, agriculture and forestry (A&F for short) remain chief among humankind's most important economic activities, providing the food, feed, fiber, and fuel necessary for our survival. (In this chapter, agriculture is understood as in the Merriam-Webster definition:

> *the science, art, or practice of cultivating the soil, producing crops, and raising livestock and in varying degrees the preparation and marketing of the resulting products.*

Therefore, the term *crop* may be used here to denote any product of an agricultural or forestry process, including grains, cereals, fruit, vegetables, nuts, trees, beef, wool, etc. Whenever necessary, we will differentiate plant from animal products appropriately.)

Over the course of history, mechanization and automation – from the manual ploughs of yore to the modern combines of today – increased crop output several orders of magnitude. This, in turn, enabled a geometric growth in population and a corresponding increase in quality of life across the globe. Rapid population growth and rising incomes in developing countries, however, require ever larger amounts of A&F output. Scientists predict that agricultural production must double to meet the demands of nine billion people in 2050 [56.1–3]. Clearly, this cannot be achieved by simply doubling the inputs (land, water, seeds, labor, etc.) because of constrained resources and environmental concerns. Therefore, the efficiency of the A&F system must increase in a sustainable and consistent manner.

According to *Reid* [56.4], global agricultural total factor productivity (TFP), or the output per unit of total resources used in production, must increase from the current 1.4 to 1.75 to double agricultural output by 2050. This requires significant scientific, technologic, and management advances in all of the factors that impact TFP – seeds, soil, water, fertilizers, herbicides, insecticides, crop architecture, cultural practices, automation, labor, public policy, etc. While robotics is but one of these factors, it is one with the potential to effect A&F in a broad, systemic way, and contribute significantly to meeting our future needs.

In this chapter we address the field of agricultural and forestry robotics *from the point of view of the applications it enables*, rather than from the point of view of the elementary technologies it comprises. Furthermore, we chose to present a limited number of case studies where robotics is being successfully applied to solve well-identified problems, rather than a comprehensive survey of all work reported in the literature. We believe the former is more meaningful as it showcases top-down, problem-oriented solutions (*market pull*) rather than bottom-up, technology-led ones (*technology push*). We focus on examples from the last five to ten years, as they represent work that leverages the most recent advances in sensors and computing.

We address applications of agricultural and forestry robotics in the form of twenty case studies where robotics is being successfully applied to solve well-identified problems. With respect to plant crops, the focus is on the in-field or in-farm tasks necessary to guarantee a quality crop and, generally speaking, end at harvest time, before the crop is transported to a packing plant or warehouse. In the livestock domain, the focus is on breeding and nurturing, exploiting, harvesting, and slaughtering and processing.

56.1 Section Scope

Before presenting the case studies that showcase recent advances in robotics applied to A&F, we must define the scope of the chapter, both in terms of crop production processes and robotic technologies.

A typical crop production cycle includes several processes, among them field preparation, seeding/breeding, transplanting, planting, growing, maintenance (including attaching plants to support structures, disbudding, removing leaves, pruning tree limbs and shoots, thinning blossoms and fruit, nurturing animals, etc.), exploiting/harvesting/slaughtering, sorting, and packing. In all of them, internal transport of people, machines, and produce play a role. Single-harvest crops such as lettuce must be replaced once harvested; multiple-harvest crops such as apples, tomatoes, and roses last a year or even several years before they need to replanted. Depending on the crop, machinery with varying levels of automation exists for some or all of these processes. In grain and cereal production, for example, farmers have access to commercial machines for tilling, seeding, transplanting, spraying, irrigating, and harvesting. In fresh fruits and vegetables, on the other hand, mechanization and automation are more prevalent at the early and late stages of the production cycle [56.5], with crop maintenance and harvest remaining for the most part manual tasks.

With respect to plant crops, this chapter's scope is limited to the in-field or in-farm tasks necessary

to guarantee a quality crop and, generally speaking, end at harvest time, before the crop is transported to a packing plant or warehouse. Of course, there are many opportunities for robotics in those latter environments, e.g., automatic sorting and grading. We exclude them from our presentation as they are currently much more the focus of automation efforts than of robotics per se. In future editions of the Handbook, we will revisit this separation as robotics advances to post-harvest tasks.

In the livestock domain, much of farming concerns the nurturing, exploitation, harvesting, and slaughtering of animals. Slaughtering extends beyond the killing of the animal to its division into marketable portions, and subsequent packing and marketing. The essential processes considered in this section pertain to land animals and birds and include:

- Breeding and nurturing: Livestock can range from poultry to cattle in feedlots, including pigs and other animals. Their care involves environment and behavior monitoring, plus fodder distribution.
- Exploiting: In many cases, products are obtained from the live animals. Sheep are shorn for wool, cows are milked, chickens lay eggs, and bees produce honey.
- Harvesting: Mustering and collecting *free-range* cattle, feral pigs, and other unconstrained animals.
- Slaughtering and processing: Cattle, poultry, pigs, other animals are killed subject to strict regulations, then divided into marketable portions.

In a future edition of the Handbook fishing should be included, since many opportunities for robotics can be imagined there.

56.2 Challenges and Opportunities

Since the end of the industrial revolution, (arguably) the three most significant impacts robotics and automation have had on agriculture and forestry are:

1. Precision agriculture, or the use of sensors to precisely control when and where to apply inputs such as fertilizers and water
2. Auto-guidance on field crop machinery, which today can drive down a field with an accuracy unattainable by human drivers
3. Machines that harvest fruits and vegetables for processing (e.g., tomato paste and orange juice).

Academic and commercial researchers are now focusing on the next wave of *sensing*, *mobility*, and *manipulation* technologies that promise to increase A&F output and productivity.

Sensing entails measuring crop temperature, humidity, pH, wetness, image, range, and other physical attributes, and combining and analyzing the data for specific purposes. One example is a camera-based system that takes pictures of an apple orchard a few weeks before harvest and produces an accurate crop yield estimate that growers can use to plan and manage the harvest operation [56.6, 7]. The utility of sensing to A&F is that it enables decision-making at a level unattainable by human sensing alone, because the latter is either inherently inaccurate or slow or both.

Mobility relates to various levels of vehicle automation that enable driverless (or driver-assistive) field coverage; the most common example is a global positioning system/global navigation satellite system (GPS/GLS)-guided combine that harvests corn with minimal intersection between passes on the field, thus optimizing coverage and minimizing fuel consumption [56.8]. More recently, auto-guidance has started to migrate to orchard vehicles as well, albeit here other navigation sensors may be required because of poor satellite reception under thick canopies (VIDEO 26, VIDEO 91). In A&F, automated vehicles equipped with the appropriate implements enable (semi-) autonomous seeding, spraying, mowing, weed removal, harvesting, and animal feeding, among other operations (VIDEO 306, VIDEO 305). Mobility requires some level of sensor-based perception, provided by GPS/GNSS, inertial units, cameras, ladars, radars, etc. These sensors are not to be confused with those described in the previous paragraph, although there certainly are situations where a sensor can perform double duty of sensing for decision making and for navigation. In general, robotic mobility technology is currently less advanced than sensing.

Manipulation refers to the various operations performed directly on the crop, including pruning, thinning, harvesting, tree training, leaf probing, tree cutting, weed removal, etc. In general, this technology requires more sophisticated sensor-based perception than mobility, and in terms of field deployment is less advanced than either sensing or mobility.

Two domains that are particularly challenging for robotics research are orchard crops and crops in protected cultivation. (We use the term *orchard* liberally to include grape vineyards and orange groves, among other similar environments where crops are grown in well-defined rows.) These are highly valuable crops, potentially generating one to three orders of magnitude more revenue per acre than field crops. They are also

characterized by the need for intensive cultivation and skilled labor. Consider, for example, apple production in the US – no less than 50 to 60% of the variable costs to produce an apple is due to labor [56.9]. On top of that, labor is needed in bursts – in Washington state, in the US, up to seven times more workers are needed during the fall harvest season than during the winter pruning season. In many developed countries, labor availability for manual orchard operations is a challenge, putting significant pressure on growers to find innovative solutions to address their labor requirements. In the 2011 Tree Fruit Industry Perspective publication, industry leaders with the US Northwest Farm Credit Services say that:

> *[...] seasonal labor force utilized by the tree fruit industry for thinning and harvest operations will always be an issue of concern [...]. Labor shortages at various times throughout the growing season have occurred and will likely occur in the future. The perishable nature of the tree fruit mandates a strict timeliness in field operations, and the generally narrow harvest windows do not accommodate labor shortages at critical stages. [...] New technology [...] could reduce labor requirements substantially over the next five to 10 years. Specifically, such technology would include the use of platforms and mechanical-harvest methods in the orchards, and increased use of robotics [...].*

Partly in response to the high labor costs and partly to increase production efficiency, the tree fruit industry is moving in the direction of highly structured planting architectures. Whereas before an apple grower had two to four hundred trees per acre, she now has 1200 apple trees per acre in a *fruit wall* configuration that is much more conducive to automation (Fig. 56.1a). This has led to the development of autonomous vehicles and tractors that drive from garage to block and traverse tree rows for hours on end, including turning from row to row, without any human assistance [56.10]. When equipped with the appropriate implements, these robotic vehicles can mow, spray, and collect tree and crop data for inventory management; and when configured as platforms, they can carry workers pruning, thinning, training trees, and harvesting the top part of the trees, thus eliminating the inefficiencies and injuries related to ladders. In the future, manipulators mounted on such vehicles will be able to probe plants for phenotyping purposes, and automatically prune, thin, and harvest.

Fig. 56.1 **(a)** Modern apple orchard with trees arranged to form a *fruit wall*. **(b)** Vine tomato in a protected cultivation environment. These new architectures open the door to robotics-based production technologies that increase efficiency and reduce labor costs

A protected cultivation system is a powerful instrument to produce crops (Figure 56.1b). Greenhouses protect crops from unfavorable climate conditions and pests and offer the opportunity to modify the climate to create an environment that is optimal in terms of both crop quality and quantity. A protected cultivation is an intensive production method with high investment and operational costs, therefore permitting only the production of high value fruit and vegetable crops such as tomatoes, sweet peppers and cucumbers, flowers such as roses, chrysanthemums and gerberas and many types of potted plants. In the past decades, in western societies, this type of production has been confronted with increasing size of production facilities, increasing labor costs, reduced availability of sufficiently skilled labor, health problems of the employees due to heavy and repetitive tasks, and growing competition on the national and international markets. Automation and robotics are considered to be a way to address these issues. Additionally, growing concerns with food safety call for such technology. Last but not least, more and more a precision horticulture approach is adopted in which plants are treated on an individual basis, to improve quantity and quality of crop production whilst using resources as efficiently as possible. Given the current constraints on human labor, this has led to an even stronger call for automation and robotics.

Within the context of agricultural production, in terms of production area, protected cultivation is worldwide a relatively small business. Total area in use with protected cultivation worldwide is estimated at roughly 740 000 ha [56.11]. In terms of added value, however, protected cultivation plays a much more important role. In The Netherlands at a few percent of the area available for agricultural production, horticulture produces roughly 35% of the economic return of total agricultural production. In that case production is very capital and labor intensive. On a global scale, the potential of protected cultivation is being increasingly acknowledged. Yet, technology levels still vary widely, largely based on significant differences in local conditions in terms of market, economy, resource availability, etc.

Robotics research in protected cultivation has a history of some thirty years [56.12], focused mainly on

harvesting and chemical spraying operations. Robotic harvesting focused mainly on tomatoes, cucumbers, eggplants, sweet peppers, and strawberries, for which multiple examples of research prototypes have been developed [56.12] (VIDEO 304). Single examples are known of a harvesting robot for roses [56.13] and gerbera [56.14]. Spraying robots have been reviewed in detail in [56.15]. Only two robotic systems, other than for harvesting or spraying, were found in the literature: a leaf picking robot for cucumber [56.16] and a robot for bagging, thinning, and spraying in grapes [56.17]. In protected cultivation robots have not yet been developed for actions like pruning, disbudding and attaching of plants to support structures.

Challenges in livestock production are of a more practical nature. When cattle roam freely they can be difficult to find and muster. Spotting by helicopter is expensive so there is great potential for the use of unmanned aerial vehicles. The method of collecting feral pigs and native kangaroos is usually by shooting, supplying markets in Germany and Russia respectively. There could in theory be an opportunity for the application of robotics, but it is difficult to see how this could be compatible with safety. It has been suggested that cattle could be located by carrying a transmitter. Radio communication involves considerable distances, so the use of GPS/GNSS collars with transmitting systems would pose interesting power and battery problems.

56.3 Case Studies

In this section we present a variety of case studies where robotics technologies are being applied successfully to problems in the agriculture and forestry sectors. The presentation follows a somewhat loose categorization by application domain, i. e., field crops, orchard crops, protected cultivation, forestry, and livestock. The point here is to group examples that are domain-specific but allow for the discussion of those that straddle across domains. Within each domain, the case studies follow (again, loosely) the progression from sensing to mobility to manipulation.

56.3.1 Optimized Coverage for Arable Farming

Agricultural researchers and practitioners have long-desired to have the capability to follow well-defined traffic lanes with common-width equipment systems to minimize soil compaction effects on plant growth [56.18]. The advent of automation and control technology (e.g., GNSS, automatic steering) eliminated the need for complex processes to define the controlled-traffic lanes by making it possible for agricultural machine systems to follow precise paths spatially and temporally.

With the rapid adoption of automatic guidance systems, automated path planning has great potential to further optimize field operations. Field operations should be done in a manner that minimizes time and travel over field surfaces and is coordinated with specific field operations, machine characteristics, and topographical features of arable lands. To reach this goal, *Jin* and *Tang* [56.19] proposed an optimal coverage path planning (OCPP) algorithm where coverage is represented by a geometric model. To determine the full coverage pattern of a given field by using boustrophedon paths, it is necessary to know whether and how to decompose a field into sub-regions and how to determine the travel direction within each sub-region. The search mechanism is guided by a customized cost function resulting from the analysis of different headland turning types and implemented with a divide-and-conquer strategy. The complexity of the algorithm is $O(n^3 \log(n^3))$ for a field with n edges. In order to reduce the total turning cost, the number of turns needed to be minimized. Besides, turns with relatively high operational costs needed to be avoided. Fields of irregular shapes had inefficiencies related to headland turns when headlands were at an angle to the machine. Two-dimensional field examples with complexity ranging from a sim-

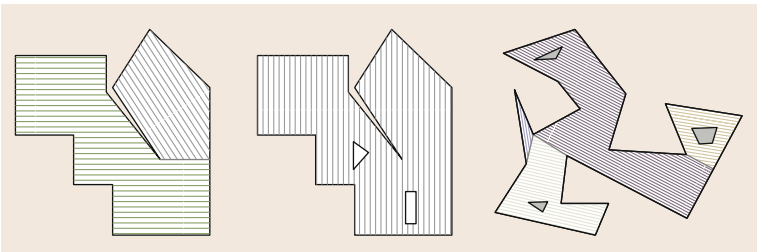


Fig. 56.2 Results obtained with the optimal coverage path planning algorithm for two-dimensional (2-D) terrains; the inner polygons indicate non-traversable obstacles

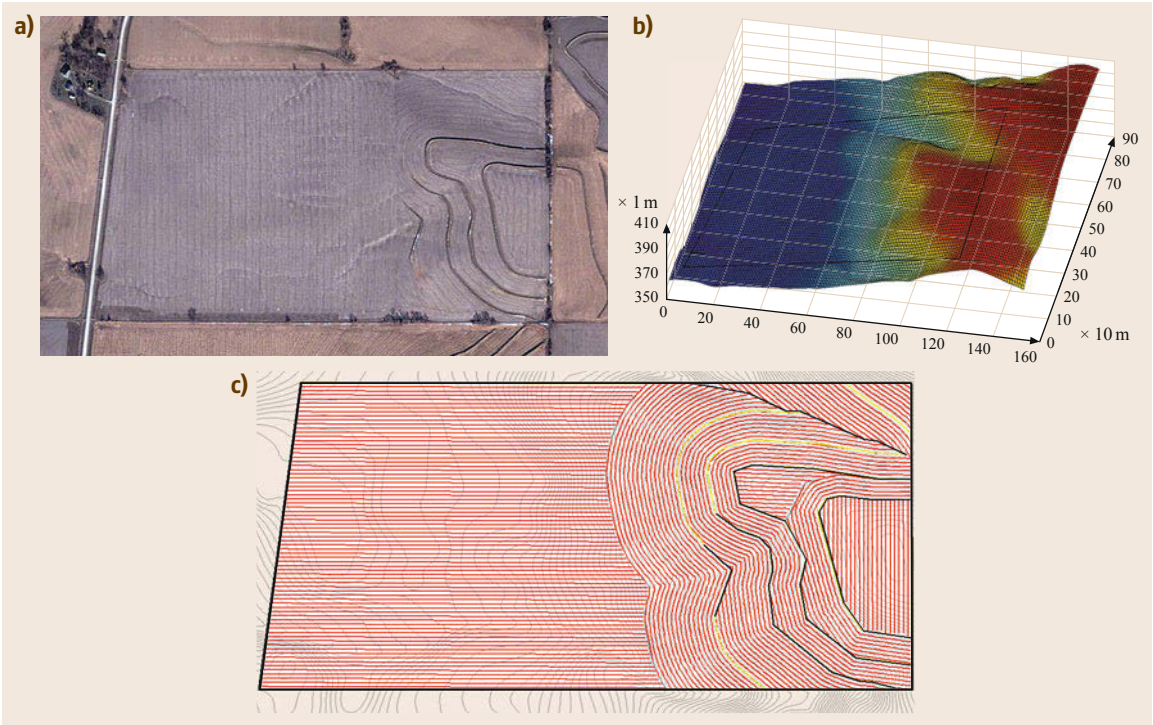


Fig.56.3a–c Results obtained with the optimal coverage path planning algorithm for a 3-D terrain with terraces and valleys. **(a)** Aerial image. **(b)** Topographic map. **(c)** Optimal coverage path

ple convex shape to an irregular polygonal shape that has multiple obstacles within its interior were tested with the OCPP algorithm (Fig. 56.2). The results show that, in the most extreme two-dimensional cases, OCPP saved up to 16% in the number of turns and 15% in headland turning cost. There were no cases where OCPP outputted solutions worse than those adopted by farmers.

When optimizing coverage path over three-dimensional (3-D) terrains, more factors need to be considered, including headland turning cost, soil erosion cost, and skipped area cost. Jin and Tang developed an analytical 3-D terrain model with B-splines; and analyzed different categories of coverage costs on 3-D terrains and developed methods to quantify soil erosion cost and curving path cost corresponding to a particular coverage solution. Similar to the 2-D coverage path optimization, terrain decomposition and classification methods are used to divide a field into sub-regions with similar field attributes and comparatively smooth boundaries. The divide-and-conquer strategy was also applied in the 3-D terrain coverage planning. The most appropriate path direction of each region was the one that achieved the minimum coverage cost. A *seed curve* searching algorithm was successfully developed and applied to several practical farm fields with various topographic features (Fig. 56.3). The 3-D planning algorithm has shown its superiority on 3-D terrain fields compared with the 2-D planner. On the tested fields, on average the 3-D version saved 10.3% on headland turning cost, 24.7% on soil erosion cost, 81.2% on skipped area cost, and 22.0% on the weighted sum of these costs, where their corresponding weights were 1, 1, and 0.5, respectively. In one of the regions, in particular, the 3-D planning algorithm generated a result with only 30.5% of the soil erosion caused by the 2-D planning algorithm. It was also observed that the skipped area resulting from the sharp turning curvature in 3-D planning is generally much smaller than the skipped area between paths when projecting 2-D planning results to a 3-D surface.

56.3.2 Weed Control

Weeds compete with the production crop for light, water, and nutrients and can have a detrimental impact on crop yield if uncontrolled [56.20]. For these reasons chemical and mechanical weed control have long received attention of agricultural engineers. Compared to current methods, robotics offers the opportunity to improve this important production task in two ways: greater precision when done mechanically, and reduced

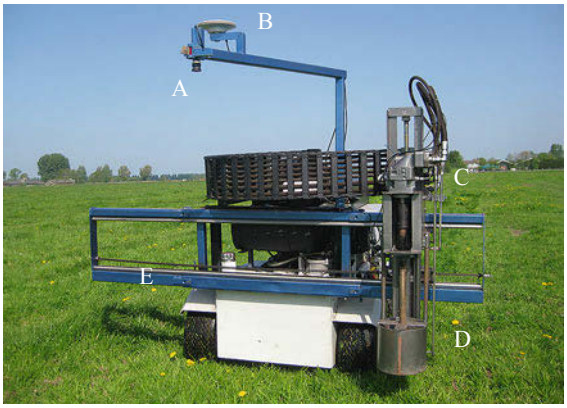


Fig. 56.4 A robot to detect and control broad-leaved dock in grassland. Weeds are detected with a texture-based algorithm using images from a color camera (A). Navigation is GPS-based (B). A diesel engine (C) provides hydraulic power for the weeding implement (D), which can be moved laterally along the rail (E)

emissions and environmental runoff when done chemically. This section describes three examples of the application of robotics to weed control; for an overview of the field see [56.20].

Broad-leaved dock is a troublesome weed in grassland. When not controlled it may reach a high density and considerably reduce grass yield. In response to a request of seventeen organic Dutch dairy farmers, a robot was constructed to detect and control broad-leaved dock in grassland [56.21] (Fig. 56.4) (VIDEO 310). It consists of a custom platform with four independently driven wheels powered by a 36 kW diesel engine. The wheels are fitted with golf cart tires that provide traction on grass with minimal impact on the sod. Reduction gears ensure high torque, allow centimeter-precision forward movement, and limit maximum speed to a safe 3 mph. Skid steering was deemed sufficiently precise under the circumstances. Traversal of a large, featureless pasture was achieved by using real-time kinematic GPS (RTK-GPS) to follow a pre-defined path consisting of parallel segments connected by half-circle turns at the ends. Because weeds and grass are both green, a texture-based image analysis method was used to detect the former [56.22, 23]. The weeder proposed by Riesenhuber was used [56.24]. The method consists of a rod weeder driven vertically into the ground to fragment the weed's taproot.

Volunteer potatoes are the leftovers of the previous year's harvest. After a mild winter these potatoes will sprout and constitute a serious weed. They not only compete with the production crop, but also potentially carry diseases. In The Netherlands, legislation requires that they be removed annually by July 1st. This labor-intensive task naturally called for automation. As a joint effort of industry, policy makers, and scientists, a project to detect and control volunteer potato plants was initiated [56.25–27]. A proof-of-principle machine for sugar beet fields has been built and tested (Fig. 56.5). Machine vision-based detection at 100 mm^2 precision was combined with a micro-sprayer with five needles and a working width of 0.2 m. The accuracy of the system was ± 14 mm in the longitudinal direction and ± 7.5 mm in the transversal direction. The main error source was the variability in micro-sprayer droplet velocity that caused longitudinal errors. Still, 77% of volunteer plants with a size larger than 1200 mm^2 were successfully controlled at velocities up to 0.8 $m\,s^{-1}$. Within the seed line, glyphosate was applied on weed potato plants, accompanied with up to 1.0% unwanted killed sugar beet plants.

The Intelligent Autonomous Weeder (Fig. 56.6) is a four-wheel steer, four-wheel drive platform to be used for autonomous weeding operations in arable farming [56.28–30]. The platform combines dual GPS-

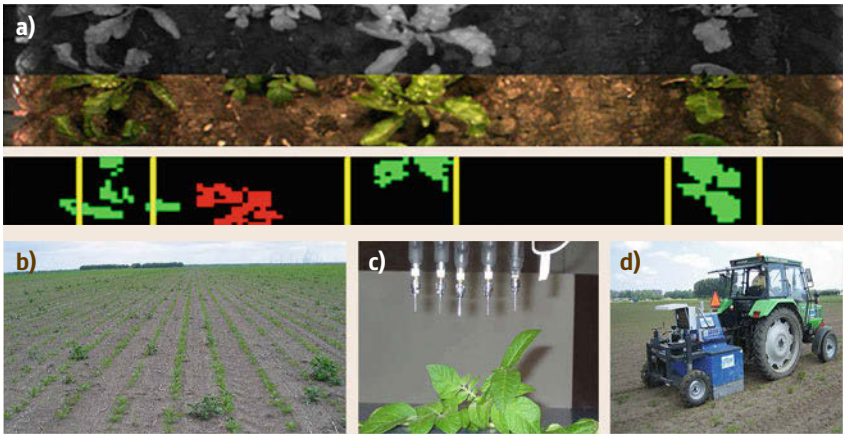


Fig.56.5a–d Precision removal of volunteer potatoes in sugar beets. **(a)** Computer vision-based detection of the potato weeds (*red*) among the sugar beet plants (*green*). **(b)** A sugar beet field infested with potato weeds. **(c)** A micro-spray system. **(d)** The removal device pulled by a tractor

Fig. 56.6 The Intelligent Autonomous Weeder operates in arable farming land

based navigation with computer vision-based row following. The four wheel steering construction offers supreme maneuverability, which is not only an advantage when precise operation within the crop is considered, but also offers the opportunity of very compact head land turns.

56.3.3 High Precision Seeding

The premise behind this work is that, if an autonomous agricultural machine could accurately follow a predefined path to carry out seeding, then the same machine could drive in the field throughout the entire growing season carrying out subsequent tasks – e.g., weeding, fertilizing, spraying, etc. – without the need for repeated crop location sensing (VIDEO 131). The primary challenge is to develop centimeter-level path tracking accuracy along straight runs by the mobile agricultural machines, and even greater accuracy (1–2 cm) in agricultural tool manipulation, e.g., controlling seeding tines for seed placement.

While there are semi-automated seeding systems currently on the market, they suffer from a number of problems with major implications for environmental sustainability, productivity, and economic return:

- They are passive, i. e., unable to take action to correct the seeding tines' path.
- The seeding implements are towed by a tractor which is GPS-guided, but with insufficient accuracy (usually 40 cm); moreover, accurate path following by a tractor does not guarantee accurate seed placement by the implement.
- The tractor does not sense deviation of the seeding implement's path, causing the crop to be laid out unpredictably.
- Even when the system senses implement deviation, it does not have the ability to correct the seeding implement's path, let alone the position of the seeding tines.
- Tractors used in current systems are massive, therefore must use fixed tracks to limit ground compaction (however still causing compaction of up to 20% of arable land).
- Insufficient seeding accuracy does not permit inter-row cropping in alternating seasons – a technique that utilizes remnant nutrients in inter-row spaces, and
- Present day systems do not have fully autonomous operational capability.

Katupitiya et al. [56.31] built and delivered a system that advanced the field in the following ways (Fig. 56.7):

- An active seeder equipped with sophisticated control systems that can take corrective action against path deviations while ensuring high accuracy seeding tine position control.
- High accuracy localization of the seeder and tines achieved via precise GPS, high-precision sensors, data fusion, and control software to locate/position the seeding tines. Control systems include those that micro-adjust the seeding tines with respect to the main seeder frame. This allows the seeding tine positions to follow a more accurate path than the seeder itself.
- The seeder is force-controlled and self-propelled, allowing for the size of the tractor to be greatly reduced. Smaller tractors also mean smaller wheels, so the tractor wheel width can be smaller than the inter-crop row width; hence the tractor can use the inter-row space as its wheel tracks without squashing the crop for subsequent operations.

Fig. 56.7 Precision seeding system for broad acre farming

These result in the complete elimination of ground compaction.

- The level of automation integrated into the tractor-seeder pair is such that the entire system is readily autonomous with no operator required.
- The seeder is modular, with either single or multiple units operating around the clock. A special purpose tandem, non-linear adaptive pursuit path tracking algorithm is used to control the steering of the tractor and the seeder wheels. The seeder has its wheels under force control based on the tension at the off-axle hitch point.

56.3.4 Crop Yield Estimation

One common desire of all fruit growers is knowledge of the crop yield. Accurate yield prediction helps growers improve fruit quality and reduce operating cost by making better decisions on intensity of fruit thinning and size of the harvest labor force. It benefits the packing industry as well, because managers can use estimation results to optimize packing and storage capacity.

Typically, yield estimation is performed based on historical data, weather conditions, and workers manually counting fruit in multiple sampling locations. Manually gathering samples is a time-consuming, labor-intensive, and inaccurate process, and the number of samples is usually far too small to capture the magnitude of the variation in yield across each block. Growers are searching for an automated and efficient alternative that can accurately capture spatial variation in yield.

To deal with this need, *Nuske* et al. [56.6, 7] developed a computer vision-based system to detect and count fruit. The system uses a camera rig for image acquisition, working at nighttime with controlled artificial lighting to reduce the variance of natural illumination. An autonomous vehicle is used as the support platform for automated data collection (Fig. 56.8). The system scans both sides of each row of trees or vines, detecting fruit captured within the image sequence, and then generates yield estimates.

The accuracy of the system was demonstrated by comparing its crop yield estimation with ground-truth recorded via careful and tedious manual measurements. Its end result is a yield map that closely resembles the true spatial distribution of yield, which growers can utilize to make critical production management decisions.

The yield estimation system was deployed in a number of vineyards, apple orchards, and strawberry ranches, and the results show that the system works well in a variety of crops and training structures (Figs. 56.9 and 56.10).

Fig. 56.8 Autonomous crop yield estimation system hardware, composed of concentric cameras and ring flashes for controlled illumination during nighttime operations

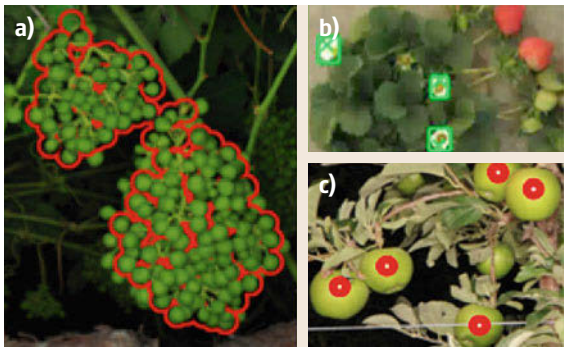


Fig.56.9a–c Output of the crop yield estimation system in vineyards **(a)**, apple orchards **(c)**, and strawberry ranches **(b)**

56.3.5 Precision Irrigation

Current irrigation practice within the agricultural community often dictates over-watering crops as opposed to under-watering. This results in wasted resource, increased leaching of fertilizers, and an increase in crop disease. There are crop models that determine adequate irrigation quantities, but they are seldom used as the constant modifying of irrigation parameters is tedious and difficult.

Kohanbash et al. [56.32] used wireless sensor networks (WSN's) to monitor environmental conditions in real time and adjust irrigation parameters on the fly. WSN's can do basic set-point control where irrigation is enabled every time the soil moisture goes below a predefined threshold. More advanced control methods are also possible by integrating crop water use models into the system. WSN systems communicate with a central base station that can be connected to the Internet, allowing remote access for viewing crop condition and modifying irrigation settings. Having a central base sta-

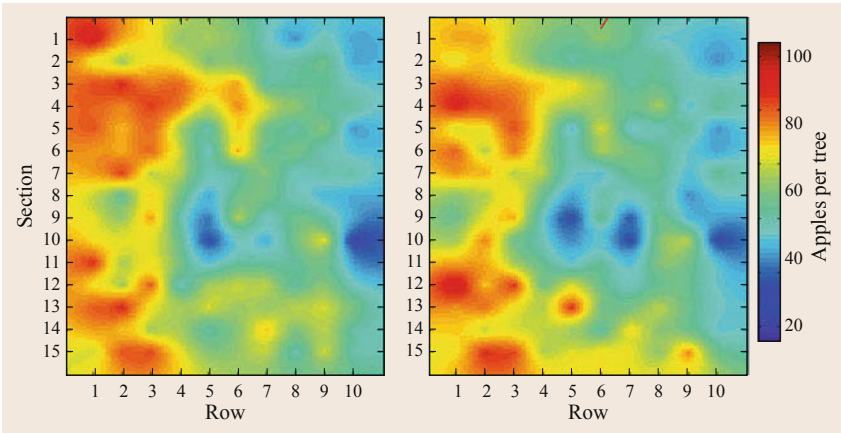


Fig.56.10a,b Spatial yield map collected by the system: **(a)** Truth manual count, **(b)** automated count

Table 56.1 Irrigation improvements at a greenhouse by using a wireless sensor network system (after [56.32])

Condition	Irrigations per day
Before WSN system installed (baseline)	8
With WSN system (monitoring)	4–5
With WSN system (monitoring and control)	2–3

tion also allows growers the ability to monitor and analyze the crop growing environment for trends and long-term changes.

Early results (Tab. 56.1) show water savings of up to 75%, fertilizer leaching reduced to near zero, increased crop quality, speed of crop growth increased by over 50%, and reduced occurrence of crop disease. In addition to these savings, growers can use the data from WSN systems to tailor their crop for specific markets. For example, by adjusting the irrigation set-point value, a grower can choose between more expensive (Grade A) product, or less expensive (Grade B) product that they can sell more of.

56.3.6 Tree Fruit Production

Autonomous orchard vehicles will radically transform tree fruit production by automating maintenance operations such as mowing and spraying and augmenting workers pruning, thinning, and harvesting. From a robotics point of view, these applications can be realized with a relatively simple, yet challenging, capability: driving along one row of trees, turning at the end of the row, and entering the next one. Challenges involved include reliably sensing the trees in the presence of sloped terrain, branches, tall grass, and missing trees; localizing the vehicle in the orchard; following trajectories inside and outside the rows; and avoiding

Table 56.2 Control modes enabled by the family of autonomous orchard vehicles in Fig. 56.11, from least to most complex

	Mule mode	Scaffold mode	Pace mode
How it works	The vehicle drives along rows of trees as workers harvest fruit, placing them in bins on the vehicle	Farm workers stand on the vehicle while it self-steers in the row	The vehicle autonomously drives an entire block at a time without requiring any further interaction
Production tasks enabled	Fruit harvesting	Pruning, fruit and blossom thinning, tree maintenance, harvesting, pheromone dispenser placing	Mowing, spraying, and scouting for disease, insects, and crop yield estimation
Autonomous functionalities	Row following (continuous or *stop-and-go*), end-of-row detection, obstacle detection		Row following, end-of-row detection, turn and enter new row, obstacle detection
Vehicle speed	0.2–2 mph	0.1–0.2 mph	2–5 mph
Interface	Control box with buttons	Control box and foot pedals	Handheld tablet or smartphone
Permanent infrastructure installed in orchard	None	None	None

Fig. 56.11 A family of autonomous orchard vehicles capable of driving among tree rows, based on the Toro Workman series. To keep cost low, the vehicles use only a laser rangefinder and steering and wheel encoder for perception and navigation

Fig. 56.12 **(a)** Autonomous orchard vehicle. **(b)** Workers on foot harvest apples and deposit them on bins towed by the vehicle, eliminating the need for and the cost of a tractor driver. **(c)** In this version the vehicle is equipped with a lift platform from where workers can prune, thin, tie trees, and place pheromone dispensers up to twice as fast as workers performing the same operations on ladders. Users on the platform control the vehicle from an interface **(d)** that communicates with the onboard computer wirelessly

obstacles. Additionally, the added cost of the autonomy components should be as low as possible, to make such vehicles commercially viable.

Figure 56.11 shows a family of autonomous orchard vehicles with a common sensing and computing infrastructure that allows them to cover entire orchard blocks continuously for several hours (VIDEO 91). To keep cost low, they are not equipped with GPS/GNSS or inertial navigation systems; instead, perception and navigation is achieved with a laser rangefinder mounted on the front of the vehicle and steering and wheel encoders. Table 56.2 summarizes the three autonomous control modes in which the vehicles can operate.

From 2008 to 2012 the vehicles drove a combined 350 km in experimental and commercial orchards, vineyards, and nurseries in several US states, including many in the largest apple producer in the United States, Washington state. The longest run covered 25 km over five hours [56.10, 33]. Time trials conducted by The Pennsylvania and Washington State Universities Extension educators showed that workers onboard an autonomous orchard platform can be twice as fast as workers using ladders or on foot when working on the top portion of apple and peach trees (Fig. 56.12) [56.34].

56.3.7 Vehicle Formation Control

Multiple robotic agricultural vehicles driving in formation offer capabilities beyond those possible by a single vehicle. *Lenain* et al. [56.35] developed a control architecture that enables accurate and stable control of several robots in a given and possibly variable formation. The method is based on a path tracking framework, defining the relative robot's positions as a lateral distance with respect to a given path, and the longitudinal distance along this path. Perturbations due to motion on natural environments (e.g., poor grip conditions and uneven terrain topography) are compensated with a nonlinear model-based controller that includes an observer for wheel sideslip. In the formation control point of view the lateral error is no longer regulated to zero but to a desired set-point, potentially variable and accounting for other vehicle deviations. In addition, the velocity of each vehicle is controlled in order to ensure a desired curvilinear distance with respect to the others, and imposing a desired speed for one of them. Similarly to lateral dynamic, the desired distance may be defined as varying, and error may be constituted of a mix between longitudinal errors of one vehicle with respect to the other.

This approach was tested in the field with two electrical off-road robots (Fig. 56.13a). They are equipped with RTK-GPS providing an accuracy of ± 2 cm, and are able to communicate their positions among each other using wireless communications. Weighing around 500 kg each, these two-meter long vehicles are able to reach speeds of $4\,\mathrm{m\,s^{-1}}$ and climb up to 20 degrees terrain. In this example, the first vehicle has to follow the trajectory depicted in black, composed of a straight line along a 15 degree slope on grass at a speed of $2\,\mathrm{m\,s^{-1}}$, followed by a half turn coming back on flat ground. The second vehicle must follow the first one with a desired curvilinear distance of 10 m and a desired lateral

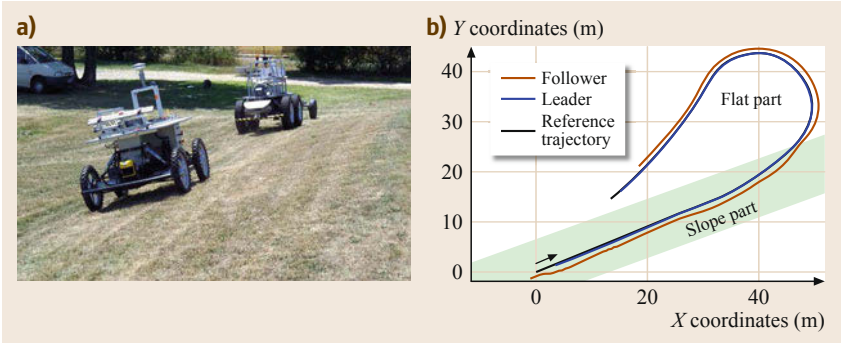


Fig. 56.13 **(a)** Two agricultural vehicles in formation control. **(b)** Reference trajectory and actual path achieved by the two vehicles

Fig. 56.14 Date tree tracking system adapted to a commercial sprayer. The system aims the sprayer nozzle toward the trees more efficiently and safely than a human operator

deviation of 1 m. The positions of the first and second vehicles are shown in blue and red dots in Fig. 56.13b, respectively. In these conditions, the accuracy of relative positioning is within ±15 cm despite the challenging terrain conditions (low grip, uneven terrain, curved trajectory). This work demonstrated the capability of maintaining a formation of several vehicles, compatible with the conditions encountered in agriculture, allowing to consider the introduction of multi-vehicle platoons in the field for precision agriculture.

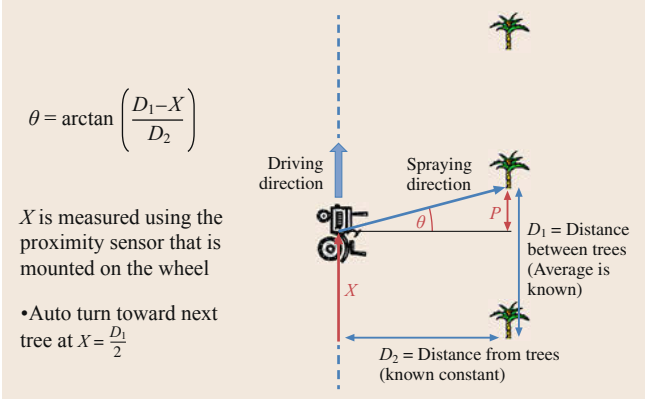


Fig. 56.15 Calculation of the desired spray angle

56.3.8 Date Palm Tree Spraying

Dates palm tree spraying is normally done manually by a team of three workers from a platform 18 m or more above the ground. In the past, many accidents have occurred due to instability of the platform when in a lifted position. Degania Sprayers Company (Israel) developed a sprayer with a tall air cannon and a pan-tilt unit at the end to control the air flow and spray direction. This system, however, requires a worker to manually aim the spray toward the tree. *Shapiro* et al. [56.36] developed an automatic tree tracking system for the sprayer (Fig. 56.14).

The tracking system is based on an ultrasonic range sensor to detect the trees and a proximity sen-

sor mounted on the sprayer's wheel to measure distance traveled. A human driver is responsible for maintaining the sprayer a distance $D_2 = 3.5\,\text{m}$ from the trees. Knowing the average distance between trees in the row ($D_1 = 9\,\text{m}$), it is possible to compute the desired spraying angle as $\theta = \tan^{-1}((D_1 - x)/D_2)$, where x is the distance traveled since the last tree (Fig. 56.15). This value is fed fed into a proportional–integral–derivative (PID) controller that outputs a pulse-width modulation (PWM) signal to an electrically-controlled hydraulic valve that controls the spraying angle. The authors opted to use an on/off valve driven by a PWM signal instead of a proportional valve to reduce the system cost and therefore make it more attractive to farmers. The rotation angle of the sprayer is measured by a potentiometer and used as feedback for the PID controller. When the sprayer reaches the midpoint between two trees, i. e., $x = D_1/2$, the sprayer is set to rotate toward the next tree in the row and start spraying it. The tracking algorithm is implemented on an Arduino microcontroller.

The system was built and deployed in date farms, showing good tracking results. It replaces a human worker which previously had to perform a dull task controlling the sprayer to track the trees using a joystick. A preliminary economic analysis indicates that the additional cost of the sprayer tracking system can be recovered in one harvest season.

56.3.9 Plant Probing

In large botanic experimentation fields, treatments (watering, nutrients, sunlight) that optimize certain desired aspects (growth, appearance) need to be determined. Towards this aim, experiments entailing many repetitive actions need to be conducted. For example, measurements and samples from leaves must be taken regularly and some pruning may need to be performed. For these tasks robots would be very handy, but difficulties arise from the complex structure and deformable nature of plants, which do not only change appearance through growing, but whose leaves also move on a daily and sometimes hourly basis. Even though recent advances in depth sensors, deformable object modeling, and autonomous mobile manipulation have brought this goal in reach for robotic applications [56.37, 38], many problems still exist, in particular concerning robust recognition and localization of plant parts (leaves, flowers, fruits, stems), and robot manipulation under weakly constrained conditions in natural environments.

In this context, the European project GARNICS (gardening with a cognitive system) aims at 3-D sensing of plant growth and building perceptual representations for learning the links to actions of a robot gardener.

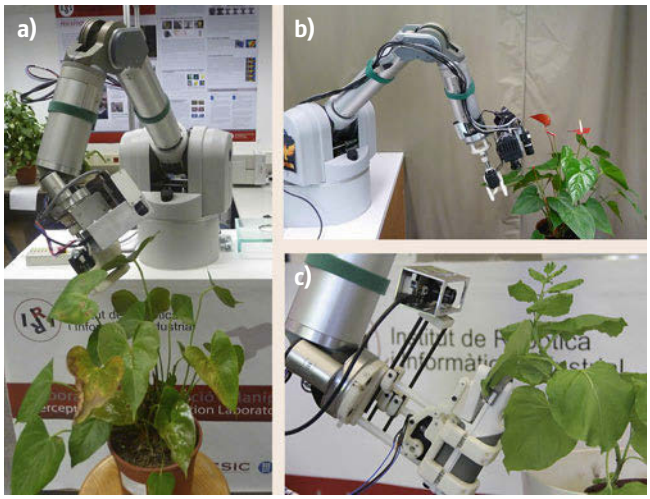


Fig. 56.16 (a) Chlorophyll measurements with a robot-mounted spad meter. (b) Cutting of leaf disks with a robot-mounted, custom-designed cutting tool. Both a color camera and a ToF camera are mounted on the end-effector of the arm. (c) Close-up of probing: Chlorophyll measurements of a tobacco plant. The lightweight time-of-flight camera is seen at the top

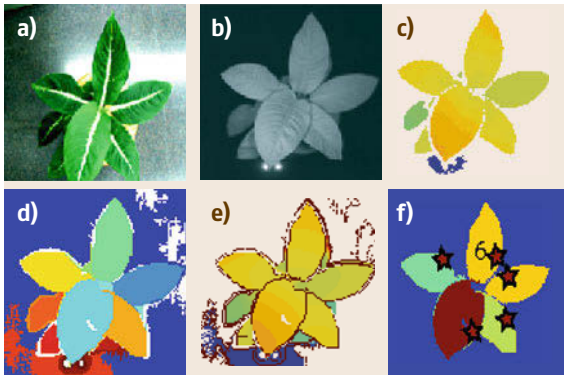


Fig. 56.17 (a) Color image. (b) Infrared image. (c) ToF depth. (d) Segmentation. (e) Surface models. (f) Color-coded leaf confidences and extracted probing points

Plant sensing and control is addressed by combining active vision with appropriate perceptual representations, which are essential for cognitive interactions (VIDEO 95).

An application of robotized phenotyping related to this project is the accurate placing of a measurement tool on a leaf in order to either cut sample disks from the leaf, or to measure chlorophyll content. The robotic arm is equipped with a time-of-flight (TOF) camera and a measurement tool (Fig. 56.16) [56.39–41]. In this approach, image segmentation and model fitting are employed to recognize and localize single leaves from depth information. 3-D data is combined with color or

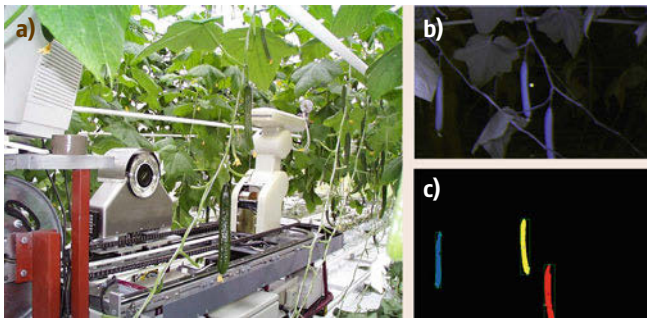


Fig.56.18a–c A cucumber harvesting robot. **(a)** The robot in the greenhouse. **(b)** Unprocessed image of cucumber crop in the near infrared spectrum. **(c)** Detected cucumbers

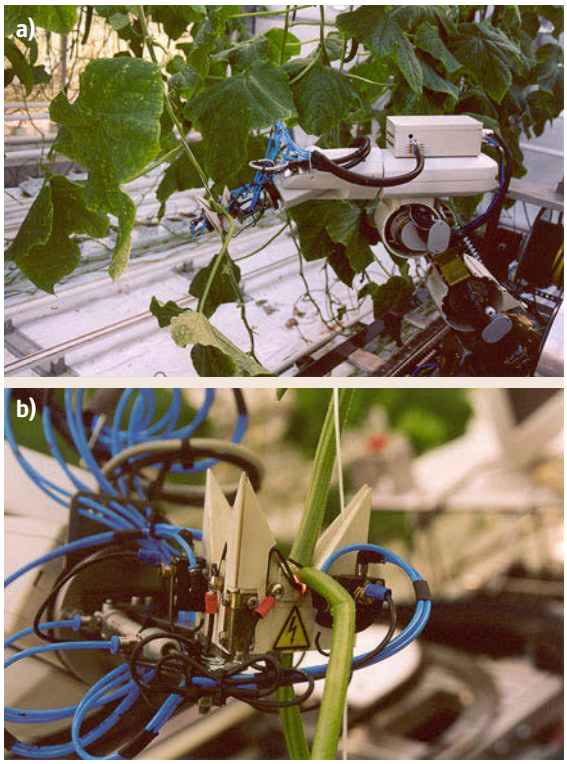


Fig.56.19a,b Cucumber leaf removal robot. **(a)** Robot operational in the crop. **(b)** End-effector (after [56.16])

infrared images and used to segment the data into surface patches, which are assumed to correspond to actual plant leaves [56.42].

In this approach, a next-best view strategy was proposed for finding a non-obstructed and frontal view of the leaf [56.41]. Initially, the robot arm is moved to a position from which a general view of the plant is obtained. The depth and infrared images acquired from this position are segmented into their composite surfaces, leaf model contours are fitted to the extracted segments, and the validity of the fit and the graspability of the leaf are measured (Fig. 56.17). A target leaf is selected and the robot moves the camera to a closer, frontal view. The validity of the target and the graspability are re-evaluated. If the leaf is considered to be suitable for sampling based on these criteria, the probing tool is placed onto the leaf following a two-step path. If the target is not considered suitable for probing, another target leaf (from the general view) is selected and the procedure is repeated.

The method is based on several assumptions:

i) The boundaries of leaves are visible in the infrared-intensity image
ii) The leaf surfaces can be modeled by a basic quadratic function
iii) Leaves of a specific plant type can be described by a common 2-D contour
iv) Leaves are large enough to allow analysis by a ToF camera, and
v) The leaves are static during probing.

These assumptions may be violated under certain conditions, and further research will have to be undertaken to solve the various problems originating mostly from the complex and deformable nature of plants.

56.3.10 Cucumber Harvesting

A prototype robotic harvester for cucumbers is presented in Fig. 56.18. The machine consists of a mobile platform that runs on rails (VIDEO 308). These rails are commonly used in greenhouses in The Netherlands for the purpose of internal transport, but they are also used as a hot water heating system of the greenhouse. Harvesting requires functional steps such as the detection and localization of the fruit and assessment of its ripeness. In case of the cucumber harvester, the different reflection properties in the near infra-red spectrum were exploited to detect green cucumbers in the green environment [56.43–45]. Whether the cucumber was ready for harvest was identified based on an estimation of its weight. Since cucumbers consist almost 95% of water, the weight estimation was achieved by estimating the volume of the fruits. Stereovision principles were then used to locate the fruits to be harvested in the 3-D environment. For that purpose the camera was shifted 50 mm on a linear slide and two images of the same scene were taken and processed. A Mitsubishi RV-E2 manipulator was used to steer the gripper-cutter mechanism to the fruit and transport the harvested fruit back to a storage crate. Collision-free motion planning based on the A* algorithm was used to steer the manipulator during the harvesting operation [56.44]. The cutter consisted of a suction cup on a parallel gripper

that grabbed the peduncle of the fruit. (The peduncle is a stem segment that connects the fruit to the main stem of the plant.) Then the action of a suction cup immobilized the fruit in the gripper. A special thermal cutting device was used to separate the fruit from the plant. The high temperature of the cutting device also prevented the potential transport of viruses from one plant to the other during the harvesting process. For a successful harvest this machine needed 65.2 s on average. The success rate was 74.4% [56.43].

56.3.11 Cucumber Leaf Removal

Harvesting has received considerable attention in robotics research focused on protected cultivation. Yet, this is not the only time- and labor-consuming cultivation operation. In cucumber production, amongst others, removal of old non-productive leaves in the lower regions of the plant is a time-consuming task. Figure 56.19 shows the same platform used for harvesting, but in this case hardware and software were used for removal of leaves from the plants [56.16] (VIDEO 309). In this system, the camera system is used to identify and locate the main stem of the plants. The gripper is sent to the plant and moved upwards. Leaves encountered during this upward motion are separated from the plant using the similar thermal cutting device as used for harvesting. An interesting feature of this machine is that with slight modifications of software and hardware, two greenhouse operations could be performed.

56.3.12 Rose Harvesting

In recent years a harvesting robot for roses has been developed and tested under practical circumstances in The Netherlands. The prototype is shown in Fig. 56.20. In this case, the rose plants are grown on moveable benches. Thus plants move to the robot instead of the robot moving to the plants in the greenhouse. During the harvest cycle, a camera system travels over the rose plants and locates the roses to be harvested. Then the harvesting operation is performed with two manipulators. One manipulator grips the rose just below the flower and pulls it gently aside to generate space for the second manipulator to travel down the stem towards the point where it will cut the stem. This manipulator carries a small-sized stereovision system that is used for real-time tracking of the stem during this downwards motion. Upon arrival, the manipulator deploys a small scissor-type cutter that will cut the stem. Then, finally, the rose is pulled out and put in a storage by the first manipulator whilst the second manipulator moves out of the crop and proceeds with the next harvest cycle. Details of this system have been covered in [56.13].

56.3.13 Strawberry Harvesting

In Japan, the market of strawberries is large, as large as the market of tomatoes, cucumbers, and mandarin oranges. The potentially high-economic return of this product together with the high labor intensity of processes like harvesting, explains the long tradition of research on robotic strawberry harvesting. As shown in Fig. 56.21, the robot consists of a 4 degrees-of-freedom (DOF) cylindrical manipulator. The robot carries 3 charge-coupled device (CCD) cameras. A square LED-array is used for illumination of the scene. Two cameras provide stereo vision for detection and localization of the fruits. Once a fruit is detected, the end-effector is positioned in front of the fruit. The third camera, mounted on the end-effector, is then used to detect the peduncle, i. e., the fruit stem, and calculate its inclination. Based on this data, the orientation of the end-effector is modified with a tilt mechanism and it then approaches the fruit. A successful approach of the strawberry is detected by a reflection-type light sensor in the end-effector. Upon successful completion of this motion, the peduncle is grasped and the stem is cut with a scissor-type cutting mechanism. A suitable manipulator motion then sends the harvested strawberry to a tray. This procedure is repeated for all detected fruits at the

Fig.56.20a,b A rose harvesting robot. **(a)** The robot with the rose crop on a mobile growing system. **(b)** The end-effector with stereo vision system and a scissors-type cutting device (courtesy Jentjens Machinetechniek BV, Veghel, The Netherlands)

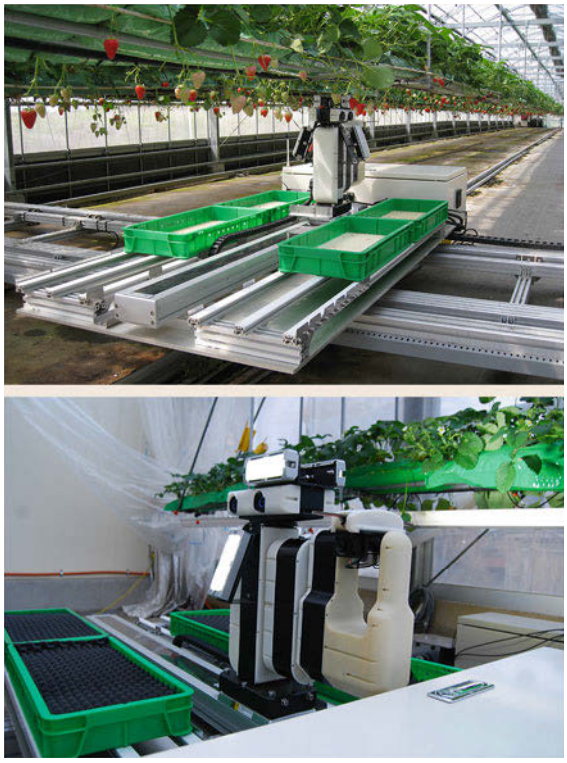


Fig. 56.21 A strawberry harvesting robot (courtesy IAM-Brain, Japan)

current position of the robot. After all picking attempts are completed the full robot platform is moved 210 mm with a gantry-type transportation system running below the strawberry benches. The current prototype has achieved a picking speed of 6.3 s with a success rate of 52.6%. For more details refer to [56.46, 47].

56.3.14 Pot Handling in Nurseries and Greenhouses

Nursery and greenhouse (N&G) farms in the United States produce over two billion potted plants annually (Fig. 56.22a). During the course of production plants are moved several times – distributed onto indoor or outdoor growing beds, repositioned to recover space as orders are filled, and collected for bulk transport. Until recently, only scarce manual labor was suitable for these jobs (Fig. 56.22b).

Harvest Automation's recently-marketed HV-100 (*Harvey*) robot automates critical N&G tasks (Fig. 56.23a). Plants are lifted and transported using a one degree-of-freedom manipulator coupled with a one degree-of-freedom gripper. The mobility system employs two differentially-controlled drive wheels balanced by a front roller.

Harvey uses a laser rangefinder to identify the containers in which plants are grown. This sensor has a horizontal field-of-view greater than 180 degrees and can detect poorly reflecting plant containers from at least 4 m – even under bright sunlight. The robot also uses the laser rangefinder to detect obstacles and robot teammates.

Four *boundary sensors*, two forward-pointing, two rear-pointing, are used to find and follow retroreflective tape that marks the edge of the bed. The tape marker performs double duty as both the robot's global reference and as part of the user interface. By positioning the boundary marker workers indicate to the robot where plants should be placed.

A user interface consisting of a dial and buttons is located on the back of the electronics box. The interface enables users to input the desired plant spacing, bed width, spacing pattern (hexagonal or rectangular) and the number of aisles the robot should instantiate.

Robust operation of the robot is enabled by a behavior-based programming scheme. Figure 56.23b illustrates *spacing*, a task that gives each plant sufficient room to grow without interfering with its neighbors. The closely packed plants in the foreground have been delivered to the growing bed and placed on the ground. Robot A identifies plant containers using the rangefinder; it selects for pick up the container farthest down field. After closing on and capturing a container with its gripper the robot will turn toward the boundary B. Nearing the tape marker the two forward boundary sensors on the robot will detect the tape and compute the relative angle between robot and tape. This enables

Fig.56.22a,b Potted plants destined for garden stores and landscapers are grown in containers on immense nursery farms. Large amounts of stoop labor is needed to place, maintain, and retrieve these plants

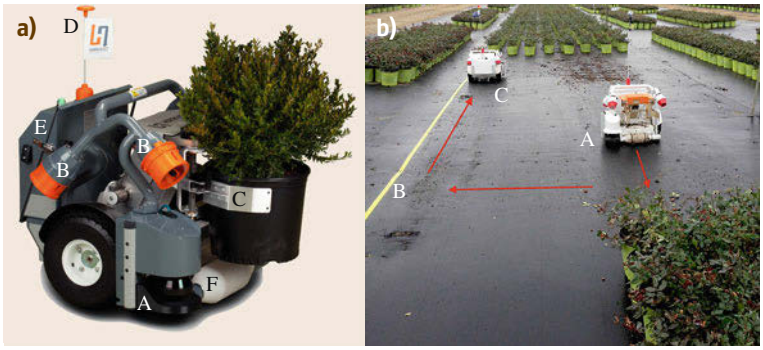


Fig. 56.23 **(a)** HV-100 (*Harvey*) nursery and greenhouse automation robot with main components labeled: (A) laser rangefinder, (B) boundary sensor (two of four), (C) gripper, (D) emergency stop flag (pulling the flag stops the robot), (E) electronics box, with user interface components mounted on the back, and (F) passive roller to balance the robot. **(b)** Two robots operating as a team perform the spacing task. Robot A approaches the source plants in the foreground. Boundary B marks the edge of the bed. Robot C carries a plant toward the destination

the robot to turn and align with the tape, keeping the tape on its left side.

After acquiring the boundary marker the robot will servo along the tape as robot C is doing. When the plant destination comes within view the laser rangefinder is used to identify the next empty space in the pattern of *spaced* plants. The robot then computes an efficient path to the put-down point, moves to that position, and drops off the plant. Afterward, it turns back toward the source of plants and repeats the process. The robot uses dead reckoning to find its way to the vicinity of the source plants. Following this strategy robots can work singly or in teams of various sizes.

56.3.15 Precision Forestry

About 30% of the land mass of the earth is covered by forests. In addition to providing the raw materials for furniture, paper, clothing, and heating, forests provide habitats to diverse animal species and form the source of livelihood for many different human settlements. In 2012, in Germany alone, an estimated 1.3 million jobs in the wood processing industry generated a revenue of more than 180 billion euros. To explore forests in an environmentally-friendly but still economical way is thus a major issue. Robotics is currently being used to preserve the forest and secure the jobs in forestry and related industries.

Today, work in forests is already highly mechanized, and in the last decade, mobile robotics know-how combined with new virtual reality and remote sensing techniques paved the way for a new *robotics view* onto work machines in the forest. Wood harvesters (Fig. 56.24) and forwarders, advanced work machines for log cutting and transport, are currently a major aim of automation efforts [56.45, 48, 49]. Based on the insight that accurate machine localization cannot solely be based on GPS, mobile robotics capabilities for localization and navigation have been introduced. Measured GPS errors of up to 50 m due to signal absorption in the canopy and to multi-path effects make GPS practically useless for precise localization and navigation. In

Fig. 56.24 **(a)** A simulated wood harvester equipped with simulated laser scanners in a virtual testbed used for algorithm development and evaluation. **(b)** The physical harvester in the woods, equipped with laser scanners mounted to the right and left door of the cabin

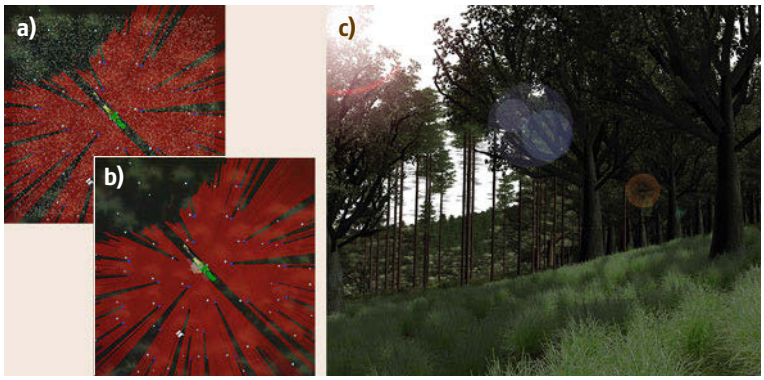


Fig. 56.25 **(a)** Top-down view of the machine, particles (*gray dots*), trees from global tree map (*light blue dots*) and trees from local tree map (*dark blue dots*). **(b)** Particles accumulate near the left edge of the cabin of the forest machine and indicate the newly calculated position in comparison to the GPS-based position (*white rectangle* below the harvester). **(c)** A semantic world model turned into a *virtual forest.*

the *VisualGPS* approach, the GPS position only serves as a starting point for a combined Kalman filter and Monte Carlo localization algorithm based on optical range measurements by laser scanners (VIDEO 96). The approach yields a machine position with an accuracy of 0.5 m and thus provides a sound basis for the development of navigation and (semi-) autonomous logging procedures [56.50, 51].

Practical experiments show that map building based on simultaneous localization and mapping (SLAM) techniques [56.52] are not very applicable in these environments, because the resulting errors in the map are not bounded and thus are not well-suited for large area operation and for a matching against parcel borders. Instead, the highly accurate position estimation from *VisualGPS* builds on a previously generated map of single trees. A multi-sensor fusion approach helps to build this map based on airborne and satellite imagery in multiple spectral ranges as well as on airborne laser scanning. The multispectral imagery data provides the basis for the tree species determination which is solved as an advanced pattern classification problem [56.53]. The next step, single tree delineation, is based on airborne imagery in combination with 3-D surface data from the laser scans. A modified watershed algorithm [56.54] delineates the tree crowns. From the tree crowns' sizes and species information, the geo-referenced stem position and even the trees' diameters can be deduced. This not only allows for generating a global, geo-referenced tree map, but is also sufficient for generating a semantic world model, the *virtual forest* (Fig. 56.25).

Each work machine in the forest also builds local maps of visible trees using machine-mounted laser scanners and a compass. For localization purposes, these local maps are matched against the previously generated global map of trees by means of a particle filter. For a moving machine, the prediction step is carried out by a Kalman filter [56.50, 51].

Figure 56.25 shows the two important results of the global map building process. The figures on the left show just the tree map with the white *noise* in the upper image denoting the particles representing potential poses for the Monte Carlo localization process. In the lower image, the particles have converged against the machine's correct position. The right image gives an impression of the *virtual forest* generated from the same data that was used for the tree map. This *virtual forest* is used to visualize the deduced information (e.g., tree species, tree height, crown shape and size, etc.) in a high-end virtual reality representation. This approach follows a general trend in robotic applications for natural or space environments: the development of virtual testbeds for world model information visualization in an intuitively comprehensible manner. Advanced virtual testbeds then provide the simulation capabilities to develop and test the robotic application against the virtual world in order to save time and costs [56.55].

The developed laser rangefinder-based *VisualGPS* algorithms are being enhanced by optical stereo image recognition capabilities and are being ported to mobile platforms like the Seekur Jr. This is due to the fact that the work that started as an automation effort for work machines in the forest has recently turned into an environment monitoring project which aids the forester to inspect, protect, and attend to the forest with increased efficiency and effectiveness.

56.3.16 Semi-Automation of Forwarder Crane

In cut-to-length harvesting, which is the predominant tree harvesting method in Europe, delimbed and bucked timber is collected from the forest to the roadside by a forwarder. For an operator of such a machine, a large part of the work cycle consists of maneuvering the onboard hydraulic crane (Fig. 56.26). The operator controls the links individually using two joysticks. The redundant kinematic design is necessary for dexterity and the large active workspace of the machine, but it

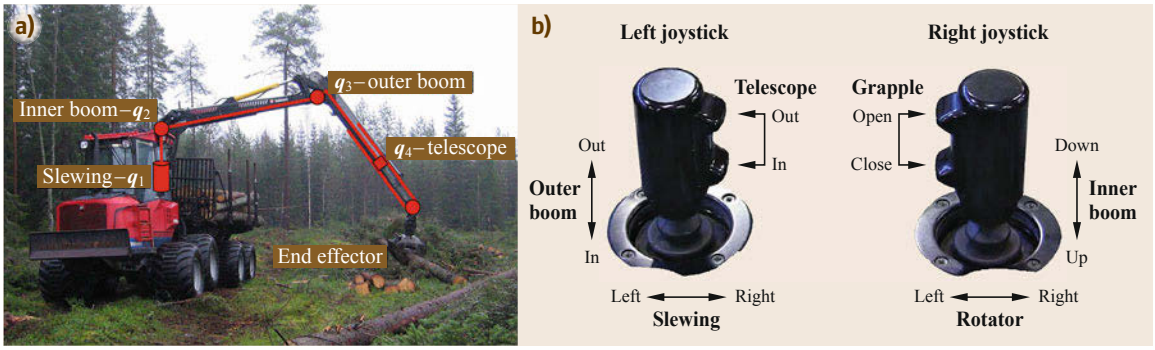


Fig. 56.26 **(a)** Valmet 830 forwarder with redundant kinematic design of the hydraulic crane. **(b)** The operator uses two joysticks to control each link individually

also makes the crane control difficult to learn and to perform in an efficient way.

Considering the technological progress and capabilities of today's forestry machines, the current manual control approach creates bottlenecks in the forwarding process. Semi-automation of these operations would thus be beneficial for productivity. Automation of some repetitive motions is also desired in order to support the operator and to reduce the physical and cognitive workload.

To this end, a small-scale hydraulic forwarder crane has been installed in a laboratory environment [56.56]. The crane is equipped with position and pressure sensors, as well as with electronics and software for rapid prototyping of automated control strategies. The same equipment has been installed on a commercial Valmet 830 forwarder by Komatsu Forest for field testing.

Using these platforms, new feedback control methods and trajectory planning procedures have been developed to construct and implement time-efficient motions [56.57]. For a given geometric path, the speed and relative use of the different crane links along this path can be shaped to achieve motions with optimal performance within the limitations of the machine. Velocity constraints for the individual joints are particularly restrictive in hydraulic manipulators.

In [56.58], trajectories planned with this approach are compared with recorded motions by professional human operators with respect to execution time. Results show that the performance can be improved significantly by path-constrained replanning of human-operated motions (Fig. 56.27). Additional replanning of the geometrical path, along with efficient velocity profiles along the path, can further improve the time efficiency of the crane motions.

Tracking the trajectories with feedback control requires sensors for measuring the joint positions. Installation of such sensors may not be desired due to high costs or durability problems in the rough outdoor environment. Using the approach in [56.59], motions can be found that are robust to uncertainties in initial conditions and therefore possible to perform in open loop. The end effector position along the trajectory can also be estimated using an observer and signals available from pressure sensors. Such design can be a more convenient solution in the harsh conditions of the forest.

The introduction of new tasks or new ways to perform tasks means new challenges for human-machine interaction (HMI). Semi-autonomous operation requires the automated parts to be well integrated with the manual work. In order to facilitate cooperation in manipulator control between the human and computer, the work distribution and work transitions between them is important. *Hansson* and *Servin* [56.60] describes an implementation of shared and traded control for this setup. The results from user tests show that semi-autonomous operation reduces the workload and shows significant potential to improve the productivity of inexperienced operators.

A reliable framework of sensors and low-level control, along with efficient motion planning strategies, allows for development of more advanced interaction technologies. One such possible future scenario is teleoperation, that presents several advantages to both machine owners and operators. Firstly, it opens up for redesign of the machine with removal of the cabin which saves weight and cost. Secondly, the working environment is improved with reduction of the noise and vibration levels. A virtual environment-based system for teleoperation of a forwarder crane was demonstrated in [56.61]; follow-up work is investigating how a virtual environment can be a useful tool for presenting feedback to the operator in different scenarios. This method does however require environment reconstruction to locate logs and potential obstacles, both of which are considered in [56.62].

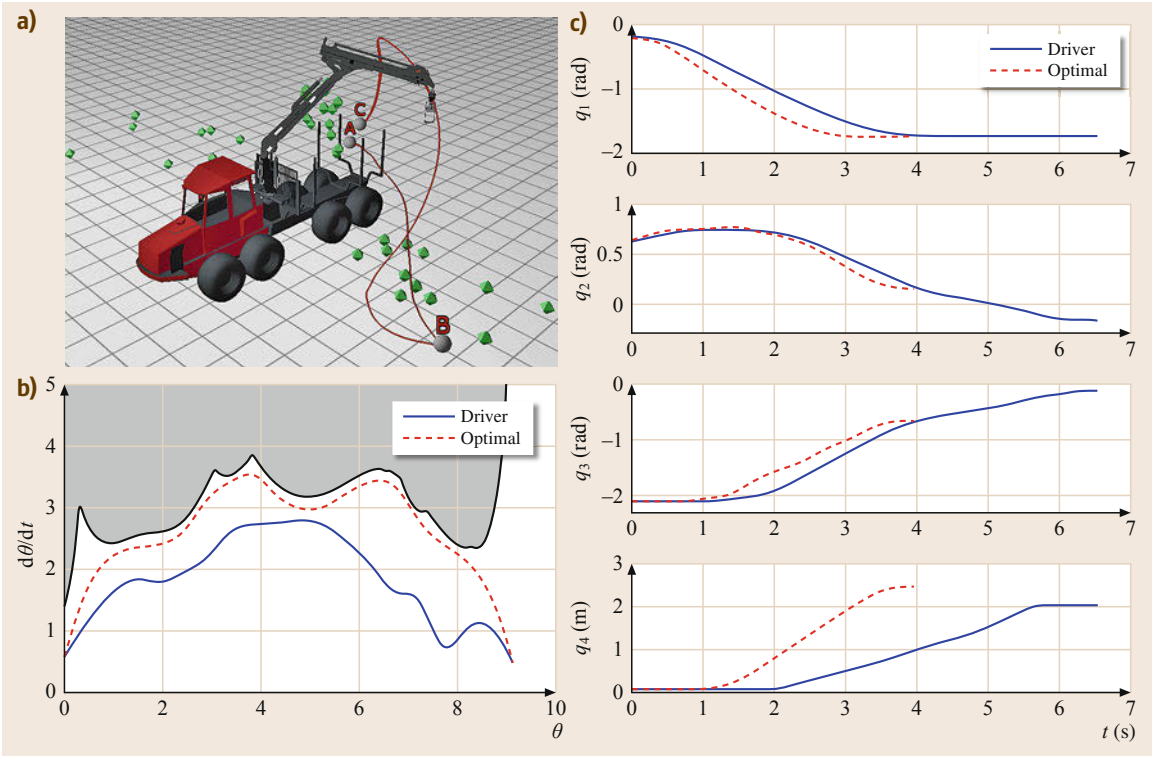


Fig.56.27a–c Path-constrained replanning of human-operated motions. **(a)** Example of two crane motions performed by a human operator: from the load bunk to the logs (A–B) and back (B–C). **(b)** Velocity profiles for the distance θ along the path A–B. A different distribution of the work between the joints allows for a higher velocity along the same path. The *gray area* indicates velocity constraints not to be violated. **(c)** The resulting evolution of joint coordinates with respect to time. The motion with a time-efficient velocity profile is clearly faster than the one of the driver

56.3.17 Livestock Breeding and Nurturing

In some cases animal breeding is a proactive process, such as in hatcheries where optimum conditions must be managed. In others nature is allowed to take its course. Sometimes intervention may be needed in the birthing process, but in the Australian outback cattle must take their chances. Nevertheless the newborn must be marked with transponder tags under the national livestock identification scheme, NLIS [56.63].

Other species such as kangaroos, feral pigs, wild camels and horses can breed and run wild until harvested or culled. When creatures are confined, tasks

Fig. 56.28 Lely Juno robot

Fig. 56.29 Lely Vector feeder

include environmental monitoring and control, feeding, cleaning and growth monitoring.

A Dutch company, Lely [56.64], is active in marketing products that have a strong robotic element. When cattle are located in stalls, the Juno [56.65] patrols the laneway at the side of the pen, pushing fodder towards the rails to put it in reach of the feeding cattle (Fig. 56.28).

The Vector [56.66] goes even further. A mobile robot carrying a hopper, it navigates automatically between the barn that holds the feed and the stalls where the cattle are waiting (Fig. 56.29).

Other equipment in Australian feedlots estimates the weight gain of the cattle, such as a system that reads the NLIS tag of a beast at a water trough and records the weight imposed by the front legs.

In the Australian outback, similar sensing systems perform a variety of functions. Water is scarce, so watering points can be fenced, giving access only via a laneway that can be monitored. Some earlier projects [56.67] have involved species identification to control access by means of an automatically operated gate, but similar technology has valuable use in cattle production.

Again the NLIS tag is read, while a *walk over* weighing system notes the weight gain of the animal. Vision can be used to identify cows that are followed by a calf. Now automatic gates can direct these to a separate paddock for tagging the calves.

Many of the opportunities are for automation, which may be a precursor for robotics. Several projects have involved machine-vision monitoring of cattle [56.68] or of pigs in piggeries [56.69]. The weight-gain is estimated by visual means using precision cameras (Fig. 56.30). Other analyses concern behavior [56.70].

Mobile robots can be involved in sheds where chickens are bred for meat, rather than laying. A mobile robot has been proposed for moving among the poultry to monitor air quality including temperature, relative humidity and the concentrations of ammonia and dust.

Fig. 56.30 Measuring the live weight of cattle with vision

Australian honeybees must also be protected from pests, carried by bees on arriving ships. Bait boxes are deployed at ports around Australia to attract any such swarms. A remote monitoring system with camera sensing gives an early alert to the apiary officer [56.71].

56.3.18 Livestock Exploitation

Traditional farming practices milk cows twice a day; in morning and in the evening. Automatic milking parlors with robotic milking stations are being widely adopted to improve diary productivity and convenience (Fig. 56.31). These allow the cows to self-determine when to come in for milking and feeding. In an automated milking station, teats are located and attachment

Fig. 56.31 Robot milking

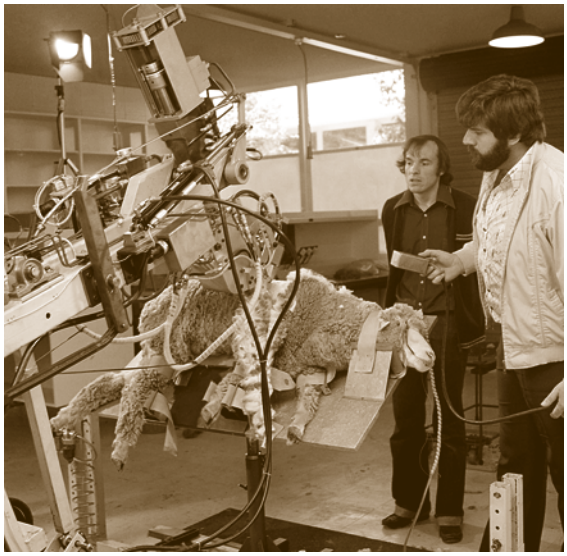


Fig. 56.32 Engineers demonstrate how their pioneering sheep shearing robot works at the University of Western Australia in 1980 (courtesy National Archives of Australia: A6180, 23/9/80/1)

of milking systems is automatic. Yield is monitored and udders are automatically inspected for injury and disease. Human intervention is minimal. The Lely company is active in this area [56.72].

Once a year sheep must be shorn of their wool. In the past, the University of Western Australia developed an automatic shearing system known as the Shear Magic [56.73] (Fig. 56.32). The commercial exploitation was not a success and there seems to have been little activity in this area in recent years.

Fig. 56.33 Prototype computer vision system for scanning the egg collection belt for potential blockages

Fig. 56.34 Precision Pastoral's mustering method

Eggs must be collected from battery hens. As they are laid, they roll from the cages onto a conveyer belt. Among other robotic applications in egg production, machine vision is used to detect foreign bodies or damaged eggs [56.74] (Fig. 56.33).

56.3.19 Livestock Harvesting, Slaughtering, and Processing

The system already described for monitoring cattle that approach a waterhole is also the subject of a funded project for mustering. Cattle, selected by means of their NLIS tags, are diverted into a compound from which they can be collected for transport to a feedlot, in preparation for slaughter [56.75] (Fig. 56.34).

The vision-based system that can control access to waterholes can also be used for the collection of feral animals such as pigs. There are said to be more wild camels in Australia than in Arabia and these too can be captured by such a system.

At the start of the 1990s, Fututech was hailed as the future of slaughterhouses [56.76]. Robotic systems to be installed at the Kilcoy pastoral company in Queensland would automate the whole process from the *knocking box* to the chillroom. Construction started in 1992, but by June 1994 the project was abandoned, at a cost of over $40 million [56.77]. It was a project that was ahead of its time, relying on a central computer and kilometers of cabling rather than distributed intelligence.

But much is different now. Robots have become commonplace in abattoirs, supported by sophisticated sensing systems to locate skeletal features [56.78] (Fig. 56.35). Pigs probably outnumber cattle and the processed animals include sheep and goats. Deboning systems are marketed for products including chicken legs and ham bones [56.79, 80].

For slaughtering quantity, the lead must be held by poultry, where a single plant can process up to 4000 birds per hour [56.81].

Fig. 56.35 Lamb processing

56.3.20 Aerial-Based Precision Agriculture

Unmanned aerial vehicles (UAVs) have recently begun being applied to precision agriculture. This is a new development and results are still preliminary, albeit very promising (VIDEO 307). In fact, the Association for Unmanned Vehicle Systems International forecasts that 80% of all UAVs sold in the United States in the 2015–2025 period will go to serve the agricultural market [56.82]. Even if the actual figure turns out not to be that high, there is no question that roboticists and agricultural engineers are investing significant time and resources to understanding how UAVs can improve agricultural efficiency and reduce costs.

Dong et al. [56.83] use high-resolution images and on-board sensor data from an aerial vehicle (Fig. 56.36) to create a sequence of dense 3-D reconstructions of the crop over time. This four-dimensional (4-D) spatio-temporal reconstruction is used to segment the canopies and estimate how the crown radius and height of each plant evolve. The processing pipeline was tested on data collected in a field planted with corn, broccoli, and cabbage in Tifton, GA, USA (Fig. 56.37).

Fig. 56.36 AscTec Pelican quadrotor used for data collection over early season sweet corn

In many potato growing areas, the potato crop does not senesce naturally and therefore tuber maturation is artificially induced by killing the haulm 10 to 25 days prior to harvest. Reglone is a widely-used potato haulm killing herbicide. Building on the knowledge of the relationship between potato crop biomass status, expressed in weighted difference vegetation index (WDVI) [56.84], and the minimum effective

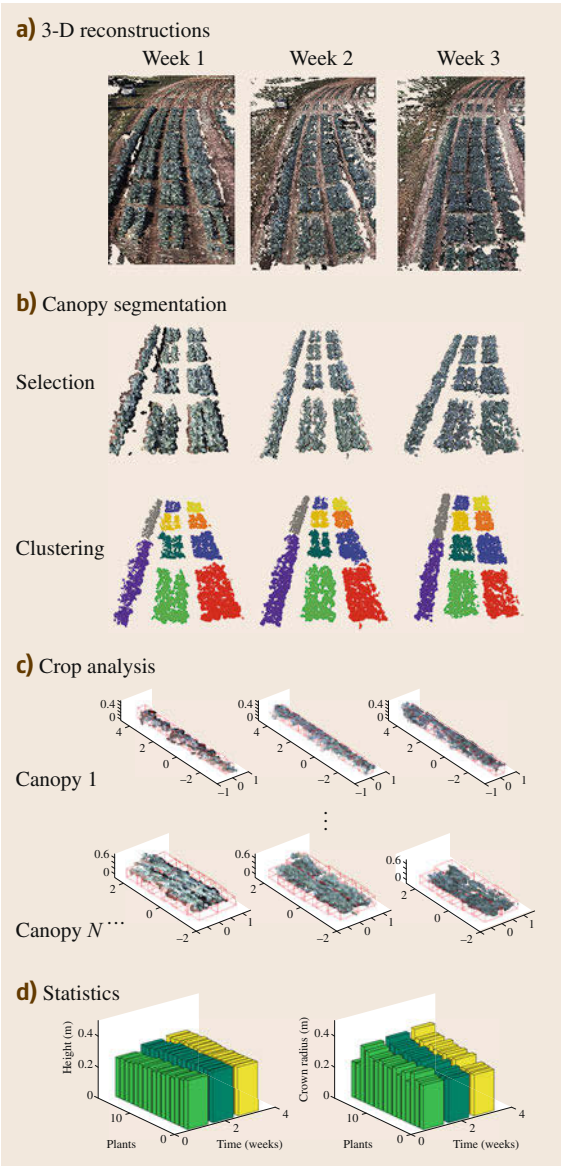


Fig.56.37a–d Dense 3-D reconstruction of the corn stalks, used to estimate how the crown radius and the height of each plant evolve over time. **(a)** 3-D reconstruction of crop, **(b)** canopy segmentation, **(c)** crop analysis and **(d)** crop statistics

Fig. 56.38 Operator launching an unmanned aircraft over a potato field via a launch rope. The plane then flies autonomously following a pre-programmed flight path

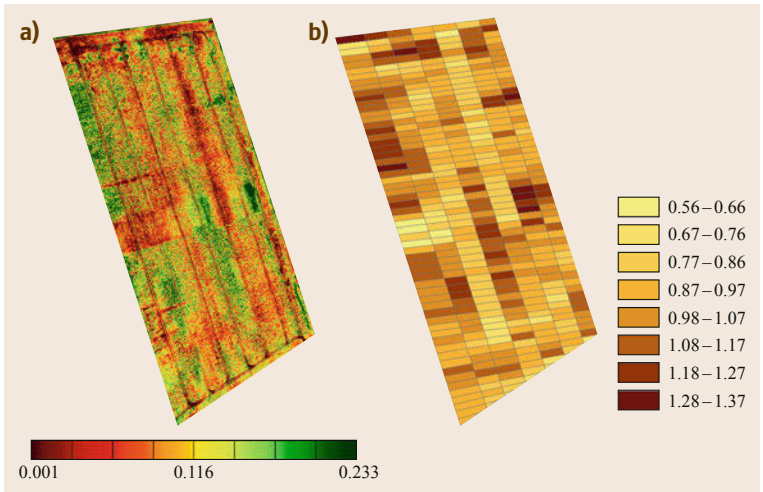


Fig. 56.39 **(a)** Weighted difference vegetation index (WDVI) map of a potato field; the legend on the bottom represents the WDVI, which varies from zero (no crop at all) to one (theoretical maximum). A vigorous potato crop has a value in the 0.6–0.7 range. **(b)** Reglone dose task map; the legend represents dose in liters per hectare

dose of Reglone, *Kempenaar* et al. [56.85] successfully demonstrated the use of crop biomass imaging with a multi-spectral camera underneath an unmanned aircraft (Fig. 56.38) for variable rate application. The WDVI map was converted into a 33 m × 10 m dose grid map (Fig. 56.39) adjusted to the boom width of the field sprayer. On the field, the average use was 0.9 liters of Reglone per ha, with satisfactory efficacy. Standard practice would have been the use of 2 liters per ha. The use of unmanned aerial imaging thus prompted a savings of more than 50%, without loss in potato haulm killing efficacy, harvestability of the potatoes, and final product quality.

Fig. 56.40 Fixed-wing UAV launched over a black-grass-infested wheat field

Wheat is the most widely grown arable crop in the UK, covering around 1.6 million hectares and producing 11.9 million metric tons in 2013. As with other crops, wheat has its share of weed competitors, one of which is black-grass. This prolific weed is highly competitive and increasingly resistant to chemical control, making it one of the biggest challenges facing UK's agricultural sector.

Researchers from URSULA Agriculture have demonstrated a system that combines multispectral sensing with UAVs to capture imagery at 10 cm resolu-

tion (Fig. 56.40) [56.86]. Using the unique spectral and morphological properties of the black-grass, they developed classification processes that are applied to the image to automatically delineate the black-grass from the host crop.

For the farmer and agronomist, the output is supplied in the form of visual maps and spatially attributed data points that are transferable through farm planning software to precision guided machinery (Fig. 56.41). URSULA Agriculture's imagery analysis is able to not only identify the location and in-field extent of black-grass infestation, but also the density and area of black-grass plants, all of which will have a bearing on control decisions. The information provides the farmer with vital data to help contain black-grass infestation in the current growing season. Crucially, it also helps with long-term control by informing management methods such as higher seed rates to increase competition in black-grass *hotspots*, alternative cultivation techniques, or adjusting cropping strategies, all of which can help control black-grass alongside chemical herbicides.

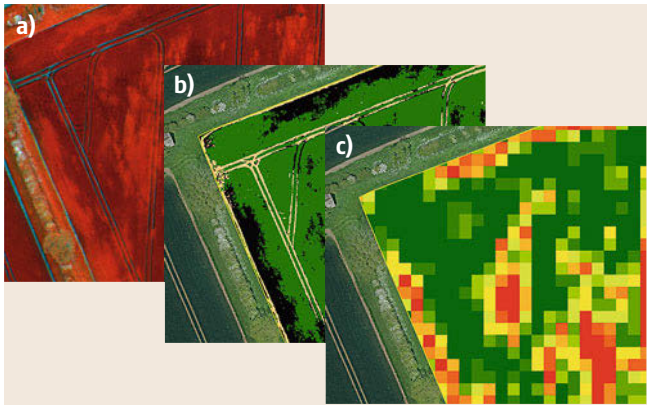


Fig.56.41a–b Black-grass analysis stack: High-resolution multispectral image **(a)**, black-grass delineation **(b)**, and density map **(c)**

56.4 Conclusion

The robotics community has made great strides toward introducing robotic systems in agriculture and forestry. The next ten years will bring improved sensing and mobility, and some advances in manipulation for crop production.

In orchard crops, there is still a need for improvement in algorithms that calculate crop yield and canopy volume or that automatically detect insects and disease. Here the breakthroughs may end up coming from physicists and engineers producing novel imaging systems rather than from roboticists. The continued conversion of existing orchards to the *tree wall* architecture means that autonomous orchard vehicles and platforms will soon have access to the vast majority of the fruit production acreage, at least in developed countries.

Manipulation in orchard environments is a growing and challenging field. The long-sought *harvesting robot* is still an object of fiction, especially when it comes to its economic feasibility. In the apple industry, for example, the best human workers are capable of picking 40 to 60 pieces of fruit per minute, keeping bruising down to a few percent of the volume picked. Our community is relatively far from achieving this level of performance, but we do not need to get there in one shot. Many value-added operations, including pruning, thinning, and harvesting, could benefit from an automated manipulation-based solution. Manipulation should be introduced into low-precision tasks first, where economic gains can be more readily realized, before the technology is refined and directed at tasks such as apple picking. One example is automatic grape shoot and fruit thinning according to each individual vine's crop load [56.87]. This is in contrast to today's methods, where crop load management is conducted uniformly over the entire vineyard, leading to significant variations in yield. Achieving vine-level manipulation in an economically sound way would help move the grape industry in the direction of variable rate management, so that one day, each vine, leaf, and fruit will be treated individually according to its own needs of water, nutrients, light, etc. [56.88].

Despite more than thirty years of research, robotic systems for plant maintenance operations in protected cultivation have not become commercially available. Clearly, the complexity of the operation is an issue. This is due to the highly unstructured working environment, the inherent natural variability of the crop, and the unfavorable environmental conditions, e.g., strong variations in lighting. In this domain, robust sensing and perception is key to successful deployments. This includes aspects such as detection of fruits, stems, and leaves, determination of their properties, e.g., ripeness, and, finally, their accurate localization in the 3-D working environment. Object detection can be quite a challenge as in some cases it boils down to finding a green object in a green background. When color differences are more pronounced detection can be easier, but still in many cases object occlusion poses another problem.

This does not only prevent proper detection of the fruits, but also adds difficulty in the proper assessment of their properties.

In the cluttered environments of protected cultivation, manipulation is also quite challenging. While performing a task, the robot should prevent damage to the crop as this will immediately reduce its value. Still, the robot should be allowed to hit objects such as leaves. Humans tend to have quite intensive contact with the crop during operations. To mimic this kind of behavior requires knowledge about whether objects should not be touched at all or whether they might be gently touched and pushed aside, employing a kind of compliant motion.

Generally speaking, robotics-based crop production can be seen as an instantiation of the hand-eye coordination problem, or the integration of effective sensing, perception, and intelligent manipulation; in other words, those aspects of a well-trained human that are hard to reproduce. Development of technology to mimic human behavior is one way to go, but it will always be limited by the environmental conditions found at the time the robot was introduced. A systemic approach where the crop production system is designed top-down with the robotic system as just one of its components is a much more promising route, yet one that is still in its infancy. For example, growing systems can be modified to make detection and the approach of fruits and flowers easier, thus allowing for more simple design and operation of automated systems. Likewise, cultural practices may be modified to include the robot as part of the task [56.89, 90]. Human-robot interaction might be an interesting and cost-effective intermediate step – allowing for human guidance and supervision only when the machine needs assistance. Such collaborative approaches facilitate data collection in real working environments, offering the opportunity for learning and improving algorithms, thus paving the way for fully autonomous operations in the future.

Another important aspect of the successful introduction of robots in agriculture and forestry is that related to the socio-economic barriers to adoption. In addition to being able to execute a task correctly, the robot must do it in a cost-effective way. Such efficiency can only be demonstrated via numerous field trials, which are time-consuming and costly. Furthermore, the robotic system must have a total ownership cost, including acquisition, maintenance, user training, and disposal, that is less than the financial gains leveraged through its introduction in the production process. Finally, one fundamental aspect is safety. Not only must the hardware and software be designed and validated based on explicit safety requirements, there must be standards and regulations that dictate how and when robots and humans can interact. For an example of a study on the socio-economic barriers to adoption in the apple industry, see [56.91].

The mechanization enabled by the Industrial Revolution and the automation enabled by the information technology era have revolutionized agricultural production to the point where a single farmer can produce grains to feed one hundred people. In the next fifty years, we will witness a similar occurrence in fruits, vegetables, and other crops, thanks to the advances in robotic technologies that our community is continually developing for and applying to agriculture and forestry (VIDEO 93).

Video-References

VIDEO 26 Autonomous orchard tractors
available from http://handbookofrobotics.org/view-chapter/56/videodetails/26

VIDEO 91 Autonomous orchard vehicle for specialty crop production
available from http://handbookofrobotics.org/view-chapter/56/videodetails/91

VIDEO 93 Autonomous utility vehicle – R Gator
available from http://handbookofrobotics.org/view-chapter/56/videodetails/93

VIDEO 95 Automatic plant probing
available from http://handbookofrobotics.org/view-chapter/56/videodetails/95

VIDEO 96 Visual GPS – High accuracy localization for forestry machinery
available from http://handbookofrobotics.org/view-chapter/56/videodetails/96

VIDEO 131 Smart Seeder: An autonomous high accuracy seed planter for broad acre crops
available from http://handbookofrobotics.org/view-chapter/56/videodetails/131

VIDEO 304 A robot for harvesting sweet-pepper in greenhouses
available from http://handbookofrobotics.org/view-chapter/56/videodetails/304

VIDEO 305 Ladybird: An intelligent farm robot for the vegetable industry
available from http://handbookofrobotics.org/view-chapter/56/videodetails/305

VIDEO 306 An automated mobile platform for orchard scanning and for soil, yield, and flower mapping
available from http://handbookofrobotics.org/view-chapter/56/videodetails/306

VIDEO 307 A mini unmanned aerial system for remote sensing in agriculture
available from http://handbookofrobotics.org/view-chapter/56/videodetails/307
VIDEO 308 An autonomous cucumber harvester
available from http://handbookofrobotics.org/view-chapter/56/videodetails/308
VIDEO 309 An autonomous robot for de-leafing cucumber plants
available from http://handbookofrobotics.org/view-chapter/56/videodetails/309
VIDEO 310 The Intelligent Autonomous Weeder
available from http://handbookofrobotics.org/view-chapter/56/videodetails/310

References

56.1 J.A. Foley, N. Ramankutty, K.A. Brauman, E.S. Cassidy, J.S. Gerber, M. Johnston, N.D. Mueller, C. O'Connell, D.K. Ray, P.C. West, C. Balzer, E.M. Bennett, S.R. Carpenter, J. Hill, C. Monfreda, S. Polasky, J. Rockström, J. Sheehan, S. Siebert, D. Tilman, D.P.M. Zaks: Solutions for a cultivated planet, Nature **478**, 337–342 (2011)

56.2 A. Alleyne: Editor's note: Agriculture and information technology, Natl. Acad. Eng. Bridge, Issue Agric. Inf, Technol. **41**(3), 3–4 (2011)

56.3 Global Harvest Initiative: The 2012 Global Agricultural Productivity Report, http://goo.gl/GBPvjK

56.4 J. Reid: The impact of mechanization on agriculture, Nat. Acad. Eng. Bridge, Issue Agric. Inf. Technol. **41**(3), 22–29 (2011)

56.5 E.J. Van Henten: Greenhouse mechanization: State of the art and future perspective, Acta Hortic. **710**, 55–69 (2006)

56.6 S.T. Nuske, S. Achar, T. Bates, S.G. Narasimhan, S. Singh: Yield estimation in vineyards by visual grape detection, IEEE/RSJ Int. Conf. Intell. Robots Syst., San Francisco (2011)

56.7 Q. Wang, S.T. Nuske, M. Bergerman, S. Singh: Automated crop yield estimation for apple orchards, Int. Symp. Exp. Robotics, Quebec City (2012)

56.8 J. Deere: Integrated AutoTrac System, http://goo.gl/CsxxfL

56.9 K. Gallardo, M. Taylor, H. Hinman: 2009 Cost estimates of establishing and producing gala apples in Washington, Washington State University Extension Fact Sheet FS005E, http://goo.gl/BHBZrb (2010)

56.10 B. Hamner, M. Bergerman, S. Singh: Results with autonomous vehicles operating in specialty crops, IEEE Int. Conf. Robotics Autom., St. Paul, MN (2012)

56.11 G. Giacomelli, N. Castilla, E.J. Van Henten, D.R. Mears, S. Sase: Innovation in greenhouse engineering, Acta Hortic. **801**, 75–88 (2008)

56.12 C.W. Bac, E.J. van Henten, J. Hemming, Y. Edan: Harvesting robots for high-value crops: State-of-the-art review and challenges ahead, J. Field Robotics **31**(6), 888–911 (2012)

56.13 J.C. Noordam, J. Hemming, C. van Heerde, F. Golbach, R. van Soest, E. Wekking: Automated rose cutting in greenhouses with 3D vision and robotics: Analysis of 3D vision techniques for stem detection, Acta Hortic. **691**, 885–892 (2005)

56.14 T. Rath, M. Kawollek: Robotic harvesting of Gerbera Jamesonii based on detection and three-dimensional modeling of cut flower pedicels, Comput. Electron. Agric. **66**, 85–92 (2009)

56.15 R. Berenstein, O.B. Shahar, A. Shapiro, Y. Edan: Grape clusters and foliage detection algorithms for autonomous selective vineyard sprayer, Intell. Serv. Robotics **3**, 233–243 (2010)

56.16 E.J. Van Henten, B.A.J. Van Tuijl, G.J. Hoogakker, M.J. Van Der Weerd, J. Hemming, J.G. Kornet, J. Bontsema: An autonomous robot for de-leafing cucumber plants grown in a high-wire cultivation system, Biosyst. Eng. **94**, 317–323 (2006)

56.17 M. Monta, N. Kondo, Y. Shibano: Agricultural robot in grape production system, Int. Conf. Robotics Autom. (1995)

56.18 Wikipedia: Controlled Traffic Farming, http://goo.gl/F0DY04

56.19 J. Jin, L. Tang: Coverage path planning on three-dimensional terrain for arable farming, J. Field Robotics **28**(3), 424–440 (2011)

56.20 D.C. Slaughter, D.K. Giles, D. Downey: Autonomous robotic weed control systems: A review, Comput. Electron. Agric. **61**, 63–78 (2008)

56.21 F.K. Van Evert, J. Samsom, G. Polder, M. Vijn, H.-J. van Dooren, A. Lamaker, G.W.A.M. van der Heijden, C. Kempenaar, T. van der Zalm, B. Lotz: A robot to detect and control broad-leaved dock (Rumex obtusifolius L.) in grassland, J. Field Robotics **28**, 264–277 (2011)

56.22 G. Polder, F.K. Van Evert, A. Lamaker, A. De Jong, G.W.A.M. Van der Heijden, L.A.P. Lotz, T. Van der Zalm, C. Kempenaar: Weed detection using textural image analysis, 6th Bienn. Conf. Eur. Fed. IT Agric. (EFITA), Glasgow (2007), available online at http://edepot.wur.nl/28203

56.23 F.K. Van Evert, G. Polder, G.W.A.M. Van der Heijden, C. Kempenaar, L.A.P. Lotz: Real-time, vision-based detection of Rumex obtusifolius L. in grassland, Weed Res. **49**, 164–174 (2009)

56.24 H. Böhm, J. Finze: Überprüfung der Effektivität der maschinellen Ampferregulierung im Grünland mittels WUZI unter differenzierten Standortbedingungen [Testing the effectiveness of mechanical control of docks in grassland with the WUZI under a variety of conditions.], http://goo.gl/BVdqgy (2004)

56.25 A.T. Nieuwenhuizen, J.W. Hofstee, E.J. van Henten: Adaptive detection of volunteer potato plants in sugar beet fields, Precis. Agric. **11**, 433–447 (2009)

56.26 A.T. Nieuwenhuizen, J.W. Hofstee, J.C. van de Zande, J. Meuleman, E.J. van Henten: Classification of sugar beet and volunteer potato reflection spectra with a neural network and statistical discriminant analysis to select discriminative wavelengths, Comput. Electron. Agric. **73**, 146–153 (2010)

56.27 A.T. Nieuwenhuizen, J.W. Hofstee, E.J. van Henten: Performance evaluation of an automated detection and control system for volunteer potatoes in sugar beet fields, Biosyst. Eng. **107**, 46–53 (2010)

56.28 T. Bakker, H. Wouters, K. Van Asselt, J. Bontsema, L. Tang, J. Müller, G. Van Straten: A vision based row detection system for sugar beet, Comput. Electron. Agric. **60**(1), 87–95 (2008)

56.29 T. Bakker, K. Van Asselt, J. Bontsema, J. Müller, G. Van Straten: A path following algorithm for mobile robots, Auton. Robots **29**(1), 85–97 (2010)

56.30 T. Bakker, K. van Asselt, J. Bontsema, J. Müller, G. van Straten: Autonomous navigation using a robot platform in a sugar beet field, Biosyst. Eng. **109**(4), 357–368 (2011)

56.31 J. Katupitiya, R. Eaton, T. Yaqub: Systems engineering approach to agricultural automation: New developments, 1st Annual IEEE Syst. Conf. (2007) pp. 298–304

56.32 D. Kohanbash, A. Valada, G.A. Kantor: Irrigation control methods for wireless sensor network, Am. Soc. Agric. Biol. Eng. Annual Meeting (2012), available online at http://goo.gl/3NJyXb

56.33 S. Singh, M. Bergerman, J. Cannons, B. Grocholsky, B. Hamner, G. Holguin, L. Hull, V. Jones, G. Kantor, H. Koselka, G. Li, J. Owen, J. Park, W. Shi, J. Teza: Comprehensive automation for specialty crops: Year 1 results and lessons learned, J. Intell. Serv. Robotics, Special Issue Agric, Robotics **3**(4), 245–262 (2010)

56.34 B. Davis: CMU-led automation program puts robots in the field, AUVSI's Unmanned Syst.: Mission Crit. **2**, 38–40 (2012)

56.35 R. Lenain, J. Preynat, B. Thuilot, P. Avanzini, P. Martinet: Adaptive formation control of a fleet of mobile robots: Application to autonomous field operations, Int. Conf. Robotics Autom. (2010) pp. 1241–1246

56.36 A. Shapiro, E. Korkidi, A. Demri, R. Riemer, Y. Edan, O. Ben-Shahar: Toward elevated agrobotics: An autonomous field robot for spraying and pollinating date palm trees, J. Field Robotics **26**(6/7), 572–590 (2009)

56.37 S. Foix, G. Alenyà, C. Torras: Lock-in time-of-flight (ToF) cameras: A survey, IEEE Sensors J. **11**(9), 1917–1926 (2011)

56.38 R. Klose, J. Penlington, A. Ruckelshausen: Usability study of 3D time-of-flight cameras for automatic plant phenotyping, Workshop Comput. Image Anal. Agric. (2009) pp. 93–105

56.39 G. Alenyà, B. Dellen, S. Foix, C. Torras: Leaf segmentation from ToF data for robotized plant probing, IEEE Robotics Autom. Mag. **20**(3), 50–59 (2013)

56.40 G. Alenyà, B. Dellen, S. Foix, C. Torras: Robotic leaf probing via segmentation of range data into surface patches, IROS Workshop Agric. Robotics, Vilamura (2012)

56.41 G. Alenyà, B. Dellen, C. Torras: 3D modelling of leaves from color and ToF data for robotized plant measuring, IEEE Int. Conf. Robotics Autom., Shanghai (2011) pp. 3408–3414

56.42 B. Dellen, G. Alenyà, S. Foix, C. Torras: Segmenting color images into surface patches by exploiting sparse depth data, IEEE Workshop Appl. Comput. Vis., Kona (2011) pp. 591–598

56.43 E.J. Van Henten, B.A.J. Van Tuijl, J. Hemming, J.G. Kornet, J. Bontsema, E.A. Van Os: Field test of an autonomous cucumber picking robot, Biosyst. Eng. **86**, 305–313 (2003)

56.44 E.J. Van Henten, J. Hemming, B.A.J. Van Tuijl, J.G. Kornet, J. Bontsema: Collision-free motion planning for a cucumber picking robot, Biosyst. Eng. **86**, 135–144 (2003)

56.45 E.J. Van Henten, J. Hemming, B.A.J. Van Tuijl, J.G. Kornet, J. Meuleman, J. Bontsema, E.A. Van Os: An autonomous robot for harvesting cucumbers in greenhouses, Auton. Robots **13**, 241–258 (2002)

56.46 S. Hayashi, K. Shigematsu, S. Yamamoto, K. Kobayashi, Y. Kohno, J. Kamata, M. Kurita: Evaluation of a strawberry-harvesting robot in a field test, Biosyst. Eng. **105**, 160–171 (2010)

56.47 S. Hayashi, S. Yamamoto, S. Saito, Y. Ochiai, Y. Kohno, K. Yamamoto, J. Kamata, M. Kurita: Development of a movable strawberry-harvesting robot using a travelling platform, Proc. Int. Conf. Agric. Eng. CIGR-AgEng 2012, Valencia, Spain (2012)

56.48 F. Georgsson, T. Hellström, T. Johansson, K. Prorok, O. Ringdahl, U. Sandström: Development of an autonomous path tracking forest machine – a status report. In: *Field and Service Robotics: Results of the 5th International Conference*, ed. by P. Corke, S. Sukkarieh (Springer, Berlin, Heidelberg 2006) pp. 603–614

56.49 M. Miettinen, M. Öhman, A. Visala, P. Forsman: Simultaneous localization and mapping for forest harvesters, IEEE Int. Conf. Robotics Autom. (2007) pp. 517–522

56.50 J. Roßmann, M. Schluse, C. Schlette, A. Bücken, P. Krahwinkler, M. Emde: Realization of a highly accurate mobile robot system for multi purpose precision forestry applications, The 14th Int. Conf. Adv. Robotics (2009) pp. 133–138

56.51 J. Roßmann, P. Krahwinkler, A. Bücken: Mapping and navigation of mobile robots in natural environments. In: *Advances in Robotics Research – Theory, Implementation, Application. German Workshop on Robotics*, ed. by T. Kröger, F.M. Wahl (Springer, Berlin, Heidelberg 2009) pp. 43–52

56.52 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)

56.53 P. Krahwinkler, J. Roßmann, B. Sondermann: Support vector machine based decision tree for very high resolution multispectral forest mapping, IEEE Int. Geosci. Remote Sens. Symp. (2011) pp. 43–46

56.54 A. Bücken, J. Roßmann: From the volumetric algorithm for single-tree delineation towards a fully-automated process for the generation of virtual forests. In: *Progress and New Trends in 3D Geoinformation Sciences. Lecture Notes in Geoinformation and Cartography*, ed. by J. Pouliot, S. Daniel, F. Hubert, A. Zamyadi (Springer, Berlin, Heidelberg 2013) pp. 79–99

56.55 J. Roßmann, T. Jung, M. Rast: Developing virtual testbeds for mobile robotic applications in the woods and on the moon, The IEEE/RSJ 2010 Int. Conf. Intell. Robots Syst. (2010) pp. 4952–4957

56.56 A. Shiriaev, L. Freidovich, I. Manchester, U. Mettin, P. La Hera, S. Westerberg: *Status of Smart Crane Lab Project: Modeling and Control for a Forwarder Crane*, Department of Applied Physics and Electronics (Umeå University, Umeå 2008)

56.57 U. Mettin, P.X. La Hera, D.O. Morales, A.S. Shiriaev, L.B. Freidovich, S. Westerberg: Trajectory planning and time-independent motion control for a kinematically redundant hydraulic manipulator, Int. Conf. Adv. Robotics (2009)

56.58 U. Mettin, S. Westerberg, P.X. La Hera, A. Shiriaev: Analysis of human-operated motions and trajectory replanning for kinematically redundant manipulators, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2009)

56.59 D. Morales, S. Westerberg, P. La Hera, U. Mettin, L. Freidovich, A. Shiriaev: Open-loop control experiments on driver assistance for crane forestry machines, IEEE Int. Conf. Robotics Autom. (2011)

56.60 A. Hansson, M. Servin: Semi-autonomous shared control of large-scale manipulator arms, Control Eng. Pract. **18**(9), 1069–1076 (2010)

56.61 S. Westerberg, I. Manchester, U. Mettin, P. La Hera, A. Shiriaev: Virtual environment teleoperation of a hydraulic forestry crane, IEEE Int. Conf. Robotics Autom. (2008)

56.62 Y.-C. Park, A. Shiriaev, S. Westerberg, S. Lee: 3D log recognition and pose estimation for robotic forestry machine, IEEE Int. Conf. Robotics Autom. (2011)

56.63 NLIS: National Livestock Identification System, NSW Department of Primary Industries, http://goo.gl/WoHUi2

56.64 Lely home page: http://goo.gl/JCl8Cl

56.65 Juno feed-pusher: http://goo.gl/V5H2Ye

56.66 Lely Vector feeder: http://goo.gl/dLiL3q

56.67 M. Dunn, J. Billingsley, N. Finch: Machine vision classification of animals, Mechatron. Mach. Vis. 2003: Future Trends: Proc. 10th Annual Conf. Mechatron. Mach. Vis. Pract. (Research Studies, Baldock 2003)

56.68 C. McCarthy, J. Billingsley, N. Finch, P. Murray, J. Gaughan: Cattle liveweight estimation using machine vision assessment of objective body measurements: First results, Proc. 28th Bienn. Aust. Soc. Anim. Production Conf., Vol. 28 (Australian Society of Animal Production, Wagga Wagga 2010)

56.69 Y. Wang, W. Yang, P. Winter, L.T. Walker: Non-contact sensing of hog weights by machine vision, Appl. Eng. Agric. **22**(4), 577 (2006)

56.70 W. Zhu, F. Zhong, X. Li: Automated monitoring system of pig behavior based on RFID and ARM-LINUX, Third Int. Symp. Intell. Inf. Technol. Secur. Inf. (2010) pp. 431–434

56.71 S. Barry, D. Cook, R. Duthie, D. Clifford, D. Anderson: Future Surveillance Needs for Honeybee Biosecurity, RIRDC Publication No. 10/107, http://goo.gl/MUrPLo

56.72 Lely Astronaut A4 milking system: http://goo.gl/wYhUVH

56.73 J.P. Trevelyan: Sensing and control for sheep-shearing robots, IEEE J. Robotics Autom. **5**(6), 716–727 (1989)

56.74 Robotics in the Poultry Industry, Poultry CRC: http://goo.gl/xH5vy6

56.75 A. J. Bubb, T. K. Driver, C. D. James: CASE STUDY - Redefining the cowboy: Precision pastoral decision making from remote monitoring and control of cattle, http://goo.gl/Vu0FIQ (2010)

56.76 R. Rankin: Further automation, Meat '93, Aust. Meat Ind. Res. Conf., Gold Coast. (1993), available online at http://goo.gl/Iv2yjh

56.77 Performance Audit, Management of Project Fututech, The Meat Research Corporation, http://goo.gl/C5jMgm

56.78 Robotic Technologies Limited: RTL – Scott and Silver Fern Farms, http://goo.gl/xZxL0Z

56.79 Automated Boning Room System, Scott Group: http://goo.gl/0zFVqi

56.80 Chicken Whole Leg Deboning Machine, Mayekawa: http://goo.gl/xhj2Lf

56.81 Poultry slaughtering, Hyfoma: http://goo.gl/dzTfVT

56.82 Association for Unmanned Vehicle Systems International: The Economic Impact of Unmanned Aircraft Systems Integration in the United States, http://goo.gl/NS5EkT (2013)

56.83 J. Dong, L. Carlone, G.C. Rains, T. Coolong, F. Dellaert: 4D mapping of fields using autonomous ground and aerial vehicles, ASABE Int. Conf., Montreal (2014), Paper No. 141912258

56.84 F.K. van Evert, P. van der Voet, E. van Valkengoed, L. Kooistra, C. Kempenaar: Satellite-based herbicide rate recommendation for potato haulm killing, Eur. J. Agron. **43**, 49–57 (2012)

56.85 C. Kempenaar, F.K. van Evert, T. Been: Use of vegetation indices in variable rate application of potato haulm killing herbicides, Proc. ICPA Conf., Sacramento, USA 2014 (2014)

56.86 C. Pugh, M. Jarman, S. Keyworth, J. Webber, I. Cameron: URSULA Agriculture, http://goo.gl/HzGkRT (2013)

56.87 S. Singh, S. Nuske, M. Bergerman, T. Bates, J. M. Peltier: Vineyard efficiency through high-resolution, spatiotemporal crop load measurement and management, http://goo.gl/iqXgWy

56.88 S. Blackmore: New concepts in agricultural automation, HGCA Conf. (2009)

56.89 J.L. Glancey, W.E. Kee: Engineering aspects of production and harvest mechanization for fresh and

processed vegetables, HortTechnology **15**, 76–79 (2005)
56.90 T. Burks, F. Villegas, M. Hannan, S. Flood, B. Sivaraman, V. Subramanian, J. Sikes: Engineering and horticultural aspects of robotic fruit harvesting: Opportunities and constraints, HortTechnology **15**, 79–87 (2005)
56.91 K. Ellis, T.A. Baugher, K. Lewis: Use of survey instruments to assess technology adoption for tree fruit production, HortTechnology **20**, 1043–1048 (2010)

Multimedia Contents

57. Robotics in Construction

Kamel S. Saidi, Thomas Bock, Christos Georgoulas

This chapter introduces various construction automation concepts that have been developed over the past few decades and presents examples of construction robots that are in current use (as of 2006) and/or in various stages of research and development. Section 57.1 presents an overview of the construction industry, which includes descriptions of the industry, the types of construction, and the typical construction project. The industry overview also discusses the concept of automation versus robotics in construction and breaks down the concept of robotics in construction into several levels of autonomy as well as other categories. Section 57.2 discusses some of the offsite applications of robotics in construction (such as for prefabrication), while Sect. 57.3 discusses the use of robots that perform a single task at the construction site. Section 57.4 introduces the concept of an integrated robotized construction site in which multiple robots/machines collaborate to build an entire structure. Section 57.5 discusses unsolved technical problems in construction robotics, which include interoperability, connection systems, tolerances, and power and communications. Finally, Sect. 57.6 discusses future directions in construction robotics and Sect. 57.7 gives some conclusions and suggests resources for further reading.

57.1 **Overview** 1494
57.1.1 Industry Description 1494
57.1.2 Automation in Construction 1496
57.1.3 Classification of Robotics in Construction 1496
57.2 **Offsite Applications of Robotics in Construction** 1499
57.2.1 Robotics in Component Production 1499
57.3 **Onsite Applications of Single Task Construction Robots** 1504
57.3.1 Steel Welding 1504
57.3.2 Reinforcement Manufacturing and Positioning 1504
57.3.3 Concrete Distribution 1504
57.3.4 Robots for Customized Construction On-Site 1505
57.3.5 Robotic Interior Finishing 1506
57.3.6 Robotic Facade Operation 1510
57.3.7 Earthmoving 1511
57.3.8 Road Maintenance 1511
57.3.9 Material Handling 1511
57.4 **Integrated Robotized Construction Sites** 1511
57.4.1 Robotic Roof Field Factory Approach 1511
57.4.2 Robotic Field Factory on-Ground Approach 1513
57.4.3 Robotic Deconstruction and Robotic Recycling 1513
57.5 **Currently Unsolved Technical Problems** . 1514
57.5.1 Interoperability 1514
57.5.2 Structural Connection Systems 1514
57.5.3 Tolerances 1515
57.5.4 Power and Communications in the Field 1515
57.5.5 Sensing 1516
57.6 **Future Directions** 1516
57.7 **Conclusions and Further Reading** 1516
Video-References 1517
References 1517

Construction is a ubiquitous human activity that relates to the creation or realization of physical artifacts or custom-made capital goods. It is distinguished from manufacturing in that the production activity normally occurs in a field setting and is undertaken in the open air, on natural terrain, and often with naturally oc-

curring materials. Typically, building and construction products are large in scale and unique in form. Moreover, the environment or field setting is typically unique and requires a, rather ad hoc, *factory* to be synthesized on site.

Over the centuries, various forms of machines and mechanical engineering systems have been introduced into the construction engineering domain and into the building and construction industry to increase production efficiency. In common with the fields of agriculture, mining, and forestry, the long-term trend has been for these fields to become increasingly mechanized [57.1].

In the last few decades, with the decrease in the relative cost of machinery to labor and with the globalization of markets, the construction industry has become significantly more capital intensive and large-scale machinery systems and pieces of construction plant – such as tunnel-boring machines and very large tower cranes – have become commonplace. This trend to mechanization is likely to continue with the progressive introduction of computer-controlled construction machinery and flexible manufacturing concepts into the industry.

With the relatively recent development of the microprocessor and the availability of low-cost computer and sensing technology, construction robots have become a technical and economic possibility and this form of technology is now gradually being used in the industry.

57.1 Overview

The application of robots in construction traditionally falls under *construction automation*.As the term implies, the field of construction automation is focused on automating construction processes, and the use of robots is but one aspect of automation. Construction processes also fall within several categories best described through a brief introduction to the construction industry.

57.1.1 Industry Description

The construction industry typically accounts for 5% of a country's gross domestic product (GDP) or gross value added (GVA) and employs a significant portion of the country's workforce. Table 57.1 presents some of these statistics for the USA, the European Union (EU), Japan and China [57.2–8]. Worldwide, construction industry *spending* was estimated at approximately 11% of the world's GDP at the end of the 20th century [57.9].

Construction is considered by many to be technologically behind other industries, such as manufacturing. In manufacturing, a product is designed for mass production, whereas construction *products* (or projects) are usually one-off and unique [57.10]. Thus the efficiencies achieved through mass production are not easily achieved in construction.

Other often attributed reasons for the construction industry's technological lag are the industry's fragmentation and aversion to the risks associated with the introduction of new technologies [57.10, 11]. For example, in the US in 2010 firms with fewer than twenty employees employed approximately 40% of all construction workers, and 63% of all construction workers worked for specialty trade contractors who accounted for 64% of construction firms [57.10]. Specialty trade contractors are usually subcontractors on a project and are not responsible for the overall outcome of the project.

In addition, unlike their manufacturing counterparts, construction sites are for the most part unstructured, cluttered, and congested, making them difficult environments for robots to operate in. Furthermore, human workers are also present in large numbers on a construction project, making safety a paramount concern.

Table 57.1 Value of construction put in place and number of construction workers for the USA, EU, Japan and China

Country Data period (year)	USA 2012	EU 2009	Japan 2011	China 2011
Value of construction put in place as percentage of national GDP/GVA (%)	5	7	10	7
Number of construction workers as percentage of the total national workforce (%)	5	11	10	5
Total number of workers in the national workforce (million)	112	175	63	784

Types of Construction

Construction projects are usually classified as residential, commercial, industrial, or civil. Residential construction generally involves single-family homes or large apartment buildings; commercial focuses on building structures such as office and retail space, warehouses, and so on; industrial is involved in building factories, power plants, and other similar structures; and civil construction focuses on public infrastructure such as highways, bridges, tunnels, and dams.

The Typical Construction Project

A construction project typically goes through six major phases, as shown in Fig. 57.1. Some projects may go through different variations of the sequence shown, but most projects include planning, design, construction, and operation phases [57.4, 10]. Phase 1 begins when a need for a project arises and the requirements are identified. Phase 2 involves developing alternative project plans that could meet the identified needs and evaluating the technological and economic feasibility of each alternative. Phase 3 develops detailed engineering designs and specifications for the plan selected in phase 2. The construction of the facility from ground-breaking through to final inspection takes place in phase 4 of the project. The facility is occupied and commences operation in phase 5 and continues operating until it is time to shut down and dismantle the facility once it becomes obsolete (phase 6).

The actual physical work (building, operating, maintaining, and dismantling) on a construction project takes place in the last three phases. Although these are the only phases in which machines are used, construction automation can also take place during the other phases of a project, as described in Sect. 57.1.2.

During the various phases of a construction project, several stakeholders may be involved at any time. The major stakeholders include the following: the *owner* and *operator*, whose needs initiated the project; the *architect* and *engineer*, who have the task of translating the owner's needs into an aesthetically pleasing and structurally sound design; and the *general contractor*, whose task is to translate the design into a physical structure. In addition several other stakeholders may be involved either independently from the above stakeholders or as part of their organizations; for example, the constructor may employ a project manager, a construction manager, a site superintendent, and others, and he/she will typically subcontract major activities such as excavation and concrete pouring. The subcontractors and sub-stakeholders are often the ones who use a new technology and can either *make or break* its implementation.

Although the builder is the most likely user of robots on a construction project, the actual work on site is often conducted by subcontractors who often are reluctant or financially unable to use advanced technologies which have not been entirely adopted by industry. A constructor's goal is often to meet the owner's requirements by the most efficient *and* least risky methods possible. Hence, traditional construction methods that have stood the test of time are preferred. Nevertheless, it has often been anecdotally reported that the owner has *the power* to require the use of certain technologies on a construction project, since that shifts part of the risk and cost associated with the use of the technology to the owner. This phenomenon may be partially responsible for the widespread penetration of laser scanning technology into the construction industry that is currently taking place [57.12].

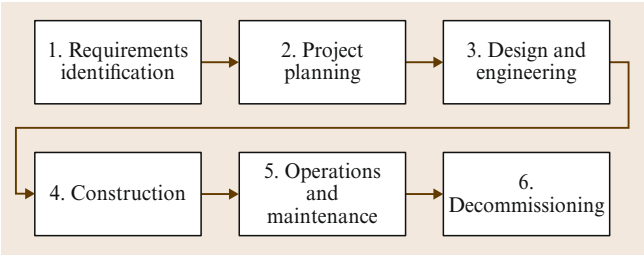


Fig. 57.1 Phases of a typical construction project

Typical Construction Processes

From a purely physical-world and practical point of view, construction may be viewed as being comprised of a finite number of elementary processes [57.13–16], which may be summarized by the following list [57.11]:

- Attaching
- Building
- Coating
- Concreting
- Connecting
- Covering
- Cutting
- Digging
- Finishing
- Inlaying
- Inspecting
- Jointing
- Measuring
- Placing
- Planning
- Positioning
- Spraying
- Spreading.

Most of these processes can also be grouped into three predominant types of functional operators as follows:

- Materials handling (by bulk and unit load)
- Materials shaping (cutting, breaking, compacting, and machining)
- Structural joining.

These functional operators are typically each applied to multiple operands. Common operands in build-

ing and civil engineering are steel and other metals, concrete, timber, earth and rock, masonry, plastic and glass, cement, aggregate, epoxy resin, bitumen, and other bulk and formed materials.

57.1.2 Automation in Construction

Construction automation describes the field of research and development focused on automating construction processes, and the use of robots is but one aspect of that field. In short, construction automation deals with applying the principles of industrial automation to the construction sector, whether in building construction, civil engineering (roadways, dams, bridges, etc.), or in prefabrication of construction components [57.17]. This can be viewed as an extension to research in field service robots generally designed to replace or assist humans in a specific construction-related task or function.

From a historical perspective, research in construction robotics and automation started in the 1980s with the introduction of single-purpose robots (principally remotely controlled, or teleoperated, machines). The Japanese led this effort, driven primarily out of concern for societal demographics, which showed a significant future shortfall in personnel available for the construction labor pool [57.18]. In the US the principal related work involved developing remote control or teleoperated machinery for hazardous work that required modified construction equipment. Example applications include robots developed for rapid runway repair and unexploded ordinance removal. In the EU, research was focused on the development of large-size masonry (brick laying, assembly) robots for residential and industrial building construction.

During the next decade, as research on task-specific construction robots continued, large Japanese construction firms introduced on-site *factories* for high-rise construction. These construction systems included just-in-time delivery of components, automated part tracking and material handling, robotic connection and assembly, and centralized control in an enclosed or semi-enclosed environment. It is reported that the systems enable better working conditions (weather invariant) and reduced project completion time [57.19]. Other benefits include improved productivity and quality, though overall construction costs are not necessarily lower [57.20].

Advanced concepts in integrated residential construction automation were developed as part of the EU FutureHome project. In this construction concept, each structure consists of several high-quality, prefabricated three-dimensional (3-D) modules and two-dimensional (2-D) panels which are assembled *production-style* on site [57.21]. An analogue of this approach for automated residential construction has been commercialized in Japan [57.22].

New methods for collecting, processing, analyzing, and communicating construction information are a significant area of construction automation research [57.23]. This research includes data interoperability and exchange through the design, construction, operations, maintenance, and decommission phases of a capital project [57.24]; advanced sensors for assessing the status of the construction process [57.25–27]; visualization systems for planning construction events, verifying constructability, and maintaining site situational awareness [57.28–30]; and information models which extend traditional computer-aided design (CAD) modeling to combine both the physical (geometric) and functional characteristics of building components [57.31].

Finally, three notable, large-scale, EU projects attempted to integrate construction automation into the way construction projects are executed: the ManuBuild (open building manufacturing) project dealt with, among other things, the development of mobile field factories, including robots, for on-site modular construction [57.32]; the I3CON (industrialized, integrated, intelligent, construction) project dealt with the indoor automation and robotization of buildings [57.33]; and the Tunconstruct project dealt with the robotization of inspection and maintenance operations in tunnels [57.34].

57.1.3 Classification of Robotics in Construction

Construction robotics is an advanced form of mechanization (automation) in which an endeavour is made to automate some industrially important operation and thereby reduce the cost of this operation by either removing a human operator *from the control loop*, or enhance operational efficiency through machine control systems. Due to the nature of construction work, most robots which have been developed for the construction industry are either mobile or relocatable systems. Some platforms, such as floor-finishing robots and machine-controlled earthmovers require mobility as a specific function of the work process to be performed. Others, such as wall and ceiling panel manipulators, require some level of mobility to extend their operating workspace.

Robot terminology can vary depending upon the research discipline. For this chapter two broad classifications of construction robotics are onsite robots and offsite robots. These are distinguished by whether they are applied at the construction site or at a factory or prefabrication facility. In addition, a further distinction can

be made depending on whether a construction robot is intended to be used for a single task (onsite or offsite) or whether multiple robots are integrated into an automated construction site.

Single Task Construction Robots Versus Integrated Robotized Construction Sites

After the initial experiments in large-scale industrialized, automated and robotized pre-fabrication of system houses were conducted successfully in Japan, and the first products (e.g. Sekisui M1) also proved successful in the market, in 1975 a Japanese construction contractor, Shimizu Corporation in Tokyo, set up a research group for construction robots. The goal was now no longer the mere shifting of complexity into a structured environment as in prefabrication, but the development and deployment of systems that could be used on the construction site to create structures and buildings. The focus initially was set on simple systems in the form of so-called single task construction robots that can execute a single, specific construction task in a repetitive manner.

Single-task construction robots are systems that support workers in executing one specific construction process or task (e.g. digging, concrete levelling, concrete finishing, painting) or completely supplement the physical activity of the human workers necessary to perform this one process or task. Further, the processes and tasks they support or supplement can be allocated to a specific profession or craft. In addition, the processes and tasks for which single task robots were developed had in common that they entailed frequent, repetitive activities. Common characteristics of single-task construction robots are as follows:

- Highly specialized not only for a profession but even for a task within a profession (e.g., concrete pouring, levelling and finishing.
- Enhanced productivity compared to conventional labour. E.g., according to [57.17] the conventional labour productivity rate for concrete floor finishing is between $100\,m^3/h$ and $120\,m^3/h$, whereas the productivity using machines is between $300\,m^3/h$ and $800\,m^3/h$.
- Improved quality through precise control of functions and operations (e.g., uniform distribution of paint) and by allowing real time monitoring (and recording) of the operation.
- Improved working conditions by removing workers from dangerous environments and reducing the amount of and heavy physical work.
- Most robots allow various operation modes such as autonomous or sensor guided, pre-programmed, or tele-operated.
- Reduced material consumption through precise control of material delivery and collection and reuse of unused material.
- Most robots have simple yet robust sensor technologies, such as gyroscopes, simple laser systems, or touch/pressure sensors.
- Most robots only require 1 or 2 persons to operate.

The evolution of industrialized and automated building prefabrication during the 1970s along with the development of single tasks robots, envisioned the concept of integrated robotized construction sites. This concept combined prefabrication technology (processing of prefabricated components instead of parts in order to reduce complexity on the site), single task construction robotic systems, and moving or stationary site factories that were able to assemble the building's main structure (e.g., steel frame or concrete structure) almost automatically. As those site factories not only automated parts of the construction process but also integrated prefabricated component technology and single task automation can thus be called as integrated robotized construction sites.

A major reason for the transition into integrated robotized construction sites was that the construction companies realized that single task construction robots which were not networked or embedded within a greater infrastructure, turned out to be incompatible with the way buildings were designed and built. Single task robots were designed to execute certain tasks meanwhile construction workers were not allowed to interfere significantly with the robots activity. However, it turned out that under these premises, only a very few number of robots could be used efficiently. The constraints for the workers, the necessary safety regulations, coupled with the unforeseen, unpredictable, and dynamic processes at the construction site, led to the implementation of individual robots working in parallel. Although single task construction robots achieved a high throughput, significant time had to be spent on-site for transportation, preparation, programming, configuration, etc. Besides the mentioned frictions single task construction robots caused on (conventional) construction sites, the emergence of concepts for integrated sites was also nurtured by the new technological possibilities. Important for the integration of such systems into larger and coordinated automated systems was the development of systems that allowed controlling and monitoring an uninterrupted flow of information and material on-site, between individual automated entities that are involved in the final assembly of the building.

The transition to integrated sites can also be explained from an evolutionary view. In many industries

the evolution from more workshop like production with individual and only loosely coupled production entities or stations, to flow-line like or production line like systems with stable processes and continuous material flow was part of the evolution of other well established and known industries,, i. e., automotive, computer industry.

Finally, three general categories of construction robots are introduced. The first class is *teleoperated systems*, which for simplicity includes *remote control systems*. The general distinction between the two terms is whether or not the equipment must be operated in line of sight from the human controller [57.35]. The second category, *programmable construction machines* (PCM), includes most construction equipment that is outfitted with sensors and mechanisms to augment operation by an onboard human operator. The final category, *intelligent systems*, relates to unmanned construction robots which operate either in a semi- or fully autonomous mode. In Sect. 57.3 this classification will be expanded to include examples based on various generic activities, materials handled (operand types), levels of onboard intelligence, levels of commercialization, and levels of system integration and computer integration.

Teleoperated Systems in Construction

In established engineering terminology, the term *teleoperation* refers to the remote control of machines and systems. In teleoperation (loosely referred to as telerobotics) the control of the machine is accomplished by the use of remote control means such an umbilical cord or wireless control. Teleoperation ideas and methods are used extensively in the space and nuclear industries (Chaps. 55 and 58, respectively).

In telerobotics, the machine does not operate autonomously but is under the control of a human. Data sensing and interpretation and cognitive activities such as task planning are done by the operator.

Recently, many telerobotic devices have appeared in the construction and mining industries. These machines have evolved in response to industrial situations where there is danger to the operator and where remote-controlled machinery is necessary (e.g., teleoperated small compactors). Situations of this kind occur in the construction, demolition, and mining industries and in other hazardous locations.

The technology for telerobotics in construction is well established, with a number of excellent examples of such activity available, for example, a sophisticated model-based supervisory-control-type distributed teleoperation system for the construction of a trench for a diversion dam in a lava field in Japan using a fleet of heavy earthmoving machines [57.36].

Programmable Construction Machines

A software-programmable construction machine is what most people would consider to be a robot. The operator of this type of machine is able to vary the task to be accomplished within certain constraints either by choosing from a preprogrammed menu of functions or by *teaching* the machine a new function. Variations in the task to be accomplished could be as simple as slight changes in the driving speed based on the current load for an automated forklift, or as complex as a change from being able to pick and place steel beams and columns using an autonomous crane to being able to deliver concrete autonomously using the same crane.

Generally, software-programmable construction machines are identical to traditional construction machines (such as an excavator), but have been modified to be controllable through a computer (similar to the way in which traditional manufacturing machines – such as mills and lathes – evolved into computer numerically controlled (CNC) machines).

Software-programmable construction machines can make use of an electronic representation of a portion of the construction site where their work is to be conducted in order to control all or part of the machine's operation. A commercialized example is *stakeless grading*, where data from a 3-D model is used in combination with global positioning systems (GPS) and/or laser measurement systems to automate the blade control for bulldozers and motor-graders.

Intelligent Systems in Construction

As opposed to a teleoperated or a software-programmable construction machine a fully autonomous construction robot is expected to accomplish its task, within a defined scope, without human intervention. A semiautonomous construction robot would be expected to accomplish its task with some level of planning interaction conducted with a human supervisor [57.35]. In each case the construction robot is expected to adapt to its sensed environment, formulate plans for the execution of its task, and replan as necessary (with possibly some human assistance in the semiautonomous mode). The intelligent construction robot should also be able to determine when its tasking is not executable and request assistance.

Example research in intelligent construction systems includes autonomous excavation [57.37, 38] and autonomous crane operations [57.30, 39–41].

57.2 Offsite Applications of Robotics in Construction

The infusion of technologies (such as computer aided design (CAD), computer aided manufacturing (CAM)) into the construction industry from other industries (such as the automotive, airplane and shipbuilding industries) are the clearest proof for a shift in building strategy. Fully automated concrete fabrication is currently a reality that are implemented by factories with customized product delivery in order to adapt to the changing requirements of the market. There are two major industrialized production methods for prefabricated concrete: Production methods that (a) use stationary single formwork and (b) that use mobile formwork.

57.2.1 Robotics in Component Production

Production methods using stationary single formwork comprise mobile workflows such as cleaning, pouring/casting and moving the final product to the storage, while the formwork is stationary during the whole process of prefabrication. In production methods using mobile formwork, the work posts are stationary while the formwork is mobile, moving into the various prefabrication posts of the assembly line.

Concrete Production

A plant for combined manufacture of 2-D and 3-D precast concrete elements includes overhead gantry cranes and electric hoists, used to transport consumable materials (such as lattice girders, steel coils, and parts to be inserted/installed) as well as to lift elements from the concreting station into the storage facility or onto transport pallets. Pallets start at the discharging station, from where they proceed via the pallet cleaning station to the first workstation. There, by means of a de-palletizing device attached to a panel-stacking crane, the finished flooring elements are lifted off the pallets and stacked directly within the working range of the discharging vehicle. A cleaning and plotting (magazining, cleaning, plotting (MCP)) robot is used for a variety of tasks: picking up, insertion of latitudinal anchors, cleaning of pallets, full scale plotting of elements, and installation of latitudinal anchors (Fig. 57.2). Figure 57.3 shows further examples of robotic concrete component manufacturing and handling.

Brickwork Component Production

Since the late 1970s, due to the lack of skilled workers in construction and increasing costs of buildings, rationalization developments were started. They mainly took place in the masonry and the formwork sector, where no ready solutions from other countries were available. In Germany and also in most parts of Europe an often used and still loved building material is brickwork. Most of the building projects are constructed with this material. Due to the lack of brickwork in Japan (not resistant to earthquakes if not reinforced), and missing developments in other countries, main efforts were made in this field.

Fig. 57.2 (a) Pallet in concreting station; (b) bridge crane; (c) pallets molding systems; (d) demolding-depalletizing device; (e) cleaning, measuring and oiling (after [57.42])

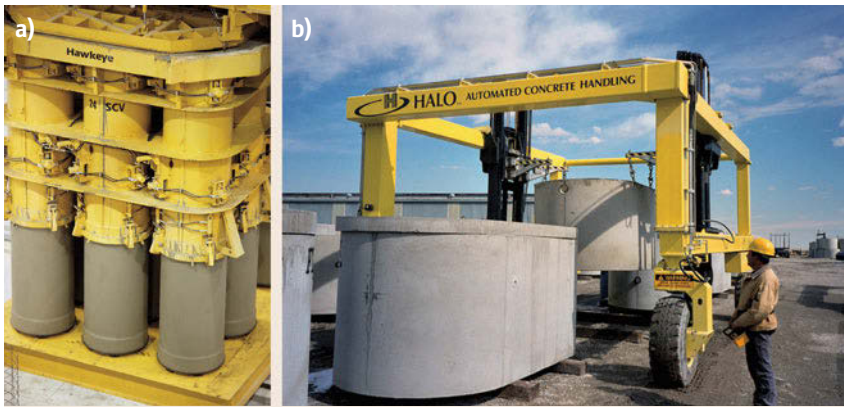


Fig. 57.3 **(a)** Automated concrete pipe manufacturing machines (courtesy Hawkeye Group USA); **(b)** precast concrete handling robot (courtesy Halo, Inc.)

In the last few decades activities to develop machines that support the laying of bricks and even automated robots have been increasing, especially in Germany. Brick laying is a very labor-intensive activity, which can lead to significant health issues and early retirement [57.43, 44]. Consequently a very small number of skilled workers is available nowadays. This in turn is reflected in ultimately high prices and wages, which will make brick building in some time prohibitively expensive. Therefore it should just be a question of time before only houses are put on the market that have not been erected on site by a team of bricklayers, but by a robot in the production shop.

Early Trials of Robotic Assembly of Modular Blocks

One idea for Robot Oriented Design (ROD) [57.45–47], in building construction is to change the conventional construction works adaptable to robotics so that construction system and advanced automation and/or robotics are co-adapted. Thus a structural system for the wall erection named SMAS (Solid Material Assembly System) [57.48] was proposed and developed already in the 80s in Japan. SMAS is a of reinforced masonry construction system. A standard building component of this system, 30 cm × 30 cm × 18 cm in size and 20 kg in weight, is made of pre-cast concrete and includes cross-shaped steel bar inside each component for the reinforced of structural wall. Components are positioned automatically by the robot one by one without arrangements of conventional bonding. Following the positioning of each components, steel bars are connected to those of adjacent components also by the robot. The joint type of steel bar for vertical direction is mechanical, and that for lateral direction is overlapping. Concrete is grouted from the top of the wall which is erected one storey high (about 3 m). The newly developed operating hand is installed to the mother robot (6-articulation-type robot) which was developed for a wide variety of applications in factory use and a series of experiments for wall erection were carried out.

The rapid progress being achieved today in the modernization and industrialization of building construction technology has triggered a trend to reduce the complicate works at construction site and increasingly produce building components at factories. It is obviously that the prefabrication has been successful in up-grading the quality of the building and in shortening the construction period. The sizes of those building components such as prefabricated structural members are also becoming larger to simplify assembling work at the construction site.

However, these movements are not necessarily oriented toward the introduction of robots. The heavy and large components are difficult to be operated by the robots and the complicated assemble techniques are sometimes too skilful for robots. Meanwhile, when one looks at the prefabrication of compact and lightweight structural components as a means of accomplishing construction work more efficiently, robots can be used for assembling these structural members efficiently.

From this point of view SMAS structural system was developed and proposed as a robot-oriented construction system. A masonry structure has not been considered as a major structural system in Japan because of earthquakes; however, it becomes to be recognized as a flexible structural system applied to various building designs when properly reinforced. In addition, it has advantages in the construction cost and in the construction period. The SMAS system itself was designed as a modular system for the purpose that single components could be developed further in the future. All system components including stones are developed as complementary parts according to the guidelines of Robot Oriented Design (ROD) (Fig. 57.4). These robotic approaches were meant to be applied on-site, but technologically they were proven to be not effective. Thus, they became better suited for offsite assembly operations.

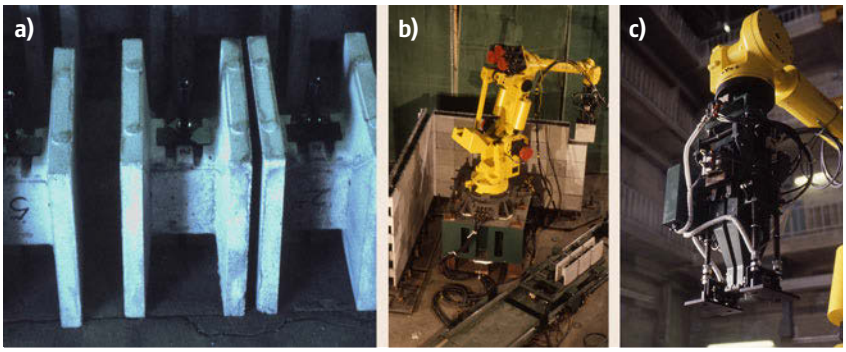


Fig. 57.4 **(a)** Building component developed by guidelines of Robot Oriented Design (ROD), **(b)** SMAS total system, **(c)** robot end-effector/gripper with optical fiber sensor guiding screwing mechanism

Automated Brickwork Plants

The adoption of CAD/CAM systems combined with implementation of Enterprise Resource Planning (ERP) Solutions has made the implementation of advanced technically refined, fully automatic production plants possible:

- High degree of information and communications technology (ICT) integrated and automated/interconnected processes
- Integration of as many devices as possible by ICT (interoperability)
- CAD/CAM Systems
- Integration of processes by ERP Systems
- Just in Time and Just in Sequence supporting customization.

Brickwork Robot Plant SüBA

Crucial technical developments preceded the construction of a fully automatic brickwork machine. The prototype of the automatic brickwork plant was developed by the company SüBA in Hockenheim, and the company Windhoff AG in Rheine at the beginning of the 1990s. The production of brickwork panes is appropriate for a capacity of 300 m^2 net area of brickwork panes – without windows and door recesses – in a shift of eight hours. The employment of CAD in the architect's offices made it possible to transfer the large data set for the production of brickwork panes directly without manual input over CAM to the brickwork robots. Considering the large number of data for the automat, which was necessary for the production of a brickwork pane, this was one of the most important problems of the automation of bricking at that time. Figures 57.5 and 57.6 give an overview over the most important SüBA factory modules and automation processes.

Brickwork Robot Plant Winkelmann (Horizontal Brickwork Panel Production)

Today's fully automated and highly robotized brickwork plants can be distinguished into two basic types: horizontal and vertical brickwork panel production. The brickwork robot plant Winkelmann is a characteristic example for horizontal panel production. All single devices are equipped with Microsystems, interconnected and part of a systemic logistic network. CAD/CAM guarantees efficient data processing between planning section and production. After the delivery standard palettes bricks are brought into processing order by an automated palletizing system with a robot for distribution of bricks (Fig. 57.7a). After that the bricks are taken up by automated de-palletizing system (Fig. 57.7b), which supplies horizontal brickwork layering robot station (Fig. 57.8a). Insertion of reinforcement and house

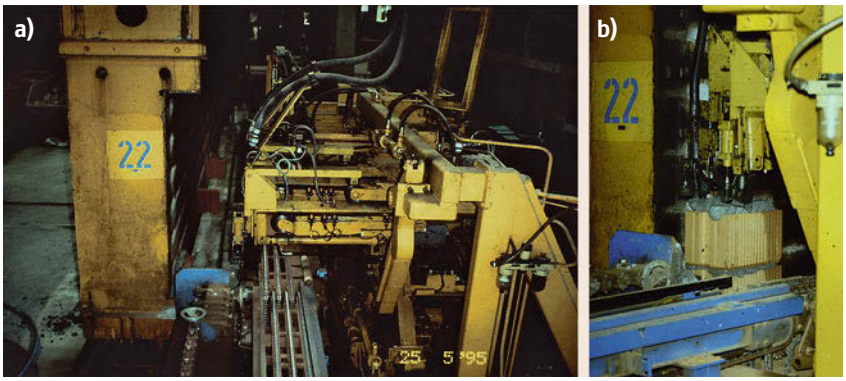


Fig. 57.5 **(a)** Automated reinforcement (horizontal) station; **(b)** mortar dispensing

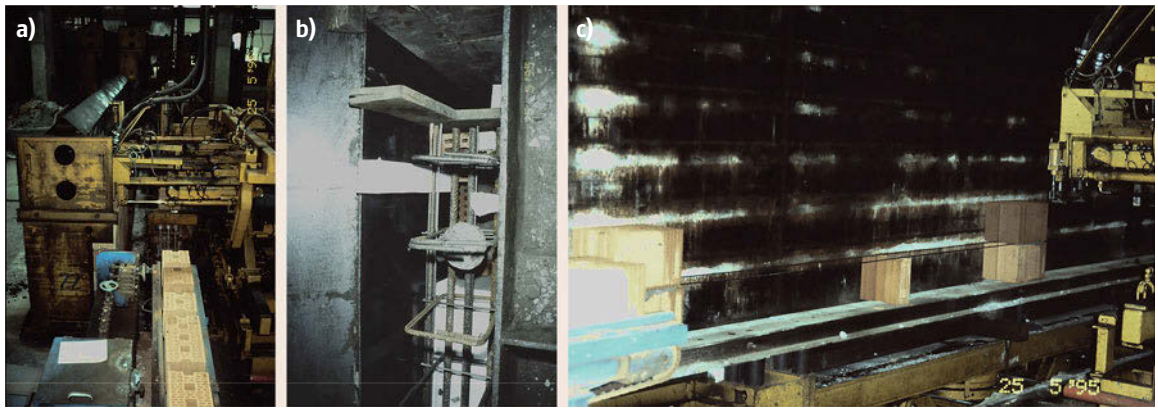


Fig. 57.6 (a) Brickwork positioning; (b) brickwork alignment; (c) automated reinforcement (vertical) station

Fig. 57.7 (a) Automated palletizing system with robot for distribution of bricks; (b) automated de-palletizing system

Fig. 57.8 (a) Automated horizontal brickwork layering robot station; (b) insertion of reinforcement and house infrastructure

infrastructure as well as plastering are done stationary in the factory (Fig. 57.8b) until a finalized and dried-out brickwork panel is delivered to the construction site.

Brickwork Robot Plant Leonhard Weiss (Vertical Brickwork Panel Production)

The brickwork robot plant Leonhard Weiss is a characteristic example for vertical brickwork panel production. Many processes as for example the palletizing and de-palletizing to bring the bricks into factory order are similar to the horizontal type. Yet the central process of positioning the bricks in given order is done vertically layer by layer. High accuracy robots combined with linear axis are here in charge for the exact positioning. The automated vertical brickwork layering has some advantages in terms of efficient use of the factory area and moreover the firmness of the wall could be improved whereas the exact positioning of reinforcement, cables and other elements of the house infrastructure is done easier with the horizontal type. CAD/CAM combined

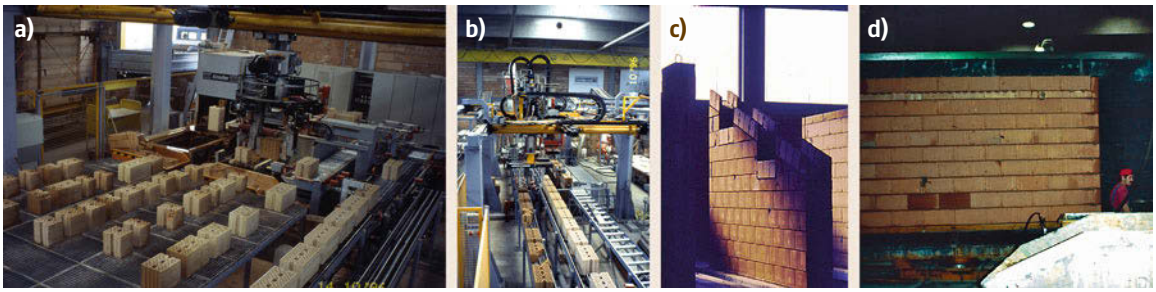


Fig. 57.9 **(a)** Sequencing station; **(b)** robotic assembly system for vertical brickwork layering; **(c,d)** final brickwork panel product

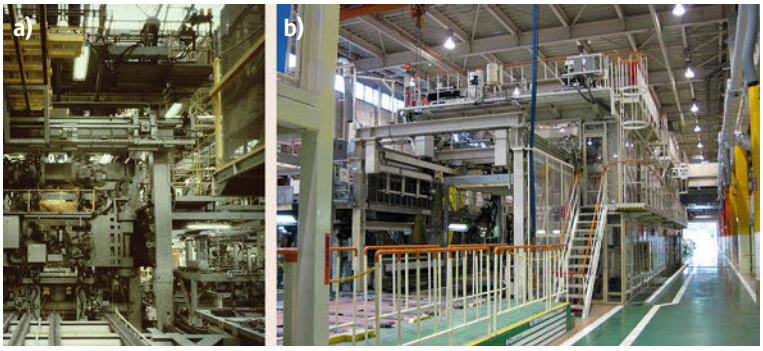


Fig.57.10a,b 3-D adjustable automated assembly welding unit (Sekisui Heim)

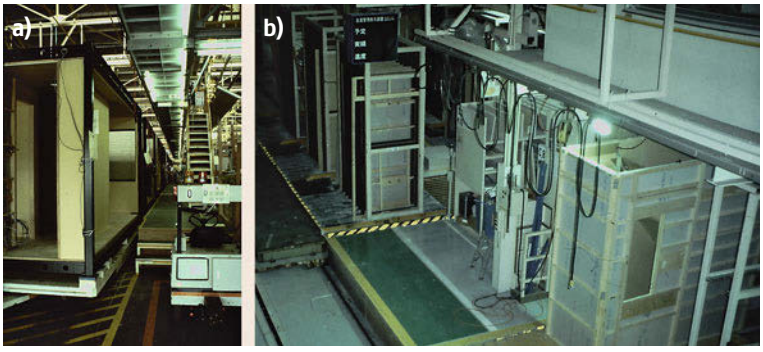


Fig. 57.11 **(a)** On the 400 m production-line, the steel frame units (*chassis*) pass several workstations; **(b)** 10 Insertion of prepared elements and by suppliers (e.g., Toto, Inax) prefabricated bath modules into the chassis unit

with variable production and ICT integrated production allows individually fabricated brickwork panels and necessary variations. Figure 57.9 shows the various steps of the automated vertical brickwork panel construction process.

Steel Component Production

Sekisui Heim introduced an automated steel frame production. One of the basic features is the automated assembling and welding station. Ceiling elements, flooring elements and columns are fed into this station, followed by automatic welding into a frame, which is used as chassis and bearing structure during the further completion process on the production line (Fig. 57.10). After the automated welding process, the steel frame chassis is streaming through the factory from work step to work step, until all installations have been completed (Fig. 57.11a). The factories of Sekisui and Toyota have gates on both sides of the assembly lines in order to receive material, parts, components and prefabricated bath or kitchen modules, required for the customized production of individual units. All of them arrive just-in-time and just-in-sequence by cooperating suppliers (Fig. 57.11b).

57.3 Onsite Applications of Single Task Construction Robots

The construction of larger buildings that use steel as the structural support system can involve a high total amount of welds. If the design of the columns and beams can be adjusted in order to reduce the amount and variety of welds, welding becomes a highly repetitive operation suitable for being automated.

57.3.1 Steel Welding

Automated welding is able to control and guarantee the quality of the connection between the welded parts to a level similar to (and sometimes better than) that achieved by professional human welders. Manual welding of these and other types of connections requires highly specialized skills that are currently in short supply. Furthermore, conventional welding can have damaging long-term effects on a worker's vision. Simultaneous automated welding on a beam (e.g., from two or three coordinated positions) is even able to ensure that the steel component is not distorted by the welding operation itself and thus improves the quality of the constructed structures. In Fig. 57.12a the Shimizu steel-welding robot is depicted. The robot can automatically weld a column including the corner portions. The configuration of a joint is detected by a laser sensor, and the welding is performed in an optimized way referring to a database. In Fig. 57.12b, the Obayashi Corporation Steel welding robot operates in a similar principle, using a circular clamp-on type fastener to the column. The welding process is computer-controlled, though workers are still involved in overseeing operations at least for the time being.

57.3.2 Reinforcement Manufacturing and Positioning

Concrete reinforcement operations involve cutting and bending of rebar (reinforcement bars), precise (relative to each other) arrangement of those rebar, binding of rebar and final positioning of the rebar elements or mesh on a floor or in a mold or formwork system. These tasks are labor intensive and hard on a worker's body (e.g., back accidents and damage to a worker's musculoskeletal system occur about ten times more during reinforcement work than during painting work [57.49]). Automated systems mitigate the risks and impacts on the health of workers and enhance the quality of the reinforced concrete structures. Figure 57.13a shows a reinforcement production robot and Fig. 57.13b shows a teleoperated rebar placing robot.

57.3.3 Concrete Distribution

Concrete distribution systems are used to distribute mixed concrete with uniform quality over large surfaces or over formwork systems. Concrete distribution

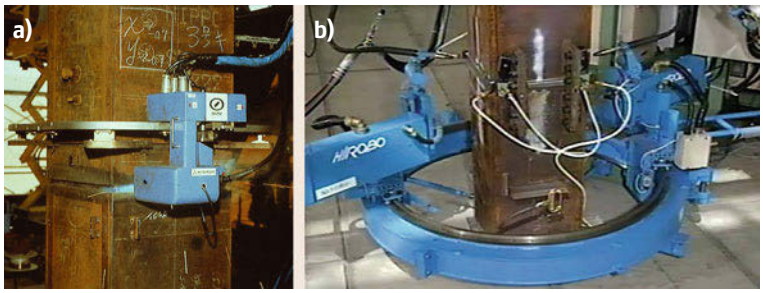


Fig.57.12a,b Steel welding robots. **(a)** Shimizu Corp.; **(b)** Obayashi Corp

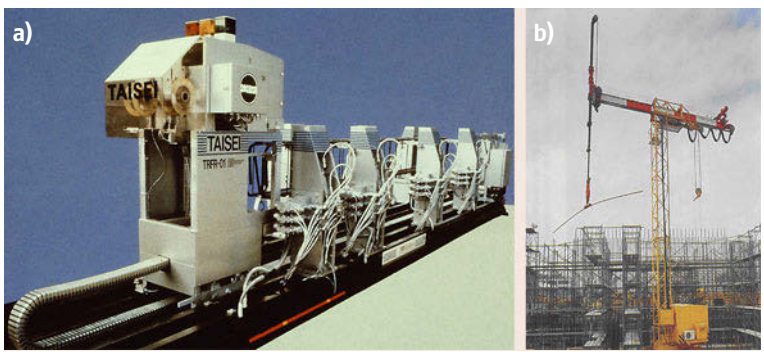


Fig. 57.13 **(a)** Automated reinforcement production and **(b)** teleoperated rebar placement (after [57.17])

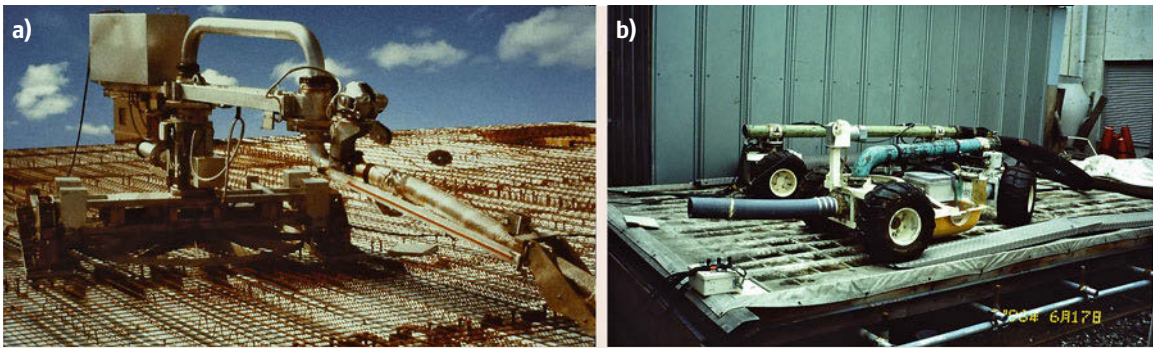


Fig. 57.14 (a) Takenaka Corp. Automatic Concrete Distributor *DB Robo*; (b) Tokyu Concrete Distribution Robots

Fig. 57.15 Concrete slipform road paving machine (courtesy GOMACO Corp.) ◄

covers the continuous supply with concrete (pumps, hoses), a system which slides in a certain pattern over the area where the concrete has to be distributed and a concrete ejection system. Systems can be operated manually, tele-operated and sometimes sensor-guided or fully automatic. Depending on the surface where the concrete needs to be distributed, concrete distribution systems can be truck mounted, stationary or mobile. In Fig. 57.14a, a track mounted automatic concrete distributor robot (DB Robo) can be seen. In Fig. 57.14b a mobile concrete distributor system can be seen, which due to its wheel-based platform, enables an enhanced operating range. Figures 57.15 and 57.16 show other examples of automated concrete distribution systems.

57.3.4 Robots for Customized Construction On-Site

Several masonry-wall erecting robots have been developed to-date. Examples of these are the SMAS in Japan

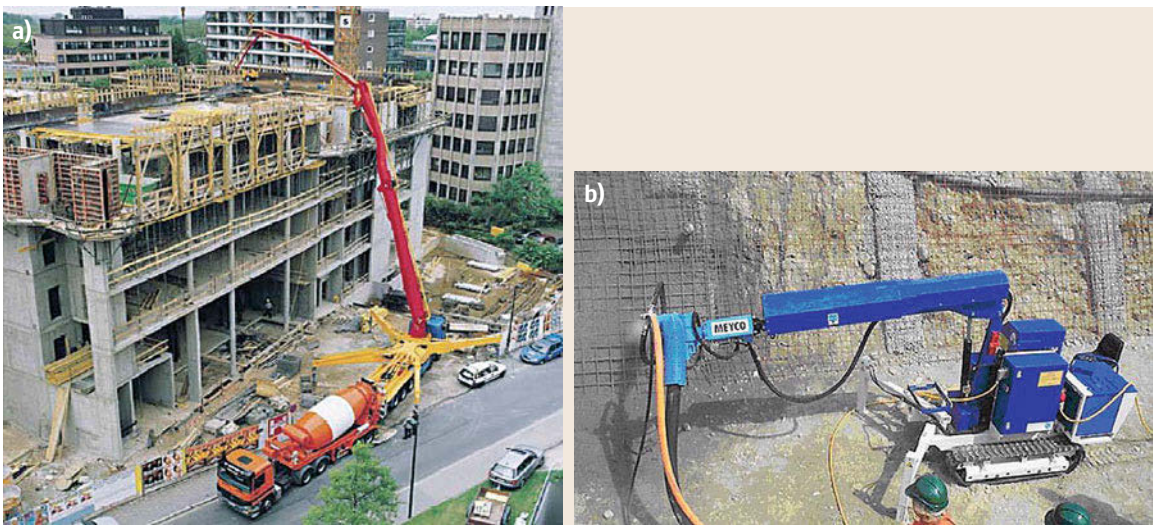


Fig. 57.16 (a) Programmable, articulated-boom machine for the pumped delivery of fresh concrete (after [57.50]) and (b) teleoperated concrete spraying robot (courtesy MEYCO Equipment)

and the ROCCO robot at the University of Stuttgart. Many of these systems are still in the prototype stage. The basic concepts that these systems have in common are as follows:

- Autonomous mobility on the construction site
- Sensor systems determine the robot's position and orientation in its environment
- Automatic pick up of the bricks from the pallets
- Automatic application of mortar
- Automatic positioning of the bricks.

ROCCO is a Robot Construction System [57.51, 52], for Computer Integrated Construction (CIC). Several companies and institutes participated in this EU funded project in an inter-disciplinary and international approach with experts in the fields of construction technology, mechanical and electrical engineering and information technology from Germany, Spain and Belgium. The goal of the project was the development of a computer integrated robot system, which also contains a continuous solution in the ICT (Information Communication Technology) for all steps from the architectural design to the automated assembly of the components on the construction site.

The main emphasis of the project lies on the realization of a mobile robot system for construction site operation as well as on the integration of a computer based system for work preparation and quality control. In the work preparation phase the necessary data is generated for the pre-fabrication and customization of the masonry blocks, for the construction site layout and for the automatic robot program generation.

Based on the CAD-representation of the building, first the walls are divided automatically into the necessary blocks. In the next step the optimal working positions of the mobile robot as well as the positions of the pallets and the arrangement of the blocks on the pallets are automatically calculated. With this information the necessary non-standard and standard blocks can be produced and palletized. Finally the robot programs are generated automatically out of the calculated geometry information. The user interface on the construction site is graphically interactive and enables the user to partially re-program the generated robot programs to deal with the uncertainties of the construction process without the need to learn a specific robot programming language.

To test different approaches based on the construction application (residential buildings vs. industrial) and the sensor integration (autonomous vehicle vs. long reach), two systems were developed within the project. The first one, for residential buildings, has a reach of 4.5 m and it is able to handle up to 400 kg. This robot is placed over an autonomous vehicle that allows the movement on the construction site, Fig. 57.17a. Its main task is the erection of walls in residential buildings. The second one is able to handle up to 1000 kg with a reach of 8.5 m. This robot, Fig. 57.17b, is placed over a towable platform and its main task is the erection of external walls in industrial buildings with typical height of up to 8 m and standardized layout.

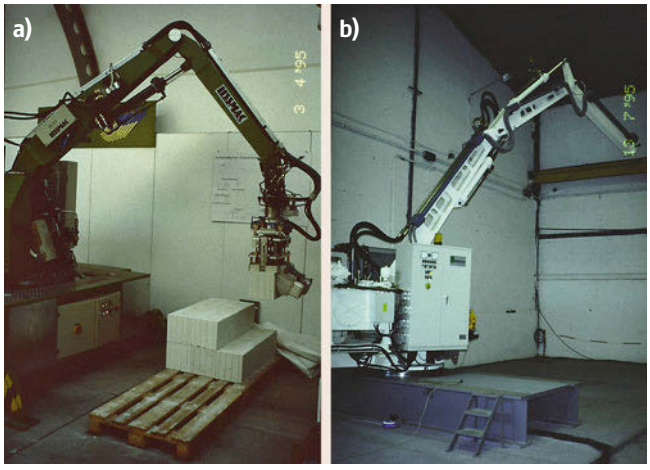


Fig. 57.17 (a) Robot for residential buildings; (b) robot for industrial use with long reach manipulator

57.3.5 Robotic Interior Finishing

Interior finishing work is work conducted inside a structure (e.g., an office building) to complete construction of the space within the structure (e.g., painting and false ceiling and gypsum board installation). Due to the confined nature of interior finishing work traditional material handling equipment (such as cranes) cannot be used. The following four systems are examples of interior finishing robots:

- The Shimizu CFR 1 (Fig. 57.18a) could be manually and tele-operated and allowed the installation of suspended ceiling panels to a height of up to 3.5 m. The robot's compliant gripper mechanism was able to finely adjust the positions of the panels.
- The Tokyu ceiling panel installation robot (Fig. 57.18b) could position and adjust the panels and also fire nails into the panels to fasten them to the underlying ceiling system.
- Two variants (Fig. 57.18c,d) of interior finishing modular mobile robot, developed in Technical University of Munich.
- The Komatsu Mighty Hand LH (Fig. 57.18e) could position and adjust interior wall panels, glass panels, door frames, window casings and the outside walls (up to 350 kg) with high accuracy.

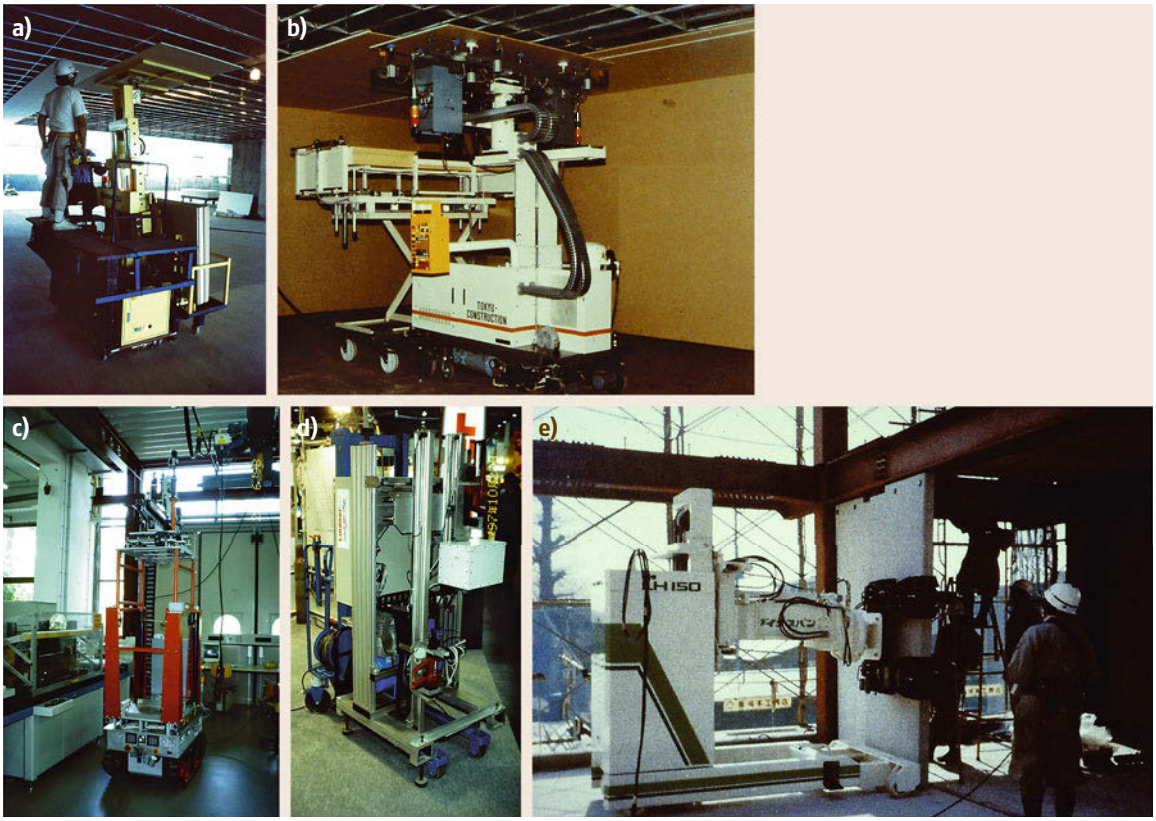


Fig. 57.18 **(a)** Shimizu CFR 1 – Interior finishing robot; **(b)** the Tokyu ceiling panel installation robot; **(c)** BR2 TU Munich – modular ceiling drilling robot system type 1; **(d)** BR2 TU Munich – modular ceiling drilling robot system type 2; **(e)** Kajima interior wall setting – KOMATSU LH 150

Concrete Finishing

Floor finishing is one of the most critical construction processes in which construction workers carry or guide trowels over an unfinished, wet concrete floor for several hours in a stooped posture. In order to reduce these physical demands on the workers and to ensure a more uniform finish quality, various companies developed and employed concepts for robots that can perform the concrete-leveling task: the Flat-Kun by Shimizu (800 m^2/h), the Kote King by Kajima (500 m^2/h), the Surfing Robo by Takenaka (300 m^2/h), the Obayashi made by Mitsubishi (500 m^2/h) and the Floor Traveling Robot MHE by Hazama (300 m^2/h). Each of these single task construction robots was able to operate on a floor where it was set up into any desired direction (e.g., not only move backwards and forwards and turning but also able to rotate 360° in place).

The finishing mechanism consisted, in most cases, of automatically controlled and operated rotating trowels. The degree of autonomy ranged from systems with human-machine interfaces for tele-operation to systems that could generate motions themselves and to preprogramming of paths for the robot to follow. In many cases gyroscopes and rotating laser levels assisted navigation and motion planning at a low level. After an intensive research and development phase that lasted until 1985, the first concrete finishing robots where deployed commercially to finish concrete floors in office buildings, factories, warehouses and shopping centers. The use of the robot systems became efficient once a floor plan allowed for a working area that could be processed without interruption of more than 500 to 600 m^2. Examples of commercial systems are depicted below in Fig. 57.19.

Tile Placement

The exterior of many buildings are sometimes finished with weather-resistant tiles. In Japan single-family buildings, factories, offices and high-rise buildings are often finished with tiles. Tiles are relatively small building elements compared to the total surface area of a building and thus a very large number of tiles have to be installed using the same, repetitive process, which involves applying mortar and positioning the tiles on

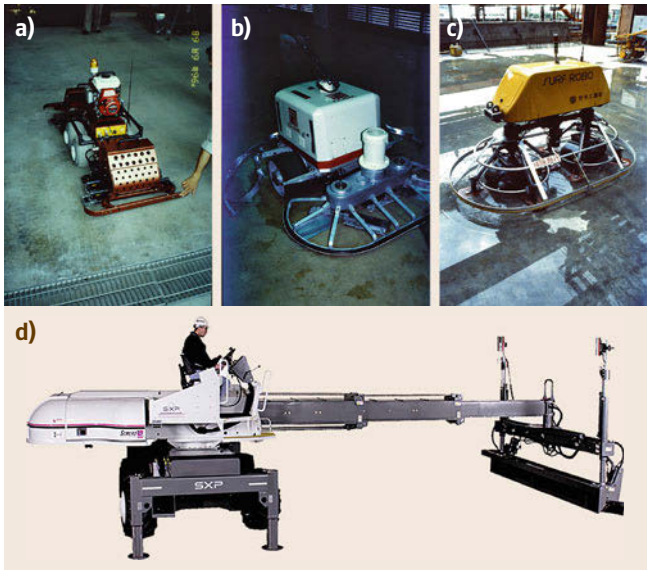


Fig. 57.19 **(a)** Hazama robot; **(b)** Kajima Kote King robot; **(c)** Takenaka Surf Robo; **(d)** auto-leveling concrete screed machine (courtesy Somero Enterprises, Inc.)

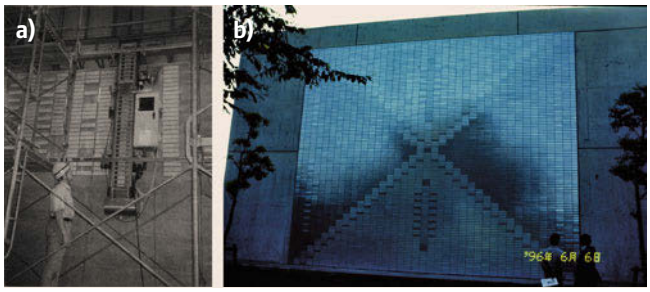


Fig.57.20a,b Hazama tile setting robot; **(a)** robot upon operation; **(b)** finished tile setting

top of the mortar. This and the fact that building facades are generally difficult to access make the use of automated systems feasible. Hazama's tile-setting robot (Fig. 57.20) also showed that accuracy could be enhanced and that the laying of patterns could be accomplished without dramatically increasing construction time.

Fig.57.21a,b Shimizu fireproof coating robots

Fig. 57.22 Concrete panel installation robot (courtesy of Fujita Research)

Fireproof Coating

In many countries building regulations require that steel structures be coated with fire proof materials and/or a fire-retarding paint. The application of the

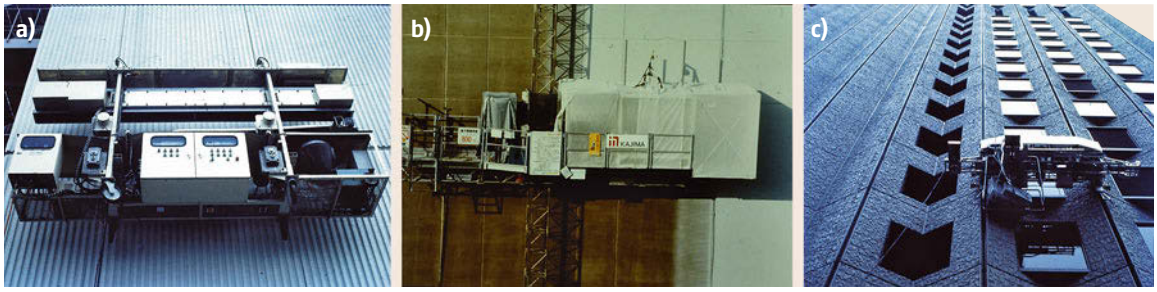


Fig.57.23a–c Facade painting robots. **(a)** Shimizu; **(b)** Kajima; **(c)** Taisei

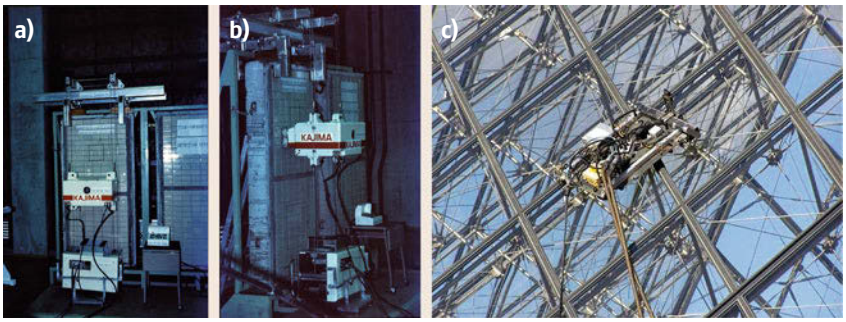


Fig.57.24a–c Kajima facade inspection robot **(a,b)**; Comatec Robosoft glass cleaning robot **(c)**

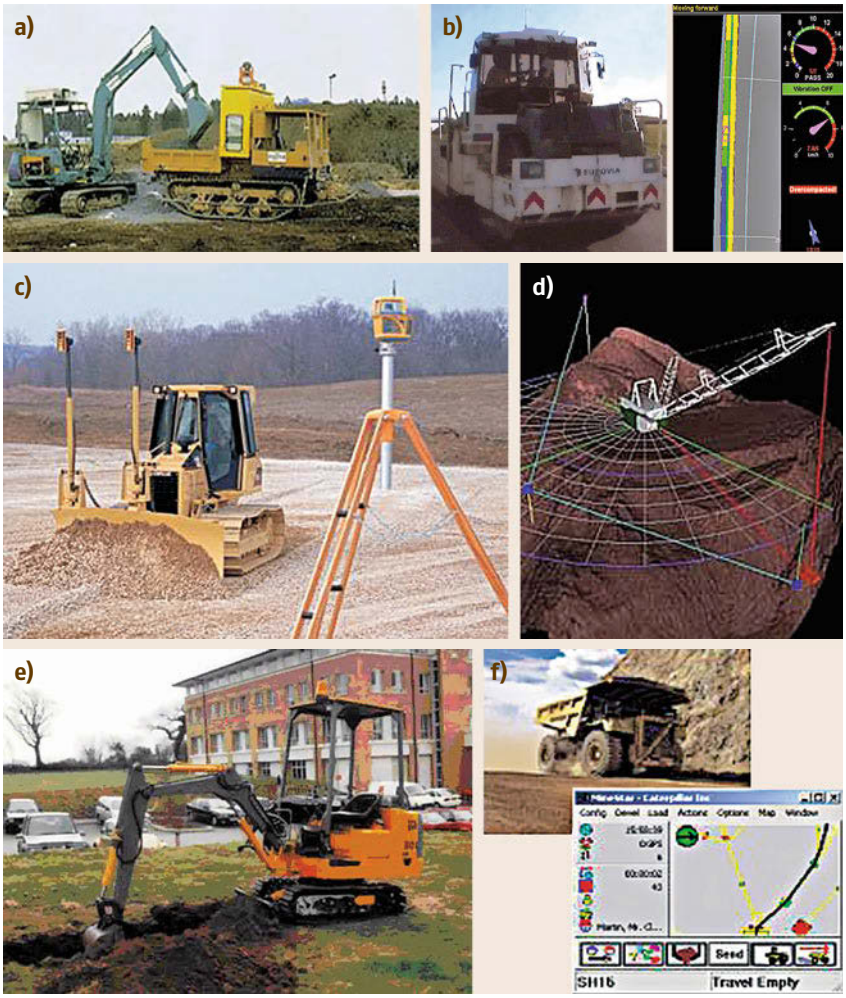


Fig.57.25a–f Earthmoving: **(a)** Teleoperated excavation system (courtesy Fujita Research); **(b)** computer assisted road compacting system (after [57.53]); **(c)** automated grading system (courtesy Caterpillar, Inc.); **(d)** automated drag line control system (after [57.54]); **(e)** autonomous excavator robot (after [57.55]); **(f)** autonomous off-road dump truck (after [57.56])

fire-roof coating and paint can only be done after the steel structure has been erected and joined on site (e.g., by welding) in order not to interfere with the joining process and in order to avoid any damage to the fire-proof coating. Therefore, it is not possible to shift the coating operations to upstream production steps that can be performed in structured factory environments where high efficiency can be achieved. Due to seismic considerations, most high-rise construction involves the use of steel structures and thus, the development and employment of automated and robotic systems that are able to apply the fire-proof coatings after the structure is erected took place in Japan. Robots devel-

oped by Shimizu (Fig. 57.21) and others were mostly autonomous.

57.3.6 Robotic Facade Operation

Facade operations involve the installation of windows, complete facade elements or building exterior walls. Facade elements are, in modern architecture and especially in high-rise construction, decoupled from the main load-bearing concrete or steel structure and can thus be considered as a type of *add-ons*. Facade installation operations are complex operations that involve the accurate positioning of heavy parts or elements at locations that are difficult to access (e.g., high elevations without scaffolding). This involves risk of injury (and thus extensive safety measures must be taken) and of damaging the building or the elements themselves. Furthermore, the positioning and alignment of prefabricated facade elements requires precision and low tolerances. A widespread trend (since the 1980s) of designing large buildings as monolithic structures that repeat the same or similar facade elements, has been a major motivation for investment in automated or robotic systems. Figure 57.22 shows an example of a tele-operated façade installation robot.

Painting

Facade-painting robots were developed to simplify the painting or re-painting of high-rise buildings during construction and operation. Facade painting robots have a particular advantage in that they can keep the paint finish quality constant. They usually have multiple spray nozzles and the spray area is either encapsulated or hermetically sealed in order to avoid streaking. A further advantage of facade painting robots is the fact that workers are not exposed to harmful paint fumes and vapours. Single task, facade-painting robots use one of the following three different strategies to move along the facade:

- Cable-suspended cage/gondola systems
- Rail guided systems
- Vacuum or other adhesion technology.

The use of façade-painting robots was not considered efficient for facades with an area $< 2000\,m^2$. Facade-painting robots were thus used primarily to paint large facades of warehouses and skyscrapers. Facades to be processed were required to have a low curvature and wherever possible no corners or lugs which can hinder the operation of the robot. Further, the design of window frames as well as the number of windows and the area they cover impact the feasibility and efficiency of facade-painting robots. Between 1984 and 1988 various companies introduced facade-painting robots. Shimizu and Kajima Corporations of Japan both applied the principle of the suspended cage or gondola (Fig. 57.23a,b, respectively). The Taisei Corporation of Japan used a rail-guided system and combined it with the gondola approach for their robot (Fig. 57.23c). The fastest of the systems (Kajima's robot) worked with a speed of $290\,m^2/h$ during the application of the primer paint, $200\,m^2/h$ for the base coat and $290\,m^2/h$ for the top coat.

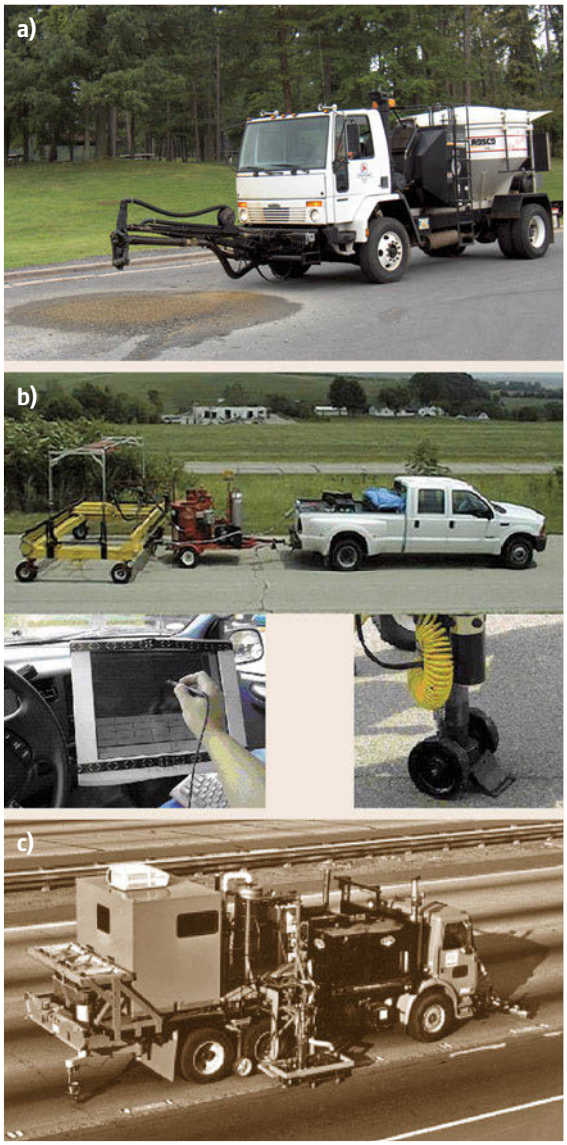


Fig.57.26a–c Road maintenance: **(a)** Teleoperated pothole patching robot (courtesy Leeboy); **(b)** semi-autonomous road crack sealing robot (after [57.57]); **(c)** semi-autonomous road crack sealing robot (after [57.58])

Fig.57.27a–c Material handling: **(a)** six degree of freedom robotic crane (after [57.59]); **(b)** large scale pipe manipulator; **(c)** large manipulator system (courtesy of Shimizu Corp.)

Robotized Inspection and Maintenance

Facades of high-rise buildings are in many cases equipped with tiles or other surface panels that have to be inspected regularly during the building's life-cycle in order to detect structural damage and in order to replace tiles or panels that might fall from the facade. Typically, workers access those tiles or panels via cages or gondolas suspended from the roof of the buildings. This work process was considered by Japanese construction firms as monotonous, inefficient and dangerous. Further, since the method for identifying damaged tiles or panels involves listening to the sound a tile or panel makes when gently impacted by a handheld tool, the sounds were difficult to classify at high elevations due to wind noise. Therefore, with substantial financial commitment, autonomous facade-inspection and maintenance robots were developed. Between 1985 and 1988, six different facade-inspection robots were developed in Japan by the Kajima, Takenaka, Obayashi, Taisei, Tamagawa and Seki Corporations. For the inspection of the facade of a 40 m high building (3000 m^2 facade) an inspection robot needed in average 8 h including ≈1 h for task as preparation, configuration, conversions, dismantling and cleaning of the robot (Fig. 57.24a,b).

Today robotic systems are not only developed for the construction of buildings but also for the operation, maintenance and decommissioning of buildings and other construction products. These types of *service* robots were developed for the construction sector for inspection of nuclear power plants, exterior walls of high-rise buildings and cleaning of high-rise facades or glass roofs (Fig. 57.24c).

57.3.7 Earthmoving

Earthmoving is a construction process for preparing project sites for construction by digging, grading, trenching, scraping and other similar tasks. There have been significant developments in many earthmoving areas and certain tasks (such as grading) can now be completely automated. Figure 57.25 shows several examples of automated earthmoving equipment some of which were still in the research and development phase while other are already commercial products.

57.3.8 Road Maintenance

Figure 57.26 shows three examples of machines that perform pothole repairs and crack sealing on road with various degrees of automation. Figure 57.26a is a commercial product while the other two machines were research projects undertaken by departments of transportation in the US.

57.3.9 Material Handling

The transport and handling of materials is a critical activity at most construction sites. Figure 57.27 shows examples of material handling robots developed under different research efforts.

57.4 Integrated Robotized Construction Sites

The first prototypes for mainly automated high-rise construction sites were put into operation in 1990 and 1991 by Shimizu (Fig. 57.28) after five years in development and a financial outlay of almost sixteen million Euros. Since then, twenty automated high-rise sites have been implemented by different companies (Taisei, Takenaka, Kajima, Maeda, and Kumagai) [57.60, 61].

57.4.1 Robotic Roof Field Factory Approach

An integrated robotized construction site involves the use of semi- and fully-automated storage, transport and assembly equipment and/or robots that are used to erect a building almost completely automatically. It is an attempt to improve the sequencing of construction

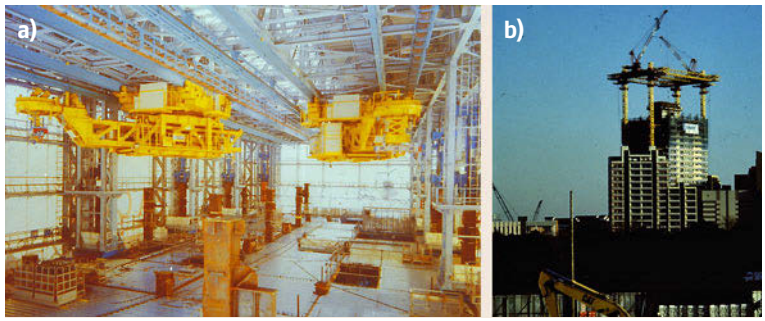


Fig. 57.28 (a) Robotic trolleys for transporting and positioning of beams, columns, floor panels, building services units and facades, in Shimizu SMART system; (b) SMART roof field factory view

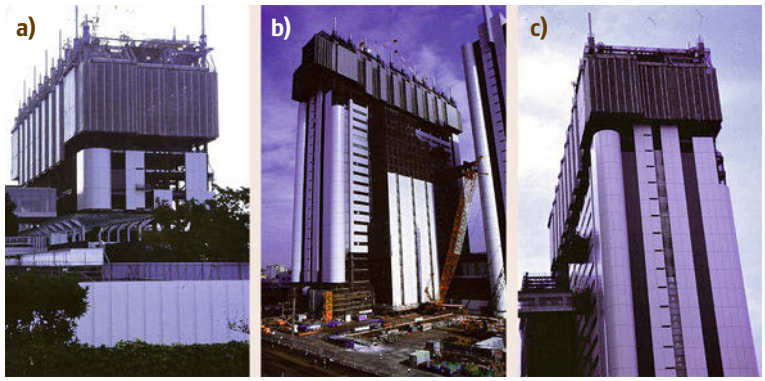


Fig.57.29a–c Obayashi ABCS (a) early construction phase; (b) intermediate construction phase; (c) final construction phase

processes and construction site management by using real-time control. This includes an unbroken flow of information from the planning and design phase through programming the on-site robots and controlling and monitoring the construction operations.

The robotic roof field factory approach is typically implemented once the building foundations have been laid. The production equipment, on which the steel construction has been installed with assembly and transport robots, is then covered completely with a roof made of plastic film. Depending on the roof system used, this process takes from three to six weeks after which the robots go into production. Due to the lack of space around building sites in Japan, steel and concrete plants are also often installed to supply parts in 10 min cycles on a just-in-time basis.

Fig.57.30a,b The AMURAD Construction Strategy: floor-wise upushing of the whole building. Kajima, Japan (a) early construction phase; (b) later construction phase

Prefabricated parts are checked and then placed in specific depots at the foot of the building or in the building itself where they are readily accessible by the robots. This is where the automated construction process actually starts. As many as 22 robots equipped with automatic crane winches deliver the columns, supports, floors, ceilings, walls and other elements to the floor of the steel skeleton under construction. These components are then positioned and fixed into place almost completely automatically. The steel columns and supports are joined together by welding robots after they have been positioned. The position and quality of the welding seams are monitored with lasers.

In the Obayashi ACBS (Automatic Constructions Building System) (Fig. 57.29 and in VIDEO 272), once a storey has been finished, the whole support structure, which rests on four columns, is pushed upwards by hydraulic presses to the next storey over a 1.5 h period. Fully extended, the support structure is 25 m high; retracted it measures 4.5 m. Once everything has been

moved up, work starts on the next storey. By constructing the topmost storey of the high-rise building as the roof at the beginning of the building process, the site is closed off in all directions, considerably reducing the effect of the weather and any damage it might cause.

57.4.2 Robotic Field Factory on-Ground Approach

The Amurad-system (Fig. 57.30), developed by the Kajima Corporation, is a way of construction based on the idea of after the first floor is built (which is the top floor), it is pushed up one floor at a time with the aid of large hydraulic cylinders. The plumbing, electrical and mechanical equipment and the interior fittings and cladding of the facade then begin. This process is repeated until the building is completed, where the last floor to be built becomes the ground floor. In order to realize the AMURAD System three automated systems were developed for pushing up the whole building, for transporting and assembling and for material handling.

57.4.3 Robotic Deconstruction and Robotic Recycling

Controlled deconstruction supported by robotic systems can be combined with component reuse systems since some elements such as structural steel have long life spans, and total recycling (i. e., scrapping, melting and recasting) consumes large amounts of energy. In a component reuse system all structural building components are accepted as trade-ins. Therefore, the deconstruction process comprises a reversed version of the construction process, which if it is based on industrialized fabrication, all deconstructed parts and elements could be directly reintroduced into the fabrication systems. The DARUMA system by Kajima Corporation (Fig. 57.31) enables such a controlled deconstruction process. Deconstructed structural steel elements are transported to special dismantling factories just-in-time

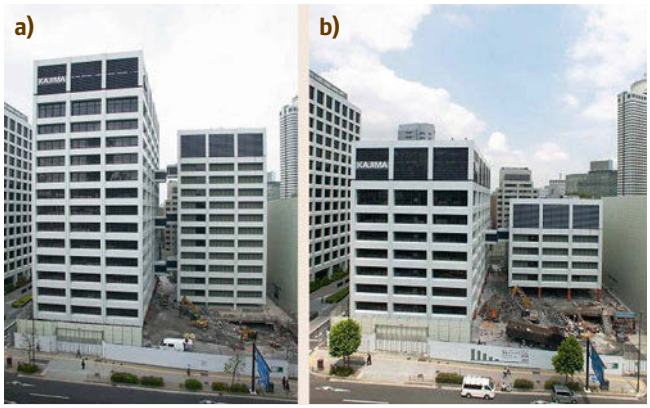


Fig.57.31a,b Systemic and automated floor-by-floor building deconstruction, Kajima, DARUMA System, Japan **(a)** early deconstruction phase; **(b)** later deconstruction phase

and just-in-sequence, where the old joints and finishes can be dismantled under factory conditions and fed into advanced reuse cycles. Modularized and standardized structural units can be inspected and renewed and then equipped with new finishes according to each customer's needs. *Component Reuse* systems could be connected to Advanced Construction ERP Systems and companies could match customers who want to sell their modular building for reuse and customers willing to buy reused building modules for further customization. Renewed building components are reorganized and customized in the factory and transported to other building sites. A combined system of controlled deconstruction and component reuse saves large amounts of materials and energy.

By adopting integrated industrialization processes the construction industry would have the chance to address all parameters relevant for sustainable economic, environmental and social development. Innovative industrialization-oriented architectural design structures, appropriate modularization and standardization of building structures, logistics, equipment and

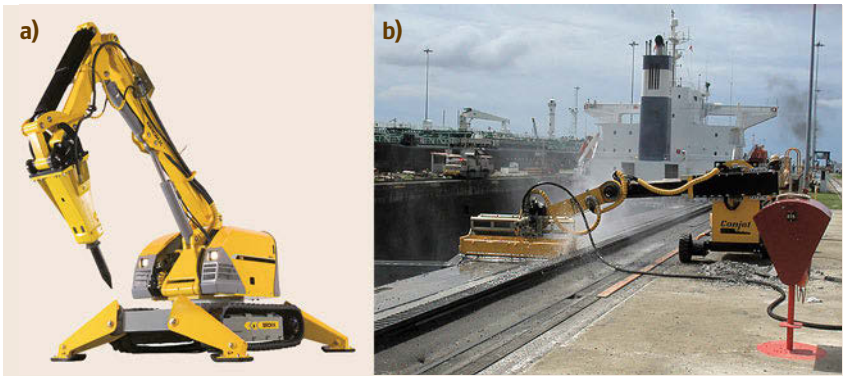


Fig. 57.32 **(a)** Teleoperated demolition robot (courtesy Brokk AB); **(b)** water-jet demolition robot (courtesy Conjet AB)

processes, could serve as a fundamental integration framework. Customized prefabrication could further be able to supply construction sites with individualized elements. Hyper-flexible robotic systems could support a limited amount of trained workers to perform positioning, joining and finishing operations. Construction ERP systems could support organization as well as a lean and demand oriented construction based on just-in-time and just-in-sequence resource supply meanwhile locally based factories grant identity and reduced logistic effort. Moreover integrated industrialization would not only be limited to the fabrication but also link systems of controlled deconstruction and component reuse to a network for continuous resource circulation.

In addition to fully automated deconstruction systems, a number of teleoperated demolition robots have been widely adopted in the construction industry. Two examples of these are shown in Fig. 57.32.

57.5 Currently Unsolved Technical Problems

A recent study found that the US construction industry does not realize approximately 15 billion US dollars per year in potential savings due to inadequate interoperability related to information exchange and management practices [57.62].

57.5.1 Interoperability

Although the lack of interoperability between the various information systems used in construction is a significant source of inefficiency for the industry, it is also a roadblock to the use of automated systems in construction. Automated systems need electronic information on past, current, and/or projected future states of a construction project to function efficiently.

For example, in order for a robotic crane to pick a steel beam from the site and deliver it to its target location, the robot must be able to know that the steel beam has been delivered to the site, as well as its current position and orientation. While information on the current inventory of parts on a site may be available on paper, it is rarely available electronically unless someone enters it manually into some computer system, which in turn may not be compatible with other systems used in that project. In many of the examples of automated construction technologies presented in this chapter, custom electronic databases and/or data formats have to be devised to demonstrate the robot's operation. In some cases, the electronic information had to be entered manually from paper into a computer for the robot to work correctly.

In addition to information exchange and management, many of the measurement instruments and sensors used in construction are not interoperable. This problem is not limited to the construction sector but is a relatively large problem in many robotic applications where different types of sensors are used. Several efforts are underway to make sensors interoperable [57.63] and construction equipment in general [57.64], but the issues have yet to be resolved.

57.5.2 Structural Connection Systems

Traditionally, structural member connections in construction have been designed for human installation. Whether using bolted, welded, or other types of connections, manual labor is usually involved in guiding the mating parts together and in establishing the connection.

For example, in structural steel erection, workers perched on the structure typically guide a crane operator through visual or auditory cues in order to maneuver a steel beam (or column) into place. The workers must then physically manipulate the beam in order to align corresponding surfaces for bolting or welding. Once the correct beam pose has been achieved, it must be maintained while the workers temporarily fasten the beam to the structure. The workers then release the beam from the crane and permanently fasten the beam to the structure at a later stage.

For automated (or robotic) construction to work, new connections that are more amenable to automation

Fig. 57.33 A drop-in, shear-load-only, steel connection

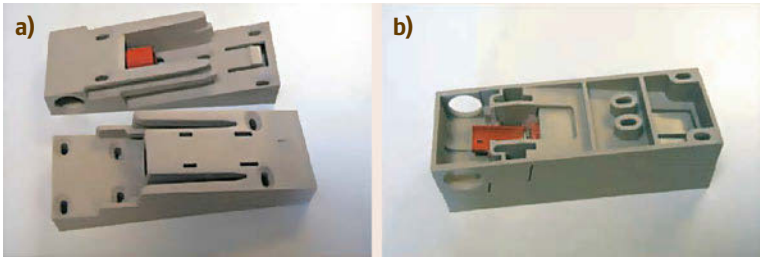


Fig.57.34a,b A piping and electrical drop-in connector shown **(a)** disassembled and **(b)** assembled

must be designed. These connections need not mimic traditional, human-installed connections, but should be optimized for use with robots instead. For example, the Lehigh University Advanced Technology for Large Structural Systems (ATLSS) Center designed a gravity-load-only shear steel connector [57.65] back in the early 1990s that is more suitable for automated construction (Fig. 57.33). This type of male–female connector for automatic assembly of building modules allows, together with an adequate control strategy, small tolerances, which permit assembly by automatic cranes [57.66].

An example of a nonstructural connector that is more amenable to automation is the piping and electrical connector developed for the FutureHome project, shown in Fig. 57.34 [57.21, 67].

Although automated welding has been applied in some limited form in construction, it is generally used to replace manual welding without changing the design of the parts being welded. In other words, no significant change in the way in which construction components are designed has occurred as a result of automated welding. Moreover, apart from a few examples of automated welding applied at the construction site, most of the other limited applications are done at the component fabrication facility.

57.5.3 Tolerances

Specifications for tolerances in the construction sector exist for most types of construction, however, they are not always achieved in practice [57.68]. For example, it has been stated that one of the biggest sources of problems in structural steel erection in the US is that fabricated pieces are often out of tolerance, and that this is only discovered during installation at the construction site [57.69]. However, since the finished facility must meet the design tolerances before it is accepted, this shifts the burden to workers and supervisors during construction. The workers are expected to handle problems as they come up rather than expect all fabricated construction components to be within tolerance. Since most construction projects are under tight schedules, it is often preferable to fix these kinds of problems on site rather than wait for replacement components to be fabricated and delivered.

However, tolerance problems are not all due to fabrication errors. Installation problems are also responsible for out-of-tolerance problems, for example, anchor rod installation has been an area of concern for structural steel erection. (Anchor rods (or bolts) are the connection interface between the concrete foundation and the structural steel columns. The rods are usually installed by the concrete foundation crew before the concrete is dry.) The situation of anchor rod patterns that do not match the hole patterns in a mating column is an identified problem in construction [57.26, 70].

Given the relatively loose achievable tolerances in construction, the application of robotics in construction faces an uphill battle. This, in addition to the unstructured nature of the construction site environment, requires that robots either be highly intelligent in order to correctly interpret and react to their surroundings or to be human assisted. However, site structure and tolerances are expected to improve as pressures to reduce costs and improve productivity continue to rise. Improvements in site organization and construction tolerances have already been proved to be achievable in a few cases, as has been demonstrated in Japan [57.71]. However, the economic case for these demonstration projects has yet to be made.

57.5.4 Power and Communications in the Field

Unlike manufacturing environments, in which specially designed factories are outfitted with the necessary power and communications installations, a construction project often begins before such resources have been installed at the site. Therefore, robots with large power requirements that need to communicate with supervisory systems located off site would be challenging to implement without significant added cost.

Although communications technologies have advanced significantly in the last few decades, it is still considered difficult to maintain a reliable local-area network at a construction site and to connect that network to the Internet. The use of cellular telephones

with a press-to-talk (the digital version of the traditional two-way radio communication method) feature has largely replaced the traditional two-way radios that also revolutionized on-site construction communications. However, in order for construction sites to become more automated, reliable interference-free high-bandwidth networks must be able to carry data transmission between sensors, machines, and supervisory systems.

57.5.5 Sensing

Field measurements are an integral part of the construction process. The *tape measure* and the *transit* have been used in construction for decades for measuring distances and angles, respectively. However, construction measurements are not limited to distances and angles, but can also include measurements of installed quantities, percentages of completion of activities, and so on. All of these measurements are necessary to be able to lay out the site where a facility is to be built, to measure the conformance of the as-built facility to the intended design, and to monitor safety, productivity, and progress.

In the past few decades, more advanced means of making measurements on construction sites have come into play. These include (but are not limited to) total stations, GPS, indoor GPS, ultra-wide band, laser scanners, ground-penetrating radar, equipment and structural health monitoring sensors, concrete maturity meters and radio frequency identification [57.72–84]. These technologies, as well as others that have not yet been adopted in the construction industry, are crucial for enabling more automation and robotics in construction. Nevertheless, despite all of these advances in technology, one of the biggest challenges for implementing robotics at the construction site is to be able to provide accurate and up-to-date information to the robots about where everything is and what other equipment are doing. This challenge will eventually be solved by the introduction of more sophisticated sensing and perception systems.

57.6 Future Directions

As previously discussed, developing new methods for collecting, processing, analyzing, and communicating construction information is a significant area of construction automation research, and will dominate near-term efforts. As this construction information becomes more readily accessible, automation of processes can be enabled directly from the combination of design information and current site status that is accurately captured and shared. Resource tracking will become ubiquitous and just-in-time delivery of needed materials and equipment will happen throughout the site.

Advances in mobility (humanoid robotics, smart cars, legged locomotion, etc.) will enable ever more automated material handling on the job site. These advances will require better control systems for the construction robots that can provide high payload and good positioning accuracy. The increased use of robots at the construction site will also drive research into safety systems for construction robots working around human workers and other machines.

Perhaps most important, more extensive automatic design systems will enable more prefabrication of building components and new methods of assembling those components on site, which in turn will provide the promise of faster, better, and cheaper construction robotics first envisioned in the 1980s.

57.7 Conclusions and Further Reading

The application of robotics to construction has yet to catch up with other industries such as automobile manufacturing. Construction presents a unique challenge for robotic applications. The construction environment is cluttered, unstructured, and teaming with human workers. In addition, construction processes are usually labor intensive and have to accommodate wide margins of error in the constructed facility. The application of robotics in construction to date has been limited to commercial teleoperated and programmable machines. Autonomous or semiautonomous machines are currently mostly limited to research projects within various nonconstruction organizations. With the increase in competition throughout the global construction market, construction companies are on the lookout for ways to improve productivity, quality, and safety. The use of automation and robotics is one answer that the industry is slowly turning toward. However, before these potential solutions can be successfully applied, much work is needed to improve construction tolerances, develop standards, and achieve real-time site status monitoring.

The International Association for Automation and Robotics in Construction holds an annual conference (the International Symposium on Automation and

Robotics in Construction) at which researchers can present the latest developments in the field. The proceedings from this conference hold a wealth of information on the state of the art of the field and are accessible to the general public through IAARC's website (www.iaarc.org).

There are several journals that publish articles on various aspects of automation in construction. Most notable of these are *Automation in Construction*, *Computer-Aided Civil and Infrastructure Engineering*, the *Journal of Computing in Civil Engineering*, and the *Journal of Construction Engineering and Management*. In addition, some journals publish special editions on robotics in construction. Some examples are *Autonomous Robots* and the *Journal of Advanced Robotic Systems*.

Video-References

VIDEO 272 Obayashi ACBS (Automatic Constructions Building System)
available from http://handbookofrobotics.org/view-chapter/57/videodetails/272

References

57.1 E. Ginzberg: The mechanization of work, Sci. Am. **247**(3), 66–75 (1982)

57.2 US Census Bureau: *Value of Construction Put in Place – Seasonally Adjusted Annual Rate* (U.S. Census Bureau, Washington DC 2012), http://www.census.gov/

57.3 Bureau of Labor Statistics: *Industries at a Glance: Construction: NAICS 23* (US Department of Labor, Washington DC 2012), http://www.bls.gov/iag/tgs/iag23.htm

57.4 US Census Bureau: *Statistics of U.S. Businesses* (U.S. Census Bureau, Washington DC 2012), http://www.census.gov/csd/susb/

57.5 European Commission: *Eurostat Regional Yearbook 2012* (Publications Office of the European Union, Luxembourg 2012), http://ec.europa.eu/eurostat/web/products-statistical-books/-/KS-HA-12-001

57.6 Statistics Bureau, Ministry of Internal Affairs and Communications: *Statistical Handbook of Japan* (Statistics Bureau, Tokyo 2012), http://www.stat.go.jp/english/data/handbook/index.htm

57.7 National Bureau of Statistics of China: *The Results of Preliminary Verified GDP for the First Three Quarters in 2012* (National Bureau of Statistics of China, Beijing 2012) http://www.stats.gov.cn/english/pressrelease/201211/t20121102_72217.html

57.8 National Bureau of Statistics of China: *China Statistical Yearbook* (China Statistics Press, Beijing 2012), http://www.stats.gov.cn/tjsj/ndsj/2012/indexeh.htm

57.9 D. Crosthwaite: The global construction market: A cross-sectional analysis, Constr. Manag. Econ. **18**(5), 619–627 (2000)

57.10 D.W. Halpin, R.W. Woodhead: *Construction Management*, 2nd edn. (Wiley, New York 1998)

57.11 K.S. Saidi: *Possible Applications of Handheld Computers to Quantity Surveying*, Dissertation (Univ. Texas, Austin 2002)

57.12 T. Greaves, B. Jenkins: *Capturing Existing Conditions with Terrestrial Laser Scanning: A Report on Opportunities, Challenges and Best Practices for Owners, Operators, Engineering/Construction Contractors and Surveyors of Built Assets and Civil Infrastructure* (Spar Point Research, Danvers 2004)

57.13 J.G. Everett, A.H. Slocum: Automation and robotics opportunities – Construction versus manufacturing, J. Constr. Eng. Manag. ASCE **120**(2), 443–451 (1994)

57.14 L.A. Demsetz: Task identification for construction automation, 6th Int. Symp. Autom. Robotics Constr. (1989) pp. 95–102

57.15 R. Kangari, D.W. Halpin: Potential robotics utilization in construction, J. Constr. Eng. Manag. **115**(1), 126–143 (1989)

57.16 R.L. Tucker: High payoff areas for automation applications, 6th Int. Symp. Autom. Robotics Constr. (1988) pp. 9–16

57.17 L. Cousineau, N. Miura: *Construction Robots: The Search for New Building Technology in Japan* (ASCE, Reston 1998)

57.18 J.G. Everett, H. Saito: Construction automation: Demands and satisfiers in the United States and Japan, J. Constr. Eng. Manag. ASCE **122**(2), 147–151 (1996)

57.19 M. Taylor, S. Wamuziri, I. Smith: Automated construction in Japan, Proc. ICE Civil Eng. **156**(1), 34–41 (2003)

57.20 J. Maeda: Current research and development and approach to future automated construction in Japan, Proc. Constr. Res. Congr. (2005) p. 2403

57.21 C. Balaguer, M. Abderrahim, J.M. Navarro, S. Boudjabeur, P. Aromaa, K. Kahkonen, S. Slavenburg, D. Seward, T. Bock, R. Wing, B. Atkin: FutureHome: An integrated construction automation approach, IEEE Robotics Autom. Mag. **9**(1), 55–66 (2002)

57.22 Y. Maruyama, Y. Iwase, K. Koga, J. Yagi, H. Takada, N. Sunaga, S. Nishigaki, T. Ito, K. Tamaki: Development of virtual and real-field construction management systems in innovative, intelligent field factory, Autom. Constr. **9**(5/6), 503–514 (2000)

57.23 C. Balaguer: Soft robotics concept in construction industry, World Autom. Congr. (2004) pp. 517–522
57.24 K.A. Reed: The role of the CIMSteel integration standards in automating the erection and surveying of structural steelwork, 19th Int. Symp. Autom. Robotics Constr. SP989 (NIST, Gaithersburg 2002)
57.25 N.J. Shih: The application of a 3-D scanner in the representation of building construction site, ISARC 2002: 19th Int. Symp. Autom. Robotics Constr. (2002) pp. 337–342
57.26 B. Akinci, F. Boukamp, C. Gordon, D. Huber, C. Lyons, K. Park: A formalism for utilization of sensor systems and integrated project models for active construction quality control, Autom. Constr. **15**(2), 124–138 (2006)
57.27 G.S. Cheok, W.C. Stone: Non-intrusive scanning technology for construction assessment, IAARC/IFAC/IEEE. Int. Symp. (1999) pp. 645–650
57.28 K. McKinney, M. Fischer: Generating, evaluating and visualizing construction schedules with CAD tools, Autom. Constr. **7**(6), 433–447 (1998)
57.29 B. Akinci, M. Fischer, J. Kunz: Automated generation of work spaces required by construction activities, J. Constr. Eng. Manag. ASCE **128**(4), 306–315 (2002)
57.30 V. Kamat, R. Lipman: Evaluation of standard product models for supporting automated erection of structural steelwork, Autom. Constr. **16**(2), 232–241 (2006)
57.31 C.M. Eastman: *Building Product Models* (CRC, Boca Raton 1999)
57.32 T. Bock, A. Malone: The Integrated Project ManuBuild of the EU, ISARC 2006 23rd Int. Symp. Autom. Robotics Constr. (2006) pp. 361–364
57.33 G. Aouad, J. Kirkham, P. Brandon, F. Brown, G. Cooper, S. Ford, R. Oxman, M. Sarshar, B. Young: Information modeling in the construction industry – The information engineering approach, Constr. Manag. Econ. **11**(5), 384–397 (1993)
57.34 G. Beer: Tunconstruct: A new european initiative, T&T Int. FEV (2006) pp. 21–23
57.35 H.M. Huang: *Autonomy Levels for Unmanned Systems (ALFUS) Framework Volume I: Terminology Version 2.0, NIST Special Publication 1011-I-2.0* (NIST, Gaithersburg 2008), http://www.nist.gov/el/isd/ks/upload/NISTSP_1011-I-2-0.pdf
57.36 Y.F. Ho, H. Masuda, H. Oda, L.W. Stark: Distributed control for tele-operations, IEEE/ASME Trans. Mechatron. **5**(2), 100–109 (2000)
57.37 S. Singh: State of the art in automation of earthmoving, ASCE J. Aerosp. Eng. **10**(4), 179–188 (2002)
57.38 H. Quang, M. Santos, N. Quang, D. Rye, H. Durrant-Whyte: Robotic excavation in construction automation, IEEE Robotics Autom. Mag. **9**(1), 20–28 (2002)
57.39 J. Albus, R. Bostelman, N. Dagalakis: The NIST RoboCrane, J. Robotic Syst. **10**(5), 709–724 (1993)
57.40 K.S. Saidi, A.M. Lytle, W.C. Stone, N.A. Scott: *Developments toward automated construction, NIST Interagency Rep. 7264* (NIST, Gaithersburg 2005)
57.41 S.C. Kang, E. Miranda: Physics based model for simulating the dynamics of tower cranes, 10th Int. Conf. Comput. Civil Build. Eng. (ICCCBE) (2004)
57.42 Weckenmann LLC: Machinery and plant systems for the production of precast concrete elements, http://www.weckenmann.com/en
57.43 M. Damlund, S. Goth, P. Hasle, K. Munk: Low back pain and early retirement among Danish semi-skilled construction workers, Scand. J. Work, Environ. Health **8**(1982), 100–104 (1982)
57.44 S. Schneider, P. Susi: Ergonomics and construction: A review of potential hazards in new construction, Am. Ind. Hyg. Assoc. J. **55**, 635–649 (1994)
57.45 T. Bock: *Robot Oriented Design* (Shokokusha Publishing, Tokyo 1988)
57.46 T. Bock: *A study on Robot-Oriented Construction and Building System, Thesis for Doctorate of Engineering, Report Number 108066* (University of Tokyo, Tokyo 1989)
57.47 T. Bock, T. Linner: *Robot-Oriented Design and Management* (Cambridge Univ. Press, Cambridge 2014)
57.48 T. Bock: The Japanese approach of SMAS-solid material assembly system and the European approach of ROCCO-robotic assembly system for computer integrated construction, EC-Japan Conf. (Reading University, Reading 1995)
57.49 G. Wickström, T. Niskanen, H. Riihimäki: Strain on the back in concrete reinforcement work, Br. J. Ind. Med. **42**(4), 233–239 (1985)
57.50 H. Benckert: Mechydronic for boom control on truck-mounted concrete pumps, Tech. Symp. Constr. Equip. Technol. 2003 (2003)
57.51 F. Gebhart, G. Mayer, F. Ott, A. Barren, B. Heid, W. Schencking, E. Andres Puente, T. Bock, A. Delchambre: Final report of the ROCCO project, ESPRIT III program of the European Union (1998)
57.52 T. Bock: Plenary paper: State of the art of automation and robotics in construction in Germany ROCCO: Robotic assembly system for computer integrated construction, 13th ISARC, Int. Conf. Autom. Robotics Constr., Tokio (1996)
57.53 F. Peyret: The Achievements of the computer integrated road construction project, 17th IAARC/CIB/IEEE/IFAC/IFR Int. Symp. Autom. Robotics Constr. (ISARC) (2000)
57.54 Commonwealth Scientific, Industrial Research Organisation: *Mining Robotics Project* (CSIRO, Clayton South 2006), https://wiki.csiro.au/display/ASL/Dragline+Automation
57.55 D.A. Bradley, D.W. Seward: The development, control and operation of an autonomous robotic excavator, J. Intell. Robotic Syst. **21**(1), 73–97 (1998)
57.56 P. Coal, C. Hughes: *Project C8001: Introduction of Autonomous Haul Trucks. Final Report* (Australian Coal Research, Brisbane 1997)
57.57 C. Haas, K. Saidi, Y. Cho, W. Fagerlund, H. Kim, Y. Kim: *Implementation of an Automated Road Maintenance Machine (ARMM), Center for Transportation Research, Project Summary Report* (NIST Interagency Rep., 7264 2005)
57.58 D.A. Bennett, X. Feng, S.A. Velinsky: AHMCT automated crack sealing program and the operator controlled crack sealing machine, Transp. Res. Board Annu. Meet. (2003)

57.59 A.M. Lytle, K.S. Saidi: NIST research in autonomous construction, Auton. Robots **22**(3), 211–221 (2007)
57.60 T. Linner: Automated and Robotic Construction: Integrated Automated Construction Sites, Dissertation (Universität München, München 2013)
57.61 T. Bock, T. Linner: *Logistics, Site Automation and Robotics: Automated/Robotic On-site Factories* (Cambridge Univ. Press, Cambridge 2014)
57.62 M.P. Gallaher, R.E. Chapman: *Cost Analysis of Inadequate Interoperability in the US Capital Facilities Industry* (National Institute of Standards and Technology, Gaithersburg 2004), US Dept. of Commerce, Technology Administration
57.63 K.B. Lee, M.E. Reichardt: Open standards for homeland security sensor networks, Instrum. Meas. Mag. IEEE **8**(5), 14–21 (2005)
57.64 E.F. Begley, M.E. Palmer, K.A. Reed: *Semantic Mapping Between IAI ifcXML and FIATECH AEX Models for Centrifugal Pumps* (National Institute of Standards and Technology, Gaithersburg 2005)
57.65 R. Fleischman, B.V. Viscomi, L.W. Lu: Development, analysis and experimentation of ATLSS connections for automated construction, Proc. 1st World Conf. Steel Struct. (1992)
57.66 S. Garrido, M. Abderrahim, A. Gimenez, C. Balaguer: Anti-swinging input shaping control of an automatic construction crane, IEEE Trans. Autom. Sci. Eng. **5**(3), 549–557 (2007)
57.67 T. Bock: Montage und Demontage im Holzbau mittels Schnellverschlüssen, BMBF Projektnummer: 0339835/5
57.68 J.K. Latta: *Inaccuracies in Construction*, Canadian Building Digest 171 (Institute for Construction, National Research Council Canada, Ottawa 1975), http://web.mit.edu/parmstr/Public/NRCan/CanBldgDigests/cbd171_e.html
57.69 A.M. Lytle, K.S. Saidi (Eds.): *Proceedings of the 23rd ISARC* (International Association for Automation and Robotics in Construction, Tokyo 2006)
57.70 A.M. Lytle, K.S. Saidi (Eds.): *Automated Steel Construction Workshop 2002* (National Institute of Standards and Technology, Gaithersburg 2004)
57.71 Y. Miyatake: SMART system: A full-scale implementation of computer integrated construction, 10th Int. Symp. Autom. Robotics Constr. (1993)
57.72 C. Lindfors, P. Chang, W. Stone: Survey of construction metrology options for AEC industry, J. Aerosp. Eng. **12**, 58 (1999)
57.73 S. Kang, D. Tesar: A novel 6-DoF measurement tool with indoor GPS for metrology and calibration of modular reconfigurable robots, IEEE ICM Int. Conf. Mechatron., Istanbul (2004)
57.74 L.E. Bernold, L. Venkatesan, S. Suvarna: Equipment mounted multi-sensory system to locate pipes, Pipelines **130**, 112 (2004)
57.75 D.A. Willett, K.C. Mahboub, B. Rister: Accuracy of ground-penetrating radar for pavement-layer thickness analysis, J. Transp. Eng. **132**, 96–103 (2006)
57.76 C.L. Barnes, J.F. Trottier: Effectiveness of ground penetrating radar in predicting deck repair quantities, J. Infrastruct. Syst. **10**, 69 (2004)
57.77 J.A. Huisman, S.S. Hubbard, J.D. Redman, A.P. Annan: Measuring soil water content with ground penetrating radar – A review, Vadose Zone J. **2**(4), 476–491 (2003)
57.78 G.W. Housner, L.A. Bergman, T.K. Caughey, A.G. Chassiakos, R.O. Claus, S.F. Masri, R.E. Skelton, T.T. Soong, B.F. Spencer, J.T.P. Yao: Structural control: Past, present, and future, J. Eng. Mech. **123**(9), 897–971 (1997)
57.79 US Department of Transportation: *Maturity Meters: A Concrete Success*, ed. by L. Pope (Federal Highway Administration (FHWA), Washington 2002)
57.80 S.V. Ramaiah, B.F. McCullough, T. Dossey: *Estimating in situ Strength of Concrete Pavements Under Various Field Conditions* (Univ. of Texas, Austin 2001), Center Transport. Res.
57.81 J. Song, C. Haas, C. Caldas, E. Ergen, B. Akinci, C.R. Wood, J. Wadephul: *Field Trials of RFID Technology for Tracking Fabricated Pipe – Phase II* (FIATECH, Austin 2003), http://www.fiatech.org/images/stories/techprojects/project_deliverables/SC_FieldTrialsofRFIDTechnologyforTrackingFabricatedPipe_PhaseII.pdf
57.82 J. Song, C.T. Haas, C. Caldas, E. Ergen, B. Akinci: Automating the task of tracking the delivery and receipt of fabricated pipe spools in industrial projects, Autom. Constr. **15**(2), 166–177 (2006)
57.83 J. Aksoy, I. Chan, K. Guidry, J. Jones, C. R. Wood: *Materials and Asset Tracking Using RFID: A Preparatory Field Pilot Study* (FIATECH, Austin 2004), http://www.fiatech.org
57.84 J. Kang, P. Woods, J. Nam, C.R. Wood: *Field Tests of RFID Technology for Construction Tool Management* (FIATECH, Austin 2005), http://www.fiatech.org

Multimedia Contents

58. Robotics in Hazardous Applications

James Trevelyan, William R. Hamel, Sung-Chul Kang

Robotics researchers have worked hard to realize a long-awaited vision: machines that can eliminate the need for people to work in hazardous environments. Chapter 60 is framed by the vision of disaster response: search and rescue robots carrying people from burning buildings or tunneling through collapsed rock falls to reach trapped miners. In this chapter we review tangible progress towards robots that perform routine work in places too dangerous for humans. Researchers still have many challenges ahead of them but there has been remarkable progress in some areas. Hazardous environments present special challenges for the accomplishment of desired tasks depending on the nature and magnitude of the hazards. Hazards may be present in the form of radiation, toxic contamination, falling objects or potential explosions. Technology that specialized engineering companies can develop and sell without active help from researchers marks the frontier of commercial feasibility. Just inside this border lie teleoperated robots for explosive ordnance disposal (EOD) and for underwater engineering work. Even with the typical tenfold disadvantage in manipulation performance imposed by the limits of today's *telepresence* and teleoperation technology, in terms of human dexterity and speed, robots often can offer a more cost-effective solution. However, most routine applications in hazardous environments still lie far beyond the feasibility frontier. Fire fighting, remediating nuclear contamination, reactor decommissioning, tunneling, underwater engineering, underground mining and clearance of landmines and unexploded ordnance still present many unsolved problems.

58.1 **Operation in Hazardous Environments: The Need for a Robotics Solution** 1521
58.2 **Applications** 1523
58.2.1 Eradicating Landmines 1524
58.2.2 Hazardous Materials Handling and Operations 1530
58.2.3 Lessons Learned from the Fukushima Daiichi Disaster 1536
58.3 **Enabling Technologies** 1537
58.3.1 Mobility Issues 1537
58.3.2 Manipulator Design and Control for Hazardous Object Handling ... 1539
58.3.3 Control for Hazardous Tasks 1540
58.3.4 Data Communications 1542
58.3.5 Energetics 1542
58.3.6 System Architectures for Real-Time Mission Control 1543
58.4 **Conclusions and Further Reading** 1544
Video-References 1545
References 1546

58.1 Operation in Hazardous Environments: The Need for a Robotics Solution

Routine work in hazardous environments presents many challenges for robotic devices that depend on both the tasks and the hazards. Hazards may be present in the form of temperature extremes, radiation, toxic fumes and materials, potential explosions, risk of electrocution, the absence of breathable air in space, and the high pressure of underwater environments. When the magnitudes of one or more hazards reach the point

that human exposure would either represent a direct threat to life or unacceptable long-term health consequences, some form of remote operations that separate humans from the hazards must be employed. A long-extant example of such operations is the nuclear industry where telerobotics has become a well-established technique for operations in environments with high levels of nuclear radiation. Many of the technical roots of modern robotics technology can be traced back to nuclear remote handling manipulators and support systems. Remote handling and operations concepts using engineered systems that allow humans to perform hazardous work while staying in a safe environment have evolved since the 1940s. Today, such remotely operated systems are widely used and recently they have become routinely used in explosives disposal, security operations, handling of dangerous biological materials, and routine inspection of industrial systems such as power lines, substations, sewers, piping systems, pressure vessels and many others.

A remote handling system will generally involve subsystems for mobility, manipulation, tooling, sensing, and human–machine interfacing (Fig. 58.1). Nominal operations involve the collective workings of these subsystems to accomplish remote operational goals. Any remote handling system will eventually experience some aspect of off-nominal operation that may be the result of unexpected environmental events or system malfunctions. The fundamental idea is to connect the human operator to the remote environment via a power and signal infrastructure that allows effective operation in the remote environment: mechanized devices and sensor systems that allow the human's perception and action capabilities to be *projected* into the hazardous environment to perform remote operations. The more realistic this projection, the more natural and effective the human remote control will be.

Because of their inherent complexity and the very nature of remote operations, operator training is a major challenge that requires the use of simulations and cold testing facilities that provide operators with comprehensive and realistic training. Such training will typically encompass all aspects of nominal and anticipated off-nominal operations.

Remote handling systems themselves will eventually experience equipment failures. Remote recovery/maintenance/operation of failed remote systems must be an integral part of their basic design and operational features. Hardware/software features must be provided for the analysis of, recovery from, and correction of problems.

Robotics researchers have worked hard to realize a long-awaited vision: machines that enable people to perform routine work without exposure to hazards. In this chapter we review progress and some of the current and future challenges that lie ahead for researchers.

Technology that specialized engineering companies can develop and sell without active help from researchers marks the frontier of commercial feasibility. Just inside this border lie teleoperated robots for explosive ordnance disposal (EOD) [58.1, 2] and for underwater engineering work. Even with the typical tenfold disadvantage in manipulation performance imposed by the limits of today's *telepresence* and teleoperation technology, compared with human dexterity and speed, robots can often offer a more cost-effective solution. Naturally, if people need to wear protective suits to protect them from hazards, their endurance and dexterity will be restricted.

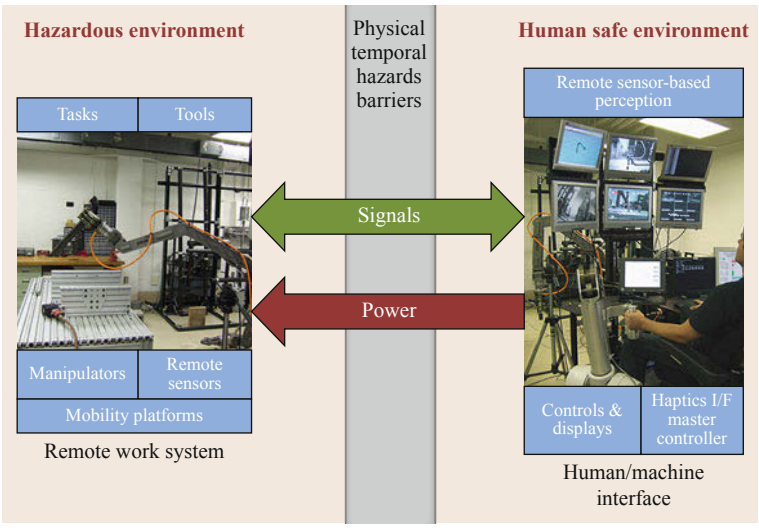


Fig. 58.1 Basic subsystems of a remote handling system

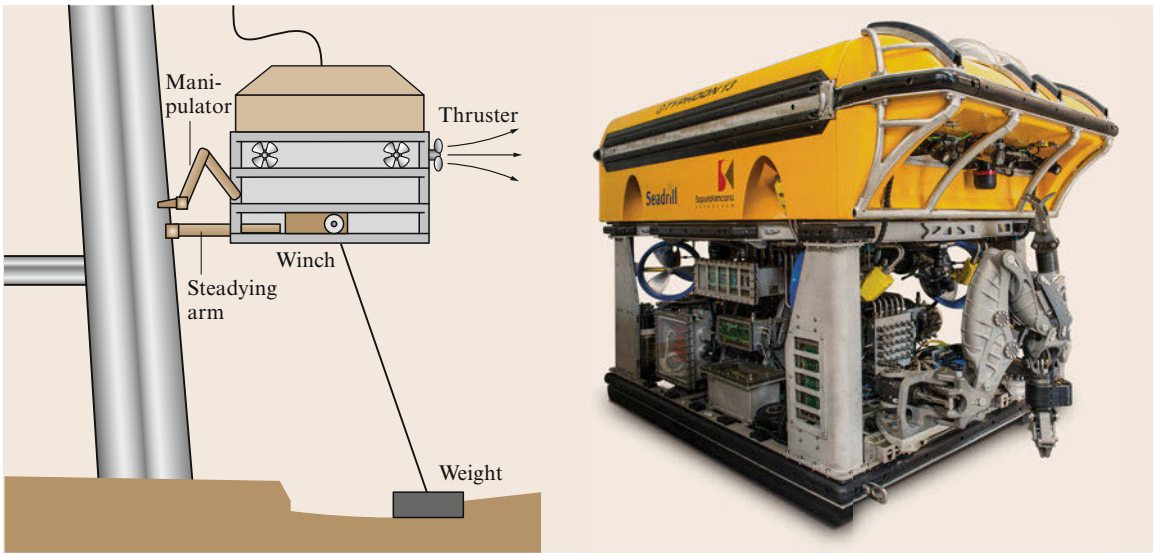


Fig. 58.2 Remotely operated vehicles designed for underwater engineering work, demonstrating an ingeniously simple technical arrangement compared with conventional free-swimming vehicles. (courtesy of Total Marine Technology, Australia)

Ingenuity stimulated by commercial incentives can greatly reduce the operating costs of remotely operated equipment. Figure 58.2 illustrates the elegant simplicity of the NOMAD remotely operated vehicle (ROV) that pulls itself deeper with a simple electric winch. It can be left in place overnight or during rough weather. Free-swimming robots are much more complicated and expensive and have to be hauled out of the water when not under active control.

Limited autonomy or autonomous operation for a restricted time can relieve operator fatigue and allows unmanned aerial vehicles (UAVs, often referred to as drones) to fly extended reconnaissance missions with occasional precision weapon delivery. Such missions would be too hazardous or too politically sensitive for manned aircraft.

Even after the first decade of the 21st century, with 60 years of remote operating experience, most hazardous applications still lie far beyond the frontier of commercial feasibility. Fire fighting, remediating high-level nuclear contamination [58.3], reactor decommissioning, tunneling, underground mining, and most landmine and unexploded ordnance problems still present many research challenges. Most attempts to use first-generation mine rescue robots reported in the press in 2000 and 2001 have merely created distractions for the people who regularly have to risk their lives saving their trapped colleagues (e.g., the Australian Numbat robot [58.4]). The Fukushima Daiichi nuclear reactor disaster triggered by the March 2011 tsunami that devastated parts of Japan serves as a reality check on our capacity to use robots effectively for hazardous tasks. Beyond the carefully engineered environments that characterize the nuclear material handling industry, the disappointing performance of several robots has demonstrated just how far we still have to go.

58.2 Applications

Potential applications of robotic systems in hazardous environments encompass an extremely wide spectrum, limited only by human imagination. The solutions for these different environments are equally diverse. In general such applications involve unique challenges associated with the uncertainty and unstructured nature of the associated tasks. In this discussion, two application areas that are very different have been selected to give the reader a deeper sense of technology evolution, accomplishments, and remaining challenges. The first application we will discuss is landmine eradication, sometimes referred to as demining, a current application domain with humanitarian importance and extremely difficult and hazardous outdoor conditions. The second application area is hazardous nuclear and biological material handling, a decades-old industry

that has strongly influenced many aspects of robot manipulation and mobility research and development. We conclude this section with a brief discussion on the lessons learned from the Fukushima Daiichi disaster, lessons that help to strengthen our argument that we are still at a very early stage of development in robotics for routine operations in hazardous environments.

58.2.1 Eradicating Landmines

Efforts to construct a practical robotic device to help with landmine clearance have met with only limited success. While remote control technology has enabled some existing machines and vehicles to be used in hazardous situations, we are still a long way from achieving reliable robotic mine clearance. It is instructive to understand why many expectations turned out to be hopelessly optimistic.

Landmines, a simple type of victim-activated explosive device, were used extensively in Europe and North Africa during the Second World War between 1939 and 1945. Extensive clearance operations in 1945 and 1946 removed nearly all of the landmines then in use [58.5, part 1, pp. 15–25].

Landmines were used extensively in subsequent decades. Along with antipersonnel cluster bombs, they caused extensive civilian and military casualties in Vietnam and Cambodia from the 1960s onwards. However, it was not until their widespread use in Afghanistan, Angola, Cambodia, and several other countries in the 1980s that they were recognized as a major humanitarian problem. Landmines blocked aid efforts needed to rebuild communities following civil conflicts.

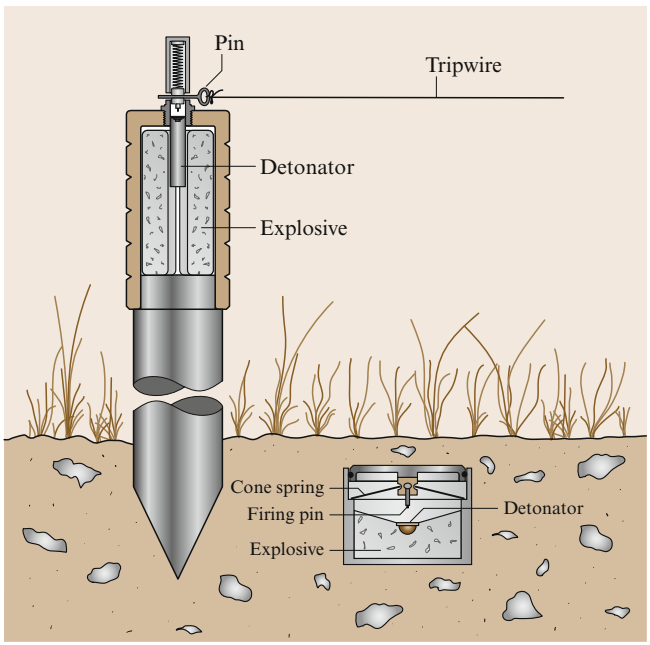


Fig. 58.3 Antipersonnel fragmentation mine (*left*) and buried blast mine (*right*)

The Red Cross and the International Campaign to Ban Landmines (ICBL) successfully promoted a ban on the use of landmines that came into effect in 1997 as the Ottawa Treaty. The knowledge that thousands of children were losing their legs, even their lives, motivated hundreds of researchers to develop new technologies to help eliminate this threat. By 2000 ICBL estimated that over 80 countries were affected by landmines and other explosive remnants of war such as cluster bombs. Although several countries that have not signed the Ottawa Treaty still have extensive stocks of landmines, the treaty has been effective in restricting the use of landmines more through peer pressure. A few countries and several non-state actors continue to deploy landmines. However, other explosive remnants of war such as cluster bombs and other munitions have become, yet again, an increasing problem in Iraq, Afghanistan, Libya and Syria.

There are several basic types of landmines and unexploded ordnance (UXO) that continue to cause problems in many countries.

Antipersonnel (AP) blast mines, made predominantly of plastic with small metal firing pins and detonator cases, typically contain between 20 and 100 g of explosive (Fig. 58.3). These mines only cause extensive injuries when they detonate within a few centimetres of a person. Typically they lie buried just below the ground surface and are activated when the victim steps on top of the mine. Shattered fragments of bone pass through the flesh of the leg at high velocity. If the victim survives long enough to reach hospital, amputation above or below the knee usually saves his or her life but the victim will need prosthetic legs, replaced at regular intervals.

Antipersonnel fragmentation mines contain similar quantities of explosive with a thick metal case that breaks into high-velocity fragments when the mine explodes just above the ground surface. Older fragmentation mines were mounted on posts; some varieties lie buried but jump into the air when activated and explode at waist height, killing or seriously wounding victims up to 200 m away. These mines are much easier to detect so they are often protected by nearby antipersonnel blast mines to deter theft.

Anti-vehicle (AV) or anti-tank mines (ATs), made predominantly of plastic with small metal firing pins and detonator cases, are large versions of the AP blast mine and typically contain 5–10 kg of explosive. Some have a thick metal plate on top that can penetrate 50 cm of armor plate on the underside of a tank. These mines cause significant damage even to mine-resistant vehi-

cles which have blast-resistant hulls, offset wheels, and additional protection for occupants.

Air-dispersed munitions such as cluster bombs (CBs) were not intended to be victim-activated. Several hundred are released at one time from a single canister and they are designed to explode on impact with the ground. Typically between 5 and 25% fail to explode immediately and lie in a partially triggered state either on or just below the ground surface. Some will detonate in response to electromagnetic fields from metal detectors, and others will detonate with the slightest movement. Most explode like powerful fragmentation mines with a lethal radius of up to 200 m.

Improvised explosive devices (IEDs), often in the form of roadside bombs, are increasingly used by insurgent groups fighting organized military forces (asymmetric warfare). They are often made from large UXOs fitted with remote controlled detonators VIDEO 572. Ironically the UXO is often unintentionally donated by the same organized military forces who become the targets of these devices. IED detection and clearance has become one of the highest priorities for military robotics research.

Evolution of Landmine Clearance Techniques

Removing landmines is usually both laborious and hazardous. It is important to distinguish between humanitarian mine clearance and military mine clearance methods (sometimes called *breeching*). Military mine clearance has to work fast, in all conditions (even under fire), and therefore it is unrealistic to aim for 100% clearance. In humanitarian operations there is less time pressure and work can be suspended in unfavorable conditions, and the aim is 100% clearance. Since the 1990s, the expectation in industrialized countries of low casualties from military operations often demands very high clearance standards even for urgent military operations.

Humanitarian mine clearance typically starts years, perhaps decades, after the mines were laid. The mines lie buried or hidden from view. They deter people from entering the land so vegetation often grows thickly (Fig. 58.4). Drainage systems rapidly become clogged, denying access in wet conditions.

The traditional *manual* method for removing landmines has been to use a metal detector to locate metal fragments close to the ground surface and then to carefully check each metal fragment to see if it is associated with a mine or explosive device VIDEO 571. Any tripwires and vegetation have to be removed, with great care, before a metal detector can be used. In many areas deminers have to investigate hundreds or thousands of metal fragments for every mine found. Manual mine clearance also requires careful organization and

Fig. 58.4 Typical ruined house overgrown by vegetation in a village in northern Croatia, possibly containing mines or booby traps. The entire village population was forced to leave in 1991 and the houses were looted and intentionally severely damaged. Vegetation problems like this must be taken into account in considering practical mine and UXO clearance devices. August 1999 (courtesy of J. Trevelyan)

marking of the ground to ensure safety and thorough clearance. Currently it is still the method that guarantees the lowest risk of residual mine contamination but it is expensive, typically costing US$ 1–5 /m^2.

Armored mine clearance machines using hammers mounted on the end of rapidly spinning chains (flails) first appeared in the 1940s but have not been able to neutralize mines with sufficient reliability for most humanitarian applications [58.6].

In the late 1990s commercial mine clearance organizations operating in thick vegetation in Bosnia Herzegovina and Croatia realized that flails spinning just above the ground could rapidly remove vegetation and trip wires to prepare the ground for manual clearance, often assisted by mine detection dogs. Clearance costs have been reduced by up to 80% (particularly in thick vegetation) using different combinations of machines, detection dogs, and manual clearance.

Ground milling machines use metal drums studded with hard cutters that shred buried objects. They require more power than flails but can operate with greater levels of reliability (Fig. 58.5). Both flails and ground milling machines have been extensively used in Croatia to recover large areas of formerly productive agricultural land. Both kinds of machines can withstand a limited number of AT and moderate-size UXO explosions before main bearings and other components need to be replaced VIDEO 574, VIDEO 575, VIDEO 576, VIDEO 577. Several of these machines have remote operation capabilities (Fig. 58.6).

Naturally, machines operate best on flat or gently sloping ground, which is also the land that is most valuable for agriculture and human habitation. Thick forest

Fig. 58.5 Flail machine using hammers on the ends of spinning chains to clear vegetation and tripwires. This machine will also detonate a proportion of buried mines (*inset*) (courtesy of Scanjack AB, Sweden)

Fig. 58.6 Bozena teleoperated demining vehicle (courtesy of Way Industry, Slovakia) VIDEO 574

and mountainous terrain still requires traditional manual clearance and, in most countries, will not be cleared of mines for a long time, if ever.

Mechanized clearance methods continue to evolve with improvements to machines and techniques. Machines can be used for survey, risk assessment, and risk reduction tasks to help determine the need for more expensive manual clearance methods. Mine action programs are gradually shifting from an emphasis on total clearance in the 1990s to one of progressive prioritized risk reduction involving a series of measures including high-security fences, mechanized survey and risk reduction methods, and selective manual clearance [58.5, part 4]. Protective measures applied to agricultural machinery offer cheaper alternatives in low-AT-risk areas [58.7].

Evolution of Demining Research Priorities

Technological development in landmine clearance from within the demining community has mainly been driven by the search for improved productivity. Many of the comments in this section are based on numerous discussions with experienced demining personnel who have tried new technologies in the field. References have been cited where further detailed written information is available.

In the mid-1990s there was the expectation that, with sufficient research, advanced technology detectors could replace eddy-current metal detector technology that had been in use since the 1940s. Metal detectors also react to metal fragments in the ground. A detector that could confirm the presence of explosive, it was thought, would save having to investigate all these false alarms. The most promising line of research seemed to be data fusion: combining signals from a metal detector, ground-penetrating radar, infrared detectors, thermal neutron detectors, and even acoustic detectors. Astute observers at research conferences have pointed out that these signals are often well correlated, even in the presence of false alarms: producing a reliable detector was going to be hard work. Their forecasts turned out to be very accurate. Only one such detector is currently in operation: the handheld standoff mine detection system (HSTAMIDS) detector used by US military forces in Afghanistan employs a combination of ground-penetrating radar and eddy-current metal detection. Little information on its effectiveness has been released and no independent trials have been reported.

Experienced research groups report that ground-penetrating radar requires accurate alignment of the detector with the ground surface (to eliminate ground surface returns) and also with the target center point to enable the target to be characterized reliably. If the principal metal component of the landmine coincides with its geometric center, a common feature of minimum metal mines, the metal detector can be used for alignment. However this is not always the case and one cannot guarantee the absence of other metal fragments near the mine. Ground-penetrating radar provides confusing returns in very dry or very wet conditions and is also susceptible to false alarm indications from underground discontinuities such as stones, sticks, animal burrows etc. Publishers mostly downplay these difficulties and prefer only to report positive results, not false negatives. These issues only emerge from discussions with developers who have seriously evaluated technology in field conditions.

The major performance improvements in sensing have been obtained by compensating eddy-current metal detectors for soil magnetization, enabling them to work in a much wider range of soil conditions. Improvements in sensitivity can help with minimum metal mines but can also result in a large number of false alarms from smaller metal fragments. Metal detector arrays have been fitted to vehicles to speed up clearance of paved areas and roads [58.8].

By the late 1990s slow progress with sensors had become more apparent and research priorities after 2000 gradually turned to mine detection dogs and large demining machines.

The Afghanistan Mine Action Center (MACA) started using mine detection dogs around 1993 but it was not until 1998 that this program was running effectively. There were several difficulties. The first challenge was that close association between humans and dogs was socially unacceptable in Afghanistan. The second challenge was to devise ways to use dogs and manual mine clearance in an effective combination providing reliable clearance with high productivity. This was much the greater challenge, but by 1998 the cost of clearance using dogs was around one-third the cost of manual clearance. It was then that the problems started to appear: the occasional missed mine that could not be explained by lack of organization or failure to follow procedures. At the same time, carefully controlled trials of mine detection dogs in Bosnia had returned highly variable results. On several occasions dogs had walked past blocks of trinitrotoluene (TNT) lying almost visible in the ground. Yet, at the same time, a number of commercial demining agencies were routinely declaring land free of mines using similar dogs. In late 1999 the Bosnian Mine Action Center ran a carefully controlled test in which around 80% of the dogs failed to achieve the required performance standard. The results were hotly contested at the time and the international community organized a systematic trial of mine detection dogs through the Geneva International Centre for Humanitarian Demining (GICHD).

By 2001 it was apparent that there had been little scientific research on the fundamental physiological mechanisms that enable dogs to locate sources of explosive vapor. Dogs had been able to find mines using explosives (such as high melting point explosives like HMX) with vapor pressure far below measurable detection thresholds. The mechanism by which TNT vapor and its breakdown products reach the ground surface was the subject of considerable scientific debate. By 2003 a systematic trial in Afghanistan, scientific studies at SANDIA Laboratories in the USA and in Scandinavia, explosive trace detection studies with dogs at Auburn University, and several other investigations provided some insight into this problem for the first time [58.9]. However, the precise physiological mechanisms for canine explosive detection remain unclear, especially for lower-vapor-pressure explosives. We still do not know for sure whether dogs are reacting to vapor, minute particles of explosive suspended in the air, biochemical breakdown products, or a combination.

In 2003 a US company, NOMADICS, demonstrated the FIDO detector, the first that could reliably measure the presence of TNT vapor with greater sensitivity than a highly trained dog. However field trials showed that TNT vapor could be detected everywhere in a mine contaminated area. An explosive vapor sensor was just the beginning of the story and warns of a complex cognitive task ahead in using the patterns of variation to localize targets.

By 2004 the international community realized that the early confidence in a breakthrough resulting from advanced sensor technology, demining machinery, and mine detection dogs had been misplaced. GICHD commissioned a detailed study of manual demining to see whether productivity improvements could be made. A systematic series of trials were conducted in Africa to determine the effectiveness of several innovations such as magnets and rakes. The final report, issued in 2005, revealed that greatly improved productivity was possible but would depend more on improving contracting arrangements, management, and training than technology.

The New York attacks in September 2001 fundamentally changed research priorities. Removing unexploded ordnance, particularly cluster bombs, became the top priority for the next 12 months in Afghanistan

(Fig. 58.7). Since then the deteriorating security and political situation in Afghanistan has focused mine clearance agencies more on maintaining security for their own workforce than trying to improve productivity and safety.

Resistance to the US and international occupation of Iraq and the easy availability of explosives both from former Iraqi armed forces and unexploded ordnance from US military operations led to the proliferation of improvised explosive devices (IED) to attack organized military forces and police. These have now become the main threat and the focus for much of the funding, and operational and research expertise formerly available to support mine clearance operations. This development has also placed ordnance disposal teams at the front line for the first time, rather than working in well-protected secure areas. Iraqi insurgent groups attack ordnance disposal teams both because they are attempting to disarm some of the insurgents' most effective weapons and also because they remove the main sources of explosives available to insurgent groups.

Improvised explosive devices, when detected, are often investigated and neutralized using remotely operated robots. While there are non-destructive methods to neutralize IEDs, the fastest method usually involves placing a small shaped charge device close to, and accurately aligned with the device. Operational details remain confidential to reduce the risk that IEDs will be modified to defeat current neutralization methods.

Paradoxically it is this development that has enabled robotics to make a greater contribution to the problem by through improvements in remote manipulation technology. These improvements come more in the form of low-cost commercial off-the-shelf components (mobile platform, motors, TV cameras etc.) than from fundamental research advances. Improvements are still being made: improved remote manipulation, blast survivability, operator interface improvements, and mobility improvements have all contributed significantly to performance and reduced operating costs.

Fig. 58.7 Unexploded BLU-97 cluster bombs in Afghanistan early in 2002. Two of the small yellow canisters with parachutes still attached lie visible in the foreground. Others lie in the houses in the distance. Some may lie up to 40 cm below the ground surface. Some can detonate when a metal detector or mobile phone is used nearby. These devices have a kill radius of 200 m and can sometimes be set off by a strong gust of wind. They readily attract the curiosity of children (courtesy of G. Zahachewsky and N. Spencer)

Advances in Demining Robotics Research

In a brief survey it is not possible to mention every contribution. We have attempted to provide a sample of research reports that illustrate the main achievements and we present brief technical discussions on mobility and manipulation dexterity.

Teleoperation remains the only robotics technology that has been used in practical application in field conditions [58.10–12], even with devices such as iRobot and machines such as MineWolf. Robotics research has not yet been able to make a significant contribution to mine and unexploded clearance work. However, the problems posed by landmine clearance have stimulated new research results that could have other applications.

Robotics researchers started their efforts in the early 1990s, for example, Havlik and *Trevelyan* independently proposed suspended cable robots to work in minefields [58.13]. However, both have later argued in favor of alternative solutions [58.10, 14].

Nicoud [58.15] wrote one of the first surveys exploring the possibility of using robotics technology for landmine clearance. Developments in robotics research since the mid 1990s have been motivated both by a genuine desire to help combat a serious humanitarian issue and also by a desire to find a justification for more fundamental research. However, most researchers still have not learned lessons from the field such as the need to remove vegetation and the variety of situations in which deminers find themselves [58.16].

One of the most prolific research areas justified in part by humanitarian demining problems has been path planning for autonomous agents [58.17]. Probabilistic approaches were explored by [58.18]. Some researchers have proposed multiple robot solutions, even *swarms* of robots [58.19] and so-called *immune systems* [58.20]. Autonomous search and mapping algorithms have also been explored [58.21], including even three-dimensional (3-D) search techniques for locating underground chemical sources [58.22].

Landmine clearance has also stimulated developments in autonomous robot vehicles, with many examples in the literature [58.23]. Tracked vehicles have been proposed for working in rubble and built environments [58.24]. Walking vehicles have been proposed,

particularly for difficult terrain in countries like Bosnia and Afghanistan [58.25] even to the extent of examining how damaged robots with missing legs could extricate themselves from a mined area [58.26].

The desire to keep human operators away from the risk of handling unexploded ordnance has stimulated research on artificial hands and telemanipulation [58.27]. Purely mechanical devices have also been explored [58.28]. Several researchers have provided detailed results of tests with manipulators mounted on autonomous vehicles [58.29] (Fig. 58.8).

Robotics solutions have also been proposed partly to overcome the limitations of handheld sensors such as ground-penetrating radar. A robot manipulator can control the motion of the sensor so much more precisely, opening the possibility of synthetic aperture techniques for both metal detection and radar [58.30].

Military research agencies in the United States, Australia, Britain, and Canada devoted large research budgets to the problem of road clearance since AT mines and IEDs accounted for a large proportion of military casualties in Somalia, Bosnia, Iraq and Afghanistan. Insurgent forces had demonstrated that they could bring organized military forces to a complete standstill overnight simply by laying anti-vehicle mines in a few road potholes. With most roads in poor condition there was no easy way to detect that the mines had been laid. Insurgents would typically use mines to stop leading vehicles in a convoy in order to increase the effectiveness of an ambush.

Most research teams proposed one or more vehicles carrying multiple sensors including ground-penetrating radar, metal detector arrays, passive and active infrared, and even some acoustic arrays. Some sensor arrangements were designed to look forward sufficiently far to allow the carrier vehicle to stop before reaching a mine. Others were to be carried by lightweight remotely controlled lead vehicles. Military planners calculated that search speeds of $\approx 30\,\mathrm{km/h}$ would be required to be able to check roads daily in time for supply convoys to use the roads in daylight hours. Many different arrangements have been reported [58.31]. Other teams proposed teleoperated devices for landmine and explosive ordnance detection and neutralization but few anticipated the requirement to operate under armed attack. Typical requirements envisaged secure rear area mine clearance in combat situations and peace-keeping on roads and tracks [58.32] or force protection roles [58.33]. Most of these efforts were initiated in the late 1990s but by 2005 it had become apparent to military planners that vehicle protection rather than mine detection was a more practical solution. Much of today's vehicle protection technology originated in southern Africa with further development in Australia,

Fig. 58.8 Gryphon experimental robot on trial in Cambodia (after [58.29] courtesy of S. Hirose)

the UK, and other countries, stimulated in part by South African expatriates.

One way to reduce the distance between researchers and field problems, at least in terms of geographic distance, has been to promote research in landmine-affected countries such as Sri Lanka and Colombia [58.34]. However this is not easy. Most countries affected by landmines have been disrupted by social conflict and destabilization that led to the military conflicts in which landmines were used. This makes it difficult for local people to create sufficient economic and physical security for researchers to pursue their work.

Future Prospects for Robotic Demining

What are the challenges for robotics researchers working on landmine clearance and other hazardous applications in the future?

We need further advances in mechatronics design, sensing, and accurate understanding of the problems to be solved using robots.

The best starting point for research is to witness people undertaking hazardous work in several different situations. Nuclear accidents, mine disasters, and burning buildings are usually off-limits to researchers. However, mine clearance operations are readily accessible in many countries. Though many researchers think a visit would be far too hazardous, there is no better way to appreciate the practical difficulties involved. Photographs taken at mine clearance operations are available to provide researchers with a web site for reference purposes, partly in answer to this need to understand the practical realities [58.35].

One of the main motivations for robotics researchers has been the perception that mine clearance is a hazardous occupation and that it would be more preferable for robots to be exposed to minefield risks

than human beings [58.36]. While mine clearance is certainly a hazardous occupation it is not necessarily dangerous. Accident records show that mine clearance in Afghanistan in 1998 resulted in about half the rate of injury of the United States forestry industry and about one third the rate of injury for the United States building construction industry per 100 000 working hours. Mine clearance agencies use advanced techniques to improve safety when possible [58.37]. In terms of deaths, demining is considerably less hazardous than mining, construction of building foundations, and especially offshore drilling rigs [58.38, pp. 11–14].

Another motivation for research is to reduce deaths and injuries among local people who have to live with the daily threat of landmines and unexploded ordnance. Again, there are misperceptions of risk. The incidence of death and injury from mine explosions is often very small compared with disease, for example. The main priorities for local people tend to be improvements for water and food supplies, education, sanitation and physical security: landmine clearance is usually a much lower priority and it is often hard to justify significant local resources.

It is also important that robotics researchers intending to contribute to the solution of this problem understand the relatively small size of humanitarian demining operations, which have been funded from a combined international humanitarian aid budget of approximately US$ 400 million. These programs spend an estimated $ 20 million annually on all equipment needs. The market for specialized humanitarian demining detectors is therefore very small and manufacturers cannot afford research and development specifically to support humanitarian demining solutions [58.39]. Adapting technology developed for other purposes, such as military equipment or civil engineering construction machinery, is more likely to be feasible.

From 1990 to 2014 there were significant improvement in mine clearance techniques but overall progress is still slow and robotics may well provide the final solution in the long term. There are still opportunities to develop robotic techniques that could ultimately provide the only cost-effective method for finally eliminating this menace.

58.2.2 Hazardous Materials Handling and Operations

The oldest application of robotics-related technology to hazardous environments is various aspects of remote nuclear operations dating back to the beginning of serious work on atomic physics in the early 1940s. This section discusses hazardous materials and operations in the context of nuclear applications and some extrapolations to other domains. These applications run the spectrum from low- to high-fidelity manipulation and multiple mobility modes.

In the 1940s, research in atomic physics led to a new era in remote handling as scientists sought to explore the nature of materials involving ionizing radiation. As experiments became more complex, mechanical manipulator systems were created which allowed operators to perform increasingly complex tasks safely behind thick biological shielding VIDEO 586, VIDEO 587, VIDEO 588, VIDEO 590. These mechanical systems then evolved into electrical systems that allowed larger work volumes to be considered. Incredible engineering achievement occurred in a 15 year period within the Remote Control Division of the Argonne National Laboratory. Even though this era represented tremendous technical achievement, it went further in illustrating the intrinsic complexity of remote operations. The equivalent work performance achieved with sophisticated teleoperated remote systems is poor in comparison to what human workers can achieve with direct contact operations and common tools. Typically, this form of teleoperation (i. e., manual control over a physical distance or barrier) is ten to hundreds of times slower than the equivalent operations performed directly by a human. Remote operations are extremely expensive and time consuming and have been the continual target of engineering improvements over the years.

Many research and development efforts have focused on different avenues for improving the work efficiency of teleoperated remote operations VIDEO 589. These efforts have included the development of better manipulators, control stations, control algorithms, etc., all intended to enhance reliability and maintainability. In the late 1960s and early 1970s, as digital electronics became more cost effective, interest began to emerge in the integration of automation with teleoperation as a scheme to effectively increase remote operations work efficiency. It was around this time that industrial robot concepts were also introduced. Combining selective automation of specific subtasks with traditional teleoperation offers the potential to reduce labor requirements and to improve the quality of repetitive task executions. This integration of automation with teleoperation became the foundation of what is now termed *telerobotics*. From the 1970s until today, telerobotics has been an active area of research and development in many different domains that include nuclear, space, and military applications. Unlike manufacturing automation, remote operations in hazardous and unstructured work task environments necessitate human-in-the-loop control, or teleoperation, at least as a backup, to assure safe operations. Teleoperation related to hazardous materials

and operations, in the most general sense, involve mobility and the use of manipulators to handle objects and tools to accomplish useful work in predominantly unstructured and uncertain environments.

Over the years, there have been numerous papers, books, and reports written concerning technical challenges, issues, and solutions. Readers will find *Vertut* and *Coiffet* [58.40] and *Slutski* [58.41] provide comprehensive and general information about manipulators and systems VIDEO 590. The performance and design of effective teleoperated systems are strong functions of human factors and *Kraiss* [58.42] and *Johnsen* and *Corliss* [58.43] provide good discussions of key principles. Finally, control system hierarchies and structures are explained in depth by *Sheridan* [58.44]. A final emphasis to all readers would be to fully explore the annals of the American Nuclear Society's Remote Systems Division, later renamed Robotics and Remote Systems Division (RRSD). Over the decades, this division hosted numerous topical meetings and workshops on all aspects of remote operations. A starting point is http://rrsd.ans.org/pages/topicals.html.

Past Perspectives

Initially, science and experiments involving dangerous radiation levels were accomplished through the innovative use of shielding walls, long-handled tools, and mirrors. As the tasks became more difficult, mechanical arms that could do a better job in emulating human motions were pursued. These ideas represented schemes whereby human sensing and handling capabilities could be more completely *projected* into the remote work environment [58.45]. One of the first challenges that the Argonne group tackled was the development of the mechanical master–slave manipulator. The basic concept was to create a mechanism that would have a master controller side where an operator could provide position and orientation commands to a slave-side mechanism/linkage system that would *replicate* motions and forces in the remote work area. It was felt that force reflection to and from the master and slave systems was essential for the operator's sensory awareness of the task execution. Experimentation has repeatedly verified the significance of both kinesthetic and tactile feedback in performing more complicated tasks [58.46].

Today, master–slave manipulator (MSMs) are used around the world in nuclear, biological, and other types of hazardous remote experimentation and operations. (They have also become increasingly used for so-called robotic surgery.) The remote work efficiency of a dual-arm MSM with shielded-window viewing is around 5–10 times slower than equivalent direct human contact operations. Because the master and slave sections of MSM's are mechanically coupled through the metal-tape drive transmission, the physical separation that can exist between the safe operating area and the hazardous remote work area is limited to a maximum of ≈ 10 m. Because of this characteristic and their kinematics, MSMs are restrictive in many applications and often have constrained the physical design of remote cells. The Argonne group and others recognized that it would be much better to have the equivalent of a fly-by-wire MSM in which the physical separation of the master and slave could be larger. This need led to the development of electrical master–slave manipulators (EMSs) that are commonly called electrical servomanipulators [58.47, 48]. Research and Development on the EMS began in the late 1940s and continued into the early 1950s.

The Argonne group was at that time limited by the available electrical control technology. Nonetheless, they made noteworthy progress toward integrated systems, as depicted in Fig. 58.9. The prototype system shown is a dual-arm anthropomorphic system with head-aiming remote television viewing and bilateral force reflection. After the Argonne Remote Control Division was disbanded, limited research and development occurred until the 1970s when commercial nuclear power growth was driving a number of research programs in the US, West Germany, France, and Japan. During this time, the programs in the US and France were focused on electrical servomanipulator systems which incorporated emerging microprocessor technology. The Central Research Laboratories model M2 system, shown in Fig. 58.10, was jointly developed with the Oak Ridge National Laboratory and was the first force reflecting servomanipulator system to use distributed digital electronics to implement position–position force reflection with multiplexed serial communications between the master and slave. The

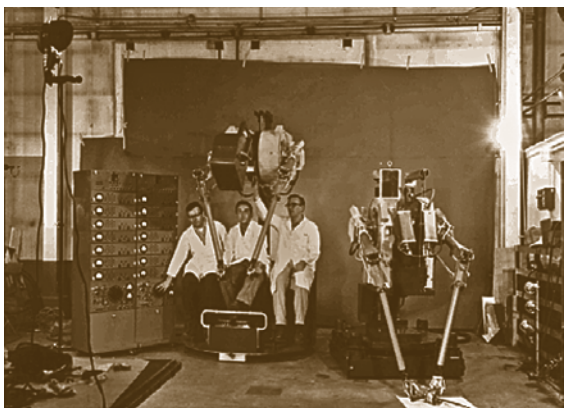


Fig. 58.9 An integrated electrical master–slave manipulator system (courtesy of Oak Ridge National Laboratory)

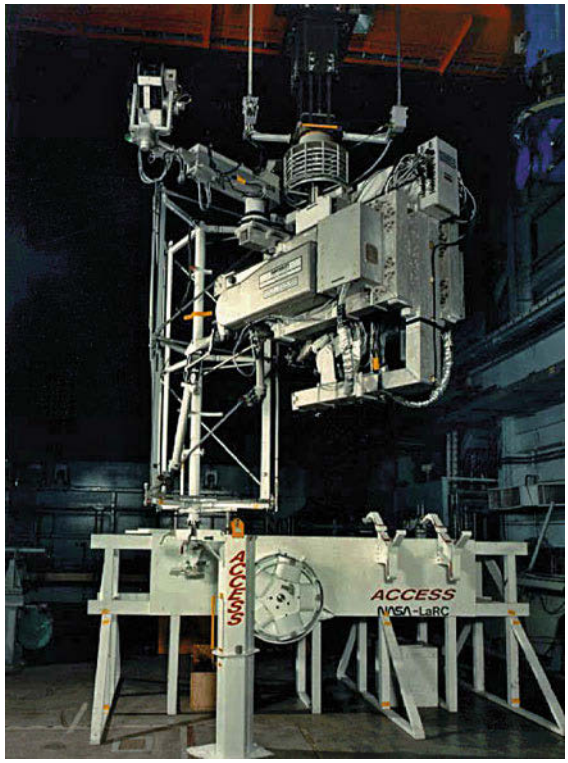


Fig. 58.10 CRL Model M2 manipulator system performing space truss assembly (courtesy of Oak Ridge National Laboratory)

Fig. 58.11 Advanced servomanipulator system (courtesy of Oak Ridge National Laboratory)

model M2 system was used over the years to explore remote operations for military, space, and nuclear applications.

The development of the advanced servomanipulator (ASM) followed the M2 in an effort to improve the remote maintainability of the manipulators themselves. This work was one of the earliest modular robotics efforts. The motivation for this work was to reduce maintenance technician radiation exposure and to increase the overall availability of the remote maintenance system. The ASM was designed from the beginning to provide a foundation for telerobotics in addition to effective teleoperation [58.49] (Fig. 58.11).

At the time of the M2 and ASM developments, Jean Vertut and his colleagues in the Commissariat à l'Énergie Atomique (CEA) focused their research on the development of telerobotic functionality for their MA-23 electrical servomanipulator systems. This research appears to include some of the earliest experimental demonstrations of telerobotic functions [58.40]. They called their concept computer-assisted teleoperation and it included both operator assists and robotic teach/playback functions. Operator assists included software jigs and fixtures designed for the improvement of the remote operation of tools such as saws, drills, etc.

In the 1980s and 1990s as nuclear power activities began to decline, nuclear remote operations technology was migrated into other areas such as space and the military. Nuclear remote operations experience with teleoperation influenced the Space Shuttle remote manipulator system and the short-lived Flight Telerobotic Servicer program.

In the mid 1990s, substantial interest developed in the application of remote handling systems to deal with problems in hazardous site/facility remediation, such as those associated with defunct nuclear facilities in the US and in Eastern Europe. The intrinsic complexity of remote clean up operations continues to drive research and development in all aspects of teleoperated and telerobotic systems. Challenges and accomplishments in this application area are discussed in the next section.

Hazardous Site Clean Up Applications

Robotic and remote systems have been used in assessing the status of sites contaminated with hazardous

materials. These surveys are essential in planning and executing subsequent clean up operations. Such systems have been used most extensively in nuclear waste sites around the world because remote techniques are more common in many nuclear operations. There has been some use of robotic survey systems in chemical and biological hazard situations as well.

Survey Systems. Numerous robotic survey systems have been developed and used. The basic idea is to integrate a sensor suite, appropriate for the contaminants of interest, with a suitable mobile platform that incorporates remote and/or autonomous driving functions and requisite navigation and control functions. The desired output of the survey process is a *precise* map of contaminant locations and concentrations. Such systems have been developed for both outside and inside operations.

It was common practice in many industries for a number of decades to bury hazardous wastes in earthen trenches in isolated burial sites VIDEO 591, VIDEO 592, VIDEO 593. Usually, useful records of what materials were buried at what locations either did not exist or were not accurate. In fact, the general conditions of such buried waste sites are often unknown to the extent that human entrance is not allowed. As a result, the first step toward remediation is to quantitatively assess the physical and hazard conditions of the site. The mobile robot shown in Fig. 58.12 is an example of a survey system used to evaluate nuclear waste buried waste sites. The robot's location is monitored with Global Positioning System (GPS). The suite of sensors includes eddy-current probes and ground-penetrating radar that reveal density contours, radiation detectors, and gas emissions monitors. This system was an initial prototype that could be operated from a remote driving station or operated in autonomous programmed trajectory mode. The system was controlled using a radio communication link. Its unique feature is that it was designed with minimal use of ferromagnetic materials to minimize interference with the magnetic subsurface sensors.

The mobile characterization system shown in Fig. 58.13 is a similar concept for radiation survey of floored areas. This particular system uses a Cybermotion commercial mobile platform, triangulation of optical fiducial markers for localization, and operates primarily in a full autonomous mode. The objective in this application is to reduce labor costs and eliminate error-prone tedious human operations.

Excavation Systems. The actual remediation of buried wastes ultimately leads to digging and object-handling type operations. Various types of systems involving integrated mobility and manipulation-like functions have been developed. One of the most popular and low-cost approaches has been to retrofit conventional excavation equipment with sensor and actuators that allow remote and robotics operations. Such systems have been used in explosive ordnance disposal also. The system shown in Fig. 58.14 is called the teleoperated small emplacement excavator (TSEE) and is a prototype remotely operable backhoe for such applications. The unit included a multi-camera remote viewing system that is the primary basis for teleoperation. Radio and tether communications connections are used to allow the human operators to be displaced up to several kilometers from the hazardous operations.

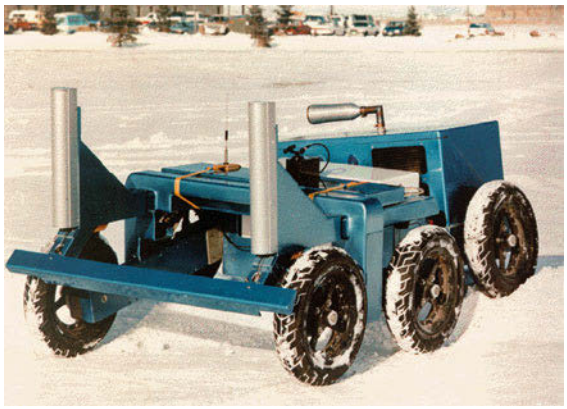


Fig. 58.12 Prototype buried waste survey robot (courtesy of Los Alamos National Laboratory)

Fig. 58.13 Mobile characterization system (courtesy of Oak Ridge National Laboratory)

Deactivation and Decommissioning Systems. In recent years, one of the most complex forms of hazardous operations is the deactivation and decommissioning (D&D) of defunct facilities where nuclear radiation or toxicity hazards preclude human presence.

Fig. 58.14 Remotely operated excavator (courtesy of Oak Ridge National Laboratory)

D&D can be thought of as remote demolition for the most part. Some operations are crude such as knocking down building structures and debris removal. Other operations may involve careful disassembly of equipment and devices, size reduction and packaging of handling/storage. These operations are essentially the inverse of remote maintenance and require the dexterous use of tools and handling of objects. Tools include saws, hydraulic shears, impact wrenches, and the like. Planning and situational awareness are very important in demolition-type operations because the physical layout and stability of the basic environment is changing during the process. Effective remote viewing (adjustable multiview capabilities) and acoustic sensing are essential. Audio feedback from the remote environment provides the human operator with the ability to monitor the normalcy of tooling operations such as sawing. D&D applications require manipulators that are both dexterous and massive. Experience has shown that payload capacities on the order of 100 kg are needed to deal with debris handling, typically with manipulators of the type shown in Fig. 58.15. Figure 58.16 is a frame view from one of the remote viewing cameras used to operate a similar system that was used to D&D an old experimental nuclear reactor. A manipulator is being used to teleoperate a conventional circular saw to segment a large-diameter aluminum tank. This operator view of the remote environment provides a realistic perception of the task environment lighting conditions and complexity.

In recent years, numerous remote manipulation systems for nuclear applications have been developed around industrial robots. These systems are augmented with sensors and master controller arrangements to facilitate teleoperation. This approach offers significant cost savings and the level of teleoperation performance achievable in complex task environments has steadily increased [58.50–54].

Fig. 58.15 Dual-arm D&D manipulator system (courtesy of Oak Ridge National Laboratory)

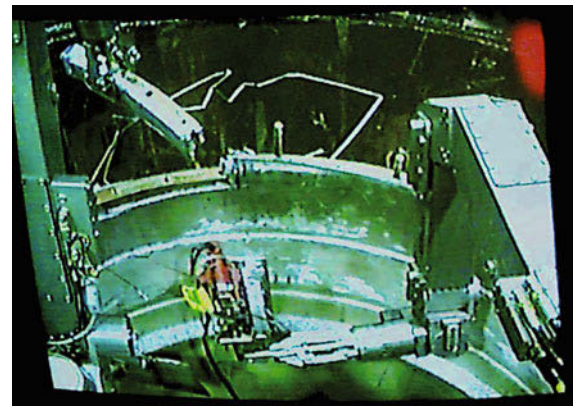


Fig. 58.16 Operator's view of a remote sawing task (courtesy of Oak Ridge National Laboratory)

Applications in Large Scale Science Experiments

High radiation levels resulting from materials activation in the target areas of high-energy particle accelera-

Fig. 58.17 SNS Telerob EMSM-2B Dual arm servomanipulator

Fig. 58.18 Telerob EMSM-2B master control station

tor experiments often require remote operations. Since these are unique scientific experiments, they involve complex and specialized equipment that results in some of the most challenging remote operations in the world. Nonetheless these highly specialized remote operations still involve the basic elements of remote operations summarized in Fig. 58.1. The most recent example of this class of remote systems is the comprehensive remote handling system developed as an integral part of the Spallation Neutron Source (SNS) located at the Oak Ridge National Laboratory [58.55].

The SNS is a neutron scattering facility in which uniquely high-flux neutron beams are generated by impinging high-energy protons on to a mercury target. The mercury is activated to the extent that the contact dose rate in the associated piping and vessels can be over $10^5\,R/h$ and are long lived. The design and operation of this target system are such that it must be periodically replaced or refurbished. The mercury target system and its support infrastructure are located in a shielded hot cell that includes a comprehensive remote handling system. The hot cell remote handling system is comprised of combinations of conventional mechanical master/slave manipulators, overhead cranes and a state of the art dual arm telerobot. Details of the overall system are provided in [58.55]. The focus of this discussion will be the telerobotic system.

Fig. 58.19 SNS remote handling control room

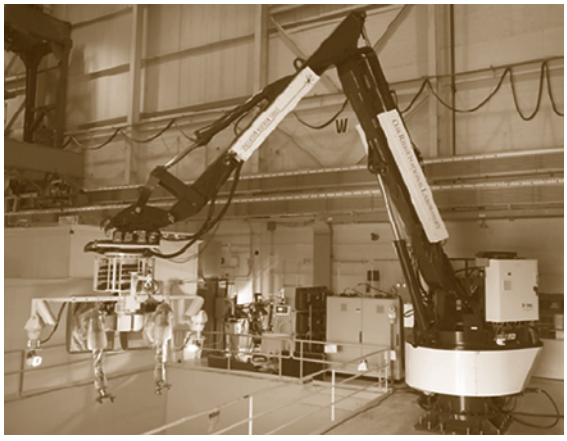


Fig. 58.20 SNS pedestal manipulator system

Figures 58.17 and 58.18 show a dual arm force reflecting servomanipulator system, Telerob EMSM-2B, that is mobile in 3-D throughout the hot cell by virtue of a overhead bridge (X,Y motions) and rigid telescoping tube (Z motion). The manipulators are 6 dof and each have a 245 N continuous capacity and 445 N peak capacity. A 2200 N auxilary hoist is integrated into the bridge trolley to facilitate handling heavy objects with the telerobot. The bridge system can be teleoperated or robotically pre-programmed. A second dual arm servomanipulator package (Fig. 58.20) is part of a mobile pedestal mounted system that is specifically designed in the high bay area over the target hot cell. The control room, Fig. 58.19 includes 21 flat panel displays

Fig. 58.21 Fukushima Daiichi Nuclear Plant after tsunami had inundated entire site to depth of over 13 m and subsequent hydrogen explosions. (courtesy of Japan Atomic Energy Agency)

that can be used with any of the 11 remote cameras located in the hot cell. The telerobot operator is located at the far end of the room and can operate either standing or sitting. This overall system is functionally similar in concept to advanced servomanipulator systems discussed in the earlier sections. The system has been in operation since 2006 and has been successfully used in several major mercury target diagnostic/refurbishment campaigns. The SNS remote system has shown that the electromechanical equivalent of traditional MSM's with full remote hot cell mobility is practical and reliable. Larger scale nuclear facilities involving higher level of radiation are certain to follow the remote handling philosophy represented by this system. Because this class of electrical servomanipulators have all of the real-time digital control features of modern robots, greater use of programmed and intelligent control concepts for certain types of remote tasks is expected in future.

58.2.3 Lessons Learned from the Fukushima Daiichi Disaster

Japan has not only been at the forefront in the development of robots to help in disaster situations, but has also, sadly, experienced a major recent disaster of the kind anticipated by robotics researchers and engineers for decades previously. This disaster provided a reality check for researchers and exposed just how far we still need to progress before robots can be relied upon to perform routinely in hazardous environments.

The March 2011 tsunami precipitated the Fukushima Daiichi nuclear reactor disaster (Figs. 58.21 and 58.22). Even though the reactors were shut down at the time, they still required significant cooling water supplies as they had been only recently shut down.

Fig. 58.22 Underwater reconnaissance robot being evaluated in search for human remains following the March tsunami, Japan. (courtesy of Prof. Shigeo Hirose)

The tsunami destroyed the diesel back-up generators required to keep the cooling water pumps operating. The resulting temperature rise in the reactors led to gas explosions that severely damaged the reactors, their cooling systems and the containment buildings, spreading radioactive contamination over a large area of nearby land and ocean.

As Kawatsuma and his colleagues explained, many different kinds of robots had been developed as a result of earlier research and development programs [58.56]. Unfortunately, at the time of the disaster most of these robots were either inoperative, unable to be repaired and made ready, or unsuitable because of the need for tether cables, or reliance on Wi-Fi communications which could not be operated satisfactorily in the contaminated nuclear environment (Chap. 60.4). Several teleoperated construction machines developed for

the 1991 Mt. Fugen volcanic disaster were successfully deployed to clear contaminated wreckage and debris VIDEO 578. Robots originally developed for reconnaissance work, primarily investigation of unexploded ordnance, were supplied by British and American sources VIDEO 579, VIDEO 580, VIDEO 581. However only limited operations could be completed with these robots as they could not reach many of the places where they were needed. In the first months following the disaster, only one robot, the Japanese designed Quince from Chiba Institute of Technology, was successfully deployed inside the reactor buildings for several weeks before it failed with a wiring fault. At the time of writing, it had not yet been recovered for repairs VIDEO 582, VIDEO 584. Since then more robots have been successfully deployed to the site, though still with limited operational capabilities.

The disaster exposed the need for prior planning and continuous training, rehearsal and cooperation between research and development agencies, defence disaster relief agencies, robotic systems manufacturers and engineers at the hazardous facilities where the robots would be used. Rather than individual robots, systems of which robots form a part need to be developed, including shielded vehicles to allow close-up monitoring and control by human operators VIDEO 583. Continuous updating of robot systems is needed: some robots that could have been deployed relied on obsolete electronic and computer components that could not be replaced. Following this disaster, we can expect a significantly increased effort to deploy robots over the coming years to deal with the consequences of similar events.

The Fukushima Daiichi disaster and other operations following the March 2011 tsunami demonstrated that, as in the other applications discussed above, robots can still only be expected to perform a niche role. Robots have only performed consistently in such environments after decades of development, and investment in developing trained operators with appropriate maintenance and support facilities. Our ability to realistically predict the capabilities of robotic operations in unforeseen hazardous environments is still very limited.

58.3 Enabling Technologies

There are a number of technology issues that are key to the design and implementation of mobile robots that can operate effectively in different types of hazardous environments. A few of them are touched upon here to give the reader additional insight into the foundational aspects of this class of robots.

58.3.1 Mobility Issues

Hazardous applications such as fire fighting, EOD, and demining require the ability to operate on uneven ground in indoor and outdoor environments where a normal wheel-type robot cannot easily operate. Most mobile mechanisms are classified into wheel type [58.57–60], track type [58.61–66], and leg type [58.67]. For negotiability in uneven ground, the wheel and track types need additional linkages with ground adaptation. There are two kinds of this adaptation: active and passive adaptation. Active adaptation uses an additional actuator to alter the linkage's motion while in passive ones the linkage motion is controlled by the ground conditions and gravity effects.

Stability Conditions

Stability should be considered in the design process to prevent rollover on uneven ground including stairways, steps, and natural terrain. Stability can be investigated in terms of three parameters: *center of mass, supporting area, and stability margin*. Static stability requires that the center of mass remains inside the supporting area, as shown in Fig. 58.23. The supporting area is a polygon built by the edges connecting each contact point on the horizontal plane. A conventional vehicle has limitations in rugged terrain due to a fixed center of mass with respect to the body coordinates. The center of mass greatly affects the stability margin (the minimum length between the center of mass and edges of a supporting area). If the center of mass is located out

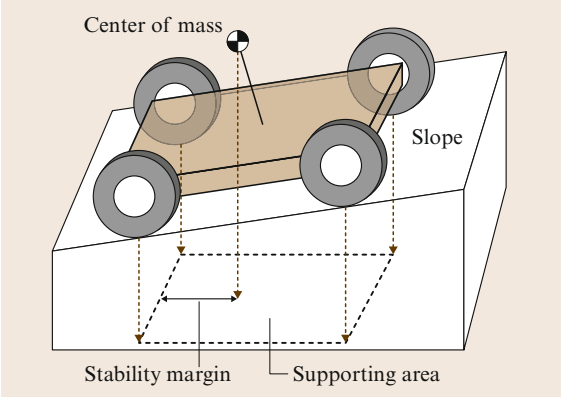


Fig. 58.23 Stability condition

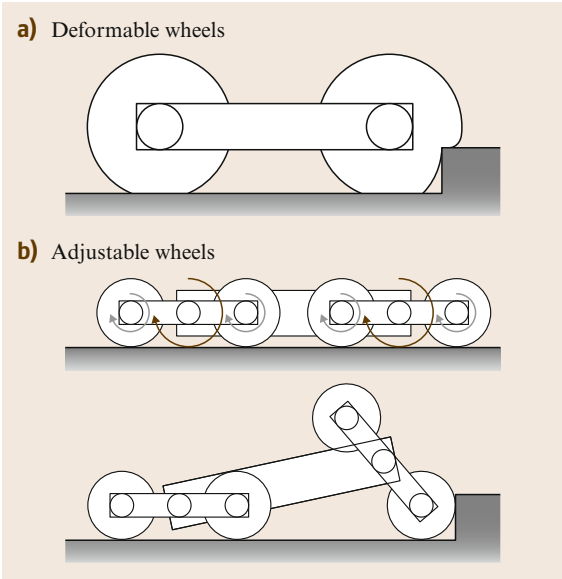


Fig. 58.24 Wheel configurations

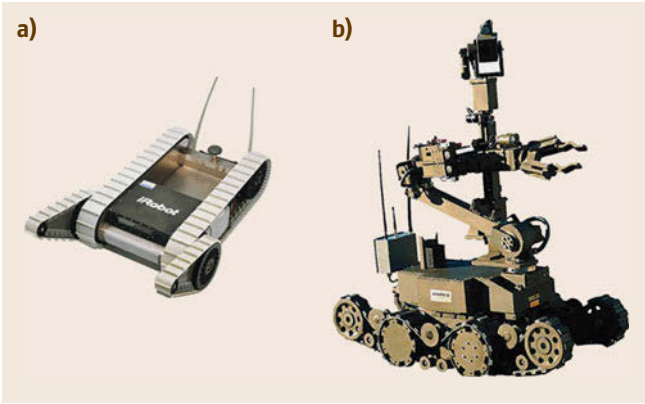


Fig.58.25a,b Mobile robots with active adaptation mechanisms: **(a)** Packbot (after [58.68]), **(b)** Andros Mark V-A1 (after [58.64])

of the supporting area, rollover can occur. The stability margin of a conventional vehicle is mainly determined by the terrain inclination. Many vehicles are designed to have a low center of mass, thereby obtaining a large stability margin on a slope. Furthermore, a vehicle is often designed to have multiple bodies to overcome this limitation. For a multiple-link mechanism, the center of mass varies and the relative motion of links when traveling over the landform also alters the supporting area.

Active Adaptation Mechanisms

An active adaptation mechanism uses an additional actuator to generate motion of the adaptation linkage on rugged ground. Although additional actuators and linkages are required, the adaptation mechanism can overcome variously shaped rugged landforms.

It is known that a vehicle can negotiate an obstacle which is smaller than the wheel radius. One may think that a vehicle which has large wheels can pass over rough terrain like a monster truck. However, most applications require a small vehicle to drive between obstacles.

A deformable wheel or its equivalent mechanism can be adopted: insufficiently inflated wheels, as shown in Fig. 58.24a, adapt to irregular terrain by deformation of wheels. However, their size should be big enough to negotiate rough terrain, so this concept still has size limitations. Their equivalent mechanisms are relatively compact and yet they need sophisticated articulation mechanisms, as shown in Fig. 58.24b. Two wheels are attached to a link that rotates about a main body. Therefore, revolution and rotation of wheels are made as shown in Fig. 58.24b. If rotation is provided by actuators at the main body (i. e., an active mechanism), the vehicle has an equivalent wheel whose radius is equal to that of the rotation. The vehicle in Fig. 58.24b can pass a step by carriage rotation, although the radius of a wheel is smaller than the height of the step.

Many track-type mechanisms have multiple linkage structures with active adaptation [58.63–66, 68]. When traveling over uneven ground, these mechanisms can negotiate rugged ground by changing the configuration of multiple track mechanisms, for example, typical multiple-tracked robots with active adaptation such as Andros [58.64] and Packbot [58.68], as shown in Fig. 58.25a and 58.25b, commonly uses a small active additional track called a flipper. When moving on relatively even ground the robot can reduce undesired track resistance by lifting up the flippers. When climbing up stairs, touching the flippers down to the ground can increase the stability by enlarging the supporting area. Snake-like robots use active interconnecting joints [58.65] and have the potential for mobility and manipulation.

Passive Mechanisms

Passive adaptation mechanisms do not use any additional actuators; rather they simply utilize the gravity effect to generate adaptation to irregular ground. This approach is less adaptive but enables the operator to drive the robot easily because he (or she) does not have to take care of controlling the adaptation mechanism. Passive mechanisms usually include various kinds of passive linkages activated by irregular ground conditions.

For the wheeled vehicle shown in Fig. 58.24b, a passive mechanism can be designed in which no additional actuators are needed, for example, when a wheel

Fig.58.26a,b Mobile platforms with passive adaptation: **(a)** Solero (after [58.60]) and **(b)** Robhaz-dt3

meets a sill whose height is greater than the radius of a wheel, it will become stuck. Then high torque is exerted at the wheel since the stall torque is usually greater than the normal rotation torque of the wheel. In this case, a closed linkage without any actuator except the driving wheels can generate a passive motion to climb over a step [58.58], for example, a smart mobile robot with passive adaptation called Solero (depicted in Fig. 58.26a) can climb up an obstacle by lowering the instant center of rotation of the passive four-bar linkage when coming into contact with the vertical surface of the obstacle [58.60].

Similarly, multiple bodies with tracked and chained passive joints can also adapt to irregular terrain. The track type is inherently insensitive to the unevenness of ground VIDEO 584. Thus passively chained multiple-tracked bodies can provide greater reliability and better capabilities in ground adaptation. For example, a mobile robot with a passively chained double-tracked mechanism called Robhaz (shown in Fig. 58.26b) is a practical design with simply adapted multiple tracked bodies with passive joints [58.66]. As depicted in

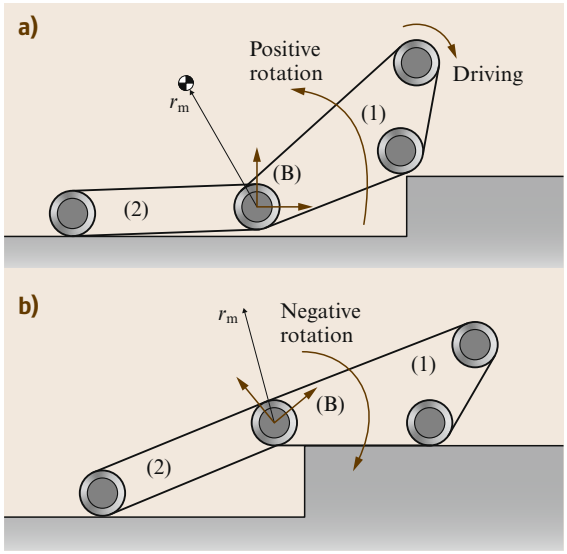


Fig.58.27a,b Passive adaptation of double-tracked mechanism: **(a)** positive deflection and **(b)** negative deflection

Fig. 58.26, two tracks of the Robhaz rotate positively or and negatively according to the contact situation between the track and the stair surface. Based on this behavior, in terms of the lower center of mass (Fig. 58.27a) and the supporting area (Fig. 58.27b) better stability in stair climbing than that of a single-tracked vehicle can be achieved.

58.3.2 Manipulator Design and Control for Hazardous Object Handling

Manipulation is essential for EOD missions. Generally, EOD missions have two phases: (1) approach, and (2) object manipulation to achieve ordnance neutralization. The second phase is generally controlled by teleoperation. Both the human's careful control and innate intelligence, plus appropriate manipulator design and control, are needed for such dangerous tasks.

Design Requirements

Unlike industrial manipulators, the design of manipulators for many robots in hazardous environments does not require very fast movement. Instead, high payload, light weight, and compactness are more important. Therefore, many manipulator designers locate actuators and reducers, which typically have high weight and large volume, on the base or lower links. This reduces the weight of the moving parts of the manipulator and offers better stability and dynamic control properties. Consequently, they have a thin profile, as shown in Fig. 58.28a,b. This characteristic is usually associated with explosive ordnance disposal-type robots that

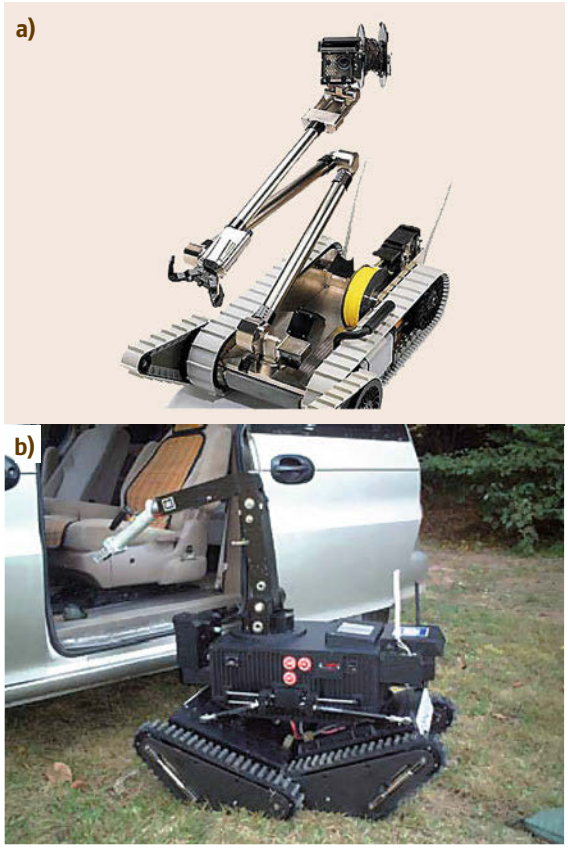


Fig.58.28a,b Hazardous application robots with manipulator: **(a)** Packbot with manipulator and **(b)** Robhaz-dt2 with manipulator

require access to congested areas and normal human living spaces. Mobile robots used in other hazardous environments can be much larger, depending on the specific applications.

58.3.3 Control for Hazardous Tasks

In most cases, the operator does not have much information about the environment around the robot and the manipulator. He may have very limited vision, and unexpected collisions between the manipulator and obstacles can easily happen, even if he (or she) tries to be very careful. The robot should be able to handle these risks by appropriate design of the manipulator and good control algorithms. There are several ways to solve these problems.

Proximity sensors that detect nearby objects can be used to avoid a collision before it occurs [58.69]. However using many proximity sensors necessitates large wiring bundles and complex manipulator design. Force control is also a good way to handle collisions. When a collision occurs, the manipulator can move away from the collision point by using force/torque sensor data. Often a six-axis force sensor attached at the end-effector of the manipulator is used for the force control [58.70]. Many researchers have developed stiffness/compliance/impedance control methods for a manipulator with a force sensor. However, this method can only handle a limited collision zone – only the end-effector or the manipulator *hand* – because the sensor cannot sense contact with intermediate links, while a collision could occur at any point on the manipulator. For this reason, some manipulators use joint torque sensors at all of their joints [58.71]. When using a joint torque sensor, contact at any point on the manipulator can be sensed, however, gravity compensation and errors in the transformation from the joints to the Cartesian space introduce further difficulties.

The stability of telerobotic systems, particularly when force feedback or reflection is involved, remains a critical issue. Systems involving significant data communication time delays are even more difficult. National Aeronautics and Space Administration (NASA) groups have studied stability and bandwidth issues numerous times over the years; one of the more recent studies is provided by *Uebel* et al. [58.72]. Energy methods have been applied to this problem showing promise for enhanced stability and performance [58.4, 73].

Master Controllers for Teleoperated Manipulation

In teleoperation, the robot is designed to be a faithful slave to deal with a dangerous task while the operator uses a control interface to direct the slave from a safe location. The user interface usually provides a means for:

- Sending position commands to the manipulator.
- Providing contact force feedback or force reflection to the user.

Ideally, the design goal of a user interface is to make it *transparent*, so the operator feels as if he or she is directly manipulating the object handled by the slave manipulator. To achieve this transparency, there are several design issues:

- Simplicity:
  - All indicators are unified as one scene.
  - All input button and joystick are integrated into one haptic device.
- Intuitiveness:
  - High-level command by speech recognition.
  - Human-friendly feedback such as graphical displays and human voice.

  - Motion command matching between the haptic device and the slave in Cartesian space.
- Portability:
  - Small/lightweight and human-friendly design for wearability
  - Tetherless operation without communication lines and additional power.

An example of wearable multimodal user interface is provided by [58.74], as shown in Fig. 58.29. The operator wears a head-mounted display (HMD), head tracker, and headset to interact with the slave. A six-degree-of-freedom haptic master is attached on his waist together with the standalone controller, and the operator grips its handle to telemanipulate in Cartesian space. With wearable hardware, the user can walk around to gain a better view during teleoperation using wireless communication. It is composed of major three operator interfaces.

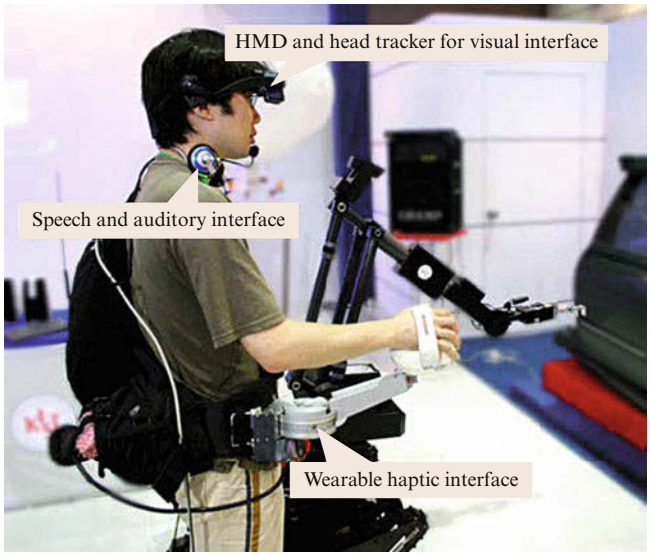


Fig. 58.29 Appearance of wearable multimodal user interface

Speech and Auditory Interfaces

An operator usually sends two types of commands to the robot: selection and continuous motion commands. For example, the selection between mobile and manipulation mode, the reset of the robot arm and mobile base, the on/off and reset of pan–tilt motors, speed selection for the mobile platform, selection among installed cameras are defined in the selection commands. Voice commands offer convenience. When the operator says a word that has been predefined as a command, the speech interface can sense that word and, if the speech recognition system successfully recognizes it, the recognized command pops up on the HMD for confirmation. Finally, the operator decides whether to execute or cancel the command with a confirmation button on the haptic master.

The auditory interface synthesizes the human voice using a speech synthesis engine. It can warn of the approach of an obstacle by sound, or reveal the distance and direction to a laser-designated object nearby.

Wearable Haptic Interfaces

A wearable haptic device is shown in Fig. 58.29. The base linkage is designed as a serial RRP mechanism to measure a translation, and an RRR z–y–z rotation mechanism is attached at the end of the base linkage.

Figure 58.30 shows a wearable haptic master device for teleoperation of mobility and manipulation systems. It has six degrees of freedom for motion input and three degrees of freedom for force feedback.

To achieve a compact design and reduce its weight, a tendon-driven mechanism is designed into each joint. Due to weight constraints, only three actuators are installed for force feedback, and each actuator is specially designed to fit the joint. Because a passive actuator is better than an active actuator with respect to power density (power per unit volume or weight), small magnetorheological (MR) brakes have been developed. They are installed at each joint of the base linkage for force feedback. Also a compact brake drive with current feedback capability has been designed, which reduces the response time of the MR brake. The controller is packed into a bag that is attached to the back side of the waist.

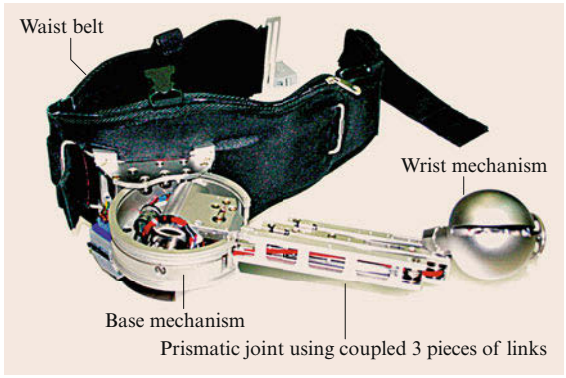


Fig. 58.30 Picture of wearable haptic device

Visual Interfaces

The visual interface integrates the robot's view, the status of sensed data, and the status of speech commands. Since the slave includes a stereoscopic camera, the user sees a three-dimensional view on the head-mounted display (HMD). The operator wears a head tracker and it generates a pan/tilt command from the two-degree-of-

freedom head motion. In the integrated system, head tracker data is used to command the direction of the remote viewing camera, thus the user easily looks around the environment of the robot, as shown in Fig. 58.31.

Moreover, the head tracker is useful to indicate an object as a target. The operator moves his head and looks at a target to get information of direction and distance, then triggers a laser displacement sensor that is placed in parallel with the pan–tilt camera. A virtual reality overlay is prepared with the reported data, as shown Fig. 58.31. The recognized speech command is highlighted on the left for confirmation. These are overlaid on the remote video source pictures and unified into a single scene. Finally the operator sees the stereoscopic picture and the status of the robot at a glance on the immersive HMD. The overlaid view is shown in Fig. 58.31c.

58.3.4 Data Communications

Robots used in hazardous environments are almost always mobile so that they can move about an area of interest with flexibility. The combination of mobility and intrinsic operation across a physical or hazard barrier introduces unique problems with respect to bidirectional data communications (Fig. 58.1). Communications from the operator location to the robot is necessary to achieve remote control or teleoperation. Communications from the robot to the operator are essential for situational awareness. The transmission of the electrical and/or optical signals comprising such data communications can be quite complex due to a host of factors such as penetration of physical barriers, signal transport delays, signal attenuation, and data throughput requirements. Data throughput requirements are dominated by the feedback of visual imagery or remote camera views from the mobile system. Channel capacity necessary for audio feedback and control are small in comparison. If the mobile system involves precision manipulation with force feedback, bidirectional data channels on the order of 1 Mb/s are required.

For a mobile robot with high-fidelity manipulation, multichannel remote viewing, and other sensors, on the order of hundreds of Mb/s are required for the data communications path from the remote environment to the operator station while on the order of ten Mb/s are needed for the bidirectional control link. Data throughput requirements can be reduced through data compression hardware provided there is no significant increase in latency (time delay) which can degrade control by humans if the delay exceeds 100 ms. The choice of communication means is not easy: while wireless communication seems attractive, when multiple robot systems need to cooperate in a shared space, the communication systems need to be designed with this in mind. All too often, communication systems have evolved to support only a single robot system within a given operating region. Tethered cables provide high bandwidth, but pose extreme difficulties in unstructured environments, and when multiple robot systems need to work together (Table 58.1).

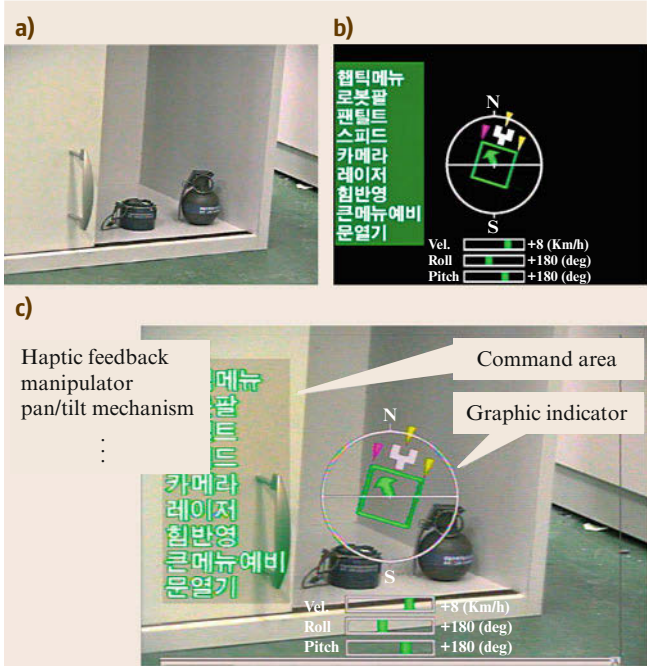


Fig.58.31a–c Appearance of integrated visual interface: (a) camera view (b) overlay source view (c) user view

58.3.5 Energetics

Another key challenge in fielding mobile robots in hazardous environments is in the area of energetics including power supply, consumption, conversion, and management. The specifics of the given environment often restrict the types of power supply/conversion that are permissible, for example, the use of combustible fuels and internal combustion engines is seldom allowed within coal mines, refineries, gas plants, offshore platforms and nuclear facilities but can be entirely acceptable in demining and other outdoor operations. The most common power systems used for research robots are electrochemical batteries and electric motor drives. The power and energy density map [58.75] given in Fig. 58.32 shows the fundamental situation for mobile robots used in hazardous environments. For perspective, a small passenger automobile is estimated to need a minimum of 200 W h/km to meet road demands, which in turn translates into 500 W h/kg. This

Table 58.1 Example data communications requirements (minimum)

Type	Single channel resolution	Typical standard system
Standard black-and-white remote television (typical for rad hard technology): 600 × 400 pixels 30 frames/s 12 bit gray scale	≈ 10 Mb/s per viewing channel	30–50 Mb/s for 3–5 channels
Color remote television: red, blue, and green = 3 × black and white	≈ 30 Mb/s per viewing channel	90–150 Mb/s for 3–5 channels
Color high definition (HD) 1080p television: 1920 × 1080 pixels 60 frames	~ 1.5 Gb/s per viewing channel with encoding compression ~ 3 Gb/s uncompressed	5.5–15 Gb/s for 3–5 channels
Control 12 bit resolution input and output 200 Hz sampling rate	≈ 4.8 kb/s per control channel	48 kb/s – ten control servo channels
Audio feedback 15 kHz signal capture 12 bit resolution	≈ 180 kb/s per audio channel	540 kb/s – three remote microphones

is why hybrid electric vehicles that combine spark ignition engines with battery-powered electric motors are becoming popular. A large mobile robot with dual manipulators operating in the payload range of hundreds of kg would have power requirements similar to these and would be well above the 100–200 W h/kg range that batteries can provide. A force-reflecting six-degree-of-freedom robot manipulator with a 20 kg payload will have a peak power consumption on the order of 10 kW. If a particular application domain is amenable to the use of spark ignition engines, the order of magnitude of power density compared with any type of battery-powered system makes implementation feasible. If pure battery power is necessary, then considerable attention must be given to the design, including state of charge monitoring and mission provisions for recharging. In the future, emerging fuel-cell technology may provide better solutions for mobile robot power systems.

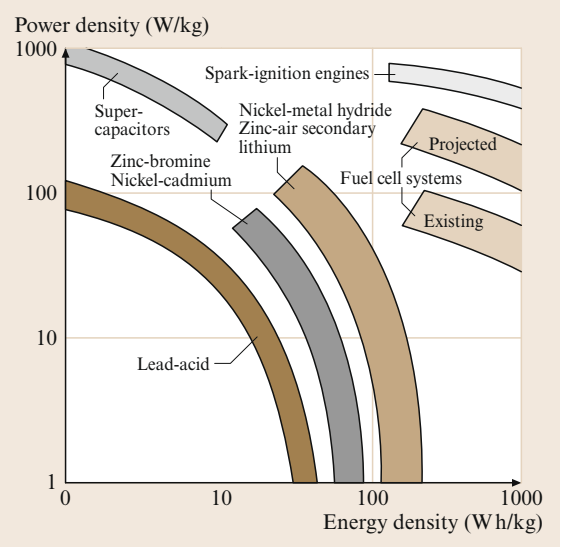


Fig. 58.32 Power and energy density for various power storage and supply systems

58.3.6 System Architectures for Real-Time Mission Control

Because of the intrinsic nature of hazardous environments where uncertainty and lack of geometric structure are prevalent in the task environment, shared control architectures are commonly used. The degree of human interaction runs the gamut from pure teleoperation (manual control) to high-level intelligent control operations (autonomous). For example, most low-cost EOD type robots utilize simple manual control architectures, while at the other end of the spectrum, the Mars rovers operate as autonomous agents responding to mission-level commands from human operators on the earth. The basic architectures for real-time mission control are summarized in Fig. 58.33 [58.76]. Also, see *Sheridan* for a comprehensive discussion of architecture principles and design [58.44]. Between the manual and autonomous control exist combinations of computer assistance and semiautonomous (i. e., selective and in situ task automation) functions intended to improve remote work performance by either reducing operator workload or allocating tasks more suitable for computer control to autonomous execution. It is important to recognize that, as one moves from manual control toward autonomous control, the data throughput rates for communication and control are reduced because high-level commands are comparatively small packets of information while force-reflection manual control of a manipulator, for example, requires visual feedback and high-bandwidth control interconnection

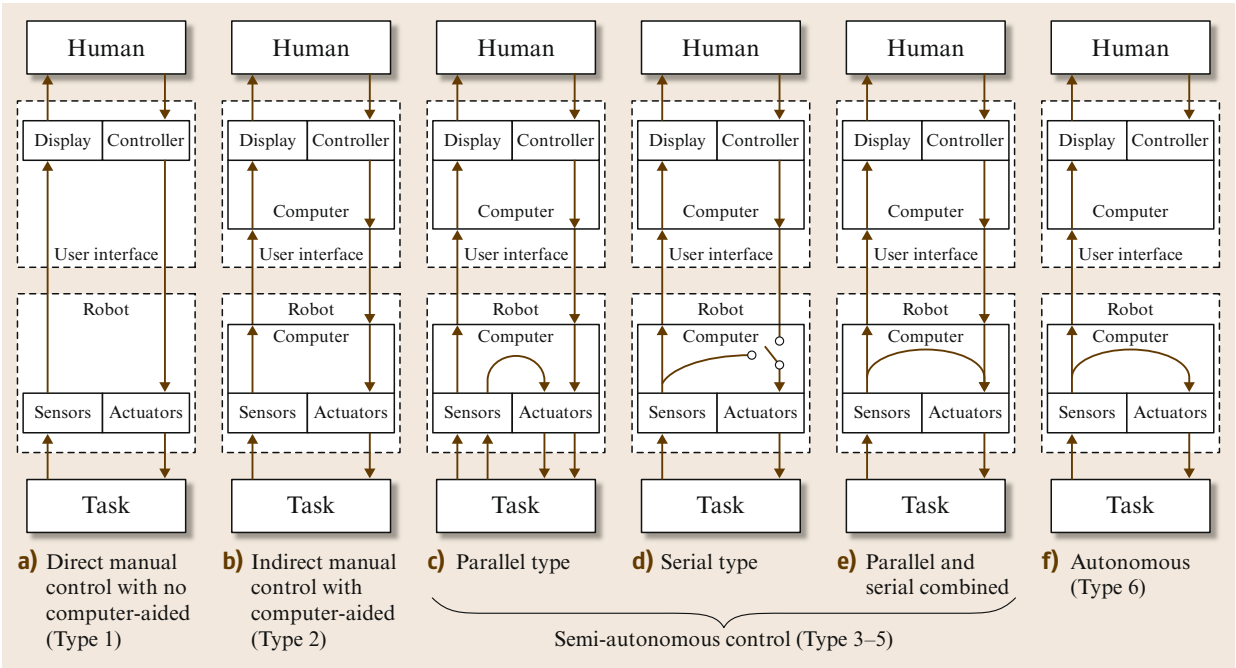


Fig. 58.33 System control architectures

of the slave and master controller. The data communication bandwidth available influences control architecture. Untethered undersea applications usually require semiautonomous architectures because of the bandwidth limitations of acoustic communications links.

As intelligent systems research advances, it is expected that, for hazardous situations, robots will employ semiautonomous and autonomous architectures to a greater degree as a fundamental approach to overall performance optimization. Operators will be able to choose the control mode based on the task needs and the likelihood that autonomous execution can be reliably achieved. In this human-centered way, task execution mode will be matched to best achievable performance.

58.4 Conclusions and Further Reading

If we are to learn anything from the experience of the last decade we must appreciate that robotics research is just one step in the development of tools that extend human capabilities. This search for improved tools is as old as humanity itself, so great patience is required [58.77].

Researchers need to make significant advances on four different fronts: mechatronics design, sensing, machine intelligence, and problem understanding:

- Mechatronics designers have to trade gains in precision, dexterity, mobility, and strength for losses in endurance and reliability. We are mostly well short of biological (e.g., human) capabilities except for high-precision applications. Machines have much the same environmental tolerance as people. Machines need special precautions against heat and cold outside the temperature range of 0–35 °C, and operation beyond −60 or +60 °C is usually impractical. Dust, radiation, low or zero air pressure, fumes, biological agents, even insects can be fatal for machines without special design features, which may result in performance reductions. Maintenance or repair work may necessitate decontamination before people can work on the machinery.
- While electronic sensors can go far beyond biological capabilities, hazardous applications still present problems far beyond present capabilities as we have seen in the case of landmine applications. Radiation levels that are lethal for humans can also quickly kill electronic sensors. Extreme heat or cold limits sensor performance, as can contamination, and even insects.

- Advances in usable machine intelligence have been much more difficult than many expected. The only real progress has been achieved with capabilities often associated with clever people: logic, mathematical manipulation, and playing games. While recent advances have enabled a remarkable degree of understanding for some insect perception and motor capabilities, the social behaviour of insects still largely defies present-day understanding. However, teleoperation and supervisory control can make up for this deficit in the medium term. Shortcomings in machine intelligence represent less of a barrier than the other three fronts.
- Understanding the intended application and economic factors has proved to be equally important for researchers. Early success from exploiting specific opportunities can generate confidence that inspires others to persevere with more difficult general problems. Failures in understanding the desired task performance, particularly with respect to economic factors, can undermine confidence and the credibility of research efforts. Attempts to develop landmine clearance robots provides a useful case study that illustrates some of the difficulties in developing robots for hazardous applications.

There is no question that future teleoperators will become hybrid telerobotic systems that allow seamless transfer between manual and autonomous operations. Specific tasks will be selectively automated by operators for the purposes of enhancing quality and/or reducing subtask execution times. Early systems will continue to incorporate high levels of human interactivity to accommodate the limitations of current intelligent systems technology. Real-time execution of complex control algorithms and virtual-reality-like graphics engines (which are really useful) have become routine. Robust machine learning that allows human task execution skills to be captured through learning by observation or imitation is gradually becoming realizable. Advances in mass production of microelectrical mechanical systems (MEMSs) technologies has greatly reduced the cost of sensing devices. In the next decade, telerobots for hazardous environments will become smarter and their relationship with their human operators will become more cooperative and less interactive. Human operators will supervise the operation of multiple telerobots: at the moment, the number of support people required for real operations is a multiple of the number of robots.

Readers are directed to the selection of references provided in this chapter for further reading.

Video-References

VIDEO 571 UNMACCA: Demining Afghanistan
available from http://handbookofrobotics.org/view-chapter/58/videodetails/571

VIDEO 572 IED hunters
available from http://handbookofrobotics.org/view-chapter/58/videodetails/572

VIDEO 574 Bozena 5 remotely operated robot vehicle
available from http://handbookofrobotics.org/view-chapter/58/videodetails/574

VIDEO 575 DALMATINO
available from http://handbookofrobotics.org/view-chapter/58/videodetails/575

VIDEO 576 PT-400 D:Mine
available from http://handbookofrobotics.org/view-chapter/58/videodetails/576

VIDEO 577 DIGGER DTR demining destroying anti-tank mines
available from http://handbookofrobotics.org/view-chapter/58/videodetails/577

VIDEO 578 Remote-control heavy equipment used in debris removal at Fukushima reactor 3
available from http://handbookofrobotics.org/view-chapter/58/videodetails/578

VIDEO 579 iRobots used to examine interior of Fukushima powerplant
available from http://handbookofrobotics.org/view-chapter/58/videodetails/579

VIDEO 580 iRobots inspecting interior of Fukushima powerplant
available from http://handbookofrobotics.org/view-chapter/58/videodetails/580

VIDEO 581 Robot being used to carry vacuum cleaner head at Fukishima powerplant
available from http://handbookofrobotics.org/view-chapter/58/videodetails/581

VIDEO 582 Views of robot control screen – inspecting Fukushima power plant
available from http://handbookofrobotics.org/view-chapter/58/videodetails/582

VIDEO 583 Promotional video of robot for cleaning up Fukushima
available from http://handbookofrobotics.org/view-chapter/58/videodetails/583

VIDEO 584 *Sukura* robot developed for reconnaissance missions inside nuclear reactor buildings
available from http://handbookofrobotics.org/view-chapter/58/videodetails/584

VIDEO 586 HD Stock Footage 1950s atomic power plants – nuclear reactors
available from http://handbookofrobotics.org/view-chapter/58/videodetails/586

VIDEO 587 Radioactive material handling 1954
available from http://handbookofrobotics.org/view-chapter/58/videodetails/587
VIDEO 588 Nuclear manipulator remote handling equipment (1960)
available from http://handbookofrobotics.org/view-chapter/58/videodetails/588
VIDEO 589 1961 nuclear reactor meltdown: The SL-1 accident – United States Army Documentary
available from http://handbookofrobotics.org/view-chapter/58/videodetails/589
VIDEO 590 Jean Vertut master-slave manipulator arms
available from http://handbookofrobotics.org/view-chapter/58/videodetails/590
VIDEO 591 NanoMag magnetic crawler for remote inspection
available from http://handbookofrobotics.org/view-chapter/58/videodetails/591
VIDEO 592 Remote handling and inspection with the VT450
available from http://handbookofrobotics.org/view-chapter/58/videodetails/592
VIDEO 593 Robots answer nuclear waste challenges at SRS
available from http://handbookofrobotics.org/view-chapter/58/videodetails/593

References

58.1 S. Kang, C. Cho, J. Lee, D. Ryu, C. Park, K.-C. Shin, M. Kim: ROBHAZ-DT2: Design and integration of passive double tracked mobile manipulator system for explosive ordnance disposal, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2003)

58.2 A. Kron, G. Schmidt, B. Petzold, M.I. Zah, P. Hinterseer, E. Steinbach: Disposal of explosive ordnances by use of a bimanual haptic telepresence system, IEEE Int. Conf. Robotics Autom. (2004)

58.3 W.R. Hamel, R.L. Cress: Elements of Telerobotics Necessary for Waste Clean Up Automation, IEEE Int. Conf. Robotics Autom. (2001)

58.4 J.C. Ralston, D.W. Hainsworth, D.C. Reid, D.L. Anderson, R.J. McPhee: Recent advances in remote coal mining machine sensing, guidance, teleoperation and field robotics, Robotica **19**(5), 513–526 (2001)

58.5 GICHD: *A Study of Manual Mine Clearance* (International Centre for Humanitarian Demining, Geneva 2005)

58.6 GICHD: *A Study of Mechanical Application in Demining* (Generva International Centre for Humanitarian Demining, Geneva 2004)

58.7 J.P. Trevelyan, S. Tilli, B. Parks, H.C. Teng: Farming minefields: Economics of remediating land with moderate landmine and UXO concentrations, Demining Technol. Inform. Forum J. **1**(3), (2002)

58.8 C.G. Bruschini, B. Gross: A survey on sensor technology for landmine detection, J. Mine Action **2**(1), (1998)

58.9 A. Göth, I.G. McLean, J.P. Trevelyan: How do dogs detect landmines? A summary of research results. In: *Mine Detection Dogs: Training, Operations and Odour Detection*, ed. by I.G. McLean (Geneva International Centre for Humanitarian Demining, Geneva 2003)

58.10 S. Havlik: A modular concept of the robotic vehicle for demining operations, Autonom. Robots **18**(3), 253–262 (2005)

58.11 J.P. Wetzel: Robotic applications in humanitarian demining, Proc. 9th Bienn. Conf. Eng. Constr. Oper. Chall. Environ. Earth Sp. (2004)

58.12 R. Bogue: Detecting mines and IEDs: What are the prospects for robots?, Ind. Robot **38**(5), 456–460 (2011)

58.13 J.P. Trevelyan: A suspended device for humanitarian demining, EUREL Int. Conf. Detect. Abandoned Land Mines Humanit. Imperative Seek. Tech. Solut. (1996)

58.14 J. Trevelyan: Robots: A premature solution for the land mine problem, Proc. 8th Int. Symp. Robotics Res. (1998)

58.15 J.-D. Nicoud: Vehicles and robots for humanitarian demining, Ind. Robot **24**(2), 164–168 (1997)

58.16 J.P. Trevelyan: Landmine research: Technology solutions looking for problems, Proc. SPIE Detect. Remediat. Technol. Mines Minelike Targets IX (2004)

58.17 H.M. Choset, E.U. Acar, A.A. Rizzi, J.E. Luntz: Sensor-based planning: exact cellular decompositions in terms of critical points, Proc. SPIE Mobile Robots XV Telemanip. Telepresence Technol. VII (2000)

58.18 Y. Zhang, M. Schervish, E.U. Acar, H. Choset: Probabilistic methods for robotic landmine search, Proc. RSJ/IEEE Int. Conf. Intell. Robots Syst. (2001)

58.19 S. Singh, S. Thayer: Inspired by immunity, Nature **415**, 468–470 (2002)

58.20 S. Sathyanath, F. Sahin: Application of artificial immune system based intelligent multi agent model to a mine detection problem, IEEE Int. Conf. Syst. Man Cybern. (2002)

58.21 D. Goldberg, M.J. Mataric: Maximizing reward in a non-stationary mobile robot environment, Autonom. Agents Multi-Agent Syst. **6**(3), 287–316 (2003)

58.22 R.A. Russell: Locating underground chemical sources by tracking chemical gradients in 3 dimensions, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004)

58.23 D.C. Conner, P.R. Kedrowski, C.F. Reinholtz, J.S. Bay: Improved dead reckoning using caster wheel sensing on a differentially steered three-wheeled autonomous vehicle, Proc. SPIE Mobile Robots XV Telemanip. Telepresence Technol. VII (2000)

58.24 S.-D. Kim, C.-H. Lee, S.-J. Yoon, H.-K. Jeong, S.-C. Kang, M.-S. Kim, Y.-K. Kwak: Variable configuration tracked mobile robot for demining operations, ASME Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf. (2004)

58.25 K. Nonami, R. Yuasa, D. Waterman, S. Amano, H. Ono: Preliminary design and feasibility study of a 6-degree of freedom robot for excavation of unexploded landmine, Autonom. Robots **18**(3), 293–301 (2005)

58.26 Y.-J. Lee, S. Hirose: Three-legged walking for fault-tolerant locomotion of demining quadruped robots, Adv. Robotics **16**(5), 415–426 (2002)

58.27 T. Wojtara, K. Nonami, H. Shao, R. Yuasa, S. Amano, D. Waterman, Y. Nobumoto: Hydraulic master-slave land mine clearance robot hand controlled by pulse modulation, Mechatronics **15**(5), 589–609 (2005)

58.28 N. Furihata, S. Hirose: Development of mine hands: Extended prodder for protected demining operation, Autonom. Robots **18**(3), 337–350 (2005)

58.29 P. Debenest, E.F. Fukushima, Y. Tojo, S. Hirose: A new approach to humanitarian demining: Part 2: Development and analysis of pantographic manipulator, Autonom. Robots **18**(3), 323–336 (2005)

58.30 H. Yabushita, M. Kanehama, Y. Hirata, K. Kosuge: 3-D ground adaptive synthetic aperture radar for landmine detection, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)

58.31 A.A. Faust, R.H. Chesney, Y. Das, J.E. McFee, K.L. Russel: Canadian teleoperated landmine detection systems. Part I: The improved landmine detection project, Int. J. Syst. Sci. **36**(9), 511–528 (2005)

58.32 A.A. Faust, R.H. Chesney, Y. Das, J.E. McFee, K.L. Russel: Canadian teleoperated landmine detection systems. Part II: Antipersonnel landmine detection, Int. J. Syst. Sci. **36**(9), 529–543 (2005)

58.33 K.N. Zachery, G.M. Schultz, L.M. Collins: Force protection demining system (FPDS) detection subsystem, Proc. SPIE Detect. Remediat. Technol. Mines Minelike Targets X (2005)

58.34 J. Coronado-Vergara, G. Avina-Cervantes, M. Devy, C. Parra: Towards landmine detection using artificial vision, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2005)

58.35 J.P. Trevelyan: *Demining Research at the University of Western Australia*, http://www.mech.uwa.edu.au/jpt/demining (Univ. Western Australia, Perth 2000)

58.36 S. Rajasekharan, C. Kambhampati: The current opinion on the use of robots for landmine detection, IEEE Int. Conf. Robotics Autom. (2003)

58.37 J.P. Trevelyan: Reducing Accidents in Demining: Achievements in Afghanistan, J. Mine Action **1**, 10–17 (2000)

58.38 T. Lardner (Ed.): *A Study of Manual Mine Clearance: Part 4 – Risk Assessment and Risk Management* (Geneva International Centre for Humanitarian Demining, Geneva 2005)

58.39 P. Newnham, D. Daniels: The market for advanced humanitarian mine detectors, Proc. SPIE Detect. Remediat. Technol. Mines Minelike Targets VI (2001)

58.40 J. Vertut, P. Coiffet: Teleoperation and robotics, evolution and development, Robot Technol. **3A**, 302–307 (1985)

58.41 L.I. Slutski: *Remote Manipulation Systems: Quality Evaluation and Improvement* (Kluwer, Dordrecht 1998)

58.42 K.F. Kraiss: *Advanced Man–Machine Interaction: Fundamentals and Implementation* (Springer, Berlin, Heidelberg 2006)

58.43 E.G. Johnsen, W.R. Corliss: *Human Factors Applications in Teleoperator Design and Operation* (Wiley, New York 1971)

58.44 T.B. Sheridan: *Telerobotics, Automation, and Human Supervisory Control, Technology and Industrial Arts* (MIT Press, Cambridge 1992) p. 432

58.45 W.R. Hamel: Sensor-based planning and control in telerobotics. In: *Control in Robotics and Automation, Sensor-Based Integration*, ed. by B.K. Ghosh, N. Xi, T.J. Tarn (Academic, New York 1999) pp. 285–309

58.46 J.V. Draper: Effects of force reflection on servomanipulator performance, Int. Top. Meet. Robotics Remote Handl. Hostile Environ. (1987)

58.47 R. Goertz: Manipulator system development at ANL, 12th Conf. Remote Syst. Technol. (1962)

58.48 R. Goertz: Some work on manipulator systems at ANL, past, present and a look at the future, Sem. Remot. Oper. Spec. Equip. (1964)

58.49 D.P. Kuban, H.L. Martin: An advanced remotely maintainable servomanipulator concept, Natl. Top. Meet. Robotics Remote Handl. Hostile Environ. (1984)

58.50 D. Sands: Cost effective robotics in the nuclear industry, Ind. Robot **33**(3), 170–173 (2006)

58.51 S. Sanders: Remote operations for fusion using teleoperation, Ind. Robot **33**(3), 174–177 (2006)

58.52 B.L. Luk, K.P. Liu, A.A. Collie, D.S. Cooke, S. Chen: Teleoperated climbing and mobile service robots for remote inspection and maintenance in nuclear industry, Ind. Robot **33**(3), 194–204 (2006)

58.53 P. Desbats, F. Geffard, G. Piolain, A. Coudray: Force-feedback teleoperation of an industrial robot in a nuclear spent fuel reprocessing plant, Ind. Robot **33**(3), 178–186 (2006)

58.54 B. De Jong, E. Faulring, J.E. Colgate, M. Peshkin, H. Kang, Y.S. Park, T. Ewing: Lessons learned from a novel teleoperation testbed, Ind. Robot **33**(3), 187–193 (2006)

58.55 M. Rennich, E. Bradley, T. Burgess, V. Graves: Spallation neutron source remote handling implementation, 1st Jt. Emerg. Prep. Response/Robotics Remote Syst. Top. Meet. (2006)

58.56 S. Kawatsuma, M. Fukushima, T. Okada: Emergency response by robots to Fukushima-Daiichi accident: Summary and lessons learned, Ind. Robot **39**(5), 428–435 (2012)

58.57 S. Hirose, H. Kuwahara, Y. Wakabayashi, N. Yoshioka: The Mobility Design Concepts/Characteristics and Ground Testing of an Offset-Wheel Rover Vehicle, Int. Conf. Mobile Planet. Robots Rover Roundup (1997)

58.58 T. Kagiwada: Robot design for stair navigation, Jpn. Soc. Mech. Eng. Int. J. C **39**(3), 629–635 (1996)
58.59 R. Volpe, R. Ohm, R. Petras, J. Welch, J. Blaram, R. Ivlev: A prototype manipulation system for Mars rover, IEEE/RSJ Int. Conf. Intell. Robots Syst. (1997)
58.60 R. Siegwart, P. Lamon, T. Estier, M. Lauria, R. Piguet: Innovative design for wheeled locomotion in rough terrain, Robot. Autonom. Syst. **40**, 151–162 (2002)
58.61 K. Yoneda, Y. Ota, S. Hirose: Development of a high-grip stair climbing crawler with hysteresis compliant blocks, 4th Int. Conf. Climbing Walk. Robots (CLAWAR) (2001)
58.62 S. Hirose, T. Sensu, S. Aoki: The TAQT carrier: A practical terrain-adaptive quadru-track carrier robot, IEEE/RSJ Int. Conf. Intell. Robots Syst. (1992)
58.63 H. Schempf, E. Mutschler, C. Piepgras, J. Warwick, B. Chemel, S. Boehmke, W. Crowley, R. Fuchs, J. Guyot: Pandora: Autonomous Urban Robotic Reconnaissance System, IEEE Int. Conf. Robotics Autom. (1999)
58.64 J.B. Coughlan: *Small All Terrain Mobile Robot* (Remo Tec, Clinton 1991)
58.65 T. Takayama, S. Hirose: Development of Souryu-I connected crawler vehicle for inspection of narrow and winding space, 26th Annu. Conf. IEEE Ind. Electron. Soc. (2000)
58.66 C.H. Cho, W.S. Lee, S. Kang, M.S. Kim, J.B. Song: Rough-terrain negotiable mobile platform with passively adaptive double-tracks and its application to rescue missions, Adv. Robotics **19**(4), 459–475 (2005)
58.67 K. Kato, S. Hirose: Development of the quadruped walking robot, TITAN-IX-mechanical design concept and application for the humanitarian demining robot, Adv. Robotics **15**(2), 191–204 (2001)
58.68 iRobot Inc.: http://www.irobot.com/
58.69 E. Cheung, V.J. Lumelsky: Proximity sensing in robot manipulator motion planning: System and implementation issues, IEEE Trans. Robot. Autom. **5**, 740–751 (1989)
58.70 B.R. Shetty, M.H. Ang Jr.: Active compliance control of a PUMA 560 robot, IEEE Int. Conf. Robotics Autom. (1996)
58.71 B.-S. Kim, S.-K. Yun, S. Kang, C.-S. Hwang, M.-S. Kim, J.-B. Song: Development of a joint torque sensor fully integrated with an actuator, Int. Conf. Control Autom. Syst. (2005)
58.72 M. Uebel, M.S. Ali, I. Minis: *The Effect of Bandwidth on Telerobot System Performance* (Goddard Spaceflight Center, Greenbelt 1991)
58.73 G. Niemeyer: Telemanipulation with time delays, Int. J. Robotics Res. **23**(9), 873–890 (2004)
58.74 D. Ryu, S. Kang, M. Kim, J.-B. Song: Multi-modal user interface for teleoperation of ROBHAZ-DT2 field robot system, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004)
58.75 A.K. Skula, A.S. Arico, V. Antonucci: An appraisal of electric automobile power sources, Renew. Sustain. Energy Rev. **5**, 137–155 (2001)
58.76 K.W. Ong, G. Seet, S.K. Sim: Sharing and trading in a human-robot system. In: *Cutting Edge Robotics*, ed. by V. Kordic, A. Lazinica, M. Merdan (pro literatur, Augsburg 2005) pp. 467–496
58.77 J. Trevelyan: Redefining robotics for the new millennium, Int. J. Robotics Res. **18**(12), 1211–1223 (1999)

Multimedia Contents

59. Robotics in Mining

Joshua A. Marshall, Adrian Bonchis, Eduardo Nebot, Steven Scheding

This chapter presents an overview of the state of the art in mining robotics, from surface to underground applications, and beyond. Mining is the practice of extracting resources for utilitarian purposes. Today, the international business of mining is a heavily mechanized industry that exploits the use of large diesel and electric equipment. These machines must operate in harsh, dynamic, and uncertain environments such as, for example, in the high arctic, in extreme desert climates, and in deep underground tunnel networks where it can be very hot and humid. Applications of robotics in mining are broad and include robotic dozing, excavation, and haulage, robotic mapping and surveying, as well as robotic drilling and explosives handling. This chapter describes how many of these applications involve unique technical challenges for field roboticists. However, there are compelling reasons to advance the discipline of mining robotics, which include not only a desire on the part of miners to improve productivity, safety, and lower costs, but also out of a need to meet product demands by accessing orebodies situated in increasingly challenging conditions.

59.1 **Modern Mining Practice** 1550
59.1.1 Stages of Mining 1550
59.1.2 Technology Drivers in Mining 1551
59.2 **Surface Mining** 1552
59.2.1 Automated Haulage 1552
59.2.2 Fleet Management 1553
59.2.3 Robotic Digging 1554
59.2.4 Robotic Dozing 1555
59.2.5 Autonomous Blasthole Drilling .. 1556
59.2.6 Telerobotic Rock Breaking 1556
59.2.7 Automated Loading Unit and Truck Interactions 1557
59.2.8 Dragline Automation 1558
59.2.9 Machine Positioning and Terrain Mapping 1560
59.2.10 Mine Safety 1561
59.3 **Underground Mining** 1562
59.3.1 Telerobotic Operations 1562
59.3.2 Autonomous Tramming 1563
59.3.3 Robotic Loading 1564
59.3.4 Longwall Automation 1565
59.3.5 Robotic Explosives Loading 1566
59.3.6 Underground Mapping, Surveying, and Positioning 1567
59.4 **Challenges and Industry Acceptance** 1568
59.4.1 Technological Challenges 1568
59.4.2 Socioeconomic Challenges 1569
59.5 **Challenges, Outlook, and Conclusion** 1569
59.5.1 Technical Challenges 1569
59.5.2 Socio-Economic Challenges 1570
59.5.3 Emerging Frontiers 1570
59.5.4 Conclusion 1571
Video-References 1571
References 1572

There was a time, in recent history, when improved productivity and lower mining costs could be achieved purely by the economies of scale that arise from the use of larger equipment [59.1]. However, this era is likely over. Today's mining companies and equipment manufacturers have been making renewed efforts in the pursuit of new and innovative approaches to the business of resource exploration and mining.

Among these efforts has been a move toward increasingly autonomous mobile equipment and processes. For example, major suppliers of mining equipment for both underground and surface mining operations now offer

robotic driving, dumping, and other materials handling functionalities. What is more, some global mining companies are now operating their own technology-focused divisions, each with goals of bringing new technological developments to their mines worldwide.

Finally, there exist several emerging frontiers for mining, where robotics is already playing a critical role. These include not only very deep and high altitude mining, but also undersea mining and extra-terrestrial mining.

59.1 Modern Mining Practice

Mining is an ancient and broad practice, dating back to Palaeolithic times (43 000 years ago) [59.2] and involves the extraction of materials from the Earth's crust for utilitarian purposes. Today we mine minerals (e.g., gemstones, metals, salt, coal, and many others), aggregates, natural gas, and petroleum. These resources are used, for instance, to make the tools we use, shelter, food products, medicine, and clothing. Try to imagine, for a moment, a life where you endeavored to use (or build a robot from) absolutely nothing derived from mining. We benefit greatly from mining, but there is also increasing pressure and international recognition that the often negative environmental, safety, and social implications of mining and mineral processing must be curtailed and balanced with economic and lifestyle considerations. Robotics can play a vital role in all of these efforts.

In this section, we briefly provide some introductory background about the fundamental stages of mining, common to most operations, and discuss what drives the field of mining robotics, as well as its unique challenges, technical and otherwise.

59.1.1 Stages of Mining

Almost all mining operations have five fundamental stages [59.3]:

1. Prospecting
2. Exploration
3. Development
4. Exploitation
5. Reclamation.

Prospecting involves the initial search for valuable materials. This stage is normally carried out by geologists, and deposits may be located at or below the surface. Prospecting usually happens by either direct visual examination or by indirect methods (e.g., through the use of geophysical techniques) that look for anomalies in seismic, magnetic, electrical, electromagnetic, and/or radiometric variables of the Earth. Geochemistry and geobotany techniques are also employed. Although robotics has not yet played a pivotal role in the search for minerals on the Earth, planetary geologists have been collaborating with roboticists for planetary prospecting; e.g., [59.4] and references therein.

In the *exploration* phase, the objective is to determine as accurately as possible the size (*tonnage*) and value (*grade*) of a deposit. This process is sometimes called *delineation* and more often than not involves collecting samples of material by drilling. Similar to prospecting, robotic exploration drills for mining are not yet common for surface exploration on the Earth. However, there has been a significant amount of work on robotic drilling in extreme environments, such as undersea and planetary exploration; see [59.5] for a good introduction and ample references.

The development and exploitation phases of mining are where tools and techniques from robotics have been, to date, applied most. *Development* is the work necessary to bring a mine into production, and usual entails planning, design, construction (e.g., overburden removal, shaft sinking, underground access) and the installation of mine services (e.g., power, water, ventilation, etc.).

It is during the *exploitation* phase that the resource is physically extracted, processed, and shipped to buyers. How exploitation occurs depends on the selected *mining methods*, which depend on many factors. These include the spatial characteristics of the deposit, geologic conditions, geotechnical conditions, economic considerations, technological considerations (e.g., the availability of advanced tools), and environmental considerations. Generally speaking, mining methods can be divided into two categories:

1. Surface mining methods;
2. Underground mining methods.

Surface-mining methods are used when the deposit lies near the surface, are usually very large scale operations and have high production rates. Underground methods are used when the deposit is deeply buried. Such methods have lower production rates and are usually more technically challenging and hazardous for mine workers.

Figure 59.1 shows examples of robotic vehicles used in surface and underground mining. These technologies, among others, are discussed further in Sects. 59.2 and 59.3.

Mining must strive to meet the economic and environmental needs of the present while at the same time ensuring (or enhancing) the ability of future generations to meet their own needs. Mine *reclamation* is the process of closing a mine, recontouring, revegetating, and restoring the land and water to an acceptable post-activity state. The objective is to minimize any adverse effects on the environment or threat to human health and safety. Some reclamation activities involve the construction and monitoring of *tailings* impoundments, where waste from mining and resource processing activities is placed.

Recently, tools from robotics have been used to conduct bathymetric surveys of tailings ponds, which are unique environments with very specific safety requirements. Figure 59.2 shows a robotic maritime research vessel performing a bathymetric survey of a mine tailings pond at a potash mining operation in eastern Canada. Performing robotic surveys has been shown to be safer and more cost-effective [59.6].

Fig. 59.1 (a) Examples of robotic vehicles in underground and surface mining. Unmanned surface haul trucks (Photo: Mobular Mining Systems, 2011); (b) an autonomous underground load-haul-dump (LHD) machine at the Kvarntorp mine, Sweden (Photo: Joshua Marshall, 2007)

59.1.2 Technology Drivers in Mining

In order to better understand the potential for applications of robotics in mining, one must appreciate what drives the development of equipment technology in the mining industry. Mining is a dynamic process that must deal with the uncertainties of *outdoor* environments, and is usually conducted in as a series of interacting unit operations (e.g., drilling, blasting, loading, hauling, and processing). The harsh, extensive, and unstructured environments found in mining often preclude the application of existing techniques from other industries (e.g., from manufacturing robotics) [59.7, p. 806]. However, the recent growth of mobile and field robotics has resulted in several opportunities for robotics in mining. These opportunities stem mainly from a set of primary technology drivers:

- *Working Environment.* Mines are often developed in harsh and remote areas. For example, several companies operate mines and processing facilities at high altitudes. Also, as mines strive to operate at greater depths, extremely high ambient temperatures and humidity pose greater health risks [59.8]. Robotics may serve to minimize infrastructure needs and physically remove people from such hostile environments.
- *Labor Shortage.* Miners have reported difficulties finding the skilled labor needed to support their operations [59.9]. The newest generation of workers is technologically astute, but holds a different attitude toward physical labor than previous generations. Robotization may be what is needed to entice this new generation into mining.
- *Health and Safety.* With increasing mine depths, growing equipment sizes and speeds, and the tight-

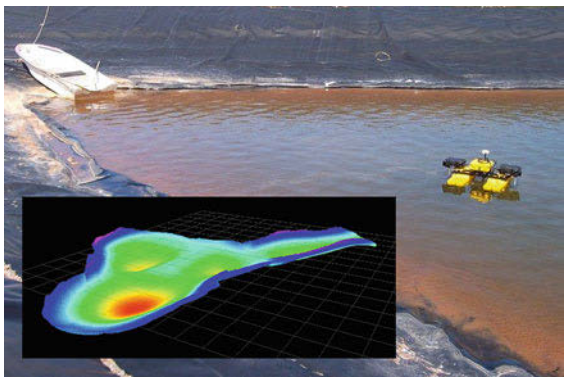


Fig. 59.2 Robotic maritime research vessel conducting a bathymetric survey of a mine tailings pond with overlaid example bathymetry map (Photos: Clearpath Robotics, 2013)

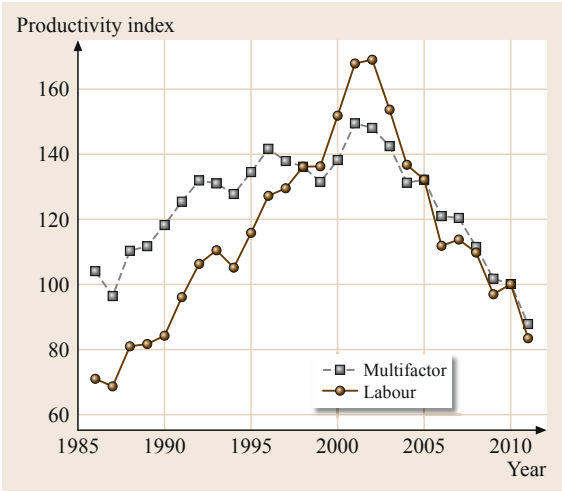


Fig. 59.3 Experimental estimates of multifactor and labor productivity for the mining industry in Australia (1986–2011) showing decreasing productivity since 2001 (after [59.10])

ening of government regulations, it stands to reason that the deployment of almost any new technology that enhances the inherent safety of mining will be viewed positively at all levels.

- *Equipment Maintenance.* Maintenance costs and machine failures can make up a very significant portion of mine operating costs [59.11]. Moreover, operators often push equipment to its performance limits. Recent wisdom from the field of reliability engineering suggests that the automation (or partial automation) of mobile equipment may help to significantly reduce downtime and maintenance costs [59.12].
- *Operational Efficiency.* There are a great number of areas in which efficiency improvements might be realized by the application robotics in mining. The most obvious lies in the fact that production time is lost during shift changes and breaks, resulting in much less than the desired 24/7 runtime. Not as obvious, and more difficult to quantify, is the less-than-optimal way in which underground mines typically operate. Without global and real-time control over the assignment (and movement) of machinery, the ability to optimize production performance through, for example, reduced dilution and increased recovery is also very difficult [59.1].
- *Sustainability.* Environmental and social responsibility are expected of modern mining companies and equipment suppliers. The alleviation of emissions (e.g., through the coordination and optimization of fleets) or reduction in power consumption by reduced operating demands (e.g., ventilation support) might be realized by way of real-time monitoring and automation of equipment.

Figure 59.3 shows multifactor and labor productivity for the mining industry in Australia between 1986 and 2011. Arguably encompassing all of the factors above, this data shows a worrisome decrease in productivity since the turn of the century, and some believe that robotics might help to reverse this trend [59.13].

59.2 Surface Mining

Surface mining is characterized by high tonnage operations, massive mobile equipment, and challenges that can include difficult environments (e.g., dust, fog, and extreme weather) together with remote locales and a strict need for keeping costs low.

59.2.1 Automated Haulage

Haulage is the process of moving material from one location to another. In the surface mining context, this is typically achieved using *haul trucks* (Fig. 59.1a), which move material from the point of blasting/excavation to the point of processing or stockpiling. The material being moved is either the desired economic quantity or overburden, and the destination is usually dependent on the load in the truck.

Haulage is a repetitive process, and the one to which automation can bring significant benefits, particularly with respect to safety. Between 1989 and 1991, haul trucks were involved in 42% of the accidents and 60% of the fatalities in surface mining [59.14]. As with most mining equipment, there are also potential benefits of automation in the areas of productivity, repeatability, and reduced maintenance costs. Although it is too early in the history of robotic mining equipment to precisely define these benefits, the lessons from port automation (which use machines of similar scale, albeit in a more benign environment) have shown that maintenance costs of autonomous straddle carriers (in particular tyre wear and fuel costs) are reduced by approximately one-third [59.15] over their conventionally operated counterparts.

Autonomous Haul Trucks

Autonomous haul trucks can be thought of as mobile robots operating in a difficult, dynamic environment. As

such, they are required to have systems to address the standard issues of sensing and perception, situational awareness, localization, and control. The unique challenges presented in surface mining not present in other robotics applications include:

- Scale (haul trucks can carry up to 400 t of material).
- Mining pits often restrict the view of the sky, which can dramatically decrease global navigation satellite systems (GLS) reliability.
- Surface mining environments are constantly changing, which limits the ability to install fixed infrastructure.
- Mixed manually operated and automated fleets of machines, which places strict requirements on safety and integrity.
- All-weather operational requirements.

In the commercial sector, both Caterpillar and Komatsu had demonstrated the ability to automate haul trucks as early as the mid-1 990s [59.16]. However, it was not until 2008 that the first commercial deployment of Komatsu driverless trucks was realised in the Gaby Mine, operated by Codelco in Chile [59.17]. This was followed closely by a deployment by Rio Tinto at their West Angelas mine in the Pilbara, Western Australia [59.18] (VIDEO 145), where they have operated continuously since and have moved more than 50 million t of waste.

The Komatsu trucks operate using a high-precision global positioning system (GPS) as the primary localization sensor, radar sensors for obstacle detection and situational awareness, a proprietary wireless communications system, and are given tasks (load, haul, dump, etc.) using a proprietary scheduling system developed by Modular Mining Systems [59.18].

At the time of writing, similar systems by Caterpillar [59.19] and Hitachi [59.20] are also in various stages of trial and deployment, although there is scant information available publicly regarding the specifics of their designs and operation.

The automation of mining haul trucks is still an area of active research, however, primarily due to the challenging nature of the mining environment as previously noted. New positioning systems [59.21], situational awareness systems (Sect. 59.2.10), and planning and scheduling algorithms [59.22] are all under active development, and are intended to improve robustness, safety, and productivity, respectively. If each of these challenges can be addressed, robotic haulage technology will be able to applied in a much greater range of operational scenarios than is currently possible.

59.2.2 Fleet Management

Fleet management is a catch-all term that may define a number of different technologies used in mining. Commonly though, fleet management is divided into three main tasks:

1. Position (and perhaps materials) monitoring
2. Production monitoring
3. Equipment task assignment.

Fleet management solutions are typically deployed to an office environment and allow operators to quickly survey large amounts of mine data so that appropriate actions can be taken in real-time on the mine-site. Computerized fleet management must integrate and evolve with the deployment of robotic mining machines.

Position Monitoring

The position of mobile equipment is commonly reported via wireless communications to a mine or region's database, where it can be queried by a visualization application. This provides a real-time picture of the movement of all equipment that has been enabled for such operation and allows an operator to quickly assess whether planned operations are proceeding as expected. Material movements may also be monitored in a similar fashion, but the material type (ore or waste) and grade must usually be entered manually by an operator. However some parameters, such as weight, are measured by the system.

Production Monitoring

Given that vast quantities of data may exist (particularly from robotic machines) that give a historical account of the evolution of a mine, or group of mines, this data can be used to provide detailed reports to mine management, and may also be useful as part of a continual productivity improvement process. Examples of production monitoring data may include machine cycle times, average maintenance cycles, machine failure events, payloads, or combinations of parameters (e.g., machine failure events vs. machine payload weight).

Equipment Task Assignment

At a typical mine, the equipment task assignment problem is addressed in a manner very similar to that of the urban taxi dispatch problem: A centralized authority decides, based on some predefined high-level goals, which machines must visit certain locations on the mine-site and perform some useful action. A machine

operator is generally given these instructions through a user interface located in the machine's cabin, which must be acknowledged and accepted prior to carrying out the task. The most typical implementation of the task assignment problem addresses the haulage scenario: when should a particular haul truck approach a given shovel, and when and where should it dump its subsequent load? Haulage is most commonly addressed, because it is here that even small changes in efficiency can have significant effects on overall profit.

Commercial Solutions

There are many commercial solutions for fleet management in the surface mining. The dominant solution is Modular Mining's DISPATCH system [59.23], which was developed in the 1970s. It implements an optimization algorithm to allocate trucks to shovels and dump sites, and contains modules for the other tasks of production and positioning monitoring. Modular Mining joined the Komatsu group of companies in 1996 and helped to deploy the FrontRunner autonomous haulage system described previously [59.24].

Other commercial systems include Caterpillar's MineStar software [59.25], Wenco's fleet management solution [59.26], Leica Geosystems Jigsaw suite of software and hardware [59.27], and Devex's SmartMine system [59.28]. These systems all now offer very similar overall functionality and differ mainly in implementation details.

Fleet Management Research

Although commercial systems for fleet management exist, there is still a substantial amount of research being undertaken, primarily in the area of task assignment optimization. Most commercial systems optimize task allocation using a heuristic that defines costs for allowing a shovel to become *idle* and/or allowing haul trucks to *wait* in a queue. In an ideal or optimal system, all items of equipment are fully utilized. However in practice, a higher cost is usually given to shovel idle time. The paper [59.29] gives a good overview of many of the commonly used heuristics used in task allocation optimization.

More recently, it has been acknowledged that heuristic, deterministic optimizers may not always produce truly optimal task assignments. This is primarily due to their inability to account for variability in the duty cycles of the items of equipment (i. e., they are stochastic rather than deterministic in nature), and any unforeseen discrete events, such as equipment failure, road blockage or operator error. The papers [59.22, 30–33] all describe recent approaches to overcoming these hurdles.

59.2.3 Robotic Digging

Mass excavation is one of the most important operations in mining. Being extremely sensitive to economies of scale and requiring a significant capital investment, it became a target application for robotic technologies starting in the mid-1990s. Surface mining typically uses large hydraulic excavators, hydraulic shovels, and electric rope shovels (Fig. 59.4). See also Sect. 59.3.3, which focuses on underground robotic loading.

Research in robotic digging involves systems design (including systems of systems), traditional robotics topics (sensing, planning, control), and tool-ground interaction problems. A framework for an intelligent earthwork system is suggested by [59.34], which discusses factors that can affect earthwork operation performance, identify key emerging technologies to support the implementation, the system architecture, and the system control strategy. A comprehensive review by [59.35] examined various aspects of sensing, planning, and control as they apply to earthmoving automation. Later developments in these topics are further discussed in [59.36]. Irrespective of the manned/unmanned aspect of the operation, a typical digging phase involves a number of critical steps: planning the bucket trajectory in contact with the ground, bank or muck pile; detecting when the bucket is full; and, detecting incipient stall and averting it. An analysis of autonomy requirements for each of these steps is detailed in [59.37] for the particular case of an electric rope shovel.

In other work, authors of [59.38, 39] proposed an autonomous excavation system for front-end-loader style machines that uses bucket force feedback, fuzzy logic, and neural networks for control. In their approach, a set of fundamental bucket action sequences,

Fig. 59.4 A rope shovel dumping into a haul truck (courtesy of Commonwealth Scientific and Industrial Research Organisation (CSIRO), 2006)

typically used by human operators, was compiled for use by the controller. A reactive approach, using fuzzy behaviours, was designed to act on force data to assess the excavation status and determine an appropriate control input. Experimental results, using a programmable universal machine for assembly (PUMA) 560 arm, were reported. Many more researchers have studied the robotic loading problem for homogeneous materials, such as soils and regolith. The majority of researchers have advocated the use of manipulator impedance or compliant motion control [59.40–43] and much of this work has focused on estimating the properties of the media [59.44] and on computing the resistive forces acting on the budget during loading [59.45].

Invariably, all researchers agree that the biggest challenge to robotic excavation is the interaction between the tool and the terrain (ground, pile). This interaction is shaped by the properties of the media (e.g., density and hardness), the rock pile geometry, and the distribution of particle sizes and shapes. The problem is compounded by the fact that it is difficult to predetermine the exact nature of interactions prior to the execution of any particular excavation operation [59.46]. Significant effort was dedicated to modeling the interaction between the bucket and the terrain in excavation. This requires estimating the soil parameters online, since these parameters can vary significantly with the strata being excavated. The classical fundamental equation of earthmoving [59.47] used to determine the forces required for digging will fail to predict these forces if the soil parameters are not estimated correctly. Models for automated excavation using online estimation of soil parameters are discussed in [59.48].

Situational awareness, which includes both workspace and machine awareness, is another key aspect of robotic digging. Workspace awareness implies a knowledge of the terrain surrounding the digging machine. At a primary level, this knowledge is gathered using a sensor suite retrofitted to the machine, which in a typical configuration consists of a number of ranging sensors, an inertial measurement unit, and a GNSS unit. The acquired raw data is subsequently processed by higher level processes to detect the location and configuration of the dig and dump regions, and the location of other mobile equipment operating the workspace (e.g., haul trucks, utility vehicles). Laser rangefinders offer adequate resolution to segment the scene. An example of a workspace image in a form of a three-dimensional (3-D) point cloud acquired with a long range laser scanner installed on a shovel is shown in Figure 59.5. Color is assigned to the points based on intensity returns from the laser scanner.

Machine self-awareness is primarily concerned with determining the position and orientation of the bucket with respect to a referential fixed with respect to the machine. It also includes all the usual status data for the main onboard systems: power plant, transmission, tool actuation, etc.. The position and orientation of the bucket are usually available from encoders and inclinometers. A better solution, however, is to use suitably placed ranging sensors that are able to track the bucket in all positions and orientations [59.49]. This latter method avoids the errors typically associated with backlash in joints and dynamic effects in inclinometers. The placement of the ranging sensors on the machine should consider the need to provide an image of the ground under the boom.

Autonomous digging systems offer the additional benefit of enabling dig-to-plan operation and reconciling actual versus planned volumes by providing accurate real-time survey information. In surface mining operations, the floor is usually mined and therefore disturbed before accurate surveys are executed, and the floor is graded on an as-needed basis to facilitate the movement of the mobile equipment operating in the pit [59.49].

59.2.4 Robotic Dozing

Dozers are used in most surface mines. Operating a dozer under conditions that are dynamic, with hazards that can vary and be subtle and difficult to recognize, adds to the complexity of the tasks that workers perform on these machines. Many of these tasks require oper-

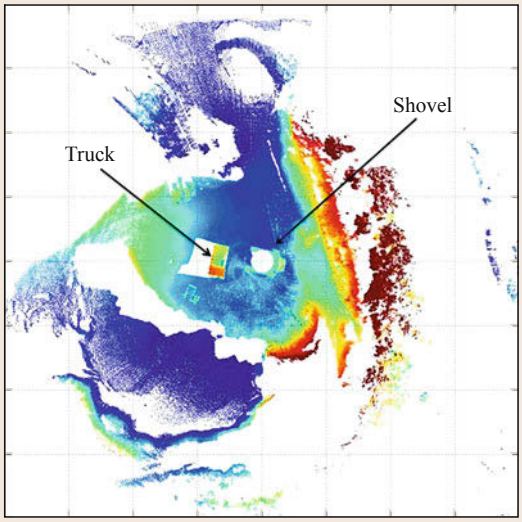


Fig. 59.5 A top view of a 3-D point cloud coloured by intensity returns from a long range laser scanner installed on a mining shovel loading a haul truck (courtesy of CSIRO, 2008)

ators that are highly skilled in ground-engaging tool control and machine positioning. In addition, dozer operators are exposed to a variety of risk factors that may lead to health problems, including whole-body vibrations, awkward postural requirements, noise, and shift work. While machine guidance systems are now ubiquitous, they still require an operator in the cab and have known problems associated with their reliance on GNSS-based localization. As alternatives, some equipment manufacturers offer tele-remote-controlled and autonomous dozers, which eliminate the need for an onboard operator.

Remote control solutions for dozers have been in use for two decades. They cannot perform in poor visibility conditions (dust, fog, night operation, heavy rain) and do not offer the possibility of automating machine guidance and tool control tasks otherwise available if operating the dozer from the cab. Other problems include loss of situational awareness, inaccurate attitude judgement, and inaccurate depth perception. More recent commercial remote control offerings include force feedback. In the case of tele-operated control with negligible latencies, imparting force feedback to the operator has the benefit of allowing the operator to *feel* and control the machine.

In view of the limitations of remote control technologies, research on dozer automation has focused on semi-autonomous and autonomous system retrofits. Autonomous systems have already made significant inroads in the mining industry, and the acceptance for this technology is growing. Leica Geosystems offers an autonomous ripping solution for track and wheel dozers. Their system, called Leica J^3dozer autorip, provides a number of automatic functions, which include engine start/stop engine, automatic control for steering, throttle, transmission and ripper, accessories control, and an emergency fail-safe system [59.50]. The system also incorporates a path planner based on area coverage algorithms.

59.2.5 Autonomous Blasthole Drilling

Blasthole drilling is a mining process typically employed during the exploitation phase of surface mining. To excavate a material it must first be blasted to fracture the rock mass into small volumes suitable for haulage and processing. This is achieved by drilling a matrix of holes from the surface down into the rock mass; these holes are then loaded with explosives and blasted. The resultant fractured rock mass can be excavated by a variety of machines. The matrix of holes (or *drill pattern*) and the amount and type of explosives used are optimized to produce a desired rock size distribution in order to minimize the cost of processing; however, there can be significant variation due to the variable nature of the underlying geology.

Blasthole drill rigs are usually large tracked vehicles with an articulated *mast* that contains the main drilling components: the drill motor, drill rods, and drill bit. To drill to depths longer than the mast, drill rods can be concatenated to form a long *drill string* as drilling is progressing. Like many home drills, most blasthole drill rigs are capable of rotary drilling, as well as hammer (or percussion) drilling. Rotary is typically preferred for softer materials, with hammer drilling primarily used for harder materials.

There are several facets of drilling amenable to automation: tramming (the process of moving the entire rig from one hole location to another), drilling, and the management of drill consumables (such as rod or bit changes). A *fully autonomous* drill rig should have all of these functions automated. The promise of automation is increased productivity and more consistency in the placement and angle of drill holes, which in turn should lead to better blasting outcomes. Consistency in post-blast fragmentation results in a significant economic impact.

There are several high-profile mining equipment manufacturers and companies that have developed fully automated drill rigs. Atlas Copco's Rig Control System (RCS) has reportedly made significant impact in increasing average tramming speeds (from 0.8 m/min to 2.6 m/min) and drilling efficiency (a savings of 1 h per 12 holes drilled) [59.51]. Flanders [59.51], and Rio Tinto [59.52] also have fully automated blast hole drill rigs in production.

59.2.6 Telerobotic Rock Breaking

Drilling and blasting results are often far from ideal and the resulting fragments of rock (called *shot muck* or just *muck*) can be larger than the projected size. Some of the shot muck can be too big for handling with a digging machine, and thus requires a secondary blasting. Other boulders can be handled by the digging and hauling machines but are still too large to fit in the mouth of a primary crusher. In this latter case, rockbreakers are used to achieve a further reduction in ore and rock size.

A rockbreaker consists of a large 4-DOF (degree of freedom) serial link manipulator arm that is fitted with a hydraulic hammer (Fig. 59.6), and is usually positioned in front of a run-of-mine (ROM) bin. The bin is fitted with horizontal bars at the bottom that prevent oversized rocks from entering the crusher below (this arrangement is called a *grizzly*). The size of the grill openings is made in such a manner that only rocks small enough to be crushed may pass through. Larger rocks do not fit and therefore must be broken. Haul trucks car-

Fig.59.6a,b Rock breaker at rest over the ROM bin **(a)** and breaking a rock on grizzly **(b)** (courtesy of CSIRO, 2007)

rying ore from a nearby quarry dump their load into the ROM bin. Typically, an operator uses a line-of-sight remote control to operate the arm.

Rio Tinto is pursuing a programme to automate the vast majority of its Pilbara iron ore operations in Western Australia, and control the operations from the Remote Operations Centre (ROC) in Perth, situated over 1000 km away from the Pilbara region. This effort includes the development of a telerobotic control system for the primary rockbreaker at the West Angelas mine in collaboration with the CSIRO [59.53]. This system addresses known drawbacks in teleoperation (limited communication bandwidth, high latency in the real-time video feedback, lack of spatial situational awareness, etc.) by improving the intelligence of the control system at the remote/machine end (e.g., Cartesian motion and collision avoidance capability) and by providing the operator with a mixed-reality interface that combines live video with 3-D computer visualization.

Each rockbreaker is fitted with a number of sensors to detect its pose in space, and actuated by signals that bypass the existing remote-control system. PTU (pan, tilt, zoom) cameras are fitted to poles on either side of the ROM bin and a pair of high-resolution digital stereo video cameras were mounted below. At the remote end, in the control room, the operator is presented with an overview of the rockbreaker from a wide-angle video stream augmented by a synthetic computer image (Fig. 59.7). The operator is able to *walk around* the rockbreaker in the virtual world and inspect the rocks from different angles, enabling the operator to determine the appropriate breaking strategy. The operator deploys the arm using commands from a joystick, and as the arm is commanded to move, the motion of the arm is replicated in the 3-D scene. Simultaneously, both PTZ cameras follow the tip of the hammer. When the operator is ready to break the rock, he/she can switch his/her attention to the live video stream, which can be used to monitor the breaking of the rock. Once complete, the arm can be automatically sent to the rest position.

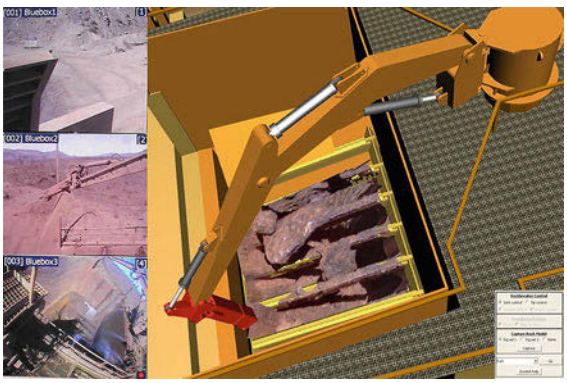


Fig. 59.7 A mixed reality user interface combines live video and 3-D stereo reconstruction of the rocks above the grizzly (courtesy of CSIRO, 2008)

59.2.7 Automated Loading Unit and Truck Interactions

The main loading units used in surface mining are electric and hydraulic shovels and hydraulic excavators. Given the large volumes of material that need to be moved, and the comparatively small size of loading and hauling units, cycle times are closely monitored. In the continuous pursuit of efficiency, mining engineers dissect cycles to look for potential savings. One area of interest is truck spotting, which is the process of manoeuvring the haul truck into a position and orientation that is ideal for the loading unit to dump material into the truck's tray (Fig. 59.4).

The ideal truck placement for loading has both productivity and safety implications. When a mining shovel is used for loading, the haul truck must be positioned in such a manner that the swing motion of the shovel is minimized and that the shovel boom does not need to lower excessively to reach the tray. If the haul truck is not placed optimally, the loader needs to reposition itself to tip with a full load in the bucket, which results in lost time. Lowering the loader arm excessively and/or an uneven distribution of load in the haul truck

tray may result in injuries to the operators and property damage.

In open-pit mining, the shovel operator usually guides the haul trucks to the desired location (spotting) by placing the bucket in the location that is optimal for the dumping phase, and waits for the truck to move into position. A key requirement for achieving automatic spotting is the availability of a robust localizer, that determines the relative pose between the shovel and the truck. Currently, most of the commercial guidance solutions are based on the use of GNSS and GNSS-aided inertial navigation systems on both the shovel and the truck. This results in an indirect determination of the relative pose between the shovel and the truck.

These systems, however, do not provide a robust solution. For the haul truck, the proximity to the shovel boom and crowd arm can lead to a degradation of the GNSS solution due to multipath, while the placement of the truck near highwalls leads to a partial obstruction of the sky, with a loss of satellite tracking and a significant geometric dilution of precision. The guidance system on the shovel can be impacted detrimentally by the proximity to highwalls. Imaging sensors (e.g., laser scanners) can be installed on both the shovel for a direct determination of the relative pose between the shovel and the truck [59.55]. However, this solution has potential pitfalls, that are related to the sensitivity of the imaging sensors to environmental conditions such as dust, rain, and variable illumination during the day (or during the 24 hour day/night cycle in cases where cameras are used).

Some researchers have looked at modalities that mitigate these pitfalls. One possible solution, suggested in [59.54], is to divide the space around the shovel into three zones: the outer zone, the handover zone, and the work zone (Fig. 59.8).

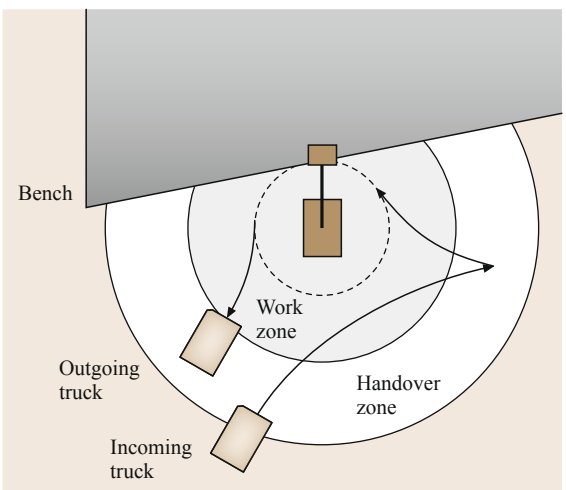


Fig. 59.8 Traffic management zones around a shovel in a double–spotting configuration, when the shovel loads on both sides (after [59.54])

The relative size of the zones depends on the truck and shovel geometries, and the chosen safe operating speed limits within each zone. As the truck crosses into the handover zone from the outer zone, the spotting system flags the truck as active and begins sending commands to it. In this zone, the relative pose between the shovel and truck is determined indirectly, based on the method discussed above. In the handover zone the truck is sufficiently far from the shovel and the highwall, and therefore the localization errors due to multipath and sky occlusion are minimized. Once the vehicle crosses into the work zone, the spotting system begins computing the relative pose using the imaging sensors on the shovel.

Since the sensors on the shovel are also able to see the bench and other obstacles in the work zone, the shovel is able to direct the truck around obstacles as they appear in the work zone, and park the trucks as close to the bench as practical. In other words, since the shovel is actively aware of its environment, it is able to deal with changes to the environment as they occur. The spotting system generates an approach path that will bring the truck in the optimum position for loading, where this optimum is determined in relation to the characteristics of the shovel and the configuration of the dig face. The path is also optimized with respect to the positioning time. Spotting times may be further minimized by double spotting, when the shovel loads on both sides, as depicted in Fig. 59.8.

The above discussion focused on shovel–truck interactions. Similar issues aise when hydraulic excavators are used for loading. Comprehensive research results in autonomous truck loading using hydraulic excavators are discussed in [59.56].

59.2.8 Dragline Automation

Mining *dragline* excavators are massive electrically powered machines used in open-pit coal mining to remove overburden and uncover coal. A dragline comprises a rotating platform that supports the house (consisting of the engine room and operator cabin), boom and a bucket rigging structure, as illustrated in Fig. 59.9. The house rotates on a base known as the tub which rests on the ground. The mechanism has three degrees of freedom:

1. Rotation with respect to the tub;
2. Hoist by a cable passing over sheaves at the tip of the boom;

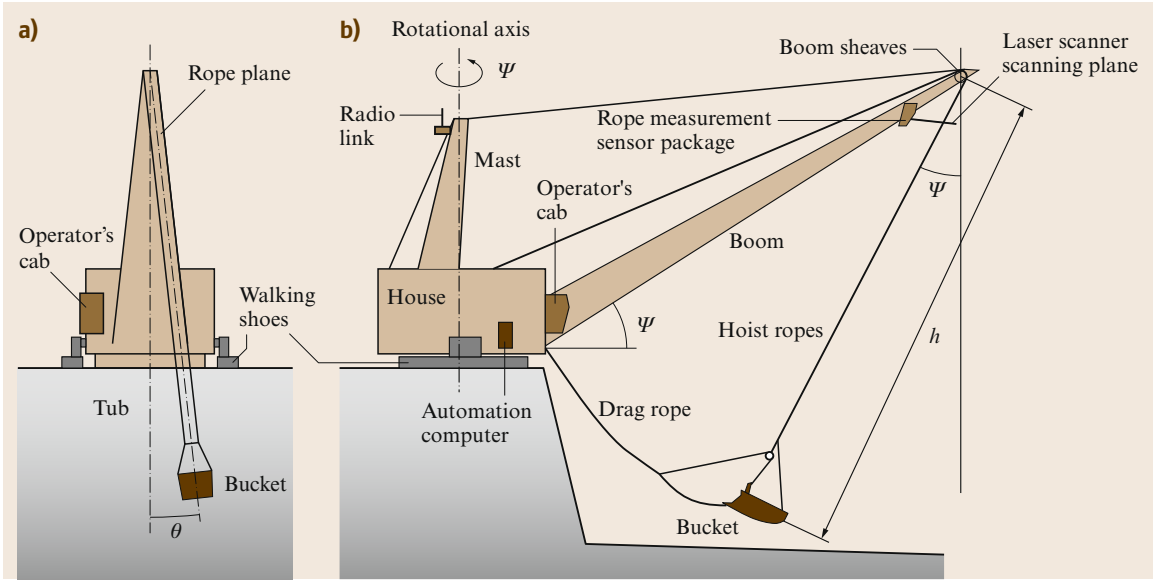


Fig.59.9a,b Schematic of a dragline excavator **(a)** front view **(b)** side view (after [59.2])

3. Drag by a cable passing over sheaves at the base of the boom.

Digging is controlled using only the drag and hoist ropes. When the bucket is filled it is hoisted clear of the ground and swung to the dump position by swinging the house and boom. The human operator fills the bucket by dragging it through the ground, lifts and swings it to the spoil pile, dumps it, and then returns to the dig point. Each cycle moves up to 100 t of overburden, and typically takes 60 s.

High capital costs are pushing the owners of these machines to continually look for avenues to maximize productivity (m^3/shift) in the presence of operator variability and maintenance costs (around 30% of operating cost) [59.2]. From this perspective, dragline operations seem ripe for deploying robotic technologies. However, the idea of completely removing the human operator from the dragline seat, while tempting from a research perspective, it is not widely entertained in the industry.

Swing control and digital terrain mapping are the two robotic technologies at the core of the dragline automation efforts. During dragline operation, the bucket naturally tends to swing with respect to the boom. A skilled operator can coordinate the boom slew motion so as to control the natural tendency of the bucket to swing during dumping and spotting operations. A novice operator typically requires six months training to become proficient. This warrants the development of a swing automation system to improve the productivity of the dig-to-dump and dump-to-dig phases of a dragline operation. Automatic swing controllers offer the additional benefit of minimizing stress fatigue on the boom.

An implementation of a dragline swing controller is discussed in [59.57]. The control system moves the bucket between operator-specified dig, and dump points, passing through intermediate (via) points. The rather rigid programming of control (dig, via, and dump) points has a number of disadvantages. Firstly, as the spoil pile grows the dump point must be raised. If the automation system does not contain a model of spoil growth, the operator has to make adjustments by retraining the system, resulting in time losses. Secondly, if the operator does not correctly identify a via point at a critical area, such as the top of the high wall, the swing automation system will not know the high-wall is there and may attempt to swing the bucket through it. Augmenting the automation system with a digital terrain mapping capability provides a common solution to these problems.

One implementation of a digital terrain mapping solution for draglines is described in [59.58]. Maps were created using data from an real-time kinematic (RTK) GNSS system and a two-dimensional (2-D) laser scanner. The laser scanner was mounted at boom tip in such a way that the scanning plane intersected the ground underneath the boom, and tilted by about 7 ° with respect to the vertical. Tilting the laser minimizes the likelihood of the bucket and ropes blocking the laser's view. The terrain is scanned as the dragline swings during the normal operation cycle, a typical map is shown in Fig. 59.10.

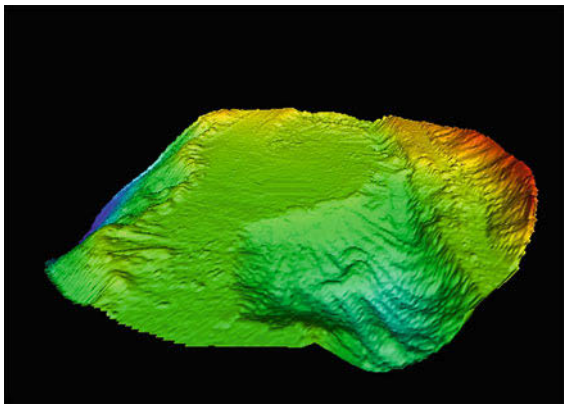


Fig. 59.10 Image of a digital terrain map created from a long range laser scanner installed at the tip of a boom dragline (courtesy of CSIRO, 2005)

The implementation of a digital terrain mapping function in dragline automation systems also enables the measurement of dragline productivity in real time. Commercial dragline monitoring systems can only measure the approximate weight of each bucket, and the approximate disengage and dump points. They cannot provide indications of where the where the spoil is located in relation to the dragline, of the actual volume of overburden moved, or how much re-handle took place [59.58].

59.2.9 Machine Positioning and Terrain Mapping

The task of establishing and maintaining the vehicle positioning (i. e., position and orientation) relative to an external frame of reference is a fundamental problem in autonomous vehicle navigation and control, and this hold true for mining applications. High-accuracy positioning technology is a dynamic area of research and development, spurred on by the advances in satellite positioning as well as by the current challenges for the technology in mining systems, especially in surface mining operations. The following initiatives are underway to improve the availability and robustness of current technologies, which have and will further enable the application of robotics in surface mining:

- Public GNSS systems (e.g., GPS and GLS (global navigation satellite system))
- Augmented public GNSS systems (e.g., GPS with WAAS (wide area augmentation system))
- Locally augmented GNSS systems (e.g., DGPS (differential global positioning system) or RTK GPS)
- Pseudolite augmented GNSS systems (i. e., through the installation of Earth-based transceivers)
- Closed systems (i. e., wholly proprietary systems relying on a constellation of locally installed Earth-based pseudolites).

For example, in the mining domain, Leica Geosystems offers the Jigsaw Positioning System (JPS), a product that offers a significant breakthrough in addressing the major shortcomings of GNSS [59.59]. Based on a technology developed by the Locata Corporation [59.60], JPS is a ground-based network for positioning that in principle mirrors the functionality of GNSS. It circumvents the fallacies of GNSS by using a terrestrial network of transmitters that can be placed in optimal locations with respect to the mobiles being tracked.

Different from pseudolites, an example of technology for positioning in GNSS-denied environments is CSIRO's wireless ad-hoc system for positioning (WASP) wireless tracking technology. The system can be configured as a mesh network and offers a unique combination of high accuracy, high resistance to multipath interference, low-cost hardware, and ability to be rapidly deployed [59.61]. The accuracy of WASP depends upon the radio propagation environment, and is typically better than 0.25 m under line-of-sight conditions, even in the presence of substantial multipath interference. The maximum range over which WASP nodes can measure range and communicate is typically over 400 m, and can exceed 1 km. As WASP can form a mesh network, the overall network size is not limited. The maximum location update rate is 200 Hz; however, there is a tradeoff between the number of nodes and update rate, and a typical update rate is 1–10 Hz.

Terrain Mapping and Surveying

The penetration of robotic and autonomous systems in mining, coupled with new computer aided tools for mine modeling and reconciliation, requires more accurate and timely information about the current state of the mine, and at higher spatial and temporal resolutions. In the traditional surveying framework, one obvious solution is to increase the number of surveys (reduce the time gap between successive surveys) and the density of survey points, while reducing the costs per survey. To address this, a number of vehicle-based scanning systems have been recently introduced to the market. The mode of operation ranges from stop-and-go-type systems (such as the Maptek I-Site), to continuous scanning mobile systems, such as those offered by Riegl (VMX 250), Topcon (IP-S2), and Trimble (MX-8). In general, these systems integrate high-end LIDAR (light detection and ranging) with GNSS-aided INS (inertial

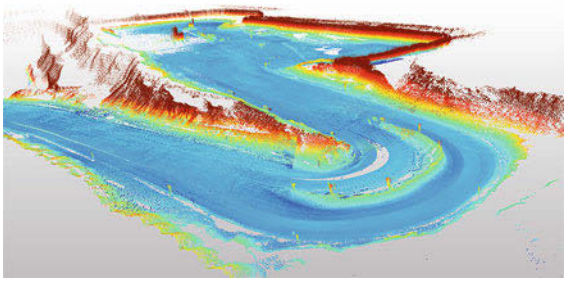


Fig. 59.11 A 3-D map of a surface mine created from a mobile mapping system installed on a four wheel drive (4WD) vehicle (courtesy of CSIRO, 2012)

navigation system) and carry a significant price tag, which can limit deployment to a single dedicated survey vehicle.

On the other hand, consider the application of simultaneous localization and mapping (SLAM) in mining. SLAM is a mature area in mobile robotics and efforts are underway to transfer the technology to mining. Data-fusion techniques used in SLAM make it possible to use lower cost mapping sensors, which in turn enables mining companies to deploy the equipment on a multitude of utility vehicles and/or heavy-duty mining equipment (Fig. 59.11). As a result, the vast majority of survey work can be performed during the normal operation of the equipment in pits, on haul roads, or other site areas, without directly exposing surveying personnel to the perils of conducting survey work in high-risk areas.

59.2.10 Mine Safety

The increase in demand for resources has posed new challenges to increase productivity, while at the same time continually improving the standard of safety. Robotics is beginning to play a role in this. There are inherent risks associated with manually operating vehicles in mining environments. Operators work long shifts and the view from the driver's cabin can be very restricted. Furthermore, vehicle interactions are inherently difficult due to the size of machines, and can be further complicated by environmental conditions such as dust, fog, and snow (Fig. 59.12).

Fig. 59.12 Complexity of mining equipment interactions affected by adverse environmental conditions (courtesy of CSIRO, 2012)

Each year there are hundreds of mine haulage accidents that result in significant numbers of injuries, even deaths, and large financial costs due to machine downtime and repair of equipment. According to the United States Mine Safety and Health Administration (MSHA) [59.62], from 2002 to 2008 the United States coal industry alone averaged 1206 accidents per year involving mobile equipment.

During the past few years, a number of approaches have been introduced to assist operators in becoming more aware of safety risks in the local environment. These technologies and methods include proximity awareness technology (PAT), proximity detection technology (PDT) and collision avoidance technology (CAT). It is important to note that, in surface mining, these existing technologies do not perform any control action. They merely provide information to the driver, who remains in total control of the vehicle.

Detection of Threats

In general, most safety incidents are from vehicle interactions with other vehicles, fixed infrastructure, or personnel. A large proportion of these incidents happen at low speed. This is most likely to occur on vehicle startup, or at the loading areas with interactions between trucks, shovels, loaders, and other ancillary equipment. These accidents are usually of low impact but can cause serious damage to equipment and significant loss of production.

The fundamental approach to reducing the likelihood of low-speed incidents is to provide the operator of large vehicles with aids to see into blind areas such at side or back of the vehicle. These aids are normally based on standard video or infrared cameras. The next level of risk reduction is achieved by incorporating automatic detection of resources and people in close proximity by utilizing range and bearing sensors. Radar is a preferred sensor modality for bad environmental conditions. Wide beam radars are very common but they are prone to false alarm. New recent developments mitigate this problem by using electronic-mechanical scanning narrow beam radar technology [59.63, 64].

High-speed incidents occur less frequently than at low speed, but often result in serious injuries and fatal-

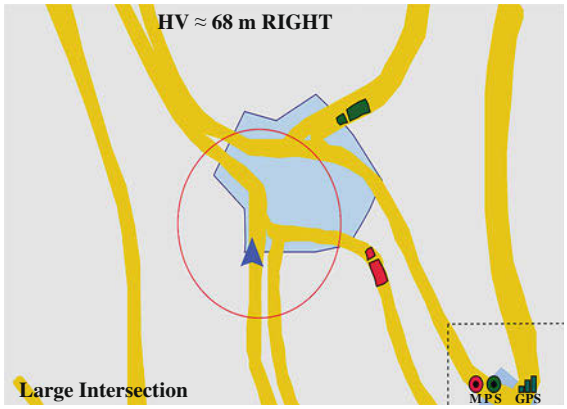


Fig. 59.13 Graphical representation of the state of the world close to the vehicle, including other vehicles in proximity, maximum risk, most likely paths, and risk areas (context) (courtesy of Eduardo Nebot, 2011)

ities. The time available to stop the vehicle or perform an evasive maneuver is reduced and consequently long-range detection of threats is necessary in order to take early preventive action.

This is addressed by incorporating vehicle-to-vehicle (V2V) communication with all nearby vehicles at sufficient range to provide early warning of potential threats. V2V networks can also form a mesh structure. The fundamental advantage of this approach is that no fixed infrastructure is required, and communication between vehicles can be guaranteed anywhere in the site when vehicles are in radio range.

Positioning, Context, and Road Maps

Positioning is one of the fundamental components of a comprehensive safety system. The incorporation of GNSS systems, integrated with velocity and inertial observations, allows for the evaluation of reliable time-to-collision (TTC), especially when combined with haulage road maps information.

Determining the risk of a situation is entirely dependent on the context of the situation. One important situation context is the location, or more specifically the type of area the vehicle is currently operating. The expected behavior and driver intent [59.65, 66] varies depending on whether they are driving on a road, parking in a designated area, or dumping ore in a crusher. Map information is required to allow the system to vary the information provided to the operator depending on the location. An example is shown in Fig. 59.13, where the map allows the operator to become aware of the risks, and complexity of the situation, independent of the weather or geometry of the terrain.

Monitoring and Design

The ultimate goal of any safety system is to find a complete solution to the safety problem by introducing new technology, methods, and algorithms. It is very unlikely that this can be achieved with a single technology alone, perhaps with the exception of a fully autonomous system implementation. An active area of research and development aims to use both reactive and proactive approaches to solve the safety problem. At the reactive level, technology is used to assist the operator in recognizing and avoiding the safety risks. At the proactive level, managers are provided with tools to monitor, analyze, design, and improve the safety of the mining operation. This new comprehensive approach to safety has the potential to eliminate all accidents in the mine by integrating technology, procedures and intelligent algorithms to make the mine safer and more productive.

59.3 Underground Mining

The especially hazardous and generally unpleasant nature of underground mining environments, operator safety and fatigue, labor costs, and the repetitive nature of the ubiquitous load-haul-dump cycle all motivate autonomous and semi-autonomous and teleoperation-based solutions for underground machines and processes.

59.3.1 Telerobotic Operations

In mining, *remote control* typically refers to the control of a machine where the operator has a direct line-of-sight view of the machine. Remote control is normally used only for excavation (or *mucking*), not for driving (or *tramming*) of loaders, for instance. Remote control systems are widely used, for example, when loaders must access underground drawpoints where there is a risk of falling ground.

By remotely operating equipment or through the partially autonomous operation of equipment, for example, it stands to reason that the exposure of personnel to unsafe circumstances might be significantly reduced. This was argued to be true at Inco Ltd. during the 1990s [59.67].

Telerobotic systems are those in which a human remains part of the control loop, although *at a distance* and not generally within direct view of the system being controlled. Many mining companies and equipment

manufacturers are now finding benefit in the application of telerobotics, particularly to underground equipment such as LHD machines (Fig. 59.1b) and drills [59.67–69]. This is normally done by the real-time transmission of video and control signals to and from a remote control room; Figure 59.14, which shows a commercial teleoperation station.

However, teleoperated equipment may not be as productive as manually operated equipment, for example, due to visibility and issues related to reduced operator feedback. Recently, some researchers have proposed the development of semi-autonomous teleoperation systems, whereby the operator's actions are filtered by a machine control system that regulates operator commands with some degree of partial or *local autonomy* [59.70].

59.3.2 Autonomous Tramming

Autonomous *tramming* refers to automating the driving task of underground mining vehicles, which is often a repetitive activity. Most robotic tramming systems have been developed for LHD machines, such as the one shown in Fig. 59.1b, although some suppliers also have systems available for their underground haul trucks.

Several challenges exist that make the autonomous tramming problem difficult. For example, the large inertia of these vehicles and their hydraulically actuated, center-articulated steering mechanisms make them difficult to control at high speeds. Another is the problem of precise and real-time underground localization in the absence of global infrastructure (most notably, GNSS available on the surface).

Underground Navigation

Early autonomous tramming systems worked by outfitting the mine with signal-emitting cables [59.71], reflective strips [59.72], light-emitting ropes [59.73], or reflective tape [59.74, 75]. Other systems required beacons, placed at strategic locations throughout the mine, which allowed the vehicle to estimate its position by measuring angles to the beacons and comparing the measured angles to those expected from a map of known beacon locations [59.76].

One drawback to these early systems is the time-consuming task of installing, localizing, and maintaining the necessary infrastructure, particularly as the mine advances. To the best of our knowledge, none of these systems ever saw widespread use in industry. The state of the art in underground autonomous vehicle systems is infrastructureless.

One of the first infrastructureless technologies was one initially developed by an Inco Ltd. spin off. The patent [59.77] describes the use of a map consisting of a set of interlinking nodes intended to represent the topology of the underground passageway environment. The nodes themselves are located at points of interest, such as at intersections, dead ends, etc. Thus, a topological map was used for navigation, which relied on an algorithm capable of classifying all of the possible tunnel geometries encountered in the mine. Steering the vehicle was then done by a lower-level reactive algorithm, which tried to keep the machine near the tunnel center. A range-sensing device was used to sense the distance to walls.

This topological and reactive approach to underground navigation and control was tried again after [59.77]; see also [59.78–81]. Recently, these methods have also been combined with radio frequency identification (RFID) tags to provide a global reference [59.82]. This general approach is what formed the basis for one of the first truly commercial autonomous LHD systems, which is now offer as a product by Caterpillar Inc. (see the following). However, the main drawback to this technology is that it relies on classifying tunnel and intersection topology. As roboticists know, misclassifications can happen in practice.

Inspired by advances in mobile robotics research, some considered the application of alternative map-based localization algorithms – the idea being to match sensor measurements to a map rather than by classifying tunnel topology [59.83, 84]. In most cases, a planar poly-line representation of the mine was used to help the vehicle navigate. This is the approach taken by Sandvik AB in their commercial offering. More recently, Atlas Copco Rock Drills AB entered the underground automation game by applying an approach similar in philosophy, which generates sequences of occupancy grid maps and subsequently uses these to estimate the vehicle's position in real time by way of an

Fig. 59.14 Teleoperator's station for control of underground robotic vehicles at the Kemi mine, Finland (courtesy of Atlas Copco, 2009)

unscented Kalman filter (UKF)-based algorithm [59.85, 86].

Vehicle Control

A machine control system capable of performing at or better than a human operator is necessary in order for robotic tramming systems to compete in industry with manual operation. Many researchers have proposed articulated vehicle steering controllers based solely on vehicle kinematics [59.87–90], while few have considered dynamics [59.91]. Others have contended that wheel slip is significant and should be explicitly accounted for [59.92]. The paper [59.86] describes a control architecture and implementation that permits high-speed tramming by handling hydraulic vehicle steering and driveline dynamics by explicitly forcing a bandwidth separation between low-level machine/actuator dynamics and the path-tracking control problem, which is solved at the kinematic level.

Commercial Products

Although product descriptions are not the primary focus of this chapter, it is worth briefly mentioning that there are (at the time of writing) three main commercially available underground vehicle automation products: Caterpillar MineStar system for underground [59.93], Sandvik AutoMine [59.94], and the Atlas Copco ST14 ARV [59.85] (VIDEO 142).

Mobile equipment vendors, not third-party suppliers, sell all three. All three systems are infrastructure free. They all operate by scanning the vehicle's local environment with a 2-D SICK laser rangefinder and combining this data with wheel encoder information. All three systems operate in conjunction with a high-speed wireless network. What is more also all perform autonomous dumping. A schematic of the Atlas Copco product is shown in Fig. 59.15, highlighting the sensor and control components used for robotic tramming.

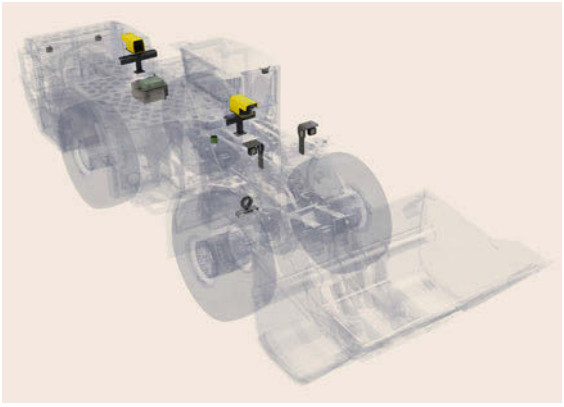


Fig. 59.15 Scooptram (or LHD) components for robotic tramming, including SICK scanning laser rangefinders (front and rear), articulation angle encoder, drive shaft encoder, onboard cameras, and computing subsystems (courtesy of Atlas Copco, 2009)

59.3.3 Robotic Loading

Despite a significant amount of research effort in the field of robotic excavation, none has yet resulted in the widespread development or adoption of autonomous or semi-autonomous excavation technologies in the mining industry. Although this section focuses on loading from an underground mining perspective, robotic loading is also applicable to surface mining operations (Sect. 59.2.3).

Robotic excavation is especially challenging because of the dynamic and unpredictable nature of the bucket-rock interactions. Also, a dig controller must manage not only the motion of the excavation arms (e.g., boom and bucket links), but also the penetration rate as determined by the motion of the mobile platform. As a result, performance is highly influenced by the conditions of interaction between the machine and its environment. For example, the forces that act on a bucket as it is actuated to penetrate a rock pile may vary significantly depending upon the properties of the media (e.g., density and hardness), the rock pile geometry, and the distribution of particle sizes and shapes [59.95, p. 562].

It is important to distinguish between excavation in homogeneous materials, such as soil, sand, or regolith, and the challenge of excavating fragmented rock as in mining, which usually has a significant distribution of particle sizes. Fragmented rock may consist of both *fines* (i. e., very small parti-

Fig. 59.16 EJC 9T LHD machine conducting robotic excavation experiments in fragmented rock at the engineering facilities of Sandvik Mining and Construction Canada, Burlington, Ontario (courtesy of Joshua Marshall, 2000)

cles), *oversize* (i. e., particles too large for the excavation tool), and a range of sizes in between. Figure 59.16 shows an EJC 9T LHD machine conducting robotic digging experiments in fragmented rock.

The robotic loading problem can be split into two fundamental tasks [59.96]:

1. Dig planning
2. Dig execution and control.

Dig planning constitutes the problem of deciding where and what to dig, possibly considering the geometry of the rock pile, the distribution of particles, and the physical characteristics of the loader and bucket. Dig control refers to algorithms for modifying plans based on the nature of the media encountered, so as to efficiently fill the loader's bucket. This task of the problem is particularly challenging in fragmented rock.

Dig Planning

Machine vision has been proposed as an enabling technology for dig planning. The paper [59.97] describes a laboratory-scale excavation system that utilized camera data for control and navigation. In [59.98], a scale-model system was built to mimic the motions of an LHD and different loading strategies were developed that depend on sensed information about the rock pile. Similar work, described in [59.99], employed a vision system to obtain images of the rock pile. In their approach, these images were used to plan the excavation task based on an estimated contour of the rock pile. In more recent work, the authors of [59.100] conducted an experimental study of two novel approaches for selecting an attack pose from 3-D data, with the objective of eventually using this for the robotic loading of sand.

Dig Execution and Control

The dig execution and control tasks have been tackled by several robotics researchers during the past two decades. Much of this work has targeted surface mining applications (Sect 59.2.3), although some concepts are transferrable to underground. Targeting underground, in [59.101] it was suggested that the trajectory of an excavator's bucket through the rock pile should not have priority in a devised control scheme, since the objective is to effectively fill the bucket, not to follow a predetermined path. This was explored further in [59.46, 95], which proposed an algorithm for robotic digging that would respond to bucket–rock interaction forces sensed as changing pressures the excavators bucket cylinder by commanding a change in the cylinders retraction velocity, rather than its position. This strategy was based on full-scale experiments with an LHD. Together with Atlas Copco AB, researchers at Queen's University in Kingston, Canada have recently demonstrated this admittance-based robotic loading strategy by way of full-scale experiments on a modified Kubota loader (on surface) and an ST14 LHD (underground), as shown in VIDEO 718.

59.3.4 Longwall Automation

Longwall mining is one of the main extraction methods for underground coal mining. This method is highly efficient in cases where thick seam reserves are present, with high-quality coal at shallow depth and benign geological conditions [59.102]. In the longwall process, a *shearer* – a machine with large rotating cutting drums – is driven back and forth across the seam, with each pass taking a massive slice of coal.

Figure 59.17 shows a schematic representation of a longwall mining equipment installation. Two long, horizontal and permanent tunnels known as *gate roads* are cut into a coal seam to form the main boundaries for a large rectangular block of coal, known as a longwall panel. The shearer travels in this panel along the face on a structure also housing an armored (or articulated) face conveyor (AFC) that carries the coal to one end of the face. The coal is then removed via other conveying systems to the surface. The roof is supported over this operation by a number of adjacent powered roof supports (also known as *shields* or *chocks*) which are each connected to the AFC. After the shearer passes, the chocks release from the roof, advance under hydraulic power into the cavity created by the removal of the coal, reposition the AFC, and re-support the roof [59.2].

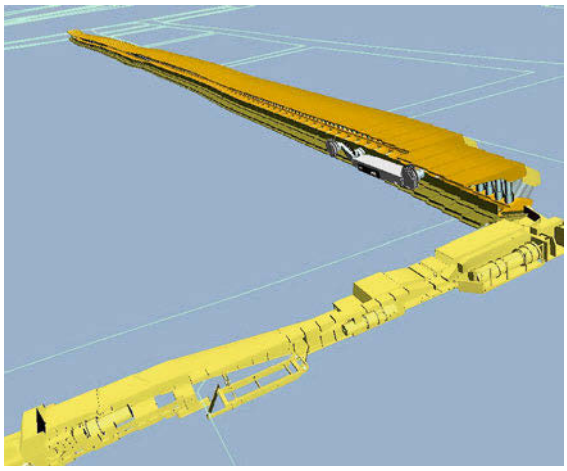


Fig. 59.17 Schematic of a longwall mining equipment installation (courtesy of CSIRO, 2008)

A concerted effort to deliver automation solutions for longwall mining saw the establishment of the Longwall Automation Steering Committee (LASC) in 2000, comprising Australian Coal Association Research Program (ACARP) industry representatives, equipment manufacturers, research providers and mine safety authorities. With funding from ACARP, CSIRO and the Cooperative Research Centre for Mining Technology and Equipment (CMTE), now CRC Mining, embarked in a major project to deliver automation solutions for coal cutting and loading, maintaining face geometry and manipulating roof supports without human intervention.

Three issues were identified as critical for longwall automation [59.103]: Face alignment, horizon control, and creep control. Until recently, efforts to develop automatic controllers for face alignment were hindered by the lack of an automatic and reliable method for measuring the actual geometric profile of the longwall face. The solution provided in this project employed a high-end inertial navigation system to provide accurate shearer position and attitude information in real time. The system is shearer-mounted and is referred to as the Shearer Position Measurement System (SPMS). One of the major outcomes of this project was the development of open specifications for the SPMS, and as a direct result, all the major longwall equipment manufacturers now offer automatic face alignment systems.

The second significant technological challenge tackled by LASC was achieving enhanced horizon control (EHC). While face alignment occurs in the horizontal plane, horizon control occurs in the vertical plane, and is significantly more difficult. In order to maximize the extraction of coal and minimize the extraction of waste, the roof and floor cutting horizons need to stay within the coal seam. The EHC incorporates two independent horizon controllers: one for the floor and one for the roof. The set points for the two controllers are generated by an integrated cut model, which sits at the core of the EHC. The cut model incorporates real-time information from navigation sensors, coal interface detectors, seam tracking sensors, as well as off-line geological and geotechnical information.

Finally, *creep control* refers to maintaining the lateral position of the longwall equipment in the panel. As the longwall mining system progresses, the assembly of supports should not creep toward either of the two gate roads. The LASC solution measures creep using laser range finders that scan in a horizontal plane in the panel and roadway directions. The creep information is provided to the longwall shearer automation processing system and generates corrections in the face profile in the form of a lead to either of the gate ends [59.103].

59.3.5 Robotic Explosives Loading

Many tasks in mining and construction require the driving of horizontal and inclined tunnels (or *drifts*) by drilling and blasting for rock fragmentation. Currently, drilling and blasting also remains the primary method for recovering ore from blocks (or *stopes*) in underground hard-rock mining. Blasting engineers design a blast hole pattern that specifies the size and location of the holes in order to achieve the desired rock fragmentation and minimize the damage to the rock mass left behind.

Blast hole charging is a two-step operation. In the first step, the operator places a primer assembly at the end of a flexible hose, inserts the hose into the blast hole, and pushes the hose until the primer reaches the end of the hole. A *primer* is a compact device containing high strength, sensitive explosives used to safely initiate the main column in the blast hole at a controlled time provided by the detonator. Initiation requires electric energy, usually delivered by detonating cords. The hole pushing is mechanized. In step two, an explosive emulsion is pumped into the hole through a hose, the hose is retracted as the explosive gradually fills the hole, leaving the trailing detonator wire. Once a ring of blast holes is filled, the detonator wires are joined and the rock is blasted.

A robotic explosive charging system (RECS) developed by the CSIRO [59.104] incorporates technologies that make the operation of loading holes safer with comparable or better productivity than otherwise possible. The system incorporates mixed teleoperation and robotic capabilities that allow the operator to load the entire pattern from the comfort and safety of the cab of the loading truck. The prototype RECS was developed for a mechanical manipulator derived from a Palfinger truck crane. The system is able to locate blast holes using the laser rangefinder attached to the arm end effector. An automatic arm pose control function is used to achieve a sweeping motion of the arm, during which the laser rangefinder collects images of the tunnel. These images are subsequently processed to extract blast hole locations, and presented to the operator via a human–machine interface.

Once the holes are identified in the scan, they are matched with the holes present in the blast plan and the operator is ready to proceed with the charging. The robot arm is sequenced to pick up the primer assembly from a magazine, and transfers it to the hole. The primer ends up at a pre-set offset from the

hole collar. In teleoperation mode the operator uses in-cab joysticks to insert the primer and the hose in the hole. The video system provides the operator with real-time video images of the relative positioning between the tip of the end effector and the blast hole collar.

In the work of [59.104], hole identification was carried out for 89 and 102 mm blast holes. While the hole diameter is used as a parameter in the identification algorithm, determining the hole length was not a requirement (this is not to say that in-situ validation of drilled versus planned is not important). During scanning, the robotic arm moves the end effector back and forth through the tunnel. The end effector is sequentially rotated, such that the laser is capable of scanning a full 360° pattern in the tunnel. Successful hole detection is influenced by the relative alignment between the laser beam and the hole axis. The larger the misalignment, the greater the chance of generating a false negative. Figure 59.18 shows a typical 3-D profile used for hole identification.

Visual servoing requires locating the tool in relation to the target hole. Cameras mounted on the end effector provide estimates of the location of blast holes with respect to the end effector. The control methodology does not require extensive knowledge of the manipulator physical parameters, and uses a minimal set of tuning parameters, to allow operators to make adjustments if and when required. However, finding an adequate solution raises a number of significant challenges when using a typical truck crane as a robotic arm.

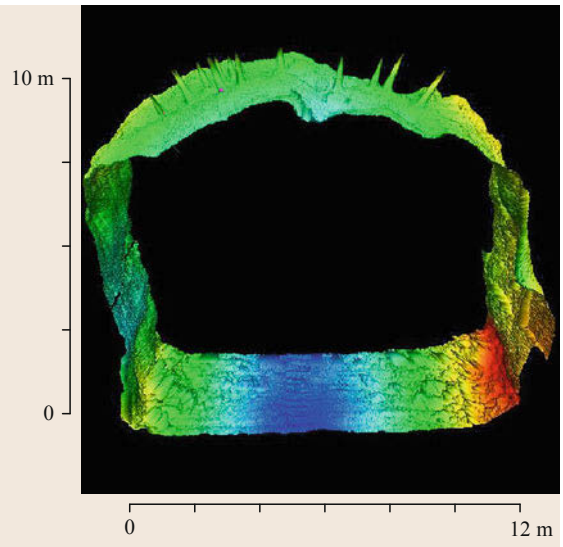


Fig. 59.18 A 3-D tunnel profile generated for identifying blast holes in a tunnel (courtesy of CSIRO, 2006)

59.3.6 Underground Mapping, Surveying, and Positioning

Mapping, surveying, and real-time equipment positioning are core services in many industries, and mining is no exception. In surface mining, the advent of the satellite-based GPS in the 1980s led to fundamental changes in the way everyday operations are carried out [59.105], with new applications being developed even more recently (Sect. 59.2). Based on GNSS, accurate site surveys, information systems, and robotic tools that enhance safety, improve productivity, and cost-saving maintenance operations, for example, have become commonplace in surface mining [59.1, 106].

However, similar progress has been made more slowly in underground mining because no directly comparable positioning technology currently exists for the accurate and real-time localization of mobile equipment in underground mining operations. This section discusses some tools and techniques from the field of robotics that have found their way into mining for such purposes as surveying, mapping, and equipment localization.

Underground Mapping and Surveying

At present, manually generated survey maps in mining are used for the design and construction of structures, ventilation, power, drainage, haulage planning, and tracking of development and ore extraction progress [59.107]. However, traditional surveying techniques are slow and laborious, involving many manual steps, infrastructure installation (e.g., retroreflective markers), and repeated measurements to obtain the necessary accuracy for the given application. Conventional survey maps are invariably a coarse approximation of the actual mine structure because, while individual observations may be accurate, few measurements are taken (i. e., poor resolution). Also, surveying is a regulated practice; thus the use of new techniques from robotics requires that current standards be updated.

Several recent advances out of the robotic mapping community show promise for applications in underground mining, and we mention just a select few here as examples. Authors from Carnegie Mellon University (CMU) [59.108, 109] describe robotic systems developed for acquiring volumetric maps of underground mine workings; specifically abandoned mines where ground conditions may be hazardous. SLAM results, based on data from 2-D scanning laser rangefinders, were reported from data obtained in two mines: a research mine in Bruceton, PA, and an abandoned coal mine in Burgettstown, PA.

Fig. 59.19 3-D SLAM-generated map of a decline at the Northparkes mine (after [59.112]); the gaps on the floor and ceiling are due to the blindspot between two vertically oriented SICK laser rangefinders; electrical cables can be seen along the left side wall (courtesy of So Jung Yun, 2012)

Some have developed techniques for robotic exploration and topological mapping [59.110] by focusing on the detection and matching of underground mine tunnel intersections as a basis for the creation of topological maps.

New scan registration techniques, such as 3-D-NDT (three-dimensional normal distributions transform) [59.111], have also been conceived specifically for underground mine mapping. Authors from the AASS in Sweden compared their results with the popular iterative closest point (ICP) algorithm on actual mine data.

Recently, researchers at CSIRO [59.112] demonstrated a 3-D SLAM solution, consisting of a spinning 2-D scanning laser rangefinder and industrial-grade micro-electromechanical system (MEMS) inertial measurement unit (IMU). The system was mounted on a site vehicle at the Northparkes mine (Parkes, NSW) that continuously acquired data at typical mine driving speeds. The deployed system mapped over 17 km of mine tunnels in under two hours, resulting in a dense and accurate geo-referenced 3-D surface model. Figure 59.19 shows an example of the generated surface model.

Underground Positioning and Tracking

The satellite-based GPS uses radio frequency signals that cannot usefully penetrate significant obstacles (e.g., rock); see any reference on the fundamentals of GPS [59.113, 114]. Although high-sensitivity receivers are being developed [59.114], these will almost certainly not function underground. For buildings, some have proposed the use of signal repeaters [59.115], but this is again not practical for most underground mine environments. These challenges necessitate a different approach for underground mines.

As a result, several alternatives for underground mining have appeared in recent years, many of which have limited functionality or are, as of yet, unproven. A recent summary, found in [59.116], presents a survey of the most commonly cited approaches:

- Employing RFID tags for event-based tracking of equipment and personnel [59.117].
- Use of radio signals (e.g., installed Wi-Fi access points) for signal strength-based positioning [59.118, 119], which is very difficult in underground mines due to severe multipath and shadowing, as well as non-line-of-sigh propagation and the need for too many devices in order to obtain GPS-like accuracies.
- Dead reckoning using inertial sensors (e.g., IMUs) and odometry for only short-term machine tracking.
- Rock-penetrating very low frequency electromagnetic signals, generally limited to near surface (< 200 m) [59.120] with questionable accuracy [59.121].

However, none of these options yet offer a solution comparable to that provided by GNSS on surface. Another issue is that mining practitioners often do not appreciate the technical limitations of these approaches [59.116, 122]. As a result, some have recently proposed the adaptation of map-based localization techniques for underground mining equipment. For example, [59.123] describes a method for large-scale mapping that employs a combination of RFID tags (as landmarks) and occupancy grid mapping specifically designed to facilitate real-time underground vehicle localization.

59.4 Challenges and Industry Acceptance

Despite the latest uptake of robotics technologies in mining, significant challenges remain. Next, we briefly review the most significant ones that are likely to affect the diffusion rate of autonomous systems in mining.

59.4.1 Technological Challenges

One expression that describes the current situation of robotics in mining is *islands of automation*. This re-

flects the significant integration issues that exist in relation to fully autonomous and mixed autonomous-manual/remote equipment. The root cause of the problem is a lack of standards to which the various equipment manufacturers should align. Although steps are now being taken to develop these standards (Sect. 59.5.3), the situation is likely to persist for a number of years. In addition to the time it will take to develop the required standards, there will be a significant lag at the equipment manufacturers' end, associated with the rolling out of new products that align with these new standards.

Other challenges to the implementation of robotics technology in mining include:

- Reliability, availability, and fail-safe operation with graceful failure
- Ability to sustain productivity at current or better levels
- Mining in extreme environments and difficulties in developing the required supporting infrastructure
- Current design methodologies for mines and mining systems are not geared to support robotic systems
- Deploying systems in the field and the fact that every mine is different
- Provision of technical support, shortfall in skilled labor, lack of adequate training
- Poorly designed or challenging man–machine interfaces.

59.4.2 Socioeconomic Challenges

The resource industry has a conservative history, and the implementation of new technology and processes must overcome significant inertia. In general, the attitude of people, from those at the working face to senior managers and company executives, mirrors the attitude of the workforce in the manufacturing industry in the early days of industrial robotics; i. e., skepticism about technology and fear of losing one's job. The generational change is likely to alleviate the fear of technology, but robots will be regarded by many in the workforce as a *competitor* for years to come.

Some researchers in human factors also point to the over-reliance on the technology by operators as a possible outcome of spreading robotic systems in the industry. As a common trend across several domains, it was noted that automation and new technologies can sometimes result in operators engaging in more risky behaviours in automated systems [59.124].

Legal issues and insurance are two other factors that could create significant headwinds to the rapid proliferation of autonomous systems in mining or in other civilian activity for that matter. Actuarial analysis is very hard to do for a green field, such as commercial autonomous operations.

59.5 Challenges, Outlook, and Conclusion

This section briefly summarizes some of the primary challenges for robotics in mining, as well as examines some emerging themes within the field of mining robotics, beginning with those on the Earth and followed by a short discussion about planetary exploration and extra-terrestrial mining, all of which will rely heavily on robotic tools. The section ends with a short conclusion about the future outlook for mining robotics.

59.5.1 Technical Challenges

Despite the recent uptake of robotics technologies in mining, significant challenges remain. One expression that could be used to describe the current status of robotics in mining is *islands of automation*. This reflects the significant integration issues that exist in relation to fully autonomous, as well as mixed autonomous/manual/remote robotic equipment. One specific challenge, for example, is the lack of standards to which the various equipment manufacturers and technology developers should align. Although, steps are now being taken to develop these standards (Sect. 59.5.3), it may take some time before equipment manufacturers roll out commercial products that adhere to these standards.

Some other key challenges for miming robotics include:

- Reliability, availability, and fail-safe operation with graceful and safe failure modes for very large equipment and (often) electric or hydraulically actuated equipment.
- Ability to sustain productivity at current or better levels when compared with current methods and systems.
- Mining in extreme and unstructured environments (e.g., far north, very deep underground, in harsh weather) and the difficulties in developing and supporting the necessary infrastructure in these environments.
- Current mining methods and design methodologies need to evolve so as to inherently incorporate robotic systems.

- Field robotics deployment challenges and the fact that every mine is different.
- Provision of technical support in remote areas, shortages of skilled labor, and the need to re-train existing personnel.
- Developing effective man–machine interfaces.
- Safely managing the colocation of robotic machines and human workers.

59.5.2 Socio-Economic Challenges

For the most part, the resource industry has a conservative history and the implementation of new technologies and processes must often overcome significant resistance to change. In many cases, the attitudes of people – from those at the working face to senior managers and company executives – mirror the attitudes of the workforce in the manufacturing industry in the early days of industrial robotics; i. e., skepticism about technology and fear of losing one's job. However, generational changes are likely to alleviate these challenges.

Some researchers in human factors engineering also point to the risk of over-reliance on technology by operators as a possible negative outcome of the proliferation of robotic systems. As a common trend across several domains, it was noted that automation and new technologies can sometimes result in operators engaging in more risky behaviours [59.124], which would not be looked favorably upon in mining.

59.5.3 Emerging Frontiers

There are several emerging frontiers for mining robotics, from the creation of standards to robots that improve safety in mining, and future-looking robots for mining asteroids and other extra-terrestrial bodies.

Emerging Standards for Mining Robotics and Automation

Due to the increasing prevalence of automation in mining, there is an immediate need to formulate standards for robotic equipment such that data may be exchanged freely among systems developed by different manufacturers. Currently, a tension exists between manufacturers, for whom it may seem the commercial imperative to design *closed* vertically integrated systems, and the mine operators, who would like any system they purchase to interoperate with all existing items of equipment.

There currently exists a data interchange standard known as the International Rock Excavation Data Exchange Standard (IREDES) [59.125]. This standard uses the extensible markup language (XML) to define a three-level architecture:

1. Administration level that defines reusable data objects such as coordinate systems
2. Application profile level that covers general information for one application purpose
3. Equipment profile level that has detailed equipment specific information.

The IREDES standard currently covers drill rigs and LHD machines, with many prospective application profiles under active development.

There is also an ISO (International Organization for Standardization) standard called *Earth-moving machinery – Autonomous machine safety* (ISO 17757) [59.126] currently under active development. However, at the time of writing, there is no public information regarding this standard apart from its December 2012 approval. The standard will likely cover the risks inherent in large equipment automation, and common ways that those risks can be mitigated through design and/or procedure.

It is still early days for the standardization of autonomous mining machines, and current efforts are far from complete. However, it is a worthy goal, and one which will only increase in importance as more mining equipment is automated.

Mine Rescue Robots

Mining is inherently risky. Although infrequent today, many fatalities have occurred in mining and the necessity of a mine rescue strategy at every operation – particularly in underground mining – is clear. So why not use robots to assist with mine rescue?

Although not mainstream, mine rescue robots have been under development at a few research institutions, including at Sandia National Laboratories where researchers have focused on coal mines under funding from the United States National Institute for Occupational Safety and Health (NIOSH) [59.127]. In coal mining, the risk of explosions is high and poisonous gases may be present, particularly after a disaster. Their tracked Gemini Scout mine rescue robot is approximately 1.2 m long, equipped with an infrared camera (to see through smoke and dust) and gas sensors that allow rescuers to assess the situation before attempting a rescue.

More about mine rescue robots can be found in [59.128], including reference to the first known mine rescue robot prototype at the CSIRO, called Numbat [59.129].

Robotic Undersea Mining

There is a vast body of robotics research related to undersea applications and remotely operated underwater

vehicles (ROVs) (Chap. 51 of this book is dedicated to this topic). However, much of this work does not yet directly target undersea mining. Nevertheless, advances in robotics – specifically computer-based mapping, multivehicle coordination, as well as telerobotic drilling and excavation – are likely to make ocean floor mining a reality in the very near future.

Although the pursuit of marine mining dates back to the 1870s, large-scale operations have not yet been realized at least in part due to technological barriers [59.130]. Today, several venture mining companies, including most notably Vancouver-based Nautilus Minerals Inc. (TSE:NUS), are targeting polymetallic seafloor massive sulphide deposits. However, these resources are only accessible by using ROVs equipped with telerobotic seafloor production tools to disaggregate and collect the rock from the seafloor [59.131]. Thus, robotics is poised to play a significant role in undersea mining.

Planetary Exploration and Mining

Perhaps the next great frontier for mining is that of planetary exploration and resource extraction, whether for in-situ resource utilization (e.g., toward development of the Moon) or for return to the Earth (e.g., mining of asteroids for valuable materials).

Several enabling technologies that support autonomous off-world robotic excavation for large-scale habitat construction and resource extraction have already been developed. For example, researchers from the National Aeronautics and Space Administration (NASA) and CSIRO demonstrated remote excavation of a trench as an example of site preparation and regolith mining [59.132]. This remote excavation system consisted of a 1/7-th scale dragline excavator located at a CSIRO test facility in Australia, a control interface located at a NASA facility in the Unites States, and a communication network between the two. The tested use case scenario involved remote initiation of terrain mapping, and a click-to-dig user interface that allowed the specification of material transfer in terms of dig and dump points on the acquired terrain map. The terrain map was used to generate a collision-free terrain-skimming path from dig to dump. The onboard path planner generated a bucket trajectory that included engaging the dig face (bank), filling the bucket and disengaging the bank. Tool force sensing was used as an input to the planner in order to constrain the dig speed and dig trajectory to remain within the operational limits of the machine and ensure adequate material flow into the bucket. The system successfully completed 50 consecutive cycles without any operator intervention. The average cycle time was approximately 63 s with the entire mission taking 52 min to complete.

There is also a thriving space exploration community, particularly within the mobile robotics community, and there is a complete chapter of this book dedicated to space robotics, Chap. 55. However, it is worth mentioning here that mining and planetary resource extraction have much in common and there is increasing cooperation between planetary geologists, mining equipment suppliers, and roboticists in problems related to Lunar and Mars exploration (e.g., see [59.4, 5, 133] and references therein for examples). Moreover, new commercial ventures, such as the widely publicized Planetary Resources, Inc., have been established with the aim of pursuing commercial extra-terrestrial resource discovery and utilization.

59.5.4 Conclusion

The desire to improve productivity, safety, and lower the costs of mining is a key motivator for the use of robotics in the mining industry. As highlighted in this chapter, several robotics-related advances have been made in recent history, for both surface and underground mining environments and, after a long courting period with robotics and automation, many mining companies are embracing robotics as a credible and practical tool. Moreover, with the emergence of *field robotics* as an increasingly popular subfield within the robotics research community, as well as the recent formation of networks, centers, and institutes that focus specifically on field robotics (some on mining), the domain of mining robotics appears poised to undergo accelerated growth on the Earth, undersea, and beyond.

Video-References

VIDEO 142 Autonomous tramming
available from http://handbookofrobotics.org/view-chapter/59/videodetails/142

VIDEO 145 Autonomous haulage system
available from http://handbookofrobotics.org/view-chapter/59/videodetails/145

VIDEO 718 Autonomous loading of fragmented rock
available from http://handbookofrobotics.org/view-chapter/59/videodetails/718

References

59.1 M. Scoble, L.K. Daneshmend: Mine of the year 2020: Technology and human resources, CIM Bull. **91**(1023), 51–60 (1998)

59.2 P. Corke, J. Roberts, J. Cunningham, D. Hainsworth: Mining robotics. In: *Springer Handbook of Robotics*, ed. by B. Siciliano, O. Khatib (Springer, Berlin, Heidelberg 2008) pp. 1127–1150, Chap. 49

59.3 H.L. Hartman, J.M. Mutmansky: *Introductory Mining Engineering*, 2nd edn. (Wiley, Hoboken 2002)

59.4 G.R. Osinski, T.D. Barfoot, N. Ghafoor, M. Izawa, N. Banerjee, P. Jasiobedzki, J. Tripp, R. Richards, S. Auclair, H. Sapers, L. Thomson, R. Flemming: Lidar and the mobile Scene Modeler (mSM) as scientific tools for planetary exploration, Planet. Space Sci. **58**(4), 691–700 (2010)

59.5 Y. Bar-Cohen, K. Zacny: *Drilling in Extreme Environments: Penetration and Sampling on Earth and other Planets* (Wiley-VCH, Weinheim 2009)

59.6 Clearpath Robotics: AMEC. Puts safety first and uses advanced robotic system for mapping potash tailings, http://clearpath.wpengine.netdna-cdn.com/wp-content/uploads/2013/02/AMEC_SuccessStory_2013e.pdf (2013)

59.7 P. Lever: Automation and robotics. In: *SME Mining Engineering Handbook*, ed. by P. Darling (SME, Enlgewood 2011) pp. 805–824, Chap. 9.8

59.8 D. Zlotnikov: Mining in the extreme, CIM Mag. **7**(5), 50–56 (2012)

59.9 P. Cross: *Recent Trends in Output and Employment*, Res. Pap. 13-604-MIE No. 054 (Statistics Canada, Ottawa 2007)

59.10 Australian Bureau of Statistics: 5260.0.55.002 – Experimental Estimates of Industry Multifactor Productivity, 2010-11, http://www.abs.gov.au/AUSSTATS/abs@.nsf/DetailsPage/5260.0.55.0022010-11 (2011)

59.11 N. Vagenas, N. Runciman, S.R. Clément: A methodology for maintenance analysis of mining equipment, Int. J. Min. Reclam. Environ. **11**, 33–40 (1997)

59.12 A. Gustafson, H. Schunnesson, D. Galar, U. Kumar: Production and maintenance performance analysis: Manual Production maintenance performance analytis: Manual versus semi-automatic LHDs, J. Qual. Maint. Eng. **19**(1), 74–88 (2013)

59.13 J. McGagh: The mine of the future: Rio Tinto's innovation pathway, http://www.riotinto.com/media/18435_presentations_22363.asp (2012), Presentation given at MINExpo 2012, Las Vegas

59.14 J.A. Aldinger, C.M. Keran: A Review of Accidents During Surface Mine Mobile Equipment Operations, Proc. 25th Annu. Inst. Min. Health Saf. Res. (1994) pp. 99–108

59.15 H. Durrant-Whyte, D. Pagac, B. Rogers, M. Stevens, G. Nelmes: Field and service applications-an autonomous straddle carrier for movement of shipping containers-from research to operational autonomous systems, IEEE Robotics Autom. Mag. **14**(3), 14–23 (2007)

59.16 J. Chadwick: Autonomous mine truck, Min. Mag. **175**(5), 287–288 (1996)

59.17 Pav Jordan: Chile's new Gaby copper mine steps into the future (Reuters), http://uk.reuters.com/article/2008/05/21/chile-codelco-gaby-idUKN2133325020080521 (2008)

59.18 Komatsu: Autonomous haulage system – Komatsu's pioneering technology deployed at Rio Tinto mine in Australia, http://www.komatsu.com/ce/currenttopics/v09212/index.html (2008)

59.19 Caterpillar: Autonomous haulage improves mine site safety, http://www.catminestarsystem.com/articles/autonomous-haulage-improves-mine-site-safety (2013)

59.20 Hitachi Construction Machinery: Hitachi chooses South Burnett for three-year automated mine-truck trial, http://www.stanwell.com/Files/Hitachi_automated_truck_trial.PDF (2013)

59.21 J. Barnes, C. Rizos, J. Wang, D. Small, G. Voigt, N. Gambale: Locata: A new positioning technology for high precision indoor and outdoor positioning, Proc. 2003 Int. Symp. GPS/GNSS (2003) pp. 9–18

59.22 G.S. Bastos, L.E. Souza, F.T. Ramos, C.H.C. Ribeiro: A single-dependent agent approach for stochastic time-dependent truck dispatching in open-pit mining, IEEE 14th Int. Conf. Intell. Transp. Syst. (ITSC) (2011) pp. 1057–1062

59.23 Modular Mining: DISPATCH, http://modularmining.com/product/dispatch/ (2013)

59.24 Komatsu: Modular mining systems unveils the latest in mining technology, http://www.komatsu.com/ce/support/v08412/index.html (2008)

59.25 Caterpillar: Track, manage and assign all types of equipment, across one site or many, https://mining.cat.com/fleet (2013)

59.26 Wenco: Wenco fleet management systems, http://www.wencomine.com/products/single-gallery/9342146 (2013)

59.27 Leica Geosystems: JOptimiser, http://mining.leica-geosystems.com/products/Jsoftware/Joptimizer/ (2013)

59.28 Devex: SMARTMINE, http://www.smartmine.com.br/eng/smartmine (2012)

59.29 S. Alarie, M. Gamache: Overview of solution strategies used in truck dispatching systems for open pit mines, Int. J. Surf. Min. Reclam. Environ. **16**(1), 59–76 (2002)

59.30 A. Arelovich, F. Masson, O. Agamennoni, S. Worrall, E. Nebot: Heuristic rule for truck dispatching in open-pit mines with local information-based decisions, Proc. 13th IEEE Int. Conf. Intell. Transp. Syst. (ITSC) (2010) pp. 1408–1414

59.31 S.G. Ercelebi, A. Bascetin: Optimization of shovel-truck system for surface mining, J. S. Afr. Inst. Min. Metall. Optim. **109**, 433–439 (2009)

59.32 R.F. Subtil, D.M. Silva, J.C. Alves: A Practical Approach to Truck Dispatch for Open Pit Mines, Proc.

2011 APCOM Symp. (2011) pp. 765–777
59.33 C.H. Ta, J.V. Kresta, J.F. Forbes, H.J. Marquez: A stochastic optimization approach to mine truck allocation, Int. J. Surf. Min. Reclam. Environ. **19**(3), 162–175 (2005)
59.34 S.-K. Kim, J.S. Russell: Framework for an intelligent earthwork system. Part I, System architecture, Autom. Constr. **12**(1), 1–13 (2003)
59.35 S. Singh: The state of the art in automation of earthmoving, ASCE J. Aerosp. Eng. **10**(4), 179–188 (1997)
59.36 S. Singh: State of the art in automation of earthmoving, Proc. Workshop Adv. Geomechatronics (2002)
59.37 M. Dunbabin, P. Corke: Autonomous excavation using a rope shovel, J. Field Robotics **23**, 379–394 (2006)
59.38 P.J.A. Lever, F.-Y. Wang: Intelligent excavator control system for lunar mining system, J. Aerosp. Eng. **8**(1), 16–24 (1995)
59.39 X. Shi, P.J.A. Lever, F.-Y. Wang: Experimental robotic excavation with fuzzy logic and neural networks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1996) pp. 957–962
59.40 W. Richardson-Little, C.J. Damaren: Position accommodation and compliance control for robotic excavation, Proc. IEEE Conf. Control Appl. (2005)
59.41 L.E. Bernold: Motion and Path Control for Robotic Excavation, J. Aerosp. Eng. **6**(1), 1–18 (1993)
59.42 Q. Ha, M. Santos, Q. Nguyen, D. Rye, H. Durrant-Whyte: Robotic excavation in construction automation, IEEE Robotics Autom. Mag. **9**(1), 20–28 (2007)
59.43 S. Tafazoli, S.E. Salcudean, K. Hashtudi-Zaad, P.D. Lawrence: Impedance control of a teleoperated excavator, IEEE Trans. Control Syst. Technol. **10**(3), 355–367 (2002)
59.44 C.P. Tan, Y.H. Zweiri, K. Althoefer, L.D. Seneviratne: Online soil parameter estimation scheme based on Newton–Raphson method for autonomous excavation, IEEE/ASME Trans. Mechatron. **10**(2), 221–229 (2000)
59.45 S. Singh: Learning to predict resistive forces during robotic excavation, Proc. 1995 IEEE Int. Conf. Robotics Autom. (1995) pp. 2102–2107
59.46 J.A. Marshall: Towards Autonomous Excavation of Fragmented Rock: Modelling, Identification and Control, Ph.D. Thesis (Queen's Univ., Kingston 2001)
59.47 A.R. Reece: The fundamental equation of earthmoving mechanics, Proc. Inst. Mech. Eng. (1964)
59.48 H. Cannon, S. Sanjiv: Models for automated earth moving, Lect. Note. Control Inform. Sci. **250**, 163–172 (2000)
59.49 E. Duff: Accurate guidance and measurement for excavators using laser scanners, Techn. Rep. C14043 (ACARP, Brishane 2006)
59.50 Leica Geosystems Mining: Jigsaw products: J^3 dozer autorip, http://mining.leica-geosystems.com/products/J3autonomous/J3dozer-autorip/ (2013)
59.51 Mining Magazine: Thinking automatically, http://www.miningmagazine.com/equipment/thinking-automatically (2012)
59.52 Ry Crozier: Gears up for expansion across Pilbara mines, http://www.itnews.com.au/News/312004,rio-tinto-advances-autonomous-drill-project.aspx (2012)
59.53 E. Duff, C. Caris, A. Bonchis, K. Taylor, C. Gunn, M. Adcock: The development of a telerobotic rock breaker, Springer Tract. Adv. Robot. **62**, 411–420 (2010)
59.54 E. Duff, K. Usher, P. Ridley: Swing Loader Traffic Control, Techn. Rep. C13041 (ACARP, Brishane 2006)
59.55 B. Owens: Concept Design and Testing of a GPS-less System for Autonomous Shovel-Truck Spotting, Ph.D. Thesis (Queen's Univ., Kingston 2013)
59.56 A. Stentz, J. Bares, S. Singh, P. Rowe: A robotic excavator for autonomous truck loading, Auton. Robots **7**(2), 175–186 (1999)
59.57 M. Dunbabin, G. Winstanley, P. Corke: Refinement of Automated Dragline Swing Control Algorithms, Techn. Rep. C13040 (ACARP, Brishane 2005)
59.58 J. Roberts: Dragline operational enhancements through the use of digital terrain maps, ACARP Report C13034 (2006)
59.59 Leica Geosystems Mining: Well positioned, http://mining.leica-geosystems.com/products/Jassist/Jps/ (2013)
59.60 Locata Corporation: Technology brief, http://www.locatacorp.com/wp-content/uploads/2011/09/Locata-Technology-Brief-13-June-2012-Public.pdf (2013)
59.61 T. Sathyan, D. Humphrey, M. Hedley: WASP: A system and algorithms for accurate radio localization using low-cost hardware, IEEE Trans. Syst. Man Cybern. C **41**(2), 211–222 (2011)
59.62 United States Department of Labor: Mine safety and health administration report, http://www.cdc.gov/niosh/mining/pubs/pdfs/mriit.pdf (2010)
59.63 B. Clark, S. Worrall, G. Brooker, J. Martinez, E. Nebot: Improving situational awareness with radar information, Proc. 2012 IEEE Intell. Vehicle Symp. (2012) pp. 535–540
59.64 K. Nienhaus, R. Winkel, W. Mayer, A. Gronau, W. Menzel: An experimental study on using electronically scanning microwave radar systems on surface mining machines, Proc. IEEE Radar Conf. (2007) pp. 509–512
59.65 G. Agamennoni, J.I. Nieto, E.M. Nebot: Estimation of Multivehicle Dynamics by Considering Contextual Information, IEEE Trans. Robotics **28**(4), 855–870 (2012)
59.66 S. Worrall, G. Agamennoni, J.I. Nieto, E.M. Nebot: A context-based approach to vehicle behavior prediction, IEEE Intell. Transp. Syst. Mag. **4**(3), 32–44 (2012)
59.67 P.V. Golde: Implementation of drill teleoperation in mine automation, Ph.D. Thesis (McGill Univ.,

Montréal 1997)
59.68 J. Appelgren: Remote control and navigation systems, Min. Constr. Mag. **2**, 16–19 (2003)
59.69 D. Hunter, D. Wells, K. Chrystall, P. Feighan: Achieving effective telerobotic control of industrial equipment, CIM Bull. **89**(1002), 83–88 (1996)
59.70 J. Larsson, M. Broxvall, A. Saffiotti: An evaluation of local autonomy applied to teleoperated vehicles in underground mines, Proc. 2010 IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 1745–1752
59.71 K. Amdahl, M. Lundström: Automatic truck saves money underground, World Mining **160**, 40–44 (1972)
59.72 G.D. Brophey: Vehicle guidance system, CA 204 1373A1 (1991)
59.73 R. Hurteau, M. St-Amant, Y. Laperriere, G. Chevrette: Optical guidance system for underground mine vehicles, Proc. 1992 IEEE Conf. Robotics Autom. (1992) pp. 639–644
59.74 J.F. Purchase, R.A. Poole: Guidance system for automated vehicles, and guidance strip for use therewith, US 616 3745A (2000)
59.75 U. Wiklund, U. Andersson, K. Hyypä: AGV navigation by angle measurements, Proc. 6th Int. Conf. Autom. Guided Veh. Syst. (1988) pp. 199–212
59.76 S. Scheding, G. Dissanayake, E.M. Nebot, H.F. Durrant-Whyte: An experiment in autonomous navigation of an underground mining vehicle, IEEE Trans. Robotics Autom. **15**(1), 85–95 (1999)
59.77 L.A. Bloomquist, E.H. Hinton: Autonomous vehicle guidance system, US 599 9865A (1999)
59.78 P. Debanné, J.-Y. Hervé, P. Cohen: Global self-localization of a robot in underground mines, Proc. 1997 IEEE Int. Conf. Syst. Man Cybern. (1997) pp. 4400–4405
59.79 J.M. Roberts, E.S. Duff, P.I. Corke: Reactive navigation and opportunistic localization for autonomous underground mining vehicles, Inform. Sci. **145**, 127–146 (2002)
59.80 J.M. Roberts, E.S. Duff, P.I. Corke, P. Sikka, G.J. Winstanley, J.B. Cunningham: Autonomous control of underground mining vehicles using reactive navigation, Proc. 2000 IEEE Conf. Robotics Autom. (2000) pp. 3790–3795
59.81 J. Steele, C. Ganesh, A. Kleve: Control and scale model simulation of sensor-guided LHD mining machines, IEEE Trans. Ind. Appl. **29**(6), 1232–1238 (1993)
59.82 J. Larsson, M. Broxvall, A. Saffiotti: A navigation system for automated loaders in underground mines, Proc. 5th Int. Conf. Field Serv. Robotics (2005)
59.83 R. Madhavan, M.W.M.G. Dissanayake, H.F. Durrant-Whyte: Autonomous underground navigation of an LHD using a combined ICP-EKF approach, Proc. IEEE Conf. Robotics Autom. (1998) pp. 3703–3708
59.84 H. Mäkelä: Overview of LHD navigation without artificial beacons, Robotics Auton. Syst. **36**, 21–35 (2001)
59.85 J. Larsson, J. Appelgren, J.A. Marshall, T.D. Barfoot: Atlas Copco infrastructureless guidance system for high-speed autonomous underground tramming, Proc. 5th Int. Conf. Exhib. Mass Min. (2008) pp. 585–594
59.86 J.A. Marshall, T.D. Barfoot, J. Larsson: Autonomous underground tramming for center-articulated vehicles, J. Field Robotics **25**(6–7), 400–421 (2008)
59.87 C. Altafini: A path-tracking criterion for an LHD articulated vehicle, Int. J. Robotics Res. **18**(5), 435–441 (1999)
59.88 A. Hemami, V. Polotski: Problem formulation for path tracking automation of low speed articulated vehicles, Proc. IEEE Int. Conf. Control Appl. (1996) pp. 697–702
59.89 V. Polotski: New reference point for guiding an articulated vehicle, Proc. IEEE Int. Conf. Control Appl. (2000) pp. 455–460
59.90 P. Ridley, P. Corke: Autonomous control of an underground mining vehicle, Proc. 2001 Austr. Conf. Robotics Autom. (2001) pp. 26–31
59.91 R.M. DeSantis: Modeling and path-tracking for a load-haul-dump mining vehicle, J. Dyn. Syst. Meas. Control **119**, 40–47 (1997)
59.92 S. Scheding, G. Dissanayake, E. Nebot, H. Durrant-Whyte: Slip modelling and aided inertial navigation of an LHD, Proc. IEEE Int. Conf. Robotics Autom. (1997) pp. 1904–1909
59.93 G.B. Smith, R.J. Butcher, A. Uzbekova, E. Mort, A. Clement: Case study comparison of teleremote and autonomous assist underground loader technology at the Kanowna Belle Mine, Proc. 11th AusIMM Underground Operators' Conference (2001) pp. 305–312
59.94 B. Cook, D. Burger, L. Alberts, R. Grobler: Automated loading and hauling experiences at De Beers Finsch Mine, Proc. 10th AusIMM Underground Operators' Conference (2010) pp. 231–238
59.95 J.A. Marshall, P.F. Murphy, L.K. Daneshmend: Toward Autonomous Excavation of Fragmented Rock: Full-Scale Experiments, IEEE Trans. Autom. Sci. Eng. **5**(3), 562–566 (2008)
59.96 S. Singh: Synthesis of Tactical Plans for Robotic Excavation, Ph.D. Thesis (Robotics Institute Carnegie Mellon Univ., Pittsburgh 1995)
59.97 Q. Ji, R.L. Sanford: Autonomous excavation of fragmented rock using machine vision. In: *Emerging Computer Techniques for the Minerals Industry*, ed. by B.J. Schneider, D.A. Stanley, C.L. Karr (SME, Littleton 1993) pp. 221–228
59.98 M.K. Petty, J. Billingsley, T. Tran-Cong: Autonomous LHD Loading, Proc. Annu. IEEE Conf. Mechatron. Mach. Vis. Pract. (1997) pp. 219–224
59.99 H. Takahashi, M. Hasegawa, E. Nakano: Analysis on the resistive forces acting on the bucket of a Load-Haul-Dump machine and a wheel loader in the scooping task, Adv. Robotics **13**(2), 97–114 (1999)
59.100 M. Magnusson, H. Almqvist: Consistent pile-shape quantification for autonomous wheel loaders, Proc. 2011 IEEE/RSJ Int. Conf. Intell. Robots Syst. (2011) pp. 4078–4083

59.101 A. Hemami: Fundamental analysis of automatic excavation, J. Aerosp. Eng. **8**(4), 175–179 (1995)
59.102 G.W. Mitchell: Longwall mining. In: *Australian Coal Mining Practice*, ed. by R.J. Kininmouth, E.Y. Baafi (AIMM, Carlton 2005) pp. 340–375
59.103 P.B. Reid, M.T. Dunn, D.C. Reid, J.C. Ralston: Real-world automation: New capabilities for underground longwall mining, Proc. Austr. Conf. Robotics Autom. (2010)
59.104 A. Bonchis, E. Duff, J. Roberts, M. Bosse: Robotic explosive charging in mining and construction applications, IEEE Trans. Autom. Sci. Eng. (2013)
59.105 D.J. Peterson, T. LaTourette: *New Forces at Work in Mining: Industry Views of Critical Technologies* (RAND Sci. Techn. Policy Inst., Santa Monica 2001)
59.106 J. Peck, J. Gray: The total mining system (TMS): The basis for open pit automation, CIM Bull. **88**(993), 38–44 (1995)
59.107 G. Schaffer, A. Stentz: Automated Surveying of Mines Using a Laser Rangefinder, Emerg. Comp. Techn. Miner. Ind. Symp. (SME) (1993) pp. 363–370
59.108 A. Nuchter, H. Surmann, K. Lingemann, J. Hertzberg, S. Thrun: 6D SLAM with an application in autonomous mine mapping, Proc. 2004 IEEE Int. Conf. Robotics Autom. (2004) pp. 1998–2003
59.109 S. Thrun, D. Hahnel, D. Ferguson, D. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, W. Whittaker: A system for volumetric robotic mapping of abandoned mines, Proc. 2003 IEEE Int. Conf. Robotics Autom. (2003) pp. 4270–4275
59.110 D. Silver, D. Ferguson, A. Morris, S. Thayer: Topological exploration of subterranean environments, J. Field Robotics **23**(6/7), 395–415 (2006)
59.111 M. Magnusson, A. Lilienthal, T. Duckett: Scan registration for autonomous mining vehicles using 3D-NDT, J. Field Robotics **24**(10), 803–827 (2007)
59.112 R. Zlot, M. Bosse: Efficient large-scale 3D mobile mapping and surface reconstruction of an underground mine, Proc. Int. Conf. Field Serv. Robotics (2012)
59.113 Garmin Ltd.: What is GPS?, http://www.garmin.com/aboutGPS/ (1996)
59.114 F. van Diggelen: Indoor GPS theory and implementation, Proc. IEEE Position Loc. Navig. Symp. (2002) pp. 240–247
59.115 H. Niwa, K. Kodaka, Y. Sakamoto, M. Otake, S. Kawaguchi, K. Kujii, Y. Kanemori, S. Sugano: GPS-based indoor positioning system with multichannel pseudolite, Proc. IEEE Int. Conf. Robotics Autom. (2008) pp. 905–910
59.116 U. Artan, J.A. Marshall, N.J. Lavigne: Robotic mapping of underground mine passageways, Trans. IMM A: Min. Technol. **120**(1), 18–24 (2011)
59.117 E. Bartsch, M. Laine, M. Anderson: The application and implementation of optimized mine ventilation on demand (OMVOD) at the Xstrata Nickle Rim South Mine, Sudbury, Ontario, Proc. 13th U.S./N. Am. Mine Venti. Symp. (2010) pp. 1–15
59.118 J.C. Ralston, C.O. Hargrave, D.W. Hainsworth: Localisation of mobile underground mining equipment using wireless ethernet, Proc. Ind. Appl. Conf. (2005) pp. 225–230
59.119 M.M. Atia, A. Noureldin, J. Georgy, M. Korenberg: Bayesian filtering based WiFi/INS integrated navigation solution for GPS-denied environments, Navigation **58**(2), 111–125 (2011)
59.120 R. Wenger: La balise de positionnement U-GPS (Underground-GPS), ISSKA Rapport Annuel (Swiss Institute for Speleology and Karst Studies, La Chaux-de-Fonds 2004), pp. 13–14
59.121 J. Chadwick: GPS for underground operations: Great potential for controlling block caves, saving trapped miners and machine automation, http://www.mining.com (2008)
59.122 J.A. Marshall: Navigating the advances in underground navigation, CIM Mag. **5**(4), 20–21 (2010)
59.123 N.J. Lavigne, J.A. Marshall: A landmark-bounded method for large-scale underground mine mapping, J. Field Robotics **29**(6), 861–879 (2012)
59.124 D. Lynas, T. Horberry: Human factor issues with automated mining equipment, Ergonomics Open J. **4**, 74–80 (2011)
59.125 IREDES: IREDES – International rock excavation data exchange standard, http://www.iredes.org/ (2013)
59.126 International Standards Organisation: ISO/NP 17757 earth-moving machinery – Autonomous machine safety, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=60473 (2012)
59.127 D. Hambling: Next-Gen coal mining rescue robot, popular mechanics, http://www.popularmechanics.com/science/energy/coal-oil-gas/next-gen-coal-mining-rescue-robot (2010)
59.128 R.R. Murphy, J. Kravitz, S. Stover, R. Shoureshi: Mobile robots in mine rescue and recovery, IEEE Robotics Autom. Mag. **16**(2), 91–103 (2009)
59.129 D.W. Hainsworth: Teleoperation user interfaces for mining robotics, Auton. Robots **11**(1), 19–28 (2001)
59.130 A. MacDonald, E. Welsch: Robotics advance ocean floor mining ventures, http://search.proquest.com/docview/1018567281 (2012)
59.131 Nautilus Minerals: Fact sheet, http://www.nautilusminerals.com/i/pdf/Factsheet-Q1-2013.pdf (2013)
59.132 M. Dunbabin, P. Corke, G. Winstanley, J. Roberts: Off-world robotic excavation for large-scale habitat construction and resource extraction, to boldly go where no human-robot team has gone before, AAAI Spring Symp. (2006)
59.133 J.E. Moores, R. Francis, M. Mader, G.R. Osinski, T. Barfoot, N. Barry, G. Basic, M. Battler, M. Beauchamp, S. Blain, M. Bondy, R.-D. Capitan, A. Chanou, J. Clayton, E. Cloutis, M. Daly, C. Dickinson, H. Dong, R. Flemming, P. Furgale, J. Gammel, N. Gharfoor, M. Hussein, R. Grieve, H. Henrys, P. Jaziobedski, A. Lambert, K. Leung, C. Marion, E. McCullough, C. McManus, C.D. Neish, H.K. Ng, A. Ozaruk, A. Pickersgill, L.J. Preston, D. Redman,

H. Sapers, B. Shankar, A. Singleton, K. Souders, B. Stenning, P. Stooke, P. Sylvester, L. Tornabene: A mission control architecture for robotic lunar sample return as field tested in an analogue deployment to the sudbury impact structure, Adv. Space Res. **50**(12), 1666–1686 (2012)

Multimedia Contents

60. Disaster Robotics

Robin R. Murphy, Satoshi Tadokoro, Alexander Kleiner

Rescue robots have been used in at least 28 disasters in six countries since the first deployment to the 9/11 World Trade Center collapse. All types of robots have been used (land, sea, and aerial) and for all phases of a disaster (prevention, response, and recovery). This chapter will cover the basic characteristics of disasters and their impact on robotic design, and describe the robots actually used in disasters to date, with a special focus on Fukushima Daiichi, which is providing a rich proving ground for robotics. The chapter covers promising robot designs (e.g., snakes, legged locomotion) and concepts (e.g., robot teams or swarms, sensor networks), as well as progress and open issues in autonomy. The methods of evaluation in benchmarks for rescue robotics are discussed and the chapter concludes with a discussion of the fundamental problems and open issues facing rescue robotics, and their evolution from an interesting idea to widespread adoption.

60.1 **Overview** 1578
60.1.1 Motivation 1578
60.1.2 Disaster Robot Tasks 1579
60.1.3 Types of Disaster Robots 1580

60.2 **Disaster Characteristics and Impact on Robots** 1581
60.2.1 Natural Disasters 1581
60.2.2 Manmade Disasters 1581
60.2.3 General Pattern of Activity During the Response Phase 1582

60.3 **Robots Actually Used at Disasters** 1582
60.3.1 Meteorological Events 1582
60.3.2 Geological Events 1584
60.3.3 Manmade Events 1586
60.3.4 Mine Disasters 1587

60.4 **Robots at the Fukushima-Daiichi Nuclear Power Plant Accident** 1588
60.4.1 Exterior Operations 1588
60.4.2 Interior Operations in Nuclear Reactor Buildings 1589

60.5 **Lessons Learned, Challenges, and Novel Approaches** 1591
60.5.1 Mobility: UGVs 1591
60.5.2 Mobility: UAVs 1592
60.5.3 Mobility: UMVs 1592
60.5.4 Communications 1593
60.5.5 Control 1593
60.5.6 Sensors and Sensing 1594
60.5.7 Power 1595
60.5.8 Manipulation 1595
60.5.9 Human–Robot Interaction (HRI) 1596
60.5.10 Multirobot Team Coordination ... 1596
60.5.11 Other Issues 1597

60.6 **Evaluation** 1598
60.6.1 Computer Simulations for Rescue Robotics 1598
60.6.2 Physical Test Beds 1598
60.6.3 Standards Activity 1600
60.6.4 Evaluation 1600

60.7 **Conclusions and Further Reading** 1600

Video-References 1601

References 1601

Robots serve as extensions of emergency professionals. During the immediate response to a disaster, robots can provide real-time data about the event and perhaps even directly aid in inserting mitigation devices and sensors, turning valves, and moving equipment. As the immediate life-saving and stabilization phase of the disaster ends, robots can help with the economic recovery. Robots for disaster are an emerging technology,

and have not yet been widely adopted by the international emergency response community. However, as of 2012, they have been used in 28 disasters in Cyprus, Haiti, Italy, Japan, New Zealand, and the United States primarily for mining disasters and structural collapses or infrastructure damage. Robots have been used for hazardous materials accidents, often referred to as chemical, biological, nuclear, radiological, or explosive (CBRNE) events, most recently the Fukushima Daiichi nuclear accident. Both structural and CBRNE events are generally handled by urban search and rescue teams unless an industry-specific team assumes command. As a result of the breadth of the term *urban search and rescue* contributes to a misperception that disaster robotics is limited to urban or constructed environments. For example, although technically an urban event, wilderness search and rescue is often conducted by a local urban search and rescue team. In general, disaster robotics connotes extraordinary situations requiring special operations; fire fighting is not considered as part of disaster robotics as it is a *routine emergency*.

Disaster robotics can contribute to broad area activities, local incidents, and routine emergencies, though this chapter will focus on the use of robots for disaster events. Interest in robotics for surveying wildland fires and flooding is emerging, and the few documented uses of robots are reported in this chapter. Robots are also being increased use in local incidents which do not qualify as true disasters. For example, several fire rescue departments in Japan and the United Kingdom routinely use small underwater robots for water-based search and recovery of drowned victims and the interest in the use of aerial vehicles for wilderness search and rescue is growing.

Disaster robots have not been widely adopted, but their use is increasing as the technology matures. The general lack of adoption is to be expected since the technology is new and the concept of operations for these novel technologies take time to evolve. Rescue applications are often similar enough to military operations that the same platforms can be adapted; however, some tasks are significantly different than their military counterpart, some are unique to rescue, and the human–robot interaction for civilian response diverges from military patterns of use.

60.1 Overview

Disaster response is always a race against time, to move as fast as possible to reach all potential survivors and yet move slowly enough to avoid creating additional collapses, damage, risk to rescuers and victims, or contention over airspace. The primary motivation is to save lives; robots can assist in meeting this goal either by interacting directly with victims or structures or automating support activities.

60.1.1 Motivation

Disasters are defined as a discrete meteorological, geological, or manmade event, that exceeds local resources to respond and contain. Disasters are distinct from an ongoing condition such as global warming. In practice, a disaster means that an agency has to engage additional specially trained experts and equipment. For example, a chemical spill may be routine and handled with existing personnel trained in hazardous materials or it may be horrific, such as the 1984 Bhopal disaster that killed thousands of people.

Disaster activities are often referred to by phases, though there is no single accepted model in emergency management. After an event, public officials are responsible for the life-saving *response* and economic *recovery* activities focused on maximizing benefit to the largest population. Nongovernmental agencies may join with other officials and citizens in humanitarian *relief* efforts which are generally focused on benefiting individuals. Emergency professionals are also tasked with *prevention* of disasters and *preparation* should there be sufficient warning.

Disaster response activities are typically executed by the urban search and rescue (USAR) task force within a fire rescue agency who work with law enforcement, transportation officials, and others. USAR is often used synonymously with any response activity. The term *urban* can be misleading as it sounds like it is restricted to building collapses. USAR includes hazardous material response, as these materials most certainly have something to do with built structures. USAR includes *wilderness search and rescue* which often involves searching tunnels and abandoned mines, avalanches, not just locating lost hikers. It also includes *water-based search and rescue* which deals with saving victims of floods or high currents (also known as *swift water rescue*) or in the aftermath of traffic accidents where victims are trapped in cars that have plunged into a river or bay. Water-based search and rescue does not necessarily have urban structures but there is a similarity of rescue techniques and re-use of equipment.

Historically, disaster robotics concentrated on the response phase of a disaster. The disaster robotics community began to form in 1995 as the outcome of the tragic loss of life in the Hanshin–Awajii earthquake in Kobe, Japan, and the bombing of the Murrah Federal Building in Oklahoma City, United States [60.1, 2]. In both cases, it was easy to imagine how small ground robots may have been able to enter rubble to find trapped victims. As a result, research efforts began in individual laboratories. Two mobile robot competitions (the AAAI (Association for the Advancement of Artificial Intelligence) mobile robot competition in the United States, and the RoboCup rescue league internationally) were started in the late 1990s to engage the scientific community in rescue research. The first use of robots for response was at the 2001 World Trade Center (WTC) disaster, where ground robots were used to penetrate the deeply packed rubble. Since then robots have been used for response to earthquakes, hurricanes, mine collapses, bridge collapses, explosions, and flooding.

The deployment of disaster robots during the response phase offers many potential benefits. The 2010 World Disasters report [60.3] suggests just how many lives have been, and will be, impacted by urban disasters. Over 1.1 million people were reported killed from 2000 to 2009, with the total amount of disaster-related damage estimated at 986.7 billion US dollars. Of the victims in urban disasters, only a small fraction may actually survive. Consider from [60.4, 5] that the majority of survivors (80%) of urban disasters are surface victims, that is, the people lying on the surface of the rubble or readily visible. However, only 20% of survivors of urban disasters come from the interior of the rubble, yet the interior is often where the majority of victims are located, providing motivation for robots that can explore deep within collapses. The mortality rate increases and peaks after 48 h, meaning that survivors who are not extricated in the first 48 h after the event are unlikely to survive beyond a few weeks in the hospital.

However, disaster robotics is no longer focused on response but has expanded to recovery. Recovery activities consist of extricating any remaining bodies of victims, re-establishing normal operations in the community, continuity of government, and resumption of the economy. The activities are typically carried out by various local, provincial, state, and federal agencies as well as insurers and banks. Each municipality may have different priorities in rebuilding schools, hospitals, ports, and industry and there is usually no centralized authority.

The benefits of disaster robotics for the recovery phase were seen in the ongoing aftermath of 2011 Tohoku earthquake and related Fukushima Daiichi nuclear accident. The tsunami generated by the earthquake swept thousands of people into the sea. It also severely damaged many fishing ports and scattered debris and pollutants throughout fishing areas. Teams of roboticists used underwater robots to search for victims and to help re-open ports and clear fishing beds. The decommissioning of the Fukushima Daiichi facility will make use of robots, possibly some of those used in the response.

Disaster robotics is also addressing prevention and preparedness. As with recovery operations, prevention and preparedness activities are similarly distributed among many groups. It should be noted that the fire rescue and law enforcement teams will be the first responders and whatever equipment they use locally will be likely to be used at a disaster. Therefore, the disaster robotics community has begun pushing for the adoption of robots into daily use by fire rescue and bomb squads so that the responders will be familiar and comfortable with the technology [60.6]. A lesson from Fukushima was that robots built by the nuclear industry in the aftermath of Chernobyl and Three Mile Island had not been used in decades, with no one expert in their use.

60.1.2 Disaster Robot Tasks

While the overall motivation for rescue robotics is to save lives, the motivation for specific robot designs and capabilities depend on their potential tasks. The types of tasks that have been proposed for rescue robots are described below. Some of the above tasks are similar to tasks for military robots, especially search and reconnaissance and mapping, but many are unique or have a different flavor. For example, structural inspection, rubble removal, and adaptively shoring rubble are rescue specific. Tasks such as casualty extraction appear to be similar but are significantly different. Consider that a wounded soldier is unlikely to have a spinal cord injury and is likely to be in a space large enough for a human to work in, so a robot entering the area and dragging the soldier to safety is appropriate. However, a crushed victim of a building collapse is physically trapped or pinned in a small space, requiring rubble removal, and the victim's spinal cord must be immobilized before extraction; clearly victim extraction in the search-and-rescue domain is more challenging.

Search is a concentrated activity in the interior of a structure, in caves or tunnels, or wilderness and aims to find a victim or potential hazards. The motivation for the search task is speed and completeness without increasing risk to victims or rescuers.

Reconnaissance and mapping is broader than search. It provides responders with general situation awareness and creates a reference of the destroyed en-

vironment. The goal is speedy coverage of a large area of interest at the appropriate resolution.

Rubble removal can be expedited by robotic machinery or exoskeletons. The motivation is to move heavier rubble faster than could be done manually, but with a smaller footprint than that of a traditional construction crane.

Structural inspection may be either conducted on the interior (e.g., to help rescuers understand the nature of the rubble in order prevent secondary collapses that may further injure survivors) or on the exterior (e.g., to determine whether a structure is safe to enter). Robots provide a means of getting structural sensor payloads closer and in far more favorable viewing angles.

In situ medical assessment and intervention are needed to permit doctors and paramedics to interact verbally with victims, visually inspect the victim or apply diagnostic sensors, or to provide life support by transporting fluids and medication through narrow tubing. The objective is to provide telepresence for medical personnel during the 4–10 h that it usually takes to extricate a victim [60.4, 5]. The lack of medical intervention was a major problem at the Oklahoma City bombing [60.7]. Preliminary work has been conducted for victim management, most notably in telemedicine (also called *medical reachback*) [60.8], general assessment and triage [60.9], and comforting trapped victims as per the Survivor Buddy project [60.10].

Medically sensitive extrication and evacuation of casualties may be needed to help provide medical assistance while victims are still in the disaster area, also known as the hot zone. In the case of a chemical, biological, or radiological event, the number of victims is expected to exceed the number that can be carried out by human rescuers in their highly restrictive protective gear; this makes robot carriers attractive. Since medical doctors may not be permitted inside the hot zone, which can extend for kilometers, robot carriers that support telemedicine may be of huge benefit. Removing survivors, particularly from a hazardous material event where there may be many people immobilized or disoriented is another area that is getting increasing attention. The US Army Telemedicine and Advanced Technology Research Center has been leading initiatives in robots such as the Vecna BEAR for autonomous casualty extraction [60.11]. Other efforts such as those by *Yim* and *Laucharoen* [60.12] have focused on civilian casualty extraction which requires stabilization of the spine.

Acting as a mobile beacon or repeater to extend wireless communication ranges, enable localization of personnel based on radio signal transmissions by providing more receivers, and to serve as landmarks to allow rescuers to localize themselves. Active or passive transponders such as wireless sensor nodes and RFIDs (radio-frequency identification) can be deployed in the field. They can also be used to support the localization of personnel, but even more to leave mission-related data such as the location of nearby victims and hazards in the field [60.13].

Serving as a surrogate for a team member, such as a safety officer or a logistics person. In this task, the robot works side-by-side with rescuers, for example, a group breaching rubble deep within the interior of a disaster may have difficulty using a radio to request additional resources because of noise. However, a team member outside of the rubble can see and hear through the robot the state of progress and anticipate needs. The objective is to use robots to speed up and reduce the demands of tasks, even if they are done by humans.

Adaptively shoring unstable rubble to expedite the extrication process. Rubble removal is often hindered by the need to adopt a conservative pace in order to prevent a secondary collapse that might further injure a trapped survivor.

Providing logistics support by automating the transportation of equipment and supplies from storage areas to teams or distribution points within the hot zone.

60.1.3 Types of Disaster Robots

Rescue robots are needed to help quickly locate, assess, stabilize, and extricate victims who cannot be easily reached. They typically do this by extending the rescuers' ability to see and act. On the ground, small unmanned ground vehicles (UGVs) can enable rescuers to find and interact with trapped victims in voids that are too small or too dangerous for human or canine entry. Large UGVs can accomplish tasks such as removing large rubble faster than humans. In the air, unmanned aerial vehicles (UAVs) robots extend the senses of the responders by providing a bird's eye view of the situation. In the water, unmanned marine vehicles (UMVs), either underwater vehicles (UUVs) or surface vehicles (USVs) robots can similarly extend and enhance the rescuers' senses.

Rescue robots can be broadly categorized into types based on *modality* and *size* [60.14], though other taxonomies that mix modality, size, and task have been proposed [60.15]. There are three modalities of robots: *ground*, *aerial*, and *marine*. The modality impacts on the basic design and capabilities of the robot. Within each modality, rescue robots can be further described as one of three sizes: *man-packable*, *man-portable*, and *maxi*. The size of the robot impacts both on the tasks for which it is suited and how soon after a disaster it might be used. In order to be man-packable, the entire robot system, including the control unit, batteries,

and tools, must fit into one or two backpacks. Manpackable robots are more likely to be used immediately after a disaster since they can be carried by responders over debris and up and down ladders into the core of the disaster, while larger equipment must wait for paths to be carved out. The next larger size is man-portable; these are robots that can be carried a short distance by two people or on a small all-terrain vehicle. Man-portable robots function in human-sized spaces such as mine tunnels. They can be used as accessibility within the hot zone improves or outside the hot zone for logistics support. Maxi-sized robots require trailers or other special transportation logistics. This inhibits their insertion into the hot zone.

Aerial rescue robots represent the most advanced robotics technology in use, and new concepts continue to emerge. Aerial vehicles can be further subdivided into *fixed wing* (plane-like), *rotary wing* (helicopters), *lighter-than-air* (blimp), and *tethered* (kite) platforms. Fixed-wing UAVs typically travel long distances and can circle points of interest, while rotary-wing platforms can hover and require a small area to launch and land. Lighter-than-air vehicles may be tethered, similar to a kite. Underwater vehicles may also be subdivided into tethered and untethered platforms.

Large unmanned helicopters, such as the Yamaha R-Max, continue to be adapted for commercial rescue-and-recovery missions, including carrying heavy payloads capable of estimating the amount of rubble and debris generated by a disaster. Small fixed-wing platforms for tactical military use may have great benefit to the response community. However, UAVs are drawing the attention of agencies that control airspace and may become heavily regulated in the future. For example, the small UAV used at Haiti in 2010 flew in violation of airspace restrictions.

60.2 Disaster Characteristics and Impact on Robots

The type of disaster as well as the general pattern of activity influence the choice of robot platforms and payloads.

60.2.1 Natural Disasters

Natural disasters, such as earthquakes, tsunamis, hurricanes and typhoons, volcanoes, avalanches, landslides, and floods, present many challenges for rescue robots. Natural disasters are usually geographically distributed, perhaps affecting a 200 km or more radius around the epicenter of the event. The sheer size of the affected area presents many challenges to the emergency response. The primary impact of natural disasters is on residences, light commercial buildings, sea walls and canals, and transportation and communications infrastructure. This implies that rescuers have thousands of structures to check quickly for survivors, but those structures will be fairly small and amenable to manual and canine search. Besides the sheer volume of structures to check, communication disruptions prevent rescuers from getting timely information as to the state of transportation access and the general needs of an area. However, designing robots to meet these challenges is important because natural disasters provide the most hope of a large number of survivors. Uninjured survivors may simply be stranded and can survive for up to 72 h. Natural disasters have generally favored a bird's eye view from a UAV invaluable in establishing situation awareness or UMVs in determining *hidden* conditions of critical transportation, power, and communication infrastructure.

60.2.2 Manmade Disasters

In comparison to natural disasters, manmade disasters (such as a terrorist bombing or an industrial accident) occur in a small area, though may have significant wide-area impacts (e.g., radiation, chemical release). The challenge is often not how to see the entire external extent of the damage, but rather to see what is not visible: the interior of the rubble, the location and condition of survivors, the state of potentially dangerous utilities (e.g., electricity, gas lines) [60.14, 16, 17], and hazardous conditions such as radiation [60.18]. The communications and power infrastructure usually exist within a 10 km range and cell phones generally work outside of the directly affected area. Voids in the rubble may be irregular in shape and vertical in orientation. Wireless communications in the interior of the rubble are unpredictable, and generally nonexistent due to the large amount of steel within commercial structures, but the combination of irregular voids and sharp rubble do not favor the use of fiber optic cables. Visibility is difficult as there is no lighting and everything may be covered with layers of gray dust, further hampering recognition of victims, potential hazards, or accurate mapping. The interiors may be wet or contain standing water due to water lines, sewers, and sprinkler systems. Survivors are more likely to be in dire need of medical attention. Small ground robots that can enter deep into the interior of the rubble have been used most frequently for structural collapses, mid-sized robots for CBRNE events, and large, intrinsically safe robots for mine collapses.

60.2.3 General Pattern of Activity During the Response Phase

In order to understand how to apply robotics to disaster response (as opposed to recovery or prevention and preparedness), it is helpful to understand the general pattern of activity, which can be summarized as [60.19, 20]:

1. Responders become aware of the existence of victims. This awareness may be generated by information from family, neighbors, and colleagues, an understanding of demographic patterns (e.g., at night, apartment buildings will be heavily occupied, while during a work day, office buildings will be occupied), or by a systematic search.
2. The response command staff attempt to understand the disaster site. They investigate the site for conditions such as hazardous materials, the risk posed to the rescuers themselves, any pending threats to trapped victims, and resource restrictions such as barriers to transporting resources to the site, nearby usable equipment and materials that can be exploited, and any other barriers to a timely rescue.
3. The command staff plans the operations.
4. Search and reconnaissance teams are sent to map the situation and assess environmental conditions. Accurate estimates of the need for emergency medical intervention are highly desirable in order to optimize allocation of medication personnel in the field and to prepare ambulances and hospitals. (In the case of the Kobe earthquake, this stage took the longest time [60.19].)
5. Excavation of rubble to extract victims begins. Note that removal of rubble in search and rescue differs from construction removal of rubble, because the safety of the victims is the top priority.
6. Responders gain access to victims and apply emergency medicine in situ.
7. Victims are transferred to hospitals.
8. Field teams report activities periodically, usually at the end of the shift, and the command staff modifies or replans accordingly.

60.3 Robots Actually Used at Disasters

Small land, aerial, or marine vehicles have been used at 28 disasters and numerous local events, starting with the first use of UGVs at the 2001 World Trade Center collapse, see Table 60.1. The majority, 16, have taken place in the United States, followed by Japan (3), Italy (3), New Zealand (2), Haiti (1), Thailand (1), Cyrus (1), and Canada (1). The May 2013 rescue by a UAV guiding searchers to an injured motorist who had wandered from his car in Canada is the first reported *live save* of a person by a robot. However, as noted earlier, directly saving lives is not the only motivation for a robot. Robots have received in general high marks for completing their missions. Table 60.1 shows 37 reported deployments of robots for a disaster or well-known local incident. The deployments are grouped by major general disaster type of event: meteorological (e.g., hurricane, earthquake), geological event (e.g., earthquake, tsunami), man-made (e.g., terrorism, industrial accident), or mining (either due to a geological or a man-made event). Mine and manmade disasters are most frequent, followed by geological and meteorological events. In 7 of the 37 events, robots were present but were not used or could not reach the area of interest and thus considered not successful. UGVs have been used most frequently, followed by UAVs, and UMVs. While the USA initially led in reported adoptions, Europe and Asia are beginning to deploy robots. This section discusses the deployments by the four disaster types, while the following section highlights the 2011 Fukushima Daiichi nuclear accident as it highlights the types of robots, missions, and challenges posed during both the response and recovery phases of a disaster.

60.3.1 Meteorological Events

Robots were used at six meteorological events and were successful at five:

- 2005 Hurricane Katrina (USA), where an Inuktun ASR Xtreme UGV [60.21], an Aerovironment Raven fixed wing UAV and a Like90 helicopter UAV, and a Silver Fox fixed wing UAV were used during the initial response phase for search, reconnaissance and mapping, and structural inspection [60.22]. Later an iSENSYS IP3 was used for structural inspection during the recovery phase [60.23]. The UGV searched an apartment building unsafe for human entry and the UAVs surveyed areas inaccessible by car. The two UAVs used during the response are shown in Fig. 60.1 – a battery-powered fixed-wing and a battery-powered rotary-wing (a Like90 T-Rex miniature helicopter modified for stability in high winds) UAVs.

Table 60.1 The modalities and numbers of robots deployed to actual disasters and whether they were successful or not

Disaster or notable incident		Disaster type of event				Robots used			Success
		Meteoro-logical	Geo-logical	Manmade	Mining	Ground	Aerial	Marine	
2001	World Trade Center (USA)			✓		4			✓
2001	Jim Walters No. 5 mine (USA)				✓	1			✗
2002	Barrick Gold Dee mine (USA)				✓	1			✓
2004	Brown's Fork mine (USA)				✓	1			✓
2004	Niigati Chuetsu earthquake (Japan)		✓				1		✓
2004	Hurricane Charley (USA)	✓				1			✗
2004	Excel #3 Mine (USA)				✓	1			✓
2005	DR No. 1 mine (USA)				✓	1			✓
2005	McClane Canyon mine (USA)				✓	1			✓
2005	La Conchita mudslides (USA)		✓			1			✗
2005	Hurricane Katrina (USA)	✗				1	3		✓
2005	Hurricane Wilma (USA)	✓					1	1	✓
2006	Sago mine (USA)				✓	1			✗
2006	California wildfires (USA)	✓					1		✓
2007	Midas gold mine (USA)				✓	2			✓
2007	Crandall Canyon mine (USA)				✓	1			✓
2007	I-35 Minnesota bridge collapse (USA)			✓				2	✓
2007	Berkman Plaza II collapse (USA)			✓		2	1		✓
2008	Hurricane Ike (USA)	✓						1	✓
2009	Cologne State Archives collapse (Germany)			✓		2			✗
2009	L'aquila earthquake (Italy)		✓				1		✓
2010	Haiti earthquake (Haiti)		✓				1	1	✓
2010	Wangjialing coal mine (China)				✓	1			✗
2010	Upper Big Branch mine (USA)				✓	1			✗
2010	Deepwater Horizon (USA)			✓				16	✓
2010	Prospect Towers (USA)			✓		2			✓
2010	Missing balloonists (Italy)			✓				1	✓
2010	Pike River mine (NZ)				✓	2			✓
2011	Christchurch earthquake (NZ)		✓			1	1		✓
2011	Tohoku earthquake (Japan)		✓			3	1		✓
2011	Tohoku tsunami (Japan)		✓					9	✓
2011	Fukushima nuclear emergency (Japan)			✓		7	2		✓
2011	Naval base explosion (Cyprus)			✓			2		✓
2011	Thailand floods (Thailand)	✓					2		✓
2012	Costa Concordia (Italy)			✓				2	✓
2012	Finale Emilia earthquake (Italy)		✓				2		✓
2012	Missing person RCMP (Canada)			✓			1		✓
Total	37	6	8	11	12	23	13	8	37

- 2005 Hurricane Wilma, Ft. Myers, Florida (USA), where an AEOS unmanned marine surface vehicle (Fig. 60.2) and an iSENSYS T-Rex unmanned aerial vehicle were used independently and cooperatively, to assist with the recovery phase for structural inspection of buildings, docks, and piers [60.24]. The robots provided useful information but the USV experienced significant wireless communications and GPS (global positioning system) signals near and underneath large structures.
- 2006–2010 California wildfires (USA), where the National Aeronautics and Space Administration Ikhana Global Hawk was used to provide recon-

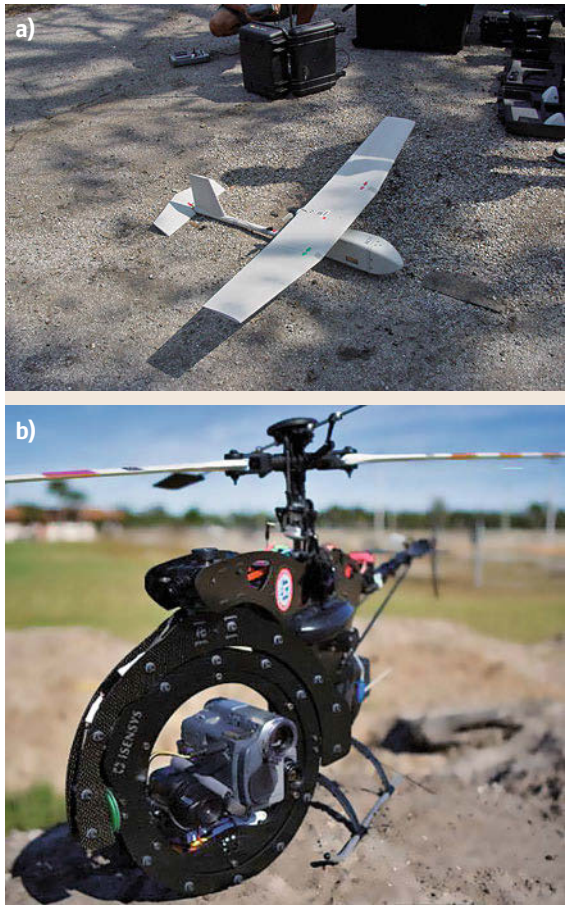


Fig.60.1a,b Man-packable UAVs used to search portions of Mississippi during the hurricane Katrina response: **(a)** an Aerovironment Raven fixed-wing UAV and **(b)** an iSENSYS IP3 rotary-wing UAV (courtesy of CRASAR)

Fig. 60.2 Unmanned surface vehicle exploring the bridge to Marco Island, Florida (courtesy of CRASAR)

naissance and mapping of 57 fires [60.25]. Ikhana was primarily used to gather data to develop new payloads but did contribute in 2010 to tactical response.

- 2008 Hurricane Ike, Rollover Pass Bridge, Texas (USA), where an AEOS unmanned marine surface vehicle, a VideoRay remotely operated vehicle (ROV), and a YSI Oceanmapper autonomous underwater vehicle (AUV) were used for structural inspection of the Rollover Pass bridge [60.26]. The USV was able to confirm the lack of scour around the pilings while the VideoRay could not be controlled in the high currents and the YSI AUV could not accurately preplan its path without colliding with the structure.
- 2011 Thailand floods (Thailand), where two unmanned UAVs provided reconnaissance and mapping of flood plains, allowing officials to concentrate evacuation or shoring efforts during the prevention and preparedness phases [60.27]. The UAVs are credited with mitigating damage to Bangkok and loss of life.

UGVs were present at 2004 Hurricane Charley (USA) but not used for the house-to-house inspection tasks that made up the bulk of ground operations [60.28].

60.3.2 Geological Events

Robots have been used at eight geological events (6 earthquakes, 1 landslide, 1 tsunami). They were successful at seven events:

- 2004 Niigata–Chuetsu Earthquake, where a variant of the Soryu III snake robot was used for search in collapsed houses. The snake robot was able to move into constricted passageways. Figure 60.3 shows the robot on site.

Fig. 60.3 IRS Soryu robot searching a house destroyed by the Niigata–Chuetsu earthquake (courtesy IRS)

Fig.60.4a–c Investigation of damaged building by Quince

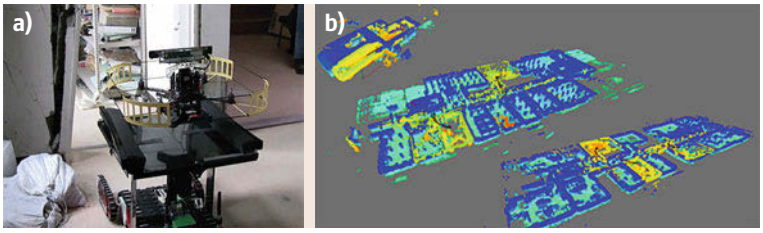


Fig. 60.5 **(a)** Docking system of Quince and Pelican, and **(b)** 3-D (three-dimensional) map measurement by collaboration

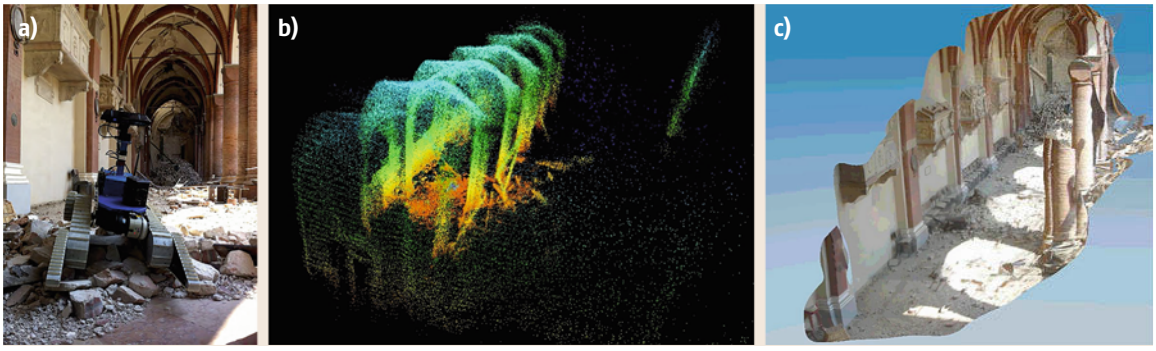


Fig.60.6a–c Finale Emilia earthquake: **(a)** the UGV platform deployed in the destroyed cathedral. **(b)** Reconstruction of the cathedral from laser measurements, **(c)** and image data

- 2009 L'aquila earthquake, L'aquila (Italy), where a customized AscTec quadrotor UAV was used for structural inspection. The UAV was able to fly successfully close to buildings and windows.
- 2010 Haiti earthquake (Haiti), where a SeaBotix LBV ROV was used for underwater reconnaissance and mapping of debris in shipping lanes and an Elbit Skylark UAV was used for reconnaissance and mapping of orphanages. Both the robots provide valuable information.
- 2011 Christchurch earthquake, Christchurch (New Zealand), where an iRobot Packbot UGV and a Parrot AR.drone UAV were used for structural inspection of the Christchurch catholic cathedral [60.29]. The Packbot was successful but the UAV's camera did not face upward and the ceiling could not be examined.
- 2011 Tohoku earthquake, Sendai (Japan), where a KOHGA3 and Quince UGVs (Fig. 60.4) and a Kenaf and Quince UGVs were used cooperatively with a Pelican UAV [60.30] (Fig. 60.5) for structural inspection of damaged buildings. The robots were able to provide information to structural engineers.
- 2011 Tohoku tsunami (Japan), where nine UMVs were used for reconnaissance and mapping, and victim recovery during the recovery phase. Five different teams deployed UMV: A SeaBotix SARbot, SeaBotix LBV-300, SeaMor, and Access ACROV were deployed by a joint IRS-CRASAR team in April 2011 and later again in October adding

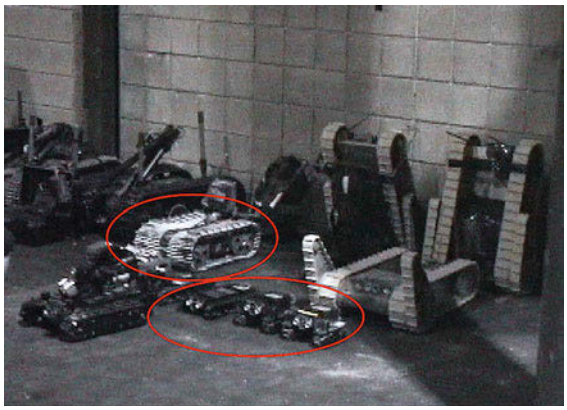


Fig. 60.7 Robots available at World Trade Center disaster. *Ovals* indicate the three types of robots used in the first two weeks (courtesy CRASAR)

a YSI Oceanmapper [60.31]; the IRS-CRASAR team was credited with reopening the port of Minamisanriku. Ura from Tokyo University fielded an unnamed ROV which recovered two dead bodies from an undisclosed location [60.32]. Prof. Shigeo Hirose at Tokyo Institute of Technology also created an experimental ROV and deployed with the Japanese Ground Self-Defense Forces [60.32]. IDEA Consultants (Japan) deployed their Mitsui ROV. Shibuya Diving Industry bought and used a VideoRay Pro 4, though the extent of the use is not known [60.33]. All UMVs were ROVs with the exception of the YSI Oceanmapper.

- 2012 Finale Emilia earthquake (Italy), where two custom unmanned aerial vehicles and two UGVs were used at Mirandola, Italy, for structural inspection of the exteriors and interiors of two churches that had not been entered due to safety reasons [60.34] (Fig. 60.6). The robots were able to provide information to the Italian National Fire Corps and state archaeologistics.

An Inunktun VGTV Xtreme UGV was unsuccessfully deployed at the La Conchita mudslide to search for possible survivors in houses collaterally damaged by the mudslide [60.35]. The robot failed on its two runs within two minutes and four minutes, respectively, due to severe mud and thick shag carpeting.

60.3.3 Manmade Events

Robots were at nine manmade events. They were successful at eight events:

- 2001 World Trade Center collapse (USA) [60.36], where Inuktun Micro-Tracks, Inuktun Micro-VGTV, Foster-Miller mini-Talon, and Foster-Miller Talon were used for search and reconnaissance and mapping in the interiors of the collapsed buildings (Fig. 60.7). The robots were able to penetrate farther than nonrobotic tools in spaces too small for human or canine entry.
- 2007 I-35 Bridge collapse, Minneapolis, Minnesota (USA) [60.37, 38], where two unspecified unmanned marine vehicles were used for search and reconnaissance and mapping underwater. The ROVs appeared to provide views without risking human divers in the fast moving currents.
- 2007 Berkman Plaza II parking garage collapse, Jacksonville, Florida (USA) [60.39, 40], where an Inuktun Micro-VGTV was used during the rescue phase for search and during the recovery phase an Active Scope Camera and an Inuktun ASR Xtreme were used for structural inspection along with an iSENSYS IP3 UAV for reconnaissance and mapping and structural inspection. The Micro-VGTV ruled out the presence of a survivor in areas too dangerous for a human to enter. The ASC and Xtreme provided structural forensic information that resolved legal liability and insurance coverage costs. The UAV provided imagery that led structural experts to hypothesize the cause of the collapse.
- 2010 Prospect Towers parking garage collapse, Hackensack, New Jersey (USA) [60.22], where an Inuktun VersaTrax 100 and an Inuktun VGTV-Xtreme were deployed for search. The robots were used to see vehicle information, license plates, makes and models in areas not safe for responders.
- 2011 Fukushima Daiichi nuclear accident (Japan) [60.18, 41], where seven UGVs and two UAVs were used for search, reconnaissance and mapping, and structural inspection. The robots provided damage assessment and radiological surveys without exposing workers to radiation. A UGV and a UAV were lost during the first year, but only after they provided important information.
- 2011 Evangelos Florakis naval base explosion (Cyprus) [60.17], where an AscTec Falcon and AscTec Hummingbird were used to inspect the damage to the adjacent Vasilikospower plant without requiring engineers to risk exposure to live unexploded ordinance. The UAVs were used for structural inspection. The operations were successful and led to a rapid repair of the power plant.
- 2012 Costa Concordia (Italy) [60.42, 43], where an Ageotec Perseo ROV and a VideoRay ROV, possible other unreported ROVs, were used for reconnaissance and mapping, structural inspections, and for victim recovery operations from the sunken cruise liner. The ROVs were considered essential in understanding the situation under the water.

- 2013 Missing person Royal Canadian Mounted Police (Canada) [60.44], where a Draganflyer X4-ES UAV with a thermal camera was used by the Royal Canadian Mounted Police to find an unconscious driver who had wandered from a car wreck. This is considered the first reported *live save* by a robot.

Robots were requested and present, but not used at the 2009 State Archives Building Collapse in Cologne, Germany, due in part to operator safety [60.45].

60.3.4 Mine Disasters

Robots were at 12 mining disasters. They were successful at seven events:

- 2002 Barrick Gold Dee mine, Elko, Nevada; 2004 Excel #3 mine, Pikesville, Kentucky; 2005 DR#1 mine, McClure, Virginia; 2005 McClane Canyon mine, Grand Junction, Colorado (USA), where a mine-permissible Remotec Wolverine (Fig. 60.8) was deployed the US Mine Safety and Health Administration (MSHA) used by to reopen mines that had been closed [60.46]. At the Barrick Gold Dee mine, the V-2 was deployed from the surface down a 16° slope and was able to navigate and to take continuous gas samples. At the Alliance Resources' Partners Excel #3 (coal) mine the robot was able to penetrate 230 m into the mine and successfully completed the objective of providing an assessment of the situation. At the DR#1 Dixon-Russell (coal) mine the robot was able to penetrate 210 to 250 m into the mine with a slope of 18° and the robot arm was used to move and realign ceiling supports in order to progress into the mine. At the McClane Canyon (coal) mine trials were conducted to establish manipulation capabilities. In this case, the robot was tasked to close five doors and pull out timbers holding up a mine fan. The robot was generally unsuccessful with manipulation tasks.
- 2007 Midas gold mine, Midas, Nevada (USA), where an Allen-Vanguard from Fallon Naval Air Station and a Inuktun VGTV-Xtreme variant from the Center for Robot-Assisted Search and Rescue (CRASAR) were lowered into the mine and used for victim recovery [60.47]. Mine-permissible robots were not needed as the gold mine was not emitting methane. The Allen-Vanguard that was lowered into the void and found the machinery but not the victim. The Xtreme was able to penetrate 35 m vertically into the void (Fig. 60.9). The robot scanned the area where the body was recovered but did not have sufficient lighting to actually see the body.
- 2007 Crandall Canyon mine, Huntington, Utah (USA), where a custom Inuktun mine Cavern Crawler robot was deployed by US mine Safety and

Fig. 60.8 The mine-permissible variant of the Wolverine robot used at the 2006 Sago Mine Disaster (courtesy of the US Mine Health and Safety Administration)

Fig. 60.9 The Inuktun VGTV-Xtreme robot being lowered into the Midas gold mine (courtesy of CRASAR)

Fig. 60.10 The Inuktun mine crawler robot being lowered into a borehole at the Crandall Canyon mine (courtesy of CRASAR)

Health Administration to search for missing miners (Fig. 60.10) [60.47]. The robot had to travel over 430 m through a 22 cm diameter borehole. Four attempts were made to enter the mine through two different boreholes, but only one was successful. In the fourth run, the robot was able to travel through the borehole then search about 2 m on the mine floor which was largely impassable due to the debris and drilling tailings. The robot provided no sign of the miners.

- 2010 Pike River coal mine, Greymouth (New Zealand), where two unknown New Zealand Defence Force bomb squad robots and Western Australia Water company pipeline inspection robot were used for search and recovery operations. The first New Zealand Defence Force (NZDF) robot ailed after reaching 550 m into the mine due to falling water, but was successfully restarted and moved out of the way of the second robot before running out of battery power. The first and second robots were presumed destroyed in a second explosion that ended any expectations of survivors. The pipeline inspection robot was later used for recovery operations.

Robots were unsuccessful at two events. At the 2004 Browns Fork mine, Hazard, Kentucky, disaster, the Wolverine robot was too tall to go into the area of interest to conduct a search. At the 2006 Sago mine Disaster, Sago, West Virginia (USA), the robot was only able to penetrate about 700 m into the mine before being damaged when it accidentally was driven off the path and flipped over [60.46].

Robots were on-site but not used at three events: the 2001 Jim Walters #5 mine fire (USA) as the robot was not mine-permissible, the 2010 Wangjialing coal mine (China) presumably because the unidentified robot could not transit high water, and the 2010 Upper Big Branch mine (USA) where the restricted confines posed too great a risk of failure. Robots were requested for the 2010 San Jose Copper-Gold mine (Chile) and the 2011 San Juan De Sabinas coal mine (Mexico) disasters but robots were not small enough and thus were not sent.

60.4 Robots at the Fukushima-Daiichi Nuclear Power Plant Accident

The Fukushima-Daiichi nuclear power plant accident merits special attention because robots have been used for both the response and recovery phases. The use of robots have been sustained over months and years, and a wide variety of robots have been used.

Control over the reactors at the Fukushima-Daiichi nuclear power plant was lost on March 11, 2011, after a 14 m high tsunami from the Eastern Japan Earthquake submerged the facility. Units 1–3 melted down, hydrogen exploded at the high floors of nuclear reactor buildings of Units 1, 3, and 4, and huge amounts of radioactive materials were released to the wide area centered by Fukushima Prefecture. Robots were used during the prolonged response phase for assessment of the emergency situation and mitigation of the event by recovering the cooling systems to enable a cool shutdown. After four months, the immediate response turned to preparation for decommission as the reactors were stabilized. The recovery phase requires operations in the contaminated area; therefore, many robotic systems have been, and are continuing to be, used for minimizing workers' exposure to radiation by substituting for human operations. In the first year, ground and aerial robots were used for exterior operations outside the buildings and ground robots for interior operations inside the reactor buildings.

60.4.1 Exterior Operations

Robots were used outside the buildings and on the grounds by different teams for four applications: debris removal, structural inspection, radiological surveys, and mitigation efforts.

Debris Removal by Unmanned Construction Machines

There was a large amount of contaminated debris outside the building producing a high dose rate of a few hundred mSv/h, preventing workers accessing the buildings. Debris removal operations were conducted starting April 1, 2011 for nine months using an adaptation of unmanned construction robot machines had been developed for the restoration work after the *Unzen Fugen-dake Volcano* eruption in 1991 (Fig. 60.11). Debris was gathered by two backhoes and a bulldozer, with wireless teleoperation, then loaded to containers on two remote-controlled crawler dump trucks, and carried to storage yards of contaminated materials. The crawler dump trucks were teleoperated from a remote operator car, with command and control signals and data from camera and sensors transmitted via a remote-controlled radio relay car. Seven camera cars were connected to the system by wire, and external image views supported the task execution. As

Fig.60.11a,b Unmanned construction machines used at Fukushima-Daiichi (courtesy of Unmanned Construction Machines Association)

a result, debris of 20 000 m^3 was removed from the area 56 000 m^2 wide, and the radiation level reduced significantly to safely accessible level for the workers.

Structural Assessment and Radiological Surveys from T-Hawk UAV

A team led by Westinghouse used Honeywell T-Hawks from April 10, 2011, to the end of July, 2011 (Fig. 60.12). The team flew approximately 40 missions with two objectives originally. One was to investigate the state of the reactor and turbine buildings and to acquire video of the general area in order to support initial assessments of the incident and to help plan for debris removal. The second was to take radiological samples to produce a survey map. The objectives later expanded to off-site debris field survey, survey of the coastal breakwater structure, a gamma radiation survey at specific locations of interest to TEPCO, and airborne-particulate sampling (flying into plumes) at select locations.

Radiation Source Measurement by Gamma Cameras

The Japan Atomic Energy Agency (JAEA) developed a robot control car *TEAM NIPPON* with a gamma camera, a 3-D camera, a thermo camera, and a QinetiQ Talon robot that were provided by Idaho National Laboratory, a US federal laboratory that conducts research in nuclear reactors, in collaboration with Tohoku University. It was used from May 5, 2011 for measuring radiation of the outdoor debris, and provided visualized data as shown in Fig. 60.13 [60.18, 41].

Remote Control of Concrete Pumping Cars

Concrete pumping cars had been used for pouring water from outside in order to cool used-fuel pools on the top floor of the exploded nuclear reactor buildings. In order to reduce the radiation exposure of workers who control their long arms, the cars were fitted for remote teleoperation by using remote controllers, cameras, and wireless equipment on May 2011.

60.4.2 Interior Operations in Nuclear Reactor Buildings

UGVs continue to be used for operations in the interior of the reactor buildings. UGVs were used to survey the interiors (iRobot Packbot, IRS Quince), monitor for gamma radiation (JAEA J-3), remove debris (QinetiQ Bobcat and Talon, Brokk), and experiment with decontamination (iRobot Warrior).

Survey of Nuclear Reactor Buildings by PackBots

On April 17, 2011, TEPCO workers and six engineers from iRobot used two donated PackBots to open the double hatch doors of Units 3 and 1, providing the first entry into those buildings (Fig. 60.14). The TEPCO workers carried the robots to the front of the hatch, and pointed an antenna to the robots from the glass window of the hatch. Based on the readings of radiation dose rate, temperature, humidity, and gas concentration, humans were allowed to enter the nuclear reactor buildings to begin the cool shutdown process. On the following day, the same mission was tried in Unit 2, but the PackBot could not complete it. The camera window of PackBot misted over under high humidity, because the vapor from boiled water on the basement floor did not escape from the building. Prior to the actual operations, TEPCO workers were intensively trained on remote teleoperation starting on March 24, 2011, and the scenario was verified in Unit 5, which was not dam-

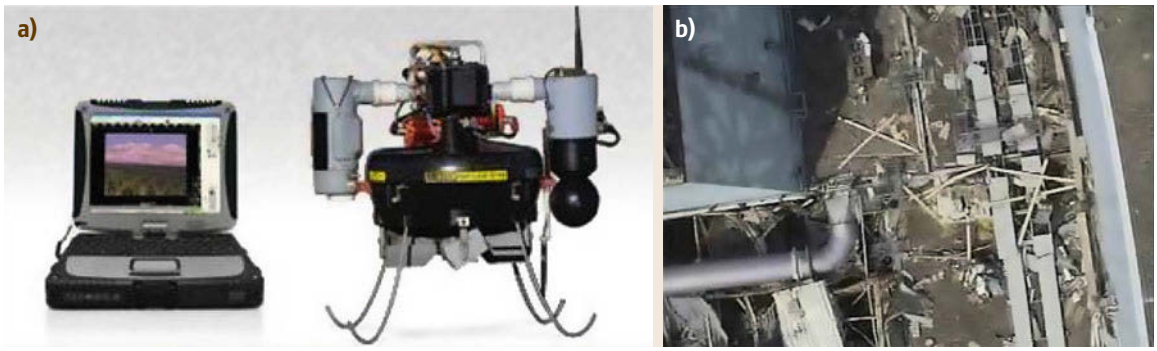


Fig.60.12a,b UAV used at Fukushima-Daiichi: **(a)** T-Hawk and **(b)** top view of nuclear reactor building of Unit 1 (photographs courtesy of TEPCO)

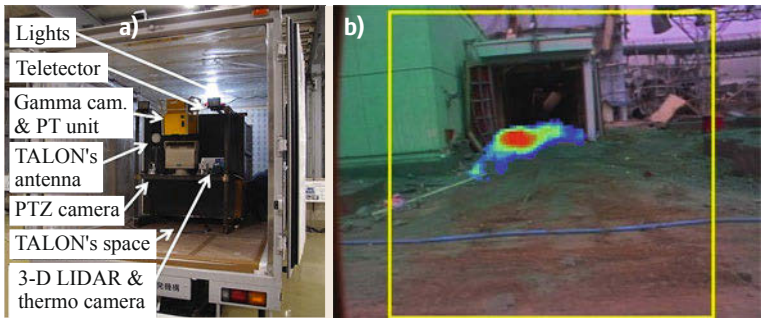


Fig.60.13a,b Gamma radiation measurement by UGVs: **(a)** *TEAM NIPPON* and **(b)** visualized radiation source by gamma camera (courtesy of JAEA)

Fig.60.14a,b Operations of PackBots in Unit 1 (courtesy of TEPCO)

aged. The PackBots continue to be used for monitoring and light tasks in the buildings.

Survey of Nuclear Reactor Buildings by Quince

The Quince robot was used in addition to the iRobot Packbots in June, 2011 (Fig. 60.14), and is now being used in cooperation with the Packbots for various missions as only Quince could climb the steps to the upper floors. Quince had been developed for CBRNE disaster response by a team from the International Rescue System Institute including Chiba Institute of Technology and Tohoku University. One unit of Quince was lent to TEPCO for free after radiation-proof test, wireless communication test, intensive refinement of reliability, and training exercises of operators [60.48]. Quince has been used for surveillance of the nuclear reactor building since June 24, 2011. On July 8, it entered Unit 2 and measured the dose rate on the second and third floors, then sampled dust for identification of nuclide (Fig. 60.15). On July 26, valves and pipes of a spray

Fig.60.15a,b Quince at Fukushima: **(a)** robot platform and **(b)** its operator station image captured in the mission on July 8, 2011 (courtesy of TEPCO)

cooling system were visually inspected by the high-density camera and dose rate was measured in Unit 3. Redundant cooling system was established on the basis of these data. On October 20, Quince surveyed fully on the third floor of Unit 2, then climbed the stairs up to the fifth floor (operation floor) for monitoring. The high dose rate, condition of the crane, etc. were investigated on the high floors at first. However, it stopped on the third floor on its return home because its communication tether was hooked and cut. If Quince had not been applied, it would have been delayed to examine situation of the upper floors, and the risk would have become higher because of late cool shutdown. Quince 2 and 3 were remodeled and lent for free of charge to TEPCO from 2012, and are used in cooperation with PackBots for various missions.

60.5 Lessons Learned, Challenges, and Novel Approaches

The experiences with disaster robotics since 2001 provide key insights into technical challenges, such as mobility, communications, sensing, control, and human factors, but also into sociotechnical issues such as training and procedures for transporting and decontaminating robots. This section highlights the general lessons learned and the fundamental problems posed in mobility for UGVs, UAVs, and UMVs, communications, control, sensors and sensing, power, human–robot interaction, multirobot team coordination, and other issues.

60.5.1 Mobility: UGVs

Mobility remains a major problem for all modalities of rescue robots, but especially for ground vehicles. The challenges for ground robots stem from the complexity of the environment, which is an unpredictable combination of vertical and horizontal elements with unknown surface characteristics and obstacles. For example, at Fukushima, the original configuration of the iRobot Packbot could not climb the steep metal staircases. In other cases, a robot could not cross a catwalk or climb stairs due to rubble and the lack of confidence that the robot could negotiate the debris. Robots such as the Inuktun VGTV-Xtreme at the La Conchita mudslides could not handle the terrain, while the Allen-Vanguard was too heavy for vertical entry into the Midas gold mine.

The field is currently lacking any useful characterizations of rubble to facilitate better design. However, even without a complete understanding of rubble environments, it is clear that more work is needed in actuation and mechanical design as well as in algorithms that would enable the robot to adapt its mobility style to the current terrain (also known as *task shaping* [60.49]).

The state of the practice in UGVs for disasters are polymorphic tracked vehicles, with a movement toward tracked robots with manipulators (e.g., the Wolverine used for mine disasters, the robots at Fukushima) and two instances of snake-like robots (Niigata–Chuetsu earthquake, Berkman Plaza II collapse). Wheeled platforms are severely limited by the roughness of the terrain and the need to overcome obstacles, steps, ramps, but combination wheeled and tracked vehicles

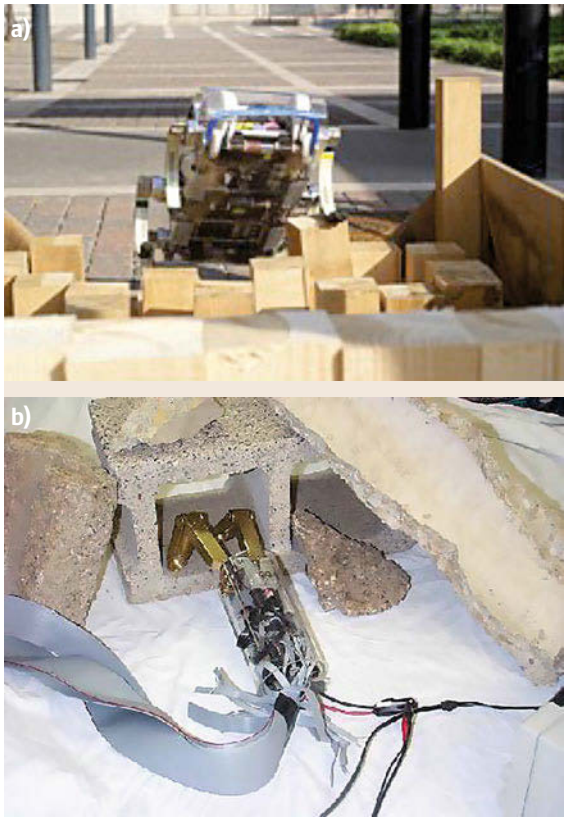


Fig.60.16a,b Examples of legged and crawler robots. **(a)** Hexapod (legged) robot from the RHex project traversing a portion of the National Institute of Standards and Technology (NIST) test bed (courtesy of R. Sheh) and **(b)** Terminatorbot being tested in rescue test bed (courtesy of R. Voyles)

are commercially available. Basic research in legged robots (Chap. 48), wheeled robots (Chap. 49), micro/nanorobots (Chap. 27), and manipulators will benefit disaster robotics.

Free serpentine robots such as the Soryu III used at the rubble at the Niigata–Chuetsu earthquake provide a fundamentally different style of mobility, while *fixed-based* snakes can be used in tandem with a more traditional platform as a highly flexible sensor manipulator [60.50]. Examples of both types of snake robots are shown in Figs. 60.3 and 60.19.

In order to overcome the difficulties posed by unknown terrain, novel legged robots and crawler robots have been proposed. In addition, some types of crawler robots can climb walls and reach locations that would otherwise be very hard to reach. Legs are interesting because they exploit biomimetic principles. The RHex hexapod robot [60.51], proposed for search and rescue among other potential applications, is shown in Fig. 60.16 climbing random step fields. Crawler robots such as the Terminatorbot [60.52] use their arms or legs to pull themselves through the rubble. The Terminatorbot is designed to withdraw itself into a cylinder that can be inserted through one of the small boreholes commonly drilled by responders to get through walls, then open up and begin moving. Other types of crawlers include lizard- or gecko-like robots that adhere to walls; these types are promising but have not been tested for the dusty, wet, and irregular conditions found in disasters.

Another novel concept for UGVs is that of *smart* tools, particularly lifts that can help stabilize collapsed structures during extrication [60.53, 54]. Extrication is one of the most time-consuming activities in rescue. Rescuers must proceed cautiously when removing rubble in order to prevent secondary collapses or slides that would further injure the survivors. Experiments suggest that roboticized lifts or shoring mechanisms would be able to sense and respond fast enough to small movements in the rubble to adaptively maintain stability [60.54].

60.5.2 Mobility: UAVs

UAVs are increasing in popularity with responders, possibly because on demand aerial access is a unique capability. Aerial vehicles, particularly helicopters, are vulnerable to wind conditions near (or in) structures and obstacles such as power lines, trees, and overhanging debris. A UAV at Hurricane Katrina hit a power line.

Novel ideas include a plane with the size of a person's hand that can fly indoors and planes with foldable wings, making it easier for responders to carry them. Quadrotor helicopters appear far more stable and easier to pilot; a design that balanced the larger size of a quadrotor with an appropriate payload could make UAVs more assessable to nonpilots. Another exciting direction is hybrid platforms that can change from fixed-wing operations, covering large distances, to rotary-wing operations, flying near or inside buildings [60.55].

60.5.3 Mobility: UMVs

Surface and underwater vehicles have to contend with swift currents and floating or submerged debris, placing significant demands on agility and control. As seen at the Tohoku Tsunami, AUVs could not be used due to the possibility of colliding with flotsam. Teleoperated ROVs can function in such conditions but have a risk of tethers becoming tangled or caught, as seen at the Tohoku Tsunami and the Hurricane Ike response.

60.5.4 Communications

Robots rely on real-time communications for teleoperation and for enabling responders to see what the robot is seeing immediately. Ground robots communicate either through a tether or via wireless radio. Aerial and surface robots are wireless, while underwater rescue robots are controlled via a tether. The communication bandwidth demands of all modalities are generally high due to the use of video imagery, and the tolerance to communications latency is low due to the control needs. In addition to communications between the tactical rescuers and their robots, it is difficult to report or transfer critical information provided by rescue robots to the strategic enterprise. Disasters typically destroy the communication infrastructure, both telephones and cellular phones, and alternatives such as satellite phones become saturated by response agencies.

Wireless communications with robots remain problematic. Operations below ground or near structures interfere with the physical propagation of radio signals. As shown in high-fidelity USAR response exercises, ad hoc wireless networks established by responders are likely to become quickly saturated, with no way of establishing priority over information. At the World Trade Center, data from the Solem robot deployed in the interior of building WTC4 returned totally black frames for 1 min 40 s of the 7 min run before wireless communications were totally lost and the robot was abandoned [60.36]. In addition, many wireless robots use lossy compression algorithms to manage bandwidth, which interfere with computer vision techniques, and/or connect through insecure links, raising the possibility that news media might intercept and broadcast sensitive video of trapped victims.

Rescue robots working underground, either for USAR or mine rescue, have two alternatives to wireless communications: either operate with a tether or deploy repeaters to maintain wireless communications. Many wireless robots now can be purchased with a fiber-optic tether; however, these tethers are fragile and may break or tangle, as seen at several mine disasters. The fiber-optic tether may also tangle with the safety rope used to support the robot during vertical drops. Data for the World Trade Center deployments indicated that a dedicated person was required to manage the tether but that 54% of tether management operations were to allow the robot to reach a more favorable position or to recover the robot after its mission [60.36].

At Fukushima, outside the buildings, allocation of wireless frequencies and accommodation of communication methods were extremely difficult, and usage of some robot frequently affected the other robots. Inside the buildings, wireless communication was sometimes unstable, and a few robots could not return. Operators always had to pay attention to radio field strength and cable tether handling.

Majority of UGV operations have used tethered robots, with the only wireless robot used at the WTC disaster being lost and the wireless NZDF robots creating significant problems with intermittent failures. Hybrid communications, in which a robot is primarily on a tether and then operates for short distances over a local wireless link before reconnecting to the tether, appear to be attractive.

A novel approach is to use other robots as repeaters, either stationary or mobile, to facilitate the establishment of communications and sensor networks. As repeaters for mobile ad hoc networks, robots on the land, sea, and air can extend the range and throughput of wireless networks [60.56]. A recent US Defense Advanced Research Projects Agency program, LANdroids, developed a set of small robots that acted as mobile repeaters for a larger robot entering the interior of a structure or tunnel. Aerial vehicles are particularly attractive as they can provide larger relay ranges while providing a bird's eye view of the disaster. However, a UAV does not always have to move: a tethered blimp or kite can support a sophisticated payload with no maintenance or support for days [60.57, 58].

60.5.5 Control

Robot control can be subdivided into *platform control*, which is usually considered by control theory, and *activity control*, which generally falls under the purview of artificial intelligence. Rescue robots are challenging both for traditional control and for artificial intelligence. The high degree of mechanical complexity of all modalities and the demands of the environment present major challenges for control theory. In all report events, the robot's activity was handled by teleoperation; a human is needed to direct the robot and to perform mission sensing. The well-documented problems of manual teleoperation (Chap. 34) argue for increasing autonomy [60.59], both in terms of navigation.

Navigational autonomy has been the primary focus within the disaster robotics community and efforts generally fall into the following three categories:

- *Simultaneous localization and mapping* (SLAM): SLAM is needed for the search and reconnaissance and mapping tasks. Whereas SLAM in office-like environments appears to be almost solved, the application of same techniques in harsh and unstructured environments turns out to be extraordinarily difficult or even impossible. One strong limitation of current mapping solutions is their inability to pro-

vide an accurate 3-D model of the environment at high data rate. For USAR such a solution has to cope with the reflection properties of debris, smoke due to fires, and the spectrum of lightning conditions. The solution also needs to be low cost in terms of required onboard CPU (central processing unit) computation and payload, due to weight and power limitations. Existing solutions rely either on automatically inclined [60.60, 61] or constantly rotating 2-D (two-dimensional) laser scanners [60.62]. In contrary to 2-D maps, which are typically generated in real time [60.63], 3-D maps are updated at low rates on the order of seconds due to the limited data rate of inclined or rotated laser scanners.

- *Exploration, planning, and path execution*: The central question in autonomous exploration is: *given what you know about the world, where should you move next to gain as much as many new information as possible?* [60.64]. Autonomous exploration can be subdivided into determining where to go next, how to plan the path to get there, and how to execute that path safely.

  The primary approach to exploration is to use a frontier-based exploration algorithm is to gain new information by moving to the boundary between open space and unknown territory, denoted as *frontier*. Frontier-based exploration, either based on 2-D [60.64] or 3-D [60.65] map representations, is an efficient technique that has been successfully deployed together with SLAM techniques for exploration in NIST-like USAR arenas [60.66].

  The objective of path planning in harsh environments is not necessarily to generate the shortest but the safest path. *Wirth* and *Pellenz* introduced an exploration strategy and path planner that utilizes occupancy grid maps for combining the distance transform and the obstacle transform (when planning to several frontier cells at the same time [60.67] in order to select the safest alternative consisting of target location and path to reach the target.

  Most methods for motion planning in USAR are still limited to static environments that can efficiently be represented in 2-D. However, since robots operate on rough terrain that may suddenly shift it is important to consider the shape of the surrounding terrain and its potential impact when moving. Several researchers have introduced solutions with short-term planning look ahead that execute specific robot behaviors with respect to the current situation of the robot. *Okada* et al. introduced an autonomous controller for tracked vehicles that is based on continuous 3-D terrain scanning [60.68]. *Magid* et al. introduced a system for keeping the robot maximally stable at every step of its path while allowing the vehicle to loose balance in a controlled manner for facilitating safe climbing over debris [60.69]. *Sheh* et al. developed a method for *behavioral cloning*, a type of learning by imitation that produces control rules that clone the skills of an expert human operator [60.70]. *Dornhege* et al. introduced the concept of behavior maps, which link certain robot behaviors on rough terrain such as climbing over stairs and obstacles, to structures detected from 3-D point clouds in real-time [60.71].
- *Object recognition and scene interpretation*: The search, reconnaissance and mapping, and structural inspection tasks for disaster robots would benefit from autonomy. They currently rely on humans to manually scan video feeds for signs of survivors, reconstruct the scene, identify potential hazards, and accurately comprehend the integrity of structures. To this end, *Andriluka* et al. evaluated various state-of-the-art techniques for vision-based victim detection from UAVs [60.72]. They concluded that by combining multiple weak models the overall detection reliability can be increased. *Kleiner* and *Kummerle* presented an approach for detecting victims based on several visual features such as body motion, skin color, and body heat [60.73]. *Hahn* et al. presented an approach for improving victim detection from low-cost thermal sensors by computing heat distributions while exploring an environment [60.74]. *Birk* et al. developed a system for recognizing humans from images taken by an infrared camera that considers plausible body shapes and postures [60.75].

60.5.6 Sensors and Sensing

Sensors, and sensing, pose the greatest mission challenge; without adequate sensing, a robot may be in an area of interest but be unable to navigate or to execute the larger mission. The physical attributes of a sensor (size, weight, and power demands) impact on whether it can be used with a particular robot platform. Currently sensors are not interchangeable between platforms; standards are needed for footprint sizes, mounting, connections, and display space.

The functionality of a sensor depends on the modality and mission [60.16]. The primary sensor missing from all robot modalities operating outside of water is a miniature range sensor. With a miniature range sensor, the success in localization and mapping seen with larger robots would be transferable to rescue robots. The sensor payloads for other missions depend on those applications; however, two sensor needs for USAR are

particularly noteworthy. One is a detector that can perceive victims obscured by rubble; current radars have not been reliable in mixed rubble. Another needed sensor is one that can tell if a victim is unconscious but alive without touching the victim. A robot may be able to see a victim, yet not be able to crawl to the victim and make contact or scrape away enough dirt or clothing to take a pulse. Stand-off detectors such as millimeter-wave radars and gas detectors appear promising but have not been validated at this time.

Smaller, better sensors are not sufficient; improvements in sensing algorithms are also needed. At this time, humans are expected to interpret all sensing data manually in real time. This is a daunting task for many reasons. Human performance is handicapped by physiological factors introduced by sensing through a computer display (also known as computer mediation), the location of sensors at viewpoints low to the ground, generally restricted fields of view, and fatigue. The modality output itself may also be nonintuitive, such as ground-penetrating radar. However, autonomous detection and general scene interpretation is considered well beyond the capabilities of computer vision. This presents a case where neither the human nor the computer can accomplish the perceptual task reliably and argues for investigation into human–computer cooperative techniques for perception. Algorithms that enhance the image for human inspection, supplement depth perception, or cue interesting areas are within the reach of computer vision.

60.5.7 Power

The robot modality and mission poses distinct challenges for power, although only one robot, the NZDF robot at the Pike River mine, has caused a failure. In general, battery power is preferred over internal combustion because of the logistics difficulties in transporting flammable liquids. While the requirements of each rescue robot application is largely unknown, a partial understanding is emerging of the power profile. For example, the operation tempo of a ground vehicle operating underground is on the order of 3–4 runs, each around 20 min in duration, over a 12–14 h shift, with the robot kept on hot standby for the majority of the shift. Rotary-wing aerial vehicles for tactical reconnaissance and structural inspection show an operations tempo of 5–8 min per face of a building, while fixed-wing vehicles are airborne for less than 20 min. Other rescue missions, such as wilderness search and rescue, will have different requirements but the need for batteries over internal combustion and for determining the power profile is the same.

60.5.8 Manipulation

Manipulation has been required for mine remediation (e.g., DR#1 and McClane Canyon mines) and at the Fukushima nuclear accident, but overall manipulation has failed in at least one task per mission. Furthermore, manipulation is almost always time consuming as all reported instances have been teleoperated. Robots with manipulators (such as that seen in Fig. 60.17) extend the capabilities of ground vehicles by allowing the robot to sample the environment, interact with survivors, move light obscurations, and add unique camera viewpoints. However, the manipulator extends the volume of the robot, impacting navigation. The arm is often at risk of being damaged in confined spaces being hitting on overhanging rubble. Manipulators also add to the control and mechanical complexity of the robot.

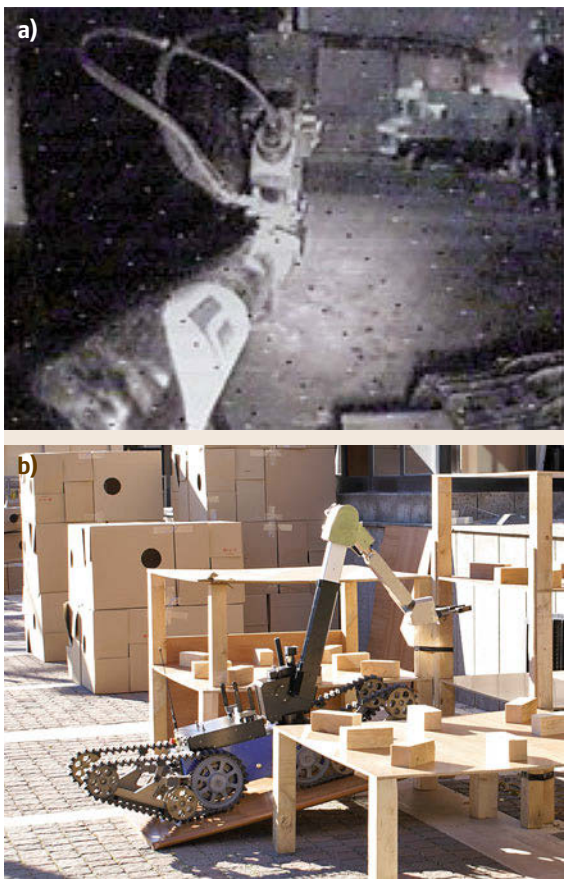


Fig.60.17a,b Examples of tracked robots with manipulator arms that have been used for disasters or rescue competitions. **(a)** View from Foster–Miller Talon at the WTC of its arm (courtesy of CRASAR) and **(b)** teleMAX by Telerob at the Rescue Robotics Camp (courtesy of R. Sheh)

60.5.9 Human–Robot Interaction (HRI)

According to the analysis by *Murphy* [60.22], over 50% of the robot failures during a disaster are due to human error, which emphasizes the importance of human–robot interaction. For the purposes of this chapter, HRI concerns fall into five topics: ergonomics, system design, human–robot ratio, situation awareness, and training.

One aspect of HRI is ergonomics. The Fukushima nuclear accident highlighted how personal protection equipment (e.g., full-face mask and rubbery gloves) precluded workers from performing delicate work and efficient task execution.

Another aspect of HRI is the larger system design. At the Fukushima nuclear accident, workers had to carry the robots to the point of entry; this sometimes increased the total radiation exposure of workers, which was clearly not the purpose of the robot. The inability to transport certain robots caused them to be excluded from the World Trade Center disaster.

The number of operators per robot has been the subject of debate, though *Murphy* and *Burke* in [60.76] offer a formula based on their comprehensive field studies for computing the baseline human-to-robot ratio

$$N_{\text{humans}} = N_{\text{vehicles}} + N_{\text{payloads}} + 1 \, .$$

The idea is to then justify why a smaller ratio would not increase unacceptable risk to the mission. For example, a UGV often does not need a safety officer, while a UAV does under most regulations. One way to decrease the number of people is to reduce their workload, such as by increasing the navigational autonomy from normal operations. However, Murphy and Burke document that if the unmanned system encounters a problem and returns control to the operator, the operator is unlikely to be able to react fast enough to avoid a crash, collision, or severing of a tether. This problem has been documented in aviation safety with initial problems with auto-pilots, where it is generally referred to as the *human out of the loop control* (OOTL) problem. The OOTL problem should not be interpreted as discouraging autonomy but rather encouraging a more comprehensive consideration of how autonomy will be handled in all conditions, not just normal operations.

Related to the number of operators is how well they can maintain situation awareness. Situation awareness is defined by *Endsley* in [60.77] as

> *the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future.*

Drury et al. [60.78] have defined types of situation awareness within search and rescue based on an analysis of RoboCup rescue tasks, while *Casper* and *Murphy* [60.79] and *Burke* et al. [60.80] have examined operator situation awareness in technology insertions. User interfaces are a key component in facilitating situation awareness. In rescue robotics, user interfaces are generally primitive and work through the operator's visual channel to provide robot, task, and situation awareness of the state of the robot, task progress, and the general operational environment. To highlight the importance of user interfaces, one robot at the World Trade Center was rejected because of the complexity of its interface [60.36]. While the user interface for fieldable rescue robots primarily display the video output from the robot, experiences from RoboCup rescue suggest that a good interface will both facilitate commanding the robot (inputs) and will provide three types of information (outputs):

- The robot's perspective: camera view(s) from the robot's current position, plus any environmental perceptions that enhance the general impression of telepresence
- Sensor and status information: critical information about the robot's internal state and its external sensors
- If possible, a map: a bird's eye view of the robot situated in the local environment.

The Fukushima nuclear accident illustrated the importance of training, as training of operators with using simulated mockups took over a month. The current state of the practice is for responders to receive training from the robot manufacturer on how to operate and maintain the robot, ignoring concepts of operations for actual scenarios. Training is a related issue in human–robot interaction [60.81]. Rescue workers have limited time compared with military operations to learn about robots and few opportunities to practice. While a bomb squad or special weapons and tactics team for law enforcement may be called out several times each month, a rescue team may be called out only a few times each year. For the near future, rescue workers may not have had prior training or experience with a robot prior to the disaster and be expected to use prototypes with only *hasty* training.

60.5.10 Multirobot Team Coordination

Multirobot teams of robots that cooperate with each other or work on the same objective have already been used or demonstrated for response and recovery activities at Hurricane Wilma (UMV-UAV) [60.24], the Tohoku tsunami (UMV-UMV) [60.31], and the To-

hoku earthquake (UAV-UGV) [60.30]. These have all been heterogeneous teams though homogeneous teams have been entered at the RoboCup rescue competition [60.82]. Only one of the three deployed teams was explicitly coordinated, where a Quince robot carried a Pelican UAV at the Tohoku earthquake and conducted cooperative mapping of the interior of a damaged building [60.30]. The current research approach to teams is focusing on adopting techniques from artificial intelligence for either centralized or distributed coordination.

Multirobot teams not only offer the possibility to field diverse capabilities, they also exhibit increased robustness due to redundancy, and superior performance by parallel task execution [60.83]. This latter aspect is as important as the first one, since the time to complete a rescue mission is of vital importance. For example, one fundamental problem when searching for survivors is to efficiently coordinate a team performing space exploration. However, the IRS-CRASAR team at the Tohoku tsunami response uncovered implicit assumptions about parallel task execution that limited performance [60.31]. The coordination of several autonomous units in search and rescue operations still represents an open issue. One reason is the fact that the higher number of degrees of freedom imposed by the robot team requires a larger number of trained human operators, which in turn makes the deployment during a rescue mission more complicated. Certainty, the number of required operators also depends on the quality of autonomous perception and decision-making of each single system. Therefore, in order to reduce the human-to-robot ratio in the field, autonomous capabilities as well as autonomous methods facilitating team coordination are needed. There are several existing techniques for team coordination in USAR, which can generally be separated into centralized and decentralized approaches.

There are three novel types of teams being pursued for disaster robotics. One type is swarms. Researchers such as [60.84] have discussed the possibility of using cost-effective, insect-sized robots to penetrate deep within a pancake building collapse and then signal the presence of a victim. One of the key features of swarm approaches is that they can scale up easily. The insect swarm scenario leaves hard problems like control, sensing payloads, localization of the victim, and communications to the imagination, but is certainly a worthy concept. However, some of the search algorithms used by insects may be adapted to single robots, for example, win-shift win-stay sampling exhibited by bees may be useful for search [60.85]. A second type of team, similar to swarms, is motes, where aerial vehicles that drop intelligent sensors called motes.

A third type of team is a hybrid robot-animal team, where search dogs carry roboticized cameras (dog-cams) [60.86] or wiring rats with controllers. Attaching cameras to search dogs has been explored for several years and such a system was used at the World Trade Center. The concept does not compete with robots, as robots are used to enter places canines cannot. The canine team handlers have generally objected to dog-cams because the cameras and communications gear interfere with the dog's mobility, pose the hazard of snagging, and the dog cannot be readily commanded to stop at points of interest beyond the line-of-sight of the handler (dogs use visual cues as commands, more than audio). However, there is less objection to placing nodes in a rat's brain to stimulate and drive the rat into a void while carrying a camera or other sensors [60.87]. The motivation for a robot rat stems in part from the rat's mobility and relative low value. While the technical feasibility of a robot rat may be within reason, assuming advances in wireless communications, the response community has been lukewarm toward the idea [60.88]. Unlike rats, robots can be kept in storage for years and can penetrate through pockets of fire or areas with no oxygen. A robot rat has all of the limitations of dog, including the problem of a handler becoming too emotionally attached, and is likely to scare a trapped survivor just as much as the other rats that swarm a disaster. The conventional wisdom is that, if the sensors, wireless, and power systems can be miniaturized and operate reliably enough to control a rat deep within rubble, those systems will enable responders and robots to work without the rats.

60.5.11 Other Issues

In addition to the challenges by the functional subsystems of a robot, three other issues should be considered.

Robots must be reliable. As noted in the discussions of the Pike River mine explosion and the Fukushima nuclear accident, *failure of a robot can obstruct the mission execution or cause it to completely fail*. A robot failing not only means the mission did not get performed, but could prevent other robots from carrying out the mission. For example, a robot getting stuck on the stairs would pose a navigational hazard to other robots or to workers in bulky safety gear. Such risk has to be minimized by thorough analysis and preparation beforehand.

Robots must be suitable for the environment. In at least two mine disasters, robots could not be used because they were not explosion proof – yet needed to be used in an explosive atmosphere. At Fukushima, robot operators had to be concerned with potential

damage of semiconductors by total dose of radiation, which could affect not only CPU and image sensors but also various components using semiconductors such as engine, battery, etc. Monitoring of the total dose using a dosimeter was necessary.

Robots must be decontaminated. Radiation contamination at Fukushima was a serious issue because it caused radiation exposure of workers in maintenance and battery exchange. However, UGVs at the World Trade Center collapse were exposed to raw sewage and body fluids. Decontamination has been discussed in standardization procedures but few platforms are built to be completely cleaned or to be easily cleaned without exposing the human.

60.6 Evaluation

Disaster robotics is still an emerging field and methods for evaluating a rescue robot or the larger mixed human–robot system are still forming. *Murphy* in [60.22] provides an analysis of the performance of disaster robots at 34 events. Evaluation is largely centered around the US government National Institute of Standards and Technology (NIST) efforts to standardize rescue robots for adoption by US responders via originally a standard test course and more recently a set of standard test methods. Each standard test method is an inexpensive, reproducible prop that tests a single capability; for example, Fig. 60.18 shows a prop designed to test a robot's ability to traverse rough terrain represented by a wooden *step field*. The standard test course and methods are duplicated via the USARsim computer simulation, and at least two sites, NERVE and SWRI, have physical testbeds consisting of the NIST standard test methods – though other physical testbeds such as Disaster City exist. The test course and methods have been used since the late 1990s for the RoboCup rescue and AAAI mobile robot competitions. This effort is being conducted through ASTM and thus, even though it is largely driven by the US, will likely result in an international standard that may be adopted by other countries.

60.6.1 Computer Simulations for Rescue Robotics

Computer simulations provide a low-cost mechanism to explore the larger behavior or a robot or system. Generally, computer simulations provide high fidelity for testing software execution, but their physical fidelity depends on the physics engine. Simulating sensors and the complex environments produced by a disaster is difficult and is rarely accurate enough to test perceptual algorithms. At the time of writing, two readily available computer simulations exist for exploring the strategic and tactical applications of rescue robots within the RoboCup rescue framework [60.89], the *RoboCup rescue simulation project* [60.90], and *USARSim* [60.91]. These simulations are well understood, accepted, open-source, and free; as such, they should be useful for most researchers or practitioners interested in ground-based rescue robotics.

The RoboCup rescue simulation project is used in the RoboCup rescue simulation league to study agent-based approaches to strategic planning for the disaster response. The simulator assumes a strong centralized response capability that is not necessarily the case for all countries or regions; the United States, for example, relies on a highly distributed organization that obviates many centralized coordination schemes. Although the simulation is focused on strategic decision making, particularly dynamic resource allocation, it does support the examination of how robot resources might be allocated during a disaster and how data from a robot might be propagated through a system. It permits the simulation of monitoring of disaster damage from reports by humans, distributed sensors, and robots and can simulate complex interactions such as telemedicine.

USARSim is a computer simulation developed by the University of Pittsburgh for physical robot simulation in disaster situations [60.91]. The simulation replicates the NIST standard test bed for search and rescue and permits efficient prototyping and testing of robot design and most aspects of control software. It uses Unreal game engine for handling physics and graphics, and virtual robots have capability of sensing (image, laser range finder, etc.) and actuation (wheel, motor, etc.) with data processing (image recognition, SLAM, etc.) in artificial environments. In 2006, the RoboCup rescue competition created a simulation league using this environment.

60.6.2 Physical Test Beds

Physical test beds provide a more realistic venue than a computer simulation for evaluating rescue robots, but may not be available to researchers, too expensive to use or to travel to, or not adequately capture some key aspect of a disaster. Physical test beds generally fall into three categories: *test beds developed for the fire rescue*

Fig.60.18a–c View of the NIST standard test bed for search and rescue used by RoboCup Rescue. **(a)** View of overall test bed, **(b)** dummy representing a victim, and **(c)** a step field challenging robot mobility (courtesy NIST)

community, *test beds developed for the robotics community*, and the *NIST standard test bed for search and rescue*.

Fire rescue training test beds occur throughout the world and are used to train human firefighters, rescue specialists, and canine teams under highly realistic conditions. The Texas A&M Engineering Extension Service's Disaster City complex has 52 ac devoted to representative structural collapses, ranging from multistory commercial building to wooden housing. Disaster City is the site for the NIST Response Robot Evaluation Exercises. The NASA Ames Research Center's Disaster Assistance Response Team facility at Moffett Field has hosted several events as well. In general, fire rescue test beds are constructed from construction and sewer debris, can introduce smoke and some simulants, and pose challenging mobility conditions, but vary in terms of fidelity. In many test beds, the density of the debris does not contain the actual amount of metals in a real collapse. This can lead to optimistic reports of success of sensors and wireless communication devices. The test beds, being designed for human training, do not replicate the conditions under which a ground robot would be used. The terrain is generally on the exterior of the rubble and does not exercise the robot in confined or vertical spaces. Depending on the size of the facility, the test bed may or may not be suitable for evaluating UAVs. An example of a fire-training test bed appeared earlier in Fig. 60.19.

Fig. 60.19 A CMU (Carnegie Mellon University) fixed-base snake being tested at a facility in California (courtesy H. Choset)

Physical testbeds for the robotics community such as the New England Robotics Validation and Experimentation (NERVE) Center at the University of Massachusetts Lowell, and the facilities at the Southwest Research Institute (SWRI) in San Antonio, Texas, incorporate both general test courses for robots and the NIST standard methods.

Perhaps the most influential physical simulation for researchers is the RoboCup rescue physical league which uses the NIST standard test bed for search and rescue, shown in Fig. 60.18. This competition started in 2001 [60.92] and has more than 40 team entries every year from all over the world. The RoboCup rescue physical league scores robot performance in terms of mobility, mapping, situation awareness, sensing, shared autonomy, etc. A robot or robot team competes in one

of three arenas which simulate disaster situations at the annual RoboCup world competition. The mission of the robot teams is to collect victim information (existence, state, and location) by sensor fusion of vital signals (heat, shape, color, motion, sound, CO_2, etc.) and report a map of victims in disaster space so that responders can efficiently arrive at the victim for rescue. In addition to the arenas, the competition and test bed contain individual skill test stations, for example, in order to test mobility, robots must traverse a random step field made of wood. The test bed was designed to be portable and reasonably inexpensive and several locations around the world have set up duplicates. As a result of the constraints of cost and portability, the test bed is not fully representative of actual disaster physical conditions and does not test the operating conditions for the human teams.

60.6.3 Standards Activity

Standards for rescue robots and systems are being generated at the time of writing. The E54.08 subcommittee on operational equipment within the E54 Homeland Security application committee of ASTM International started developing an urban search and rescue (USAR) robot performance standard with the National Institute of Standards and Technology (NIST) as a US Department of Homeland Security (DHS) program from 2005 to 2010. It plans to cover sensing, mobility, navigation, planning, integration, and operator control in order to ensure that the robots can meet operational requirements under the extreme conditions of rescue. The standards will consist of performance measures that encompass basic functionality, adequacy and appropriateness for the task, interoperability, efficiency, and sustainability. The components of the robot systems include platforms, sensors, operator interfaces, software, computational models and analyses, communication, and information. Development of requirements, guidelines, performance metrics, test methods, certification, reassessment, and training procedures is planned.

60.6.4 Evaluation

Evaluation of rescue robots is difficult not only because of the diversity of platforms and missions but also because each disaster is truly different. In addition, robots are part of a human-centric system: they are operated by humans in order to provide information to humans. Evaluation of the performance of a rescue robot system at an actual disaster is currently ad hoc. No computer or physical simulation for predicting the performance of robots and humans in a disaster has been validated; indeed, there is little argument that simulations are far easier than a real response. The difficulties of simulation are exacerbated by the differences between disasters. For example, the World Trade Center was unique in terms of the large amount of steel and the density of the collapsed material, while earthquakes and hurricanes are different from terrorist events.

Metrics for measuring performance remain a worthwhile quest. Quantitative metrics, such as the number of survivors or remains found, do not capture the value of a robot in establishing that there are no survivors in a particular area. Performance metrics from psychology and industrial engineering are only now beginning to be applied. These methods require enhanced computer and full-scale simulations in order to collect data. Data collection on human and overall system performance during a disaster has been done through ethnographic observations and are now moving to direct observations of situation awareness during demonstrations [60.80, 93]. Direct data collection during a disaster may not be possible as methods may interfere with performance (and therefore be unreasonable, if not unethical) and arouse fears by operators of *Big Brother* and being held liable for any errors in operation.

60.7 Conclusions and Further Reading

Rescue robots are making the transition from an interesting idea to an integral part of emergency response. Aerial and ground robots have captured most of the attention, especially for disaster response, but water-based vehicles (both surface and underwater) are proving useful as well. Rescue robots present challenges in all major subsystems (mobility, communications, control, sensors, and power) as well as in human–robot interaction. In terms of size, man-portable and man-packable systems are the most popular because of their reduced logistics burden, but the size of the platforms exacerbates the need for miniaturized sensors and processors. Wireless communications remains a major problem. While recent deployments have relied on polymorphic tracked vehicles, researchers are investigating miniature planes and helicopters along with new ground robots designs, particularly biomimetic. Research is also exploring alternative concepts of operations and user interfaces. Standards are currently under development and this

will help accelerate the adoption of rescue robots. The annual IEEE (Institute of Electrical and Electronics Engineers) International Symposium on Safety, Security and Rescue Robotics is currently the primary conference and clearinghouse for research in rescue robotics.

Video-References

VIDEO 140 Assistive mapping during teleoperation
available from http://handbookofrobotics.org/view-chapter/60/videodetails/140

References

60.1 A. Davids: Urban search and rescue robots: from tragedy to technology, IEEE Trans. Intell. Syst. **17**(2), 81–83 (2002)

60.2 A. Davids: Urban search and rescue robots: from tragedy to technology, IEEE Trans. Intell. Syst. **17**(2), 1541–1672 (2002)

60.3 D. McClean: *World Disasters Report 2010. Focus on Urban Risks* (IFCR, Geneva 2010)

60.4 National Fire Protection Association: *Standard on Operations and Training for Technical Rescue Incidents* (NFPA, Avon 1999)

60.5 United States Fire Administration: *Technical Rescue Program Development Manual* (USFA, Avon 1996)

60.6 R.R. Murphy, A. Kleiner: A community-driver roadmap for the adoption of safety security and rescue robot, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics, Linköping (2013) pp. 1–4

60.7 J.A. Barbera, C. DeAtley, A.G. Macintyre, D.H. Parks: Medical aspects of urban search and rescue, Fire Eng. **148**, 88–92 (1995)

60.8 R. Murphy, D. Riddle, E. Rasmussen: Robot-assisted medical reachback: A survey of how medical personnel expect to interact with rescue robots, Proc. IEEE Int. Work. Human Robot Interact. Commun. (HRI) (2004) pp. 301–306

60.9 R. Murphy, M. Konyo, P. Davalas, G. Knezek, S. Tadokoro, K. Sawata, M. Van Zomeren: Preliminary observation of HRI in robot-assisted medical response, Proc. 4th ACM/IEEE Int. Conf. Human Robot Interact. (HRI) (2009) pp. 201–202

60.10 R. Murphy, A. Rice, N. Rashidi, Z. Henkel, V. Srinivasan: A multi-disciplinary design process for affective robots: Case study of Survivor Buddy 2.0, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 701–706

60.11 A.C. Yoo, G.R. Gilbert, T.J. Broderick: Military robotics combat casualty extraction and care. In: *Surgical Robotics. Applications and Visions*, ed. by J. Rosen, B. Hannaford, R.M. Satava (Springer, New York 2011) pp. 13–32

60.12 M. Yim, J. Laucharoen: Towards Small Robot Aided Victim Manipulation, J. Intell. Robotic Syst. **64**, 119–139 (2011)

60.13 A. Kleiner: Mapping and Exploration for Search and Rescue with Humans and Mobile Robots (University of Freiburg, Freiburg 2008)

60.14 R.R. Murphy, S. Stover: Gaps analysis for rescue robots, Proc. ANS Shar. Sol. Emerg. Hazard. Environ. (2006)

60.15 C. Schlenoff, E. Messina: A robot ontology for urban search and rescue, ACM Work. Res. Knowl. Represent. Auton. Syst., New York (2005) pp. 27–34

60.16 R. Murphy, J. Casper, J. Hyams, M. Micire, B. Minten: Mobility and sensing demands in USAR, Proc. IECON Sess. Rescue Eng., Vol. 1 (2000) pp. 138–142

60.17 M. Angermann, M. Frassl, M. Lichtenstern: Mission review of aerial robotic assessment – ammunition explosion Cyprus 2011, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2012) pp. 1–6

60.18 S. Kawatsuma, M. Fukushima, T. Okada: Emergency response to Fukushima-Daiichi accident: summary and lessons learned, Ind. Robot **39**(5), 428–435 (2012)

60.19 S. Tadokoro, T. Takamori, S. Tsurutani, K. Osuka: On robotic rescue facilities for disastrous earthquakes – from the great Hanshin-Awaji (Kobe) earthquake, J. Robotics Mechatron. **9**(1), 46–56 (1997)

60.20 R. Murphy: Human-robot interaction in rescue robotics, IEEE Trans. Syst. Man Cybern. **34**(2), 138–153 (2004)

60.21 M.J. Micire: Evolution and field performance of a rescue robot, J. Field Robotics **25**(1-2), 17–30 (2008)

60.22 R.R. Murphy: *Disaster Robotics* (MIT Press, Cambridge 2014)

60.23 K. Pratt, R. Murphy, S. Stover, C. Griffin: Conops and autonomy recommendations for vtol small unmanned aerial systems based on Hurricane Katrina operations, J. Field Robotics **26**(8), 636–650 (2009)

60.24 R. Murphy, E. Steimle, E. Griffin, C. Cullins, M. Hall, K. Pratt: Cooperative use of unmanned sea surface and micro aerial vehicle at hurricane Wilma, J. Field Robotics **25**(3), 164–180 (2008)

60.25 V.G. Ambrosia, S. Wegener, T. Zajkowski, D.V. Sullivan, S. Buechel, F. Enomoto, B. Lobitz, S. Johan, J. Brass, E. Hinkley: The Ikhana unmanned airborne system (UAS) western states fire imaging missions: From concept to reality (2006–2010), Geocarto Int. **26**(2), 85–101 (2011)

60.26 R.R. Murphy, E. Steimle, M. Hall, M. Lindemuth, D. Trejo, S. Hurlebas, Z. Medina-Cetina, D. Slocum:

Robot-assisted bridge inspection, J. Intell. Robotic Syst. **64**(1), 77–95 (2011)
60.27 P. Srivaree-Ratana: Lessons learned from the great Thailand flood 2011: How a UAV helped scientists with emergency response and disaster aversion, Proc. AUVSI Unmanned Syst. North Am. (2012)
60.28 R. Murphy, S. Stover, H. Choset: Lessons learned on the uses of unmanned vehicles from the 2004 Florida hurricane season, AUVSI Unmanned Syst. North Am. (2005)
60.29 J. Lester, A. Brown, J. Ingham: Christchurch cathedral of the blessed sacrament: Lessons learnt on the Stabilization of a significant heritage building, Proc. Annu. Conf. N. Z. Soc. for Earthq. Eng. (NZSEE) (2012) pp. 1–11
60.30 N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, K. Ohno, E. Takeuchi, S. Tadokoro: Collaborative mapping of an earthquake-damaged building via ground and aerial robots, J. Field Robotics **29**(5), 832–841 (2012)
60.31 R.R. Murphy, K.L. Dreger, S. Newsome, J. Rodocker, B. Slaughter, R. Smith, E. Steimle, T. Kimura, K. Makabe, F. Matsuno, S. Tadokoro, K. Kon: Marine heterogeneous multi-robot systems at the great eastern japan tsunami recovery, J. Field Robotics **29**(5), 819–831 (2012)
60.32 F. Ferreira: ICRA Japan Forum: Preliminary report on the disaster and robotics in Japan, IEEE Robotics Autom. Mag. **18**(3), 116 (2011)
60.33 M. Shibuya: Using micro-rov's in the aftermath of japan's tsunami. Underwater Intervention (2012)
60.34 G.-J. Kruijff, V. Tretyakov, T. Linder, F. Pirri, M. Gianni, P. Papadakis, M. Pizzoli, A. Sinha, E. Pianese, S. Corrao, F. Priori, S. Febrini, S. Angeletti: Rescue robots at earthquake-hit Mirandola, Italy: A field report, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2012) pp. 1–8
60.35 R. Murphy, S. Stover: Rescue robots for mudslides: A descriptive study of the 2005 La Conchita mudslide response, J. Field Robotics **25**(1–2), 3–16 (2008)
60.36 R.R. Murphy: Trial by fire, IEEE Robotics Autom. Mag. **11**(3), 50–61 (2004)
60.37 L. Goldwert: Minneapolis honors bridge collapse victims, http://www.cbsnews.com/news/minneapolis-honors-bridge-collapse-victims/ (2007)
60.38 FBI: Photo gallery FBI response to minneapolis bridge collapse, http://www2.fbi.gov/page2/august07/bridge1.htm (2007)
60.39 K. Pratt, R. Murphy, J. Burke, J. Craighead, C. Griffin, S. Stover: Use of tethered small unmanned aerial system at Berkman Plaza II collapse, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2009) pp. 134–139
60.40 S. Tadokoro, R. Murphy, S. Stover, W. Brack, M. Konyo, T. Nishimura, O. Tanimoto: Application of active scope camera to forensic investigation of construction accident, Proc. IEEE Int. Work. Adv. Robotics Its Soc. Impacts (ARSO) (2009) pp. 47–50
60.41 K. Ohno, S. Kawatsuma, T. Okada, E. Takeuchi, K. Higashi, S. Tadokoro: Robotic control vehicle for measuring radiation in Fukushima Daiichi nuclear power plant, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2011) pp. 38–43
60.42 ROV World: Raising the Costa Concordia live – ROV-world subsea information (2012)
60.43 Hydro International: ROV survey of the Costa Concordia grounding site, http://www.hydro-international.com/news/id5324-ROV_Survey_of_the_Costa_Concordia_Grounding_Site_video.html (2012)
60.44 Draganfly: RCMP corporal Doug Green interviewed on CKOM John Gormley Live – Draganflyer X4-ES used in life-saving mission, http://www.draganfly.com/news/2013/05/13/rcmp-corporal-doug-green-interviewed-on-ckom-john-gormley-live-draganflyer-x4-es-used-in-life-saving-mission/ (2013)
60.45 T. Linder, V. Tretyakov, S. Blumenthal, P. Molitor, D. Holz, R. Murphy, S. Tadokoro, H. Surmann: Rescue robots at the collapse of the municipal archive of Cologne city: A field report, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2010) pp. 1–6
60.46 R.R. Murphy, R. Shoureshi: *Emerging Mining Communication and Mine Rescue Technologies* (Mine Safety and Health Administration, Arlington 2008)
60.47 R.R. Murphy, J. Kravitz, S. Stover, R. Shoureshi: Mobile robots in mine rescue and recovery, IEEE Robotics Autom. Mag. **16**(2), 91–103 (2009)
60.48 K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, M. Fukushima, S. Kawatsuma: Emergency response to the nuclear accident at the Fukushima Daiichi nuclear power plants using mobile rescue robots, J. Field Robotics **30**(1), 44–63 (2013)
60.49 G.M. Kulali, M. Gevher, A.M. Erkmen, I. Erkmen: Intelligent gait synthesizer for serpentine robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 1513–1518
60.50 A. Wolf, H.B. Brown, R. Casciola, A. Costa, M. Schwerin, E. Shamas, H. Choset: A Mobile hyper redundant mechanism for search and rescue tasks, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Vol. 3 (2003) pp. 2889–2895
60.51 D. Campbell, M. Buehler: Stair descent in the simple hexapod "RHex", Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 1380–1385
60.52 R.M. Voyles, A.C. Larson: Terminatorbot: A novel robot with dual-use mechanism for locomotion and manipulation, IEEE/ASME Trans. Mechatron. **10**(1), 17–25 (2005)
60.53 J. Tanaka, K. Suzumori, M. Takata, T. Kanda, M. Mori: A mobile jack robot for rescue operation, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2005) pp. 99–104
60.54 R. Murphy, T. Vestgaarden, H. Huang, S. Saigal: Smart lift/shore agents for adaptive shoring of collapse structures: A feasibility study, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2006)

60.55 W.E. Green, P.Y. Oh: A fixed-wing aircraft for hovering in caves, tunnels, and buildings, Proc. Am. Control Conf. (2006) pp. 1–6

60.56 V. Kumar, D. Rus, S. Singh: Robot and sensor networks for first responders, IEEE Pervasive Comput. **3**(4), 24–33 (2004)

60.57 D. Kurabayashi, H. Tsuchiya, I. Fujiwara, H. Asama, K. Kawabata: Motion algorithm for autonomous rescue agents based on information assistance system, Proc. IEEE Int. Symp. Comput. Intell. Robotics Autom. (2003) pp. 1132–1137

60.58 K.W. Sevcik, W.E. Green, P.Y. Oh: Exploring search-and-rescue in near-earth environments for aerial robots, Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (2005) pp. 693–698

60.59 A. Birk, S. Carpin: Rescue robotics: a crucial milestone on the road to autonomous systems, Adv. Robotics **20**(5), 596–605 (2006)

60.60 A. Kleiner, C. Dornhege, R. Kuemmerle, M. Ruhnke, B. Steder, B. Nebel, P. Doherty, M. Wzorek, P. Rudol, G. Conte, S. Durante, D. Lundstrom: RoboCupRescue – Robot league team RescueRobots Freiburg (Germany), Proc. 10th RoboCup 2006 (2006)

60.61 J. Pellenz, D. Paulus: Stable mapping using a hyper particle filter, Proc. 13th RoboCup 2006 (2010) pp. 252–263

60.62 A. Nuchter, K. Lingemann, J. Hertzberg: Mapping of rescue environments with kurt3d, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2005) pp. 158–163

60.63 A. Kleiner, C. Dornhege: Mapping for the support of first responders in critical domains, J. Intell. Robotic Syst. **64**(1), 7–31 (2011)

60.64 B. Yamauchi: A frontier-based approach for autonomous exploration, Proc. IEEE Int. Symp. Comput. Intell. Robotics Autom. (1997) pp. 146–151

60.65 C. Dornhege, A. Kleiner: A frontier-void-based approach for autonomous exploration in 3d, Adv. Robotics **27**(6), 459–468 (2013)

60.66 A. Jacoff, E. Messina, B. Weiss, S. Tadokoro, Y. Nakagawa: Test arenas and performance metrics for urban search and rescue robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2003) pp. 3396–3403

60.67 S. Wirth, J. Pellenz: Exploration transform: A stable exploring algorithm for robots in rescue environments, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2007) pp. 1–5

60.68 Y. Okada, K. Nagatani, K. Yoshida, S. Tadokoro, T. Yoshida, E. Koyanagi: Shared autonomy system for tracked vehicles on rough terrain based on continuous three-dimensional terrain scanning, J. Field Robotics **28**(6), 875–893 (2011)

60.69 E. Magid, T. Tsubouchi, E. Koyanagi, T. Yoshida, S. Tadokoro: Controlled balance losing in random step environment for path planning of a teleoperated crawler-type vehicle, J. Field Robotics **28**(6), 932–949 (2011)

60.70 R. Sheh, B. Hengst, C. Sammut: Behavioural cloning for driving robots over rough terrain, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2011) pp. 732–737

60.71 C. Dornhege, A. Kleiner: Behavior maps for online planning of obstacle negotiation and climbing on rough terrain, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2007) pp. 3005–3011

60.72 M. Andriluka, P. Schnitzspan, J. Meyer, S. Kohlbrecher, K. Petersen, O. Von Stryk, S. Roth, B. Schiele: Vision based victim detection from unmanned aerial vehicles, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2010) pp. 1740–1747

60.73 A. Kleiner, R. Kummerle: Genetic mrf model optimization for real-time victim detection in search and rescue, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2007) pp. 3025–3030

60.74 R. Hahn, D. Lang, M. Haselich, D. Paulus: Heat mapping for improved victim detection, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2011) pp. 116–121

60.75 A. Birk, S. Markov, I. Delchev, K. Pathak: Autonomous rescue operations on the iub rugbot, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2006)

60.76 R.R. Murphy, J. Burke: The safe human-robot ratio, Human-Robot Interactions in Future Military Operations, ed. by M. Barnes, F. Deutsch (Ashgate, Farnham 2010) pp. 31–49

60.77 M. Endsley: Design and evaluation for situation awareness enhancement, Proc. 32nd Annual Meet. Hum. Factors Soc. (1988) pp. 97–101

60.78 J.L. Drury, J. Scholtz, H.A. Yanco: Awareness in human-robot interactions, Proc. IEEE Int. Conf. Syst. Man Cybern. (2003) pp. 912–918

60.79 J. Casper, R. Murphy: Workflow study on human-robot interaction in usar, Proc. IEEE Int. Conf. Robotics Autom. (2002) pp. 1997–2003

60.80 J. Burke, R. Murphy, M. Coovert, D. Riddle: Moonlight in miami: An ethnographic study of human-robot interaction in usar, Hum.-Comput. Interact. **19**(1/2), 85–116 (2004)

60.81 R. Murphy, J. Burke, S. Stover: Field Studies of Safety Security Rescue Technologies Through Training and Response Activities, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Vol. 2 (2004) pp. 1089–1095

60.82 N. Sato, F. Matsuno, T. Yamasaki, T. Kamegawa, N. Shiroma, H. Igarashi: Cooperative Task Execution by a Multiple Robot Team and its Operators in Search and Rescue Operations, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Vol. 2 (2004) pp. 1083–1088

60.83 T. Arai, E. Pagello, L. Parker: Editorial: Advances in multi-robot systems, IEEE Trans. Robotics Autom. **18**(5), 655–661 (2002)

60.84 D.P. Stormont, A. Bhatt, B. Boldt, S. Skousen, M.D. Berkemeier: Building better swarms through competition: Lessons learned from the AAAI/Robocup rescue robot competition, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2003) pp. 2870–2875

60.85 J. Suarez, R. Murphy: A survey of animal foraging for directed, persistent search by rescue robotics, Proc. IEEE Int. Symp. Saf. Secur. Rescue Robotics (2011) pp. 314–320

60.86 A. Ferworn, A. Sadeghian, K. Barnum, H. Rahnama, H. Pham, C. Erickson, D. Ostrom, L. Dell'Agnese: Urban search and rescue with canine augmentation

technology, Proc. IEEE/SMC Int. Conf. System Syst. Eng. (2006) pp. 334–338
60.87 L. Yihan, S.S. Panwar, S. Burugupalli: A mobile sensor network using autonomously controlled animals, Proc. 1st Int. Conf. Broadband Netw. (BROADNETS) (2004) pp. 742–744
60.88 R. Murphy: Rats, robots, and rescue, IEEE Intell. Syst. **17**(5), 7–9 (2002)
60.89 S. Tadokoro, H. Kitano, T. Takahashi, I. Noda, H. Matsubara, A. Hinjoh, T. Koto, I. Takeuchi, H. Takahashi, F. Matsuno, M. Hatayama, J. Nobe, S. Shimada: The robocup-rescue project: A robotic approach to the disaster mitigation problem, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 4089–4094
60.90 T. Takahashi, S. Tadokoro: Working with robots in disasters, IEEE Robotics Autom. Mag. **9**(3), 34–39 (2002)
60.91 I.R. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, S. Burion: Human-robot teaming for search and rescue, IEEE Pervasive Comput. **4**(1), 72–79 (2005)
60.92 H. Kitano, S. Tadokoro: Robocup-rescue: A grand challenge for multi-agent and intelligent systems, AI Magazine **22**(1), 39–52 (2001)
60.93 J. Scholtz, B. Antonishek, J. Young: A field study of two techniques for situation awareness for robot navigation in urban search and rescue, Proc. IEEE Int. Work. Robot-Hum. Interact. Commun. (2005) pp. 131–136

Multimedia Contents

61. Robot Surveillance and Security

Wendell H. Chun, Nikolaos Papanikolopoulos

This chapter introduces the foundation for surveillance and security robots for multiple military and civilian applications. The key environmental domains are mobile robots for ground, aerial, surface water, and underwater applications. Surveillance literally means *to watch from above*, while surveillance robots are used to monitor the behavior, activities, and other changing information that are gathered for the general purpose of managing, directing, or protecting one's assets or position. In a practical sense, the term surveillance is taken to mean *the act of observation from a distance*, and security robots are commonly used to protect and safeguard a location, some valuable assets, or personal against danger, damage, loss, and crime. Surveillance is a proactive operation, while security robots are a defensive operation. The construction of each type of robot is similar in nature with a mobility component, sensor payload, communication system, and an operator control station.

After introducing the major robot components, this chapter focuses on the various applications. More specifically, Sect. 61.3 discusses the enabling technologies of mobile robot navigation, various payload sensors used for surveillance or security applications, target detection and tracking algorithms, and the operator's robot control console for human–machine interface (HMI). Section 61.4 presents selected research activities relevant to surveillance and security, including automatic data processing of the payload sensors, automatic monitoring of human activities, facial recognition, and collaborative automatic target recognition (ATR). Finally, Sect. 61.5 discusses future directions in robot surveillance and security, giving some conclusions and followed by references.

61.1 **Overview** 1605
61.2 **Application Domains** 1607
61.3 **Enabling Technologies** 1608
61.3.1 Mobile Navigation 1608
61.3.2 Payload Sensors 1609
61.3.3 Detection and Tracking Algorithms 1611
61.3.4 Human–Machine Interface 1614
61.4 **Active Research** 1617
61.4.1 Solving the Data Processing Problem 1618
61.4.2 Detecting Human Activity in Support of Surveillance 1618
61.4.3 Facial Recognition for Security Identification 1619
61.4.4 Collaborative ATR 1621
61.5 **Conclusion** 1622
Video-References 1623
References 1623

61.1 Overview

Surveillance robots and Security robots are tactically similar in design, but their specific purposes are very different. The surveillance robot uses its payload to map or search an area, while the security robot is used to protect and safeguard an asset such as the perimeter of an airport. In comparing these two categories of robots, the difference in operations would be analogous to comparing surveillance to a global problem and security to a local problem. Operationally, the surveillance robot collects data with its electronic payload

Fig.61.1a–d Examples of surveillance and security robots. **(a)** Aerial-General Atomics Predator, **(b)** ground-QinetiQ TALON, **(c)** ground-General Dynamics MDARS-E, and **(d)** surface water-Elbit Stingray

(also known as the robot's external sensor) and relays the raw data back to its control station or preprocesses the data with other information such as geographic information system (GIS) data and relays that information to an exploitation element. Eventually as more autonomy is integrated into the vehicle, the robot will be able to plan its own paths and point its sensor with minimal human intervention. The quality of surveillance is a function of target range, the amount of target clutter, sensor resolution, and its probability of detection of a target. Security robot operations consist of a mobile robot patrolling a defined area (or perimeter), detecting an intruder or suspicious activities, and initiating a response. A security robot can conduct predefined or random patrols, and automatically perform surveillance tasks such as checking for intruders or assessing the status of the perimeter defenses. Input from the operator is required only if an intruder is detected or the robot encounters a situation it is not programmed to handle. Once contact is made, the operator is able to see, hear, and talk to the intruder. If the intruder tries to avoid the robot, the intruder can track them with its onboard sensor and maneuver the robot to follow the suspected intruder. Search and Rescue Robotics (Chap. 58) uses some aspects of surveillance due to the potentially wide area to be searched and the actions of security where upon detection, an action is required.

The basic security robot [61.1] can be as simple as a fixed-sensor, mounted up high and able to pan and tilt without being able to move around. There are concepts to use a low-cost sensor (to sense motion, detect vibration, or break a sensor-beam) in order to cue a more sophisticated second sensor (e.g., radar or video) for confirmation. Depending on the complexity of the security system, the fixed sensor can alert a mobile robot, which can otherwise be defeated by just waiting for the robot to pass. A combination of fixed and mobile assets can be a very robust solution system. Again, the nominal robot configurations are still similar (Fig. 61.1): robot with sensor payload, an operator station, and a communication system that links the robot with its control station [61.2–5]. In some cases, there may be two distinct control stations, one for controlling the robot and one for handling the payload data as seen on military drones. A surveillance robot can be more data intensive as compared to the security robot which often performs in a structured environment and has a narrower data load to process. Central to both applications are the robot or robots, and the control station. Robot surveillance is in greater use today due to the war in the Middle East (Iraq and Afghanistan) with the proliferation of military drones. The majority of unmanned aerial vehicles (UAVs) are controlled via teleoperation or by flying automatically through waypoints. UAVs are now commonplace in the military, and slowly making inroads into the commercial sector with applications in agriculture, forest fire mapping, and weather analysis. The next technological step forward for surveillance robots is autonomous operations and multi-vehicle coordination (i. e., swarming). Robot security is less developed with some on-going research programs within the military, and a few commercial options available on the market today. Early mobile robot companies in the 1980s were formed to address these same security concerns such as Cybermotion Inc. and Denning Mobile Robotics, Inc. but did not survive over time. The next logical progression for robot security is to bring the costs down to make the systems more viable, reduce false-positives to make detection reliable, and take unnecessary personnel out of daily control through the careful insertion of automatic and autonomous technologies.

There are some fundamental issues that need to be addressed in the design and development of the surveillance or security robot system. For example, each specific applications will dictate if surveillance occurs indoors or outdoors, and similarly for indoor or outdoor security, such that sensors are different to address weather or lighting concerns. There are numerous applications for either case, and will impact the type of mobility chosen, and the type of sensors used for monitoring and detection. As noted earlier, robot surveillance typically requires mobility; while security robots can be either stationary or mobile. In the later, the environment is known (or classified as being struc-

tured) with set vehicle trajectories defined a priori in the majority of applications. Some robot security applications may involve a combination of both fixed and mobile robots depending on the size of the facility or building. The first issue to contend with is having to select between teleoperated and autonomous control of the robot. There is a scenario to start with a teleoperated system that can eventually evolve toward a more capable automatic or autonomous capability. However, experience has shown that this upgrade seldom occurs naturally without planning and when the capability does mature, it will require large software block upgrades and the hardware will normally require upgrading also to increase overall capability.

61.2 Application Domains

Surveillance and security robot designs begin by first understanding its intended operational environment, and simultaneously designing for data collection or various protection schemas. The typical surveillance robot requires a surveillance payload, while a security robot integrates a security sensor. This sensor suite is in addition to a navigation sensor suite required by the robot for mobility. There may be an opportunity to use the same sensor for both navigation and payload, but this option is very isolated due to disparate range and resolution requirements. Since there are two different sensors, some coordination is required between mobility of the robot and the desired viewing angle of the payload sensor.

Surveillance and security robots (Fig. 61.2) can be found in all the normal environments (e.g., aerial, ground, surface water, underwater, or space), for example by watching from above from either space or from an aerial elevation. Robots patrols are found in ground [61.6, 7] and aerial applications:

- *Aerial Domain*: The majority of long distance aerial applications utilize fixed wing vehicles flying at a set altitude and monitoring large surface areas. Aerial vehicles come in all sizes from full-size platforms that are the size of passenger jetliners to small quad-rotors, vertical take-off and landing vehicles, and microflapping wing aircraft as described in Chap. 50. Electro-optical (EO), infrared (IR), or synthetic aperture radar (SAR) are sensors nominally used to collect surveillance data. A fixed-wing aircraft can also be used for security, but rotorcraft platforms (especially a quad-rotor) are the mobility of choice for its hovering capability as shown in VIDEO 554. The weakness in rotorcraft platforms is its weight limitations, e.g., sensor payload weight and vehicle flight time. Key aerial design constraints are Reynolds number, Lift-to-drag (L/D) ratio of the wings, Breguet range analysis, vehicle endurance, and vehicle aspect-ratios. Communications between aircraft and ground control station is direct line-of-sight or over-the-horizon using satellites or other UAVs when functioning as a communication repeater.
- *Ground Domain*: Ground vehicles are either custom-designed or based on retro-fitting an existing commercial or military vehicle. They come in all sizes from automobile-size to desk-size and shoebox-size such as the Dragon Runner as described in Chap. 48 and potentially walking as described in Chap. 65. An EO, IR, or range finder is used to collect surveillance data. For robot security, a similar EO, IR, lidar, or range finder sensors are often used as can be seen in VIDEO 677. Key ground design constraints are types of vehicle steering modes, type of propulsion (wheeled, track, legged, or hybrid) system, range, suspension, and draw-bar pull performance. The standard packet-radio used for communications is bandwidth limited, thus ground robots would benefit from and automatic RSTA capability (RSTA: reconnaissance, surveillance, and target acquisition).
- *Surface Water Domain*: Surface vehicles are either custom-designed, or based on retro-fitting an existing boat such as a rigid hull inflatable boat (RHIB). They vary in sizes from a small jet ski to an 11 m RHIB. Surface water robots used for surveillance and security are in its infancy. Potentially, a radar can be used to conduct surveillance, data and for robot security, a combination of radar and EO sensors are envisioned. Key surface water design constraints are vehicle speed, range, and draft and beam dimensions for flotation. Surface navigation requires GPS for localization when visual techniques are ineffective and when water features are indistinguishable.
- *Underwater Domain*: Unmanned underwater vehicles (UUVs) vary in size from man-portable lightweight platforms to large diameter vehicles of over 10 m in length. The larger vehicles have an advantage in terms of endurance and sensor payload weight capacity as described in Chap. 51. Sonar is the payload sensor (sidescan or other) and is used to collect surveillance and support navigation data.

Fig.61.2a–e Wide ranging operational environments: **(a)** Air-Tacocopter, **(b)** Ground-Asian Forum for Corrections' Robo-Guard, **(c)** Underwater-Bluefin 9, **(d)** Surface water-Rafael Protector, and **(e)** Space-Ball Aerospace SBSS Satellite (SBSS: space based space surveillance)

The majority of operational UUVs are tethered, but untethered vehicles have a larger presence in the research world (better known as autonomous underwater vehicles or AUVs). Key underwater design constraints are Reynolds Number, buoyancy, endurance, pressure, and being tethered or untethered. Secure underwater communications is an issue for connecting the robot with its operators.

- *Space Domain*: Spacecraft by nature are considered robotic. There is a combination of onboard automation and a mix of commanding from the ground during critical events. Surveillance spacecraft range in size from microsatellites (50–100 kg) to traditional (approximately 15 000 kg) surveillance-size spacecraft, and the typical payload sensors include high-definition cameras and SAR. Security satellites are used to protect assets on the ground, and having the ability to detect and track objects from space is crucial to preventing threats, whether real or accidental. Key spacecraft design constraints are weight, power usage, thermal cycling, surviving launch loads, and operating at cryogenic temperatures.

As shown, there is a wide range of surveillance and security robots in multiple domains. Domains have specific individual environmental concerns. A domain not discussed is underground, as experienced with mining applications or the surveillance of caves with the iRobot 110 FirstLook or other small unmanned ground vehicle. It is clear with all of these robots that there is a navigational component and a payload/sensor component. Coordination between navigating and sensing for surveillance and security is of interest to robotic researchers. As noted earlier, the dominant robot applications are aerial surveillance, which covers large surface areas by a fast moving aircraft, and ground (and surface water) vehicles for security that move slower on predefined paths.

61.3 Enabling Technologies

Robot surveillance and Security requires sensing, data collection, mobility (optional), navigation, communication, control, computer vision, and having a preplanned response (i. e., using deterrence) as a security action. Acceptable responses can be as benign as alerting a human guard, to immobilizing the threat, or potentially incorporating a weapon to remove the threat. There are many relevant technologies needed for this application. In this section, we will discuss the selected enabling technologies of: mobile navigation, payload sensors, applicable algorithms, and the human–robot interface at the operator control console.

61.3.1 Mobile Navigation

Teleoperated mobility is a well-developed technology as long as the communication bandwidth is available between the assets. For fast moving surveillance robots such as large aerial platforms [61.8], teleoperation is an option but today's technology is to fly via GPS waypoints. Waypoints are sets of coordinates that identify a point in physical space such as X, Y, Z, or longitude, latitude, and altitude. Navigation systems such as GPS (or other radio triangulation systems, i. e., Loran or Kaman) are used to maneuver the aircraft to fly-over or fly-by the waypoints. Fly-by waypoints are used when an aircraft should begin a turn to its next course prior to reaching the waypoint separating the two route segments, and fly-over waypoints are used when the aircraft must fly over the point prior and start its turn. In practice, the aerial robot will fly directly overhead of a fly-over waypoint, and can miss a fly-by waypoint by a couple of miles due to a practice called *cutting the corner*. The critical phases for all UAVs are during its launch and at its recovery, and would benefit from emerging sense-and-avoid technologies. For underwater applications where GPS is not accessible unless the platform surfaces periodically, the robot [61.9] navi-

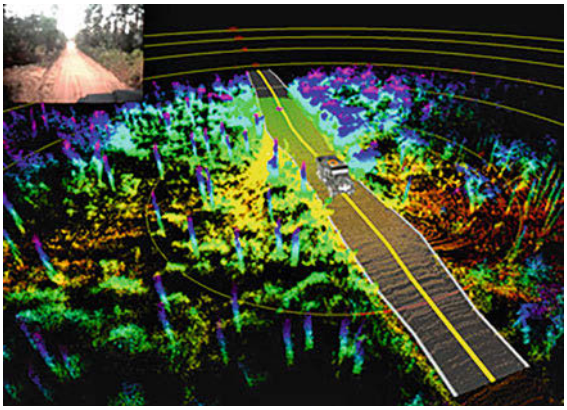


Fig. 61.3 Autonomous navigation in outdoor terrain (courtesy of Carnegie Mellon University National Robotic Engineering Center)

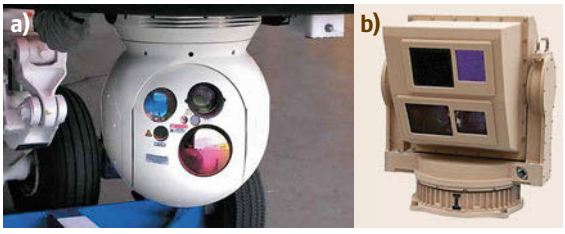


Fig.61.4a,b Typical EO sensors: **(a)** UAV EO gimbal sensor suite (courtesy of forward looking infrared (FLIR) systems) and **(b)** (UGV) EO/IR gimbal sensor suite (courtesy of L3 – Wescam)

gates through the use of dead reckoning with frequent landmark corrections.

For unknown and unstructured environments, teleoperation for security robots is the state of practice for aerial, ground, and surface water applications. When there is a setup period and the path is known a priori, as within a *structured* environment, preplanned paths can be incorporated. For a more advanced ground robot, autonomous navigation [61.10] is being researched and developed (predominantly in the outdoor environment). Benchmarks for autonomous navigation (Fig. 61.3) in complex, outdoor environments were established on the DARPA PerceptOR and LAGR programs (LAGR: learning applied to ground robots; PerceptOR: perceptions for off-road robotics). Refer to Chap. 35 for techniques that contribute to an autonomous navigation capability.

61.3.2 Payload Sensors

The end-user for surveillance data are the commanders, decision makers, analysts, targeteers, weaponeers (when a response is separate from the UAV), and cartographers. The information sought is timeliness, synoptic coverage, sensor resolution, sensor accuracy, interior details, and the differing elevations. Some basic payload definitions are:

- Real-time: information arrives after the event at the speed of electrons.
- Near Real-time: information arrives between real time and 20 min after the event.
- Accuracy: level of agreement with standard map datum.
- Precision: fineness of resolution to which measurements can be made.
- Field of View: angular area observed by a fixed (nonscanning) sensor.
- Field of Regard: angular area observed by a scanning sensor.

Surveillance sensor options include passive imaging [61.11], active imaging, and nonimaging sensors. Film and EO sensors are passive imaging. An example of passive imaging on the DARPA UGV Demo II program is shown in VIDEO 679 In addition to EO, IR optical sensors often accompany its daylight counterpart in order to be also able to see in darkness (commonly denoted as EO/IR). Active imaging sensors include radar and laser, while nonimaging sensors are SIGINT, MASINT, and chemical/biological. Passive imaging sensors are compared in Table 61.1.

There are many active imaging sensors (Fig. 61.4) such as SAR, a radar that uses its forward motion to synthesize the equivalent of a large side-looking antenna to produce high-resolution ground mapping. Moving Target Indicator (MTI) is a radar mode that uses differences in Doppler frequency due to the target's motion to distinguish it from stationary backgrounds (i. e., ground clutter). MTI has a wide field of view. Foliage penetration is radar using VHF/UHF frequencies and complex algorithms optimized to detect targets through foliage (VHF: very high frequency; UHF: ultra high frequency). Trees and leaves cover a good portion of the world's terrain. Inverse SAR is a radar mode using a target's angular movement (instead of using the platforms as for SAR) to detect moving objects, and Interferometric SAR is a radar technique using widely separated receivers to receive the same signal and combining them to produce three-dimensional (3-D) maps with highly accurate terrain relief. High resolution radar is radar using short wavelengths (millimeter wave) and sophisticated algorithms to produce high resolution (i. e., measured in inches) imagery. Laser imaging radar is a radar using light (laser), rather than radio frequency (RF) energy, for its illumination source. Robot researchers recognize LIDAR to refer to the acronym *Light Detection and Ranging*, and typically used in

Table 61.1 Passive imaging sensors

Sensor	Advantages	Disadvantages
Film	Best resolution Easily archived	Processing time compromises availability Dissemination copies degrade with reproduction
Electro-optical (visible)	Easy to data link while in-flight High resolution Easy to interpret (literal presentation) High fidelity reproduction	Cannot penetrate inclement weather Volume of data produced (current big data problem)
Infrared (cooled)	Provides imagery during the dark Penetrates some weather or obscurants Detects targets otherwise invisible due to range or background Provides evidence of recently past activity (IR traces), current activity level (engines running), or status (fuel tank levels)	Effectiveness diminished in warmer seasons, desert locales, hotter hours of the day, in Sun's direction Resolution typically 1 NIIRs poorer than that of visible
Infrared (uncooled)	Same operational advantages of cooled IR sensors, but: More compact packaging Reduced power requirements Increased reliability Lower costs	Less sensitive to thermal differences Larger pixel size = less resolution
Spectral	Provides additional operational information about targets–counter CC&D (camouflage, concealment, and deception) techniques	Bandwidth intensive for data relay

Table 61.2 Active imaging sensors

Sensor	Advantages	Disadvantages
Synthetic aperture radar (SAR)	Sees at greater distances than visible Sees through (most) weather and night Provides constant perspective imagery	Power and processing intensive Large aperture
Moving target indicator (MTI)	Detects and tracks moving air/ground objects Builds history of traffic flow over set intervals	A SAR-dependent technique requires further processing of SAR returns
Foliage penetration (FOPEN)	Defeats most uses of foliage as camouflage Provides limited ground penetration capability	Resolution limited due to frequencies used
Inverse SAR (ISAR)	Detects small objects on moving backgrounds Examples: boats, periscopes, and icebergs	Performance varies with aspect angle
Interferometric SAR (IFSAR)	Achieves locational accuracies of 5–30 m Maps large amounts of terrain in a short time	Requires relative positions of receivers be known and maintained precisely Requires extensive post-processing of data
High resolution radar (HRR)	Range limited by frequencies used	Susceptible to intervening weather
Laser imaging radar (LIDAR)	Useful for bathymetry, pollution monitoring, and altimetry Potential for 2-D range-gated and 3-D imaging	Limited in range and by intervening weather

robot navigation. Active imaging sensors are compared in Table 61.2.

Examples of Nonimaging sensors are SIGINT, MASINT, and chemical/biological. SIGINT stands for Signals Intelligence. Electronic signals are passive and the emissions of electronic systems (from radars, navigational aids, and others) can be intercepted and interpreted. The challenge with SIGNIT is frequency hopping and spread spectrum. Another facet of SIGNIT is the interception of spoken communications signals (through radios, etc.). The common challenges with communication signals are digitization, encoding, enciphering, and/or frequency hopping. Measurement and Signatures Intelligence (MASINT) can be either active or passive. MASINT adds additional dimensions to target data, but requires the intelligent interpretation of this data. Finally, chemical/biological agent sensors can also be passive or active. There are two types of chemical/biological (Chem/bio) sensors: point and remote, and these types of sensors perform based on air sampling and can detect when previous chemical or biological weapons have been used. Thus, Robots have many options for payloads, dominated by weight, power consumption, and field of view. EO/IR are the most commonly used sensors for surveillance. Its selection is based on resolution, optical efficiency, and detector responsiveness. Its resolution is a function of its detector array size and the field-of-view of the lens

used. Figure 61.5 is used to determine the desired resolution. Optical efficiency is dependent on lens diameter, lens focal length, and pitch. Finally, detector responsiveness is based on the detector's material, wavelength, and temperature. These are all physical parameters. A figure of merit used for a better payload is the product of its normalized resolution with a normalized optical efficiency with normalized detector responsiveness coefficient.

In summary, both EO and SAR sensors are approaching their limits of *immutable laws of physics*. The challenge today is in data reduction and its exploitation in support of surveillance. As a rule of thumb, surveillance sensors are selected based on:

- VHF/UHF SAR is used for foliage penetration and large area coverage.
- Microwave SAR for high resolution imaging from standoff ranges.
- Microwave ground moving target indicator (GMTI) radars for detecting moving targets and large area coverage.
- Electric support measures (ESM), SIGINT, and communications intelligence (COMINT) for intercepting, locating, recording, and analyzing radiated electromagnetic signals.
- Film for high resolution target classification/identification.
- EO/IR for targeting and target classification/identification.

A similar security payload also incorporates EO/IR sensors, but with resolution that commensurate with its shorter range to targets. As a rule of thumb, security sensors are selected as:

- Radar is used to detect moving targets at a long range.
- Radar for close-in target tracking.
- EO/IR for close-in target tracking.

Surveillance data is either stored for later use, or transmitted to exploitation centers for processing. Security data can be processed onboard the robot or transmitted to the operator control console for real-time observance or for limited post-processing. The key for security is in its algorithms with respect to detecting, identifying, and tracking a potential target or suspicious activities. If weight and processing power is not an issue, multiple overlapping sensors are used on robots to reduce false detections and confirm target identification [61.12]. This is a layered approach. Performance is increased when a wide area surveillance system is used for initial detection or cross-cueing. For example, an

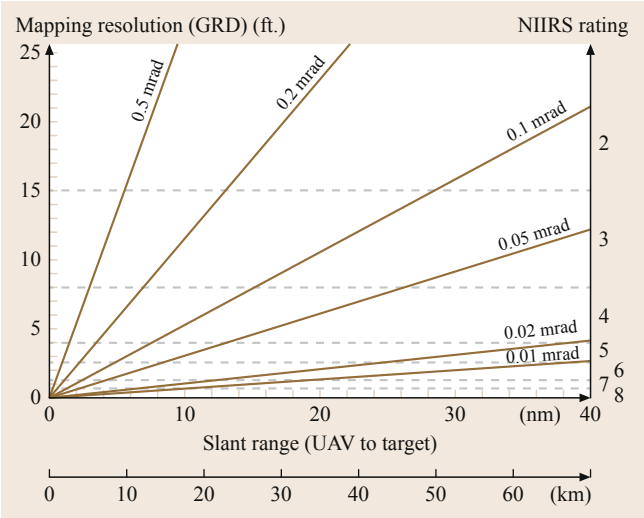


Fig. 61.5 Determining sensor resolution for ATR or RSTA (courtesy of L. Newcome)

MTI sensor can provide a wide area search and collect coarse information about potential targets sufficient to cue the narrow field of view EO and IR sensors that can continue the process of tracking, classifying, and identifying potential targets. Payload sensors collect data, but it is the processing of the data into information that can be used by others that is the nugget for both the surveillance and security communities.

61.3.3 Detection and Tracking Algorithms

An important technology in surveillance reconnaissance, surveillance, and target acquisition (RSTA). This topic is also closely associated with intelligence, surveillance, and reconnaissance (ISR) and automatic target recognition (ATR). ATR, RSTA, and ISR are age-old problems in the military to identify and classify activities of interest in the environment [61.13–16]. During surveillance, the user has an option to acquire a target at a specified location, search an area or sector for either stationary or moving targets, perform a panoramic reconnaissance of an area or sector, initiate fire control, or manually control its sensors by a human operator to acquire surveillance imagery as depicted in Fig. 61.6. For example, if the sensor is stationary and the target is stationary, than a maximum-average-correlation-height approach where a bank of linear correlation filters are optimized to respond to the presence of a specific target object by producing a peak at its corresponding location can be very effective. If a stationary target search is used, the platform is directed to search a specified area for targets using stationary target detection and recognition algorithms.

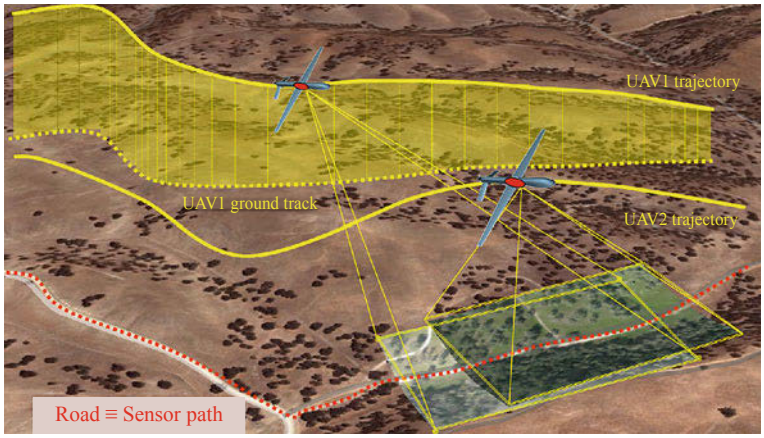


Fig. 61.6 UAV sensor coverage with aircraft trajectory (courtesy of University of Illinois – Urbana Champaign)

These algorithms can detect both stationary and moving targets, but moving target detection does not make use of motion related information that is associated with a robot. Moving objects in the scene are the subjects of interest in most visual surveillance applications using EO/IR sensors. Specifically, the actions of the moving objects need to be monitored and appropriate flags need to be raised when events of interest occur in the scene. The mobile platform needs to be controlled so as to aid the surveillance task. Thus, the objectives of the surveillance task and the associated real-time computation constraints involve difficult challenges in building highly robust yet computationally cheap algorithms for detection, classification, and correspondence.

Manually, the user can select several options including the type of search modes to be used in which the sensor is assumed to be either stationary or moving, and the target is assumed to be either stationary or moving. For example, a UAV and satellite are always in motion, while the UGV/USV/UUV can either be either stationary or mobile. The selection of the stationary target mode causes stationary target detection and recognition algorithms to run, while the selection of the moving target mode causes moving target detection and tracking algorithms to be executed. The user would benefit from an automatic selection feature. The demanding nature of the required detection criteria for surveillance and security necessitates the integration of complementary technologies which can sense motion, characterize patterns, find thermal signatures, and capture temporal behavior as can be seen in VIDEO 678.

The RSTA or ATR algorithm sequence [61.17] include reconnaissance, acquire target through search, and finally target tracking. A panoramic reconnaissance command can direct the robot's payload to collect a mosaic of images over a wide field of regard and send it to the operator for manual viewing. There are several techniques used to create an image mosaic from a sequence of images. The registration between images is obtained by minimizing the sum-squared error of image intensities at each pixel. Although this technique produces very accurate registration results, it tends to be slow, and typically require operator interaction to initialize the registration function. The collected mosaic may be used to assess trafficability, check out regions of interest such as roads and bridges, and manually look for enemy targets. To acquire targets, the vehicle is directed to look at a particular map location or in a particular direction, acquire an image, and perform stationary target detection on that image. An operator can select several search options in which the robot is assumed to be either stationary or moving, and the target is assumed to be stationary or moving. The search area determines whether to search a map region or an azimuth-elevation wedge. An azimuth may be specified relative to the world (map) or specified in vehicle coordinates. The selection of the stationary target function causes stationary target detection and recognition algorithms to run, while selection of a moving target mode causes moving target detection and tracking algorithms to be executed. The search mode determines whether to search an area in a single time or multiple times frames. An EO/IR sensor can use a combination of image processing algorithms. Probing-type algorithms include connected-component analysis or a blob analysis.

A large body of image processing algorithms has been developed to detect targets using techniques including histogram manipulation, convolution, morphology, over- and under-sampling, quantization, and spectral processing using Fourier transforms and discrete cosine transforms (DCTs). These algorithms tend to be computationally intensive [61.18, 19]. Other popular ATR algorithms include binary template matching, multispectral imaging, and wavelet transformation techniques. Of the many types of image enhancement algorithms, a spatial convolution kernel filtering tech-

nique produces the most dramatic results. A convolution kernel generates a new pixel value based on the relationship between the value of the pixel of interest, and the values of those that surround it. In convolution, two functions are overlaid and multiplied by one another. One of the functions is the video frame image and the other is a convolution kernel. ATR algorithms can be broadly described sequentially as: target detection, a segmentation function, feature evaluation, and classification. Target detection approaches can be either image-based or model-based.

If stationary target search is used, the platform is directed to search a specified area for targets using stationary target detection and recognition algorithms. These algorithms can detect targets using blob extraction and blob tracking algorithms. When a potential target is detected, images are sent to a target recognition algorithm for additional processing. Target recognition can be accomplished through the use of a *hashing* algorithm. Various target detections are passed to the hashing algorithm for labeling as one of several target types contained in a database or perception library. The hashing algorithm also produces an estimate of target pose and when the processing is complete, the results are sent to the operator for verification. Moving objects are typically detected using background subtraction. For this type of search, the platform is directed to look at a specified area for targets using a moving target detection algorithm based on temporal differencing. Most surveillance systems use background subtraction as an efficient means of motion detection with a stationary camera. Unfortunately, background subtraction techniques are not always robust under camera jitter, varying lighting conditions, and moving foliage. Techniques such as affine transformations are used for image stabilization. Problems of a periodic nature such as jitter and moving foliage induce multi-modal distribution of pixel intensity values. If the classification of the background environment can be calculated, then the background can be subtracted from the image to enable the use of motion detection and segmentation algorithms.

One popular method for background model generation is the incorporation of autoregressive (AR) or infinite impulse response (IIR) filters. A single AR or IIR filter is used for each pixel to estimate the dominant mode of the background. The AR filter estimates the center and width of the dominant mode of the background, and it is associated with each filter per pixel is a value that approximates the probability that the mode represented by the filter is seen by the pixel. When an intensity value is seen by the pixel falls within the mode of one of the pixel's filters and the associated probability is greater than a preset threshold, then the pixel is declared as background. When a particular intensity value is not represented by any of the filters for that pixel, a new filter is added with an associated low probability value. When a filter matches intensity value, its probability rating is increased and the probabilities of the rest of the filters associated with that pixel are decreased. Thus, at each detection cycle, the filters corresponding to a pixel adapts to better fit the values of its background. This motion detection technique is enhanced by the use of feedback from higher-level algorithms such as a classifier and/or a correspondence algorithm. The classifier has the capability to reject spurious detections. This information can be used either to create a new filter for the pixel or to increase or decrease the probability threshold for detection. Information from its correspondence can be used to predict future locations of detections. This considerably improves the quality of segmentation. Another correlation-based tracker function can be added, which controls the payload pan/tilt gimbal to keep a detected target centered in the image. A harder problem is when the target is moving and the sensor is also moving.

For security robots, camera-based EO/IR sensors have long been used for security and observation purposes as depicted in VIDEO 681. Surveillance cameras are typically fixed at known positions and have coverage of a circumscribed area defined by the fields-of-view of the cameras. Although some recent vision research has addressed autonomous surveillance, in most cases, humans perform the sensory processing. For an autonomous surveillance example, a vision system can provide the geometric shape of the target and its angular location relative to a radar sensor, while its onboard radar system can measure both the distance to the target and its relative velocity radial to the vehicle. The target's perceived geometric shape and distance are then used to calculate its cross-section for purposes of human target classification. The complementary mix of using multiple sensor (camera and radar) technologies allows for greater than 99% probability of detection with less than 1% nuisance alarm rate (also known as false-positives) for fielding a robust security system. This is a good example of cross-cueing. Camera control can be seen in VIDEO 702. Once a target is detected, it is tracked in the center of the collective field-of-view in order to provide verification using successive vision and radar-based detections. A track file is maintained on each detected target to correlate size, distance, speed, and other parameters over an extended period of time. This time-based integration approach allows the system to significantly reject nuisance alarms while using extremely high gains during the detection process. The classification procedure in a security application maps

a given image sequence of a moving object to the most likely class label by classifying each image independently and then classifying the resulting set of class labels. The classifier can also reject classifying an object. For example, a classifier on the mobile detection assessment and response system (MDARS) program has demonstrated class-conditional sequence error rates of less than 5% and a false alarm rejection rate of 80% in disjoint testing. Unknown objects and novel views of known objects are detected by considering the class label history over an image sequence. Image sequences that yield a significant fraction of rejections or cause atypical classifier confusion are saved for later human interpretation.

In the last step, the temporal correspondence of a moving object plays a very important role in classifying, tracking, and interpreting an object's actions. Large number of targets of varying sizes in a scene preclude the use of simple positional correspondence, i. e., correspondence based purely on the positions of moving objects. In such situations, other features of the moving objects, such as different appearance traits, need to be considered for robust correspondence. For example, the measure of goodness of the features can be chosen not only on the object in question, but also on other objects in the scene. A globally *good* set of features can be estimated a priori, but only a subset of these features might be relevant to the correspondence of a particular object. Carnegie Mellon University (CMU) has developed a technique called *differential discriminative diagnosis* in which it provides a systematic method for estimating the relevance of features and checking the temporal consistency of these features for a particular object. The correspondence task is performed with a two-part process. The first part relies on positional correspondence using linear prediction. This part nominates a set of likely candidate matches for a reference object. The second part uses appearance-based correspondence to find the best match among those nominated by linear predictions. A simple linear classifier is trained based on learning techniques to determine whether or not two images are of the same object. Differential discriminative diagnosis identifies those features that are most relevant to the correspondence problem. Efficient correspondence is achieved by enforcing the temporal consistency of the *relevances* determined for a particular object, and this technique has demonstrated correspondence among moving objects with an accuracy of 96%. Target correspondence across multiple sensors can be achieved using a maximum a posteriori (MAP) estimate of the observation sequence to maximize target appearance and spatiotemporal probabilities. A correspondence problem is also used in stereo vision.

Another approach is to perform the correspondence step using a multiple robots configuration as in a swarm, which is addressed further in the multiple vehicle ATR active research area to follow (in Sect. 61.4.4). A more complex scenario involves multiple sensors detecting several targets simultaneously, and a moving sensor payload detecting a moving target [61.20, 21]. A moving target causes the radar signature to shift outside the normal return of a radar image. This shift in frequency (a Doppler shift) is used to distinguish stationary targets from moving targets. The detection of a moving target is more difficult when the sensor is also in motion. Augmenting data such as speed and heading of the moving target can be used to better *profile* potential targets. Other factors such as size, radar cross-section, and environmental clutter are used to make assumptions governing the type and intent of the target. MTI cross-cueing data has been overlaid in real-time on to 3-D visualization of the terrain to allow an operator to quickly locate a target such that the EO/IR sensor could be rapidly slewed to the target of interest. Test results have shown that wide field-of-view MTI radar that is used to cue a limited narrow field of view EO/IR sensor has drastically increased situational awareness on the battlefield. The next evolution is to automate the cross-cueing process in order to track, classify, and identify potential stationary and moving targets [61.22].

61.3.4 Human–Machine Interface

Unmanned aerial robots that are used for surveillance come in all types from the smaller, hand launched Raven, Puma, and Wasp drones to the mid-size Aerosonde and ScanEagle to the larger Predator, Reaper, and Global Hawk platforms that can stay airborne for 24 to greater than 28 h missions [61.23]. The Global Hawk architecture is depicted in Fig. 61.7 with its many communication links. The larger UAVs and their ground segment components interact through an over-the-horizon (OTH) satellite relay and line-of-sight (LOS) links to maintain command and control and sensor data dissemination communication paths for launch, flight, and recovery. Bandwidth compression is applied to the sensor data to maximize area coverage and data throughput. The communications systems are being designed to minimize susceptibility to jamming and interception. Dissemination of UAV collected intelligence is made through direct downlinks to national and theater intelligence centers and other exploitation systems, or through established *hard* communications between UAV ground control stations, and the exploitation systems. The UAV ground control station has limited capabilities to disseminate collected intelligence directly to an exploitation ground system

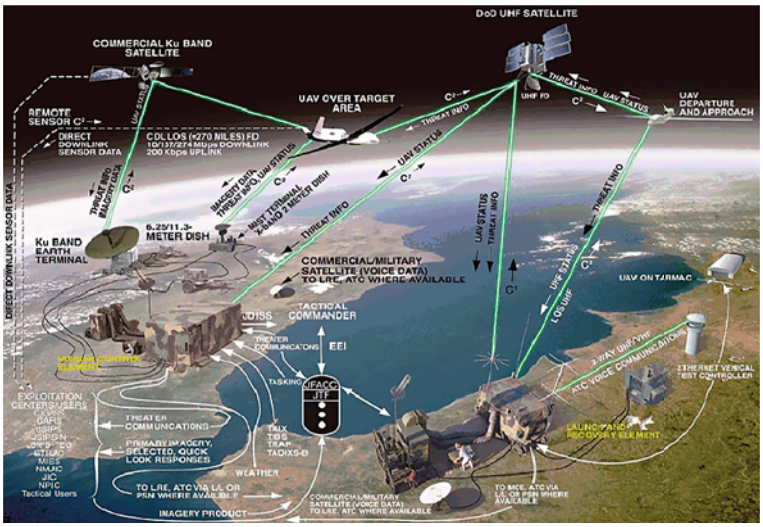


Fig. 61.7 Global Hawk UAV architecture (courtesy of Northrop Grumman and Raytheon)

Fig.61.8a–c Global Hawk ground elements: **(a)** ground station, **(b)** operator seats, and **(c)** UAV display (courtesy of Northrop Grumman and Raytheon)

or battlefield customer. When deployed, a fully operational system consists of four Predators (with EO/IR sensors), a ground control station (GCS) that houses the pilots and sensor operators for human–machine interface (HMI), and a primary satellite-link communication suite.

The Predator UAV can run autonomously, executing simple missions such as reconnaissance on a software program, or it can run under the control of its crew (Fig. 61.8). The crew of a single Predator consists of one pilot and two sensor operators. The pilot drives the aircraft using a standard flight stick and associated controls that transmit commands over a C-Band line-of-sight data link. When operations are beyond the range of the C-Band link, a Ku-Band satellite link is used to relay commands and responses between a satellite and the aircraft. Onboard, the aircraft receives orders via an L-3 Communication satellite data link system. The pilots and crews use the images and radar received from the aircraft to make decisions about controlling the aircraft. The pilot's main mission is to fly the plane, while the sensor operator controls the cameras that bring the battlefield into full view to gain intelligence. Unlike other manned missions, pilots for large UAVs only control the plane during landing and takeoff. However, sensor operators do much more than watch the feedback from their cameras [61.24]. Working together with pilots, they do preflight checks, coordinate with the air control tower, and provide feedback to pilots. The intelligence data is streamed back to exploitation elements where the data is analyzed by people on the ground or with automatic detection algorithms. Not including the exploitation element, it currently takes approximately 82 personnel to operate four of the larger UAVs for a 24 h period including technicians and support personnel.

Typically, unmanned ground robots used for security [61.25, 26] are available in limited options from the commercial Vigilus system, Reborg-Q security robot, OFRO (by Quadretec Ltd.) and patrolbot, to the army's mobile detection assessment and response system (MDARS) and its family of integrated rapid response equipment (FIRRE). The MDARS [61.27] system consists of ground robotic platforms capable of

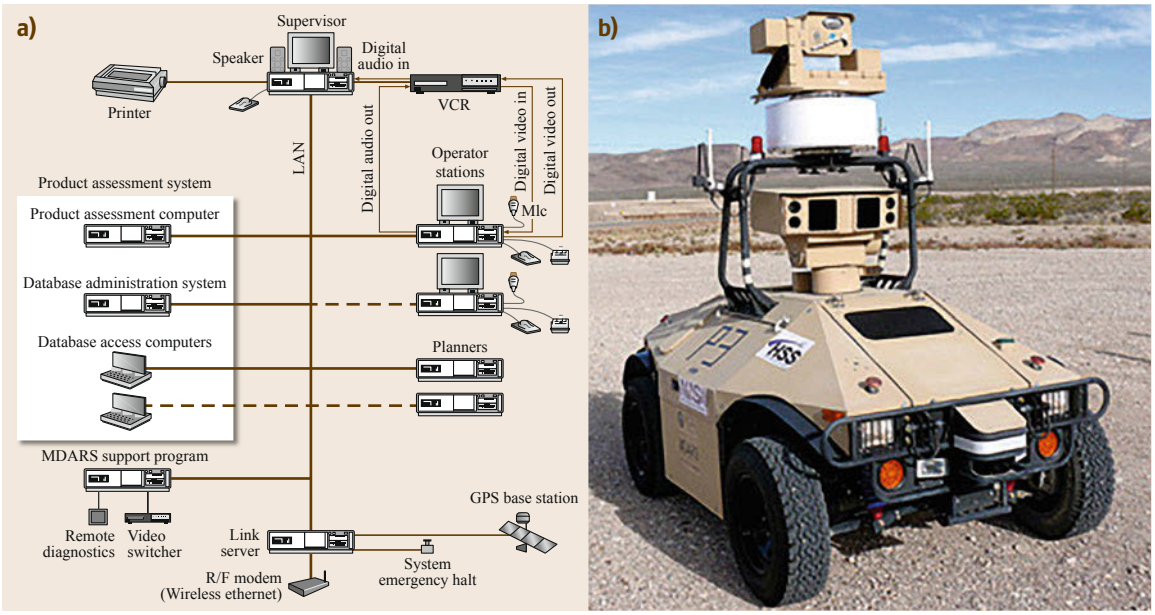


Fig. 61.9 MDARS operational concept and system components (courtesy of Space and Naval Warfare Systems Center (SPAWAR), Pacific)

preprogrammed autonomous movement, motion detection, an incident assessment subsystem, and two way communications capability between operator and robot. Its operational configuration will consist of pair(s) of mobile platforms employing a suite of sensors that are preprogrammed or remotely controlled by an operator located at a remote monitoring and control station. The actual number of fixed and mobile patrol units for a given location will be site specific. The MDARS Increment 1 system components are depicted in Fig. 61.9 with its various interfaces. All such robots and sensors (known collectively as resources) are controlled by the multiple resource host architecture (MRHA), with minimal human (i. e., security personnel and operators) supervision required [61.28]. The individual processors are connected via an Ethernet local area network (LAN) that supports a peer-to-peer communications protocol (expandable to 32 platforms or robots). The user interface provides a *big picture* representation of secured areas and system resources. User intervention is required only when a platform encounters an exceptional condition such as an environmental hazard or a security breach.

The operator control unit is either in monitor mode, in which the operator observes and monitors the status of the robot and sensors, or the operator is in direct control of a single resource such as a single robot in a multi-robot system. Moreover, the robot itself carries a number of sensory systems, each of which must also be controlled individually. For example, MDARS may have eight cameras, two radars, and four lights, but only a single camera, radar, and light is controllable at any given time as shown in VIDEO 680. Multiple cameras can be viewed simultaneously, but the operator cannot pan two of them at the same time. A joystick button or dialog button can be configured to cycle between the various payloads. The control of the resources and each of its selected subresources is integrated in such a way that a single joystick can be used to perform all relevant joystick tasks. As an example, the operator can pan the selected camera with the joystick, and then press a button to put the robot in teleoperation mode and drive the robot using the same joystick. The concept of modes is used to provide rich control features for each resource. This paradigm reduces the complexity of the software and simplifies the user interface presented on its displays, while still giving the operator full control of each asset. The MDARs control station allows the operator to execute preplanned patrols, random patrols, or user-directed patrols as seen in VIDEO 701. Automation of sensor selection would improve operations.

The controller displays can perform basic image processing, which means they take in video at a certain resolution, and display it in the display's native resolution [61.29]. Video processing is the act of scaling, which is converting an incoming video signal from one size or resolution to another in order to work with the display panel. This is what is usually referred to as *video processing*, and it is a minor feat compared to video enhancement. Video enhancement techniques can

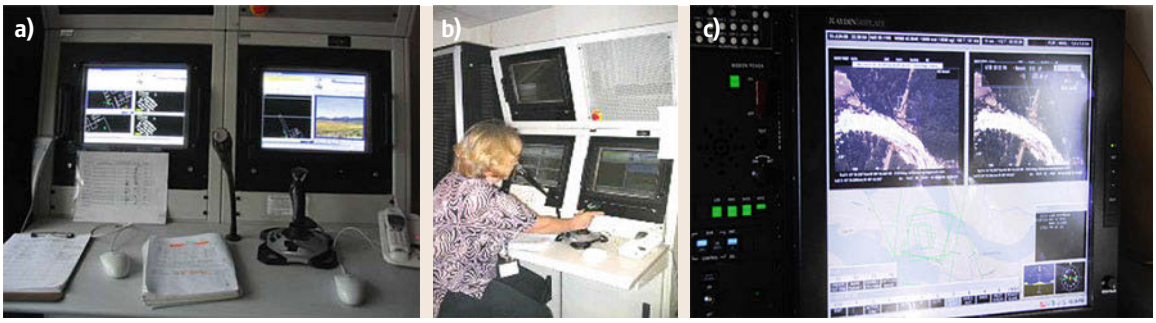


Fig.61.10a–c MDARS mission control center. **(a)** Control station, **(b)** MDARS operator, and **(c)** display (courtesy of SPAWAR)

be based on spatial domain methods or frequency domain methods [61.30]. Image enhancement algorithms offer a wide variety of approaches for modifying images to achieve visually acceptable images. The choice of techniques implemented is a function of the specific task, image content, observer characteristics, and viewing conditions. Mosaic images provide better visualization of a tracking scenario. All security sensors are fused together to form a composite *threat score* on potential targets, and intruder alarms are reported to the user on the MDARS console (Fig. 61.10). Also, the intruder's range and bearing are displayed on a map window and a digital stream from the camera is shown in a video window. For ATR, the operator receives notification of a target by the appearance of an icon on the map display at the operator interface. Selecting an icon brings up a target verification panel and a target response panel showing the imagery associated with the target that was provided by the sensors. With experience, most operators became comfortable and competent with normal operations, and each acquiring variable levels of proficiency in responding to exceptional events such as dealing with intruders or taking manual control of the robot.

61.4 Active Research

In addition to enabling robotic surveillance and security technologies, there are many active areas of robot research, such as being able to produce high-quality imagery from a mobile platform that poses a number of challenges. In addition to issues related to camera motion and the resulting image perspectives, the quality of the video imagery can also be compromised by poor environmental conditions, data link degradations, and bandwidth limitations. Atmospheric factors such as poor lighting at dawn, dusk, or night time, the presence of shadows, target occlusion and adverse weather, including sandstorms and variable clouds, can obscure important details. Some identified areas of interest include: Sect. 61.4.1 in solving the data processing problem due to the large amounts of sensory information collected, Sect. 61.4.2 detecting human activity in

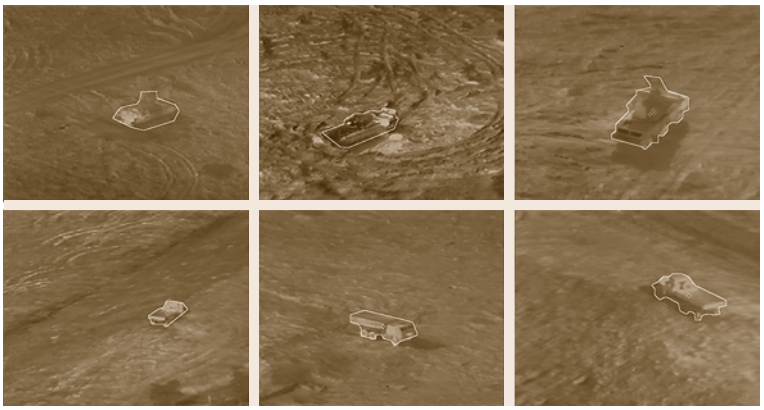


Fig. 61.11 Automatic target recognition (courtesy of Bilkent University)

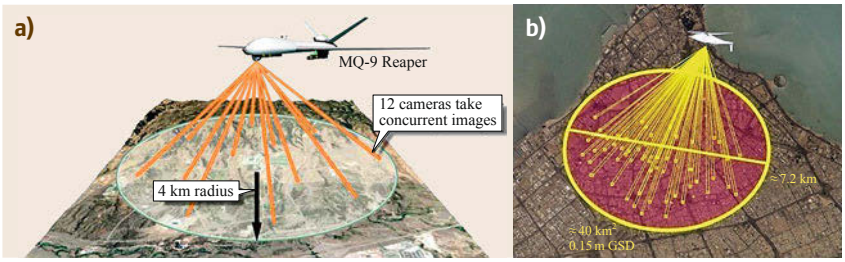


Fig.61.12a,b Multiple real-time video stream sensors. **(a)** Gorgon Stare (courtesy of Sierra Nevada Corp) and **(b)** ARGUS-IS (courtesy of BAE Systems)

support of surveillance, Sect. 61.4.3 facial recognition for security identification purposes, and Sect. 61.4.4 collaborative ATR by multiple robot in either heterogeneous or homogeneous systems.

61.4.1 Solving the Data Processing Problem

With all the useful information that UAVs provide comes the problem of how to sort through it all and find actionable data. This is the *Big Data* problem for the military. One manual solution to the data explosion problem is to tag the data, store it, and retrieve it when needed. Unfortunately, this information would not be timely. Data fusion involves the use of techniques and software that combine data from multiple sources and analyze the data to make it useful for the end user [61.31]. Data fusion can involve the combining of data (such as UAV video) with a GIS, which adds location and time data to the images gathered (Fig. 61.11). To accomplish this, the raw data has to be combined with metadata, which is information about the data that enables the data to be combined. On-going research such as with *intelligent search* is another tool that can be used to make UAV data more accessible. Based on modern artificial intelligence (AI) techniques such as machine learning, depth first searches using iterative deepening, or depth first searches based on space complexity, users can quickly personalize a search to make relevant data more easily available [61.32].

To help users retrieve information, one promising method, called natural language processing, discovers user preferences and needs by either extracting knowledge from what users are looking for or by interactively generating explanatory requests to focus users on the information they are interested in [61.33]. The use of natural language to improve the performance of the intelligent search engines is one aspect of AI. Other related AI technologies for automated processing include object recognition and statistical machine learning, or application of AI in intelligent search is facial recognition. One promising solution that could help with the data overload issue is more efficient techniques for sensor fusion. For example, a multisensor fusion algorithm that can bring together information from multiple platforms about a particular target of interest in a unified display would be beneficial to the operator. Researchers will need to develop smart technologies that automate the entire process of archiving, tagging, retrieving, managing, and displaying data and other information gathered by increasingly sophisticated sensors. As an adjacent topical area, machine-to-machine interfaces will need to become much more transparent in order to handle the data deluge [61.34].

A future technology called wide area surveillance (Fig. 61.12) is based on a new sensor system called Gorgon Stare, a video capture technology that uses a spherical array of nine to twelve cameras attached to an UAV. With so many sensors, Gorgon Stare needs to utilize a system of tagging and incorporating metadata to be fully effective. Initial reports on the progress of Gorgon Stare is that there are deficiencies in its IR performance, having *numerous* interoperability problems, having a lack of stability of the sensors, and encountering reliability problems. These issues can be overcome as the program matures. Unfortunately, the new sensor system is still limited by the human's inability to process the information gathered from many cameras, or rely on advanced automatic processing techniques to fuse the information into the coherent picture.

61.4.2 Detecting Human Activity in Support of Surveillance

The problem of using computer vision to track and understand the behavior of human beings is a very important tool for surveillance and security operations. At its highest level, this problem tries to recognize human behavior and understanding intent and motive from visual observations [61.35, 36], and [61.37]. This is a difficult task, even for humans to perform, and misinterpretations are common. The problem has several subareas of human motion recognition, surveillance, tracking, and activity detection. In the area of surveillance, automated systems to observe pedestrian traffic areas and detect dangerous action are becoming important. Many such areas currently have stationary surveillance cameras in place. However, all of the image understanding

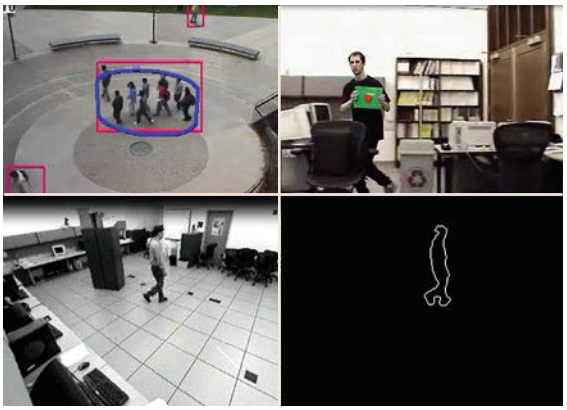


Fig. 61.13 Monitoring individual and group human activities (courtesy of University of Minnesota)

and risk detection is left to human security personnel. This type of observation task is not well suited to humans, as it requires careful concentration over long periods of time. Therefore, there is clear motivation to develop automated, intelligent, vision-based monitoring systems that can aid a human user in the process of risk detection and analysis. Eventually, this capability will migrate from stationary cameras to mobile security applications.

Most of these human activity applications require an automated recognition of high-level activities, composed of multiple simple (or atomic) actions of persons as depicted in Fig 61.13. The goal of human activity recognition is to automatically analyze on-going activities from EO/IR sensors through a sequence of image frames [61.38]. There are various types of human activities and depending on their complexity, human activities can be categorized into four different levels based on: gestures, actions, interactions, and group activities. Gestures are elementary movements of a person's body part and actions, and is defined as single-person activities that may be composed of multiple gestures organized temporally. Interactions are human activities that involve two or more persons and/or objects as shown on VIDEO 683, while group activities are the activities performed by conceptual groups composed of multiple persons and/or objects. Promising events for activity monitoring include the detection of loitering individuals [61.39], and the detection of unusual crowd activities [61.40], which can be difficult to discern from normal crowd activities. It would be useful for security robots to detect motion in restricted areas [61.41], detect perimeter breaches, and detect when security cameras are tampered with [61.42]. Also of interest which may or may not involve the human include the throwing of an object or the detection of an abandoned object. The University of Minnesota has done work in detecting abandoned objects as referred to in VIDEO 682.

There are several approaches used in the recognition of different levels of activities. Single-layered approaches are approaches that represent and recognize human activities directly based on sequences of images while hierarchical approaches represent high-level human activities by describing them in terms of other simpler activities. Single-layered approaches are divided into two classifications depending on how human activities are modeled directly from the video data: space-time approaches (a sequence of two-dimensional (2-D) images placed in chronological order) and sequential approaches (a sequence of observations). Unfortunately, the majority of activity monitoring algorithms is far from being real-time, which is an issue for surveillance. Robot surveillance is demanded in today's environment where the tracking and monitoring of people is an integral part of everyday activities. Monitoring is typically accomplished with multiple cameras [61.43] and from multiple vantage points. Activities of interest vary widely from throwing an object [61.44] to the subtle task of abandoning an object such as a backpack or leaving on unknown package [61.45]. In many instances, it would be important to estimate or count the size of a gathering [61.46–48]. Activity monitoring suffers from the same issues associated with ATR such as signed noise, tracking issues, and segmentation issues arising out of the stabilization of video images. There are a large number of new and promising improvement methods on the horizon, including space-time feature-based approaches, manifold learning, rigid/nonrigid motion analysis, and hierarchical approaches.

61.4.3 Facial Recognition for Security Identification

In order for a security robot to perform its mission to protect and safeguard, sometimes it will encounter people which have to be discerned as either friend or foe. In order to make such a determination, face recognition is critical [61.49]. Most of the current face recognition algorithms can be categorized into two groups: image template-based or geometry feature-based [61.50, 51]. The template-based methods computes the correlation between a face and one or more model templates to estimate the face's identity. For the other approach, the geometry feature-based methods analyze explicit local facial features, and their geometric relationships. Face recognition is a two-step procedure consisting of first detecting a face and followed by recognition. Top researchers have argued that the accurate detection of human faces in arbitrary scenes is the most impor-

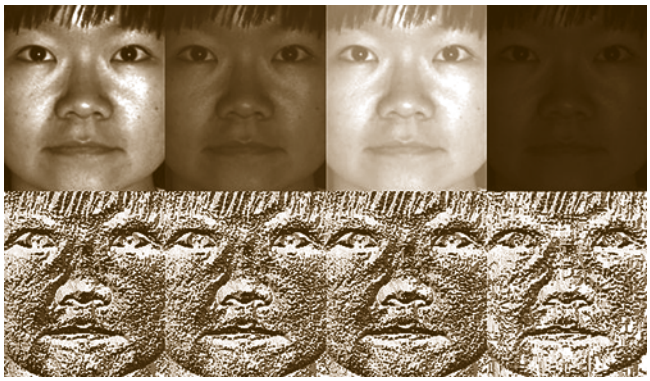


Fig. 61.14 Local binary patterns using OpenCV (courtesy of Yale University)

tant process involved, followed by the face recognition process. When faces could be located exactly in any scene, the recognition step makes the final determination of classifying the person as being either friend or foe.

Algorithms are used to know how to differentiate between a basic face and the rest of the background. If the background is controlled (referred to as being highly structured and uncluttered), an algorithm to remove the background will produce face boundaries. Faces can be found by either using color, motion, or a mixture of both types of information. The detailed steps include acquisition, normalization, and recognition of moving faces in dynamic scenes. Color as a cue can be used for detection and tracking, and provides a computationally efficient yet effective method which is robust under rotations in depth and during partial occlusions. This method can be combined with both motion- and appearance-based face detection. The main idea is to model a class-conditional density for each person in a representation space of relatively low dimensionality. Unfortunately, color detection doesn't work on all kinds of skin color, and is still susceptible to varying lighting conditions.

Humans must periodically blink to keep their eyes moist, and detecting a blinking pattern in an image sequence is a reliable means of detecting the presence of a face. Blinking provides a space-time signal which is easily detected and unique to faces. Detection by blinking is precise, but requires capturing an image pair during a blink. A common technique for recognition is the correlation with a set of Eigen images. Better results have been demonstrated by combining stereo, color, and face detection techniques into an integrated approach [61.52]. Depth estimation using stereo cameras is used to eliminate background effects. Skin-hue classification identifies and tracks likely body parts within the silhouette of a user, while face pattern detection discriminates and localizes the face within the identified body parts. Other approaches to finding faces include the use of neural networks, taking advantage of statistical cluster information, and model-based face tracking. Related work continues using edge-based methods and geometric models such as edge-orientation matching and using the Hausdorff distance measurement. The leading face finding algorithm in unstructured environments is based on a series of weak classifiers such as Haar-like features, representation by integral images, and a method for combining successively more complex classifiers in a cascading structure which can increase the speed of detection by focusing attention on promising regions of the image. An example of facial recognition can be seen in VIDEO 553.

Upon detecting faces, facial recognition has numerous distinguishable landmarks, where different peaks and valleys make up facial features (Fig. 61.14). Measurements of these features include:

- Distance between the eyes
- Width of the nose
- Depth of the eye sockets
- The shape of the cheekbones
- The length of the jaw line.

The 2-D (or 3-D) image of the face is compared to a database of known people's faces. After detection, the image is analyzed to determine the head's position, size, and pose estimation. The sensor measures the curves of the face to a submillimeter scale to create a template. This face template is translated into a unique code that represents the features on the face. This is followed by a matching function that compares the image taken with a database of known faces. This is followed by a verification step that identifies the person in the image.

Skin biometrics can be used to verify the match. Techniques such as Surface Texture Analysis, Local Feature Analysis, and Vector Templating are used to improve the match. There are some impediments to facial recognition such as a lack of resolution in the sensor, glare from glasses, obscuration due to long hair hiding features of the face, and poor lighting conditions. The wider issue is to obtain and maintain an up-to-date database for matching. It would be simple to keep records on who belong to the facility, and to declare unknown faces as foes. However, there will be instances that the human is not a foe, but does not exist in the database either. Another issue would be a scenario in which there are multiple human intruders that stretches the capability of the robot security system. As a final verifier, the human operator can be used to make the final determination as friend or foe, visually or through an interrogation process.

Fig. 61.15 Collaborative ATR (courtesy of NATO)

61.4.4 Collaborative ATR

Swarm robots are a new approach to the coordination of multi-robot systems that consist of large numbers of robots, either homogeneous or possibly heterogeneous robots, i. e., team of air and ground platforms as depicted in VIDEO 700. Swarming implies that a collective behavior emerges from the interactions between robots or the interactions of robots with its environment [61.53, 54], and [61.55]. In a coordinated team approach, individual platforms are able to pass target location information to one another during a target search operation in an attempt to get confirmation of a suspected target. Thus, a robot that detects a target passes the target's location (and a surrounding map-based uncertainty region) to another robot for confirmation. The confirming robot searches the nonoccluded portion of the given map region for the detected vehicle and returns the results of its search to the originating vehicle where they are compared with the original search results (Fig. 61.15). A final target's result is compiled and sent to the operator for verification.

The intent of cooperative target search is to increase the accuracy and reliability of the target search process. The conditions under which this search is guaranteed to be accurate must deal with the following issues:

- It is unclear under what conditions the originating vehicle can be sure that a positive response from the confirming vehicle corresponds to the same target or to any target at all.
- If the orientations of either or both robots are off significantly from their true orientations, it may be that the vehicles are not searching the same map region or that different targets within the uncertainty region are reported as the same target.
- Cooperative target searches puts stringent demands on the accuracy of vehicle orientation estimates. This problem can be partially alleviated by having reliable, long-range laser rangefinders on all participating robots.
- Cooperative target search also puts stringent demands on the accuracy of target detection and recognition algorithms. For example, it is not unusual for an unsophisticated target detection algorithm to generate a positive return from a warm rock or large animal.
- The fact that the confirming vehicle may return a positive response does not increase the probability that the object is actually a valid target. In trials, confirmation of nontargets has actually been observed in field testing.
- It is also difficult to tell whether the confirmation of a valid target results from observation of the same target or detection of a false target. To alleviate this kind of problem, accurate and sophisticated target recognition algorithms are required.

Another concern is when the confirming robot returns a negative response. This lack of confirmation may indicate nothing more than that the confirming robot is looking in the wrong area or that it simply failed to detect a target that is present in its sensor's field-of-view. At this point, it is unclear under what conditions the originating vehicle should disregard its own initial detection. These issues provide motivation for researchers to develop more sophisticated ATR algorithms to obtain higher confidence reports and reduce the number of false alarms. These same issues also provide motivation for the development of context understanding algorithms.

Since cooperative target searches place these stringent demands on the accuracy of vehicle orientation estimates, teams of surveillance platforms must solve the multi-robot distributed estimation problem [61.56] and the related source localization problems [61.57]. Each of the robots collects sensor data regarding its own motion and shares this information with the rest of the team during periodic update cycles. A single estimator, possibly in the form of a Kalman filter, processes the available positioning information from all the members of the team and produces a pose estimate for every one of the robots. The equations for this centralized estimator can be written in a decentralized form, therefore allowing this single Kalman filter to be decomposed into a number of smaller communicating filters. Each of these filters processes sensor data collected by its host robot. The exchange of information between the individual filters is necessary only when two robots detect each other and measure their relative pose. The resulting decentralized estimation schema, which referred to as *collective localization*, constitutes a unique means for fusing mea-

surements collected from a variety of sensors with minimal communication and processing requirements. What complicates the estimation is that the robots are moving and the targets (or multiple targets) may also be moving.

Researchers have introduced a simple updating rule defining a distributed estimation scheme that balances consensus with tracking. It was shown that the probability distribution of the vector of estimates obeys a *Master Equation* derived from first and second moment dynamics. From this master equation, the evolution of its mean estimate and its variance can be computed. If the individual robots ignore the discrete state and simply do consensus, the estimates converge with the probability of *one* to the average of the initial values of the estimates. Moreover, by appropriately weighting the discrete state in the update rule, the average can be tracked with an arbitrarily small variance at steady state conditions.

Source localization is addressed by incorporating a visual gradient-based approach for a robot to adaptively search a location-unknown source. While moving, the robot only measures the received image quality and adaptively decides the motion direction in each time instant. Simulation results indicate that the robot can reach the sensor when the initial signal-to-noise ratio (SNR) is as low as 0 dB and the standard deviation of motion errors is 10% of the motion step size. Due to the adaptive nature of this solution, this approach is robust to measurement noise and motion errors. Since the robot moves along the direction of the gradient estimated at each time, the approach can be applied to an arbitrary signal strength distribution in an area [61.58]. One of the scenarios can be in an environment of obstacles so that the robot at the initial location does not have a line of sight to the sensor. The other scenario to consider is that the source is a cluster of activated sensors that send signals at the same time. Thus, the gradient approach can be extended from 2-D to 3-D space for applications incorporating aerial vehicles [61.59]. A multi-robot system can be applied to both the surveillance problem and the security scenario, but continued research is required to make such a system robust and practical.

61.5 Conclusion

Robots for surveillance typically involve unmanned aerial vehicles that can cover large areas and sit on regions of interest to provide a persistent presence [61.60]. UAVs today utilize teleoperation or flies via way-points on longer traverses to search selected regions. During flight, there are instances in the mission profile where launch and recovery operations are also autonomous. With EO/IR and SAR payloads, the data is streamed back to processing elements for post-processing. This becomes a bottleneck due to the high number of platforms in theater and the enormous volume of data that cannot be processed in real-time. These operations are also highly susceptible to weather issues, and impacted by delays and bandwidth limitation issues associated with the many communication links utilized. Collaborative ATR with swarms of vehicles is an emerging capability, but introduces its own set of issues in terms of consistently and correctly identifying targets of interest. Finally, there are related efforts in ground and unmanned surface water (USV) RSTA, and present different environmental-type challenges such as handling sensor motion issues due to buoyancy or flying indoors (VIDEO 703).

Security robots are a maturing capability, but yet to be commonplace. The vision is to have a single operator commanding multiple stationary and mobile robots to deter criminal activities, especially the destructive actions of some people. The overall mission is to detect, assess, and take action in order to deter a potential threat. Typical robots include ground platforms, hovering type vehicles, and surface water vehicles of all sizes. Key issues include the monitoring of human activities, detecting faces, and determining the extent of the potential threat. With multiple sensors available to

Fig. 61.16 Humanoid KUBO as a security guard (courtesy of Korea Advanced Institute of Science and Technology (KAIST))

the operators, there will be limited opportunities to confirm the target or set of targets due to time sensitivities. Moreover, sensor fusion can deliver a more complete picture of the situation for human or automatic confirmation. Affectivity of the system is quantifiable based on measuring coverage, having a high recognition rate, and demonstrating a low false alarm rate. A future direction is to potentially arm the security robots, but this introduces a new series of issues such as liability and ethics. Chapter 80 (Roboethics: Social and Ethical Implications) expands on these larger societal issues.

A future of humanoids (Fig. 61.16) and social robots used for surveillance and security is an inevitable possibility [61.61]. Humanoids (Chap. 67) provide bipedal mobility as a ground robot, and have an inherent sensor payload with its head. However, the head sensors must be used to both navigate and perform reconnaissance. Its surveillance capability is limited by the field of view and resolution of its head cameras, but has expanded orientation capabilities based on its anthropomorphic neck and torso gimbals. This type of robot can take advantage of existing RSTA algorithms such as human activity monitoring, face recognition, and collaborative ATR for target confirmation as discussed previously. However, all these features are dependent on having adequate onboard processing capability in order to make real-time confirmation and decisions. Who is to say a future humanoid security robot cannot pick up a remote sensor suite similar to binoculars in order to search and detect a potential intruder?

Video-References

VIDEO 553 Security: Facial recognition
available from http://handbookofrobotics.org/view-chapter/61/videodetails/553

VIDEO 554 Surveillance by a drone
available from http://handbookofrobotics.org/view-chapter/61/videodetails/554

VIDEO 677 Ground security robot
available from http://handbookofrobotics.org/view-chapter/61/videodetails/677

VIDEO 678 People detection from a UAV
available from http://handbookofrobotics.org/view-chapter/61/videodetails/678

VIDEO 679 UGV demo II: Outdoor surveillance robot
available from http://handbookofrobotics.org/view-chapter/61/videodetails/679

VIDEO 680 MDARS I: Indoor security robot
available from http://handbookofrobotics.org/view-chapter/61/videodetails/680

VIDEO 681 Scout robot for outdoor surveillance
available from http://handbookofrobotics.org/view-chapter/61/videodetails/681

VIDEO 682 Detection of abandoned objects
available from http://handbookofrobotics.org/view-chapter/61/videodetails/682

VIDEO 683 Tracking people for security
available from http://handbookofrobotics.org/view-chapter/61/videodetails/683

VIDEO 700 Collaborative robots
available from http://handbookofrobotics.org/view-chapter/61/videodetails/700

VIDEO 701 Multi-robot operator control unit
available from http://handbookofrobotics.org/view-chapter/61/videodetails/701

VIDEO 702 Camera control from gaze
available from http://handbookofrobotics.org/view-chapter/61/videodetails/702

VIDEO 703 Indoor, urban aerial vehicle navigation
available from http://handbookofrobotics.org/view-chapter/61/videodetails/703

References

61.1 B. Shoop: Product manager force protection systems: Equipment that protects and secures, Chem-Bio Def. Quart. **4**(2), 8–11 (2007)

61.2 I. Pavlidis, V. Morellas, P. Tsiamyrtzis, S. Harp: Urban surveillance systems: From the laboratory to the commercial world, Proceedings IEEE **89**(10), 1478–1497 (2001)

61.3 H.R. Everett, D.W. Gage: From laboratory to warehouse: Security robots meet the real world, Int. J. Robotics Res. **18**(7), 760–768 (1999)

61.4 A. Birk, H. Kenn: RoboGuard, a teleoperated mobile security robot, Control Eng. Pract. **10**(11), 1259–1264 (2002)

61.5 M. Saptharishi, K.S. Bhat, C.P. Diehl, J.M. Dolan, P.K. Khosla: CyberScout: Distributed agents for autonomous reconnaissance and surveillance, Mechatron. Mach. Vis. **93**, 100 (2000)

61.6 T.A. Heath-Pastore, H.R. Everett, K. Bonner: Mobile robots for outdoor security applications, Am. Nucl. Soc. 8th Int. Top. Meet. Robotics Remote Syst., Pittsburgh (1999) pp. 25–29

61.7 H.R. Everett, G. Gilbreath, T.A. Heath-Pastore, R.T. Laird: Controlling multiple security robots in a warehouse environment, AIAA-NASA Conf. Intell. Robots, Houston (1994)

61.8 E. Kuiper, S. Nadjm-Tehrani: Mobility models for UAV group reconnaissance applications, Int. Conf. Wirel. Mob. Commun. (ICWMC) Bucharest, Romania (2006)

61.9 B. Fletcher: New roles for UUVs intelligence, surveillance, and reconnaissance, 9th Pac. Congr. Mar. Sci. Technol., Honolulu (2000)

61.10 A. Zelinsky, R.A. Jarvis, J.C. Byrne, S. Yuta: Planning paths of complete coverage of an unstructured environment by a mobile robot, Proc. 8th Int. Conf. Adv. Robotics (ICAR), Tsukuba (1993)

61.11 D. Stein, J. Schoonmaker, E. Coolbaugh: *Hyperspectral Imaging for Intelligence, Surveillance, and Reconnaissance* (Space and Naval Warfare Systems Center, San Diego 2001), Biennial Review

61.12 D. W. Gage, W. D. Bryan, H. G. Nguyen: Internetting tactical security sensor systems, Proceedings SPIE **3393** (1998) doi:10.1117/12.317683

61.13 V. Morellas, I. Pavlidis, P. Tsiamyrtzis: DETER: Detection of events for threat evaluation and recognition, Mach. Vis. Appl. **15**, 29–45 (2003)

61.14 W. Severson, R. Rimey: Reconnaissance, surveillance, and target acquisition in the UGV/Demo II program, Proc. 4th ATR Syst. Technol. Symp., Vol. I – Unclassif. (1994) pp. 129–142

61.15 D. Hougen, R. Rimey, W. Severson: Description of the RSTA subsystem. In: *Reconnaissance, Surveillance, and Target Acquisition for the Unmanned Ground Vehicle: Providing Surveillance 'Eyes' for an Autonomous Vehicle*, ed. by O. Firschein, T.M. Strat (Morgan Kaufman, New York 1997)

61.16 P.Y. Oh, W.E. Green: CQAR: Closed quarter aerial design for reconnaissance, surveillance and target acquisition tasks in urban areas, Int. J. Comput. Intell. **1**(4), 353–360 (2004)

61.17 B. Bhanu: Evaluation of automatic target recognition algorithms, Proceedings SPIE **0435** (1983) doi:10.1117/12.936960

61.18 C. Olson, D. Huttenlocher: Automatic target recognition by matching oriented edge pixels, IEEE Trans. Image Proc. **6**(1), 103–113 (1997)

61.19 A. Vasile, R. Marino: Pose-independent automatic target detection and recognition using 3-D ladar data, Proceedings SPIE **5426** (2004) doi:10.1117/12.546761

61.20 H. Andreasson, M. Magnusson, A. Lilienthal: Has something changed here? autonomous difference detection for security patrol robots, Proc. IEEE/RSI Int. Conf. Intell. Robots Syst. (IROS), San Diego (2007) pp. 3429–3435

61.21 A. Muccio, T.B. Scruggs: Moving target indicator (MTI) applications for unmanned aerial vehicles (UAVs), Proc. Int. Conf. Radar (RADAR), Adelaide (2003)

61.22 E. Ribnick, N. Papanikolopoulos: Estimating 3D trajectories of periodic motions from stationary monocular views, Proc. Eur. Conf. Comput. Vis. (ECCV) (2008)

61.23 C. Nehme, J. Crandall, M. Cummings: An operator function taxonomy for unmanned aerial vehicle missions, Proc. 12th Int. Command Control Res. Technol. Symp. (2007)

61.24 T.E. Noell, F.W. DePiero: Reduced bandwidth video for remote vehicle operations, Proc. Assoc. Unmanned Veh. Syst. (AUVS) (Association for Unmanned Vehicle Systems, Washington 1993)

61.25 T. Fong, C. Thorpe, C. Baur: Collaboration, dialogue, and human–robot interaction, 10th Int. Symp. Robotics Res., Lorne, Victoria (2001)

61.26 Carroll, D., Nguyen,. C., Everett, H.R., and B. Frederick: Development and testing for physical security robots, SPIE Proceedings **5804** (2005) doi:10.1117/12.606235

61.27 B. Shoop, M. Johnston, R. Goehring, J. Moneyhun, B. Skibba: Mobile detection assessment and response systems (MDARS): A force protection, physical security operational success, Proceedings SPIE **6230** (2006) doi:10.1117/12.665939

61.28 H.R. Everett, R.T. Laird, D.M. Carroll, G. Gilbreath, T.A. Heath-Pastore, R.S. Inderieden, T. Tran, K. Grant, D.M. Jaffee: *Multiple Resource Host Architecture (MRHA) for the Mobile Detection Assessment Response System (MDARS)* (Space and Naval Warfare Systems Center, San Diego 2000), Technical Document 3026, Revision A

61.29 Y. Takahashi, I. Masuda: A visual interface for security robots, Proc. IEEE Int. Workshop Robot Human Commun., Tokyo (1992)

61.30 R. Maini, H. Aggarwal: A comprehensive view of image enhancement techniques, J. Comput. **2**(3), 8–13 (2010)

61.31 H. Warston, H. Persson: Ground surveillance and fusion of ground target sensor data in a networked based defense, Proc. 7th Int. Conf. Inf. Fusion, Stock. (2004) pp. 1195–1201

61.32 J. Loyall, J. Ye, S. Neema, N. Mahadevan: Model-based design of end-to-end quality of service in a multi-UAV surveillance and target tracking application, 2nd RTAS Workshop Model. Embed. Syst. (MoDES), Toronto (2004)

61.33 A.J. Joshi, F. Porikli, N. Papanikolopoulos: Breaking the interactive bottleneck in multi-class classification with active selection and binary feedback, IEEE Conf. Comput. Vis. Pattern Recogn. (2010)

61.34 S. Balakirsky: Semi-autonomous mobile target engagement system, Proc. Assoc. Unmanned Veh. Syst. (AUVS) (1993) pp. 927–946

61.35 P. Turaga, R. Chellappa, V. Subrahmanian, O. Udrea: Machine recognition of human activities: A survey, IEEE Trans. Circuits Syst. Video Technol. **18**(11), 1473–1488 (2008)

61.36 A. Shee, W.-Y. Chung: High accuracy human activity monitoring using neural network, IEEE 3rd Int.

Conf. Converg. Hybrid Inf. Technol., Busan, South Korea (2008)
61.37 J. Aggarwal, M. Ryoo: Human activity analysis: A review, ACM Comput. Surv. **43**(3), 16 (2011)
61.38 A. Fernadez-Caballero, J. Castillo, J. Rodriguez-Sanchez: Human activity monitoring by local and global finite state machine, Expert Syst. Appl. **39**, 6982–6993 (2012)
61.39 N. Bird, O. Masoud, N. Papanikolopoulos, A. Isaacs: Detection of loitering individuals in public transportation areas, IEEE Trans. Intell. Transp. Syst. **6**(2), 167–177 (2005)
61.40 S. Saxena, F. Bremond, M. Thonnat, R. Ma: Crowd behavior recognition for video surveillance, Lect. Notes Comput. Sci. **5259**, 970–981 (2008)
61.41 Z. Zhou, X. Chen, Y.-C. Chung, Z. He, T. Han, J. Keller: Activity analysis, summarization, and visualization for indoor human activity monitoring, IEEE Trans. Circuits Syst. Video Technol. **18**(11), 1489–1498 (2008)
61.42 E. Ribnick, S. Atev, O. Masoud, N. Papanikolopoulos, R. Voyles: Real-time detection of camera tampering, IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS), Sydney, Aust. (2006)
61.43 L. Fiore, D. Fehr, R. Bodor, A. Drenner, G. Somasundaram, N. Papanikolopoulos: Multi-camera human activity monitoring, J. Intell. Robotic Syst. **52**(1), 5–43 (2008)
61.44 E. Ribnick, S. Atev, O. Masoud, N. Papanikolopoulos, R. Voyles: Detection of thrown objects in indoor and outdoor scenes, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007)
61.45 N. Bird, S. Atev, N. Caramelli, R. Martin, O. Masoud, N. Papanikolopoulos: Real time, online detection of abandoned objects in public areas, IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 3775–3780
61.46 P. Kilambi, E. Ribnick, A. Joshi, O. Masoud, N. Papanikolopoulos: Estimating pedestrian counts in groups, Computer Vis. Image Underst. (CVIU) (2008)
61.47 D. Fehr, R. Sivalingam, V. Morellas, N. Papanikolopoulos, O. Lotfallah, Y. Park: Counting people in groups, Proc. 6th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS) (2009) pp. 152–157
61.48 G. Somasundaram, V. Morellas, N. Papanikolopoulos, L. Austin: Counting pedestrians and bicycles in traffic scenes, Proc. 2009 IEEE Conf. Intell. Transp. Syst. Conf., St. Louis (2009)
61.49 J.N.K. Liu, M. Wang, B. Feng: Ibotguard: An internet-based intelligent robot security system using invariant face recognition against intruder, IEEE Trans. Syst. Man Cybern C **35**(1), 97–105 (2005)
61.50 W. Zhao, R. Chellappa, A. Rosenfeld: Face recognition: A literature survey, ACM Comput. Surv. **35**(4), 399–458 (2003)
61.51 R. Gross, J. Shi, J. Cohn: *Quo vadis Face Recognition?* CMU-RI-TR-01-17 Report (Carnegie Mellon University, Pittsburgh 2001)
61.52 P. Viola, M. Jones: Robust real-time face detection, Int. J. Comput. Vis. **57**(2), 137–154 (2004)
61.53 J. Sauter, R. Mathews, A. Yinger, J. Robinson, J. Moody, S. Riddle: Distributed pheromone-based swarming control of unmanned air and ground vehicles for RSTA, Proceedings SPIE (2008) doi:10.1117/12.782271
61.54 Y. Guo, L.E. Parker, R. Madhavan: Towards collaborative robots for infrastructure security applications. In: *Mobile Robots: The Evolutionary Approach*, *Book Series on Intelligent Systems Engineering Series*, ed. by N. Nedjah, L. dos Santos Coelho, L. de Macedo Mourelle (Springer-Verlag, Berlin 2006) pp. 185–200
61.55 J. Feddema, C. Lewis, P. Klarer: Control of multiple robotic sentry vehicles, Proc. SPIE Unmanned Ground Veh. Technol. **3693**, 212–223 (1999)
61.56 F. Shaw, E. Klavins: Distributed estimation and control for stochastically interacting robots, 47th IEEE Decis. Control Conf., Cancun (2008)
61.57 S. Roumeliotis, G. Bekey: Distributed multirobot localization, IEEE Trans. Robotics Autom. **18**(5), 781–795 (2002)
61.58 Y. Sun, J. Xiao, X. Li, F. Cabrera-Mora: Adaptive source localization by a mobile robot using signal power gradient in sensor networks, IEEE Glob. Commun. Conf., New Orleans (2008)
61.59 X. Zhou, S. Roumeliotis: Robot-to-Robot relative pose estimation from range measurements, IEEE Trans. Robotics **24**(6), 1379–1393 (2008)
61.60 D. Carroll, H.R. Everett, G. Gilbreath, K. Mullens: Extending mobile security robots to force protection missions, Proc. Progr. Robotics FIRA RoboWorld Congr. (2009)
61.61 N. Hopper: Security and complexity aspects of human interactive proofs, 1st Workshop Human Interact. Proofs (HIP), Palo Alto (2002)

Multimedia Contents

62. Intelligent Vehicles

Alberto Broggi, Alex Zelinsky, Ümit Özgüner, Christian Laugier

This chapter describes the emerging robotics application field of intelligent vehicles – motor vehicles that have autonomous functions and capabilities. The chapter is organized as follows. Section 62.1 provides a motivation for why the development of intelligent vehicles is important, a brief history of the field, and the potential benefits of the technology. Section 62.2 describes the technologies that enable intelligent vehicles to sense vehicle, environment, and driver state, work with digital maps and satellite navigation, and communicate with intelligent transportation infrastructure. Section 62.3 describes the challenges and solutions associated with road scene understanding – a key capability for all intelligent vehicles. Section 62.4 describes advanced driver assistance systems, which use the robotics and sensing technologies described earlier to create new safety and convenience systems for motor vehicles, such as collision avoidance, lane keeping, and parking assistance. Section 62.5 describes driver monitoring technologies that are being developed to mitigate driver fatigue, inattention, and impairment. Section 62.6 describes fully autonomous intelligent vehicles systems that have been developed and deployed. The chapter is concluded in Sect. 62.7 with a discussion of future prospects, while Sect. 62.8 provides references to further reading and additional resources.

62.1 **The Motivation and Approaches to Intelligent Vehicles** 1628
62.1.1 Brief History 1628
62.1.2 Benefits of Intelligent Vehicles ... 1631

62.2 **Enabling Technologies** 1632
62.2.1 Vehicle State 1632
62.2.2 Environment State 1632
62.2.3 Digital Maps and Satellite Data ... 1633
62.2.4 Driver State 1634
62.2.5 Communication 1634

62.3 **Road Scene Understanding** 1635
62.3.1 Road/Lane Tracking 1636
62.3.2 Road Sign Detection 1637
62.3.3 Traffic-Light Detection 1637
62.3.4 Visibility Assessment 1637
62.3.5 Vehicle Detection 1638
62.3.6 Pedestrian Detection 1639

62.4 **Advanced Driver Assistance** 1639
62.4.1 Collision Avoidance and Mitigation 1640
62.4.2 Adaptive Cruise Control 1643
62.4.3 Stop and Go 1644
62.4.4 Parking Assistance 1644
62.4.5 Lane Keeping 1644
62.4.6 Lane Changing 1645

62.5 **Driver Monitoring** 1645
62.5.1 Driver Fatigue, Inattention, and Impairment 1646
62.5.2 Driver and Passenger Protection . 1647
62.5.3 Emergency Assistance 1647

62.6 **Towards Fully Autonomous Vehicles** 1647
62.6.1 Operating Safely 1647
62.6.2 Traffic Congestion 1647
62.6.3 Environmental Factors 1648
62.6.4 The Automobile of the Future 1648
62.6.5 Automated Vehicle Deployment .. 1649

62.7 **Future Trends and Prospects** 1650

62.8 **Conclusions and Further Reading** 1651

Video-References 1651

References 1652

62.1 The Motivation and Approaches to Intelligent Vehicles

An important field of application of robotics has emerged in the last 20–25 years which is centered on the automobile, named *intelligent vehicles*. The automobile has been one of the most important products of the twentieth century. It has generated an enormous industry and has given individuals a freedom of movement that has completely changed our way of life. Indeed, the automobile has been a key factor in the large change in the way our urban societies are structured. Today, there are more than 800 million vehicles on the planet, and this number is expected to double in the next 10 years. This challenge has led to the development of an active research domain with the ultimate goal of automating the typical tasks that humans perform while driving. An intelligent vehicle is defined as a vehicle enhanced with perception, reasoning, and actuating devices that enable the automation of driving tasks such as safe lane following, obstacle avoidance, overtaking slower traffic, following the vehicle ahead, assessing and avoiding dangerous situations, and determining the route. The overall motivation for building intelligent vehicles has been to make motoring safer, and more convenient and efficient.

Approaches to this problem revolve about three main architectures [62.1], according to the degree of autonomy they aim for (Fig. 62.1):

a) Driver-centric approaches are built around the idea of having a human in the loop supervising the functions of the vehicle.
b) Network centric-approaches assume that vehicles can share information with other vehicles or the infrastructure.
c) Vehicle-centric approaches aim to build fully autonomous intelligent vehicles with no human in the loop.

62.1.1 Brief History

Futurama was an exhibit at the 1939 World Fair designed by Norman Bel Geddes that presented a possible model of the world 20 years into the future (1959–1960) with automated highways. The exhibit was sponsored by the General Motors Corporation and was characterized by its vision of automated highways. Technological development in support of this vision began in the late 1940s and early 1950s, with pioneering efforts by RCA Laboratories and GM Research. By 1960, GM had tested full-scale cars driving automatically on test tracks, with lane tracking, lane changing and car following functions automated. The goal of these efforts was the development of an automated highway system (AHS). In the 1960s, a key effort was research and development under the leadership of *Fenton* and his team at The Ohio State University [62.2]. During these years, vehicles were developed to test steering and speed controllers. Runs were accomplished at the Transportation Research Center (TRC) grounds, using active sensing (a live wire imbedded in the roadway) and with steering control done with analog feedback controllers. In Japan, research on automated highway systems was started in the early 1960s in the Mechanical Engineering Laboratory (MEL) and the National Institute of Advanced Industrial Science and Technology (AIST). The automated driving system in 1960s employed a combination of an inductive cable

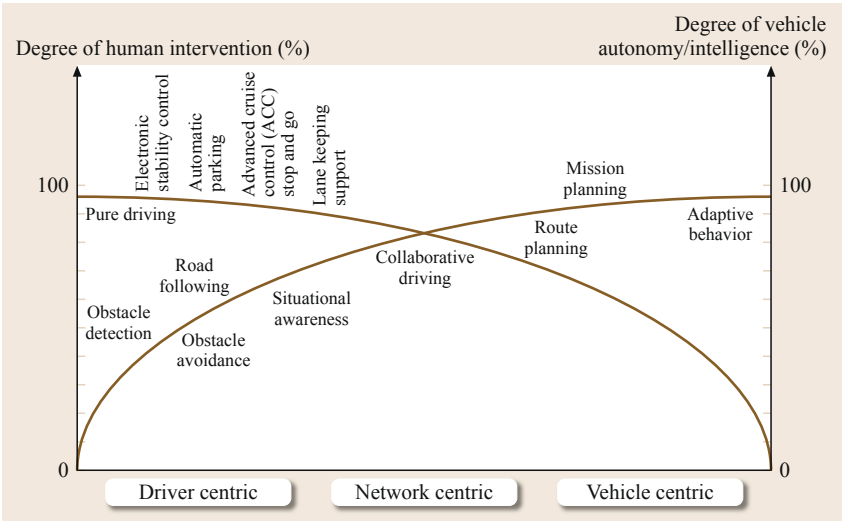


Fig. 62.1 Architectures for autonomous vehicles (after [62.1])

embedded under a roadway surface and a pair of pickup coils at the front bumper of a vehicle for lateral control, and it was a cooperative system between vehicle and infrastructure. In addition to the free agent, a rear-end collision avoidance system between two automated vehicles based on roadway vehicle sensors was developed. The automated vehicle drove up to 100 km/h on a test track in 1967 [62.3].

The history of intelligent vehicles developed further over the ensuing decades. Although the first ideas were born in the 1960s, the level of maturity of the technology at that time did not allow pursuit of the original goal of implementing fully autonomous all-terrain, all-weather vehicles. The first documented prototypes of such vehicles were fielded by a few groups in the military arena in the mid 1980s [62.4–6]. The initial stimulus that triggered these innovative ideas was provided by the military sector, which was eager to provide complete automation to its fleet of ground vehicles.

Testing of autonomous vehicles on real roads in a real environment was one of the most important milestones in the history of intelligent vehicles. In Germany, in the beginning of the 1980s *Dickmanns* and his team equipped a Mercedes-Benz van with cameras and other sensors. For safety reasons, initial experiments in Bavaria took place on streets without traffic. Since 1986 the robot car *VaMoRs* from the same group manages to drive all by itself, since 1987 at speeds up to 96 km/h. A number of these capabilities were also highlighted in the European PROMETHEUS Project. (for example, [62.7])

This happened in the mid to late 1990s. Figure 62.2 shows the first motor vehicles that pioneered the development of intelligent vehicles. In the summer of 1995, the Carnegie Mellon Navlab group ran their *No Hands Across America* experiment [62.8]. They demonstrated automated steering, based solely on computer vision,

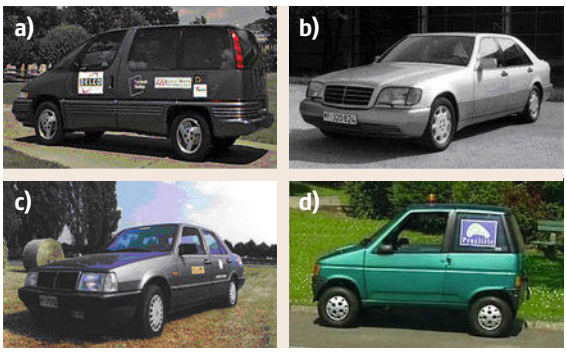


Fig.62.2a–d Pictures of pioneering autonomous vehicles **(a)** NAVLAB, **(b)** UBM, **(c)** Argo, and **(d)** Ligier-INRIA electric vehicles

over 98% of the time on a 2800 mile trip across the United States. At the same period the EU funded the first large and ambitious project for autonomous car navigation (the EUREKA Project *Prometheus*) involving the major European car constructors and research institutes. In the scope of this project, UBM (Bundeswehr Universität Munich) fielded a vehicle that was demonstrated on a 1758 km-trip from Munich to Copenhagen and back. The vehicle was able to drive autonomously for 95% of the trip. The car suggested and executed manoeuvres to pass other cars. Unlike later robot cars, this car located itself on the current road and followed it until instructed otherwise. It did not localize itself in global coordinates and could drive without a global positioning system (GPS) and road maps as found in a modern automotive navigation systems. The vehicle used dedicated purpose-built computers and hardware.

A different approach was followed by VisLab at the University of Parma within the Argo project. The passenger car that was designed and developed was based on a low-cost approach. An off-the-shelf Pentium 200 MHz personal computer (PC) was used to process stereo images obtained from low-cost cameras installed in the driving cabin. The vehicle was able to follow the lane, locate obstacles, and – when instructed – change lane and overtake slower vehicles. The main milestone of this project was the successful test of the Argo vehicle on a tour of Italy of more than 2000 km called *Millemiglia in Automatico*, in which the vehicle drove itself for 94% of the total distance. Another low-cost approach was developed at INRIA in 1995 for its *dual mode* system, in which a small dedicated electric vehicle was able to park itself autonomously using only ultrasonic sensors and a linear camera.

Through the years, many demonstrations have been held showing the public the capabilities of autonomous vehicles, and by extension, underlining the sensor capabilities at the autonomous driving demonstrations. One of the most comprehensive highway-based demonstrations was held in 1997 on I-15 in San Diego, USA, and showed the capabilities of cars, buses, and trucks in various automated highway scenarios. The demonstration (called Demo'97) was organized by the National Highway Systems Consortium. The key technologies were lane-following using roadway imbedded magnets, roadway laid radar-reflective stripes or using lane markers with vehicle mounted cameras; car following using laser or radar, with or without help of inter-vehicle communication. Scenario maintenance was accomplished by either total preplanning and timing, or by GPS and map-based triggering, or by situation-based triggering [62.9, 10].

The Demo'97 was followed by a number of demonstrations around the world, notably in Japan, Holland, France, and the US. In each case, not only were state-of-the-art technologies demonstrated, but the public was informed about what to expect in future cars.

In November 2000, the demonstration *Demo 2000 Cooperative Driving* was organized by the National Institute of Advanced Industrial Science and Technology in Tsukuba, Japan. Several platooning scenarios were accomplished by five automated vehicles in the enclosed test field. This platoon of five automated vehicles performed stop-and-go operations, platooning, splitting into two platoons demonstrating an exit ramp, merging into one platoon from two platoons imitating highway traffic at a ramp, and passing by the last vehicles. Furthermore, obstacle detection, platoon leaving, and joining tasks were accomplished. Autonomy was guaranteed by inter-vehicle communication. In addition, vehicles were equipped by differential global positioning system (DGPS) to assure localization by the laser radar to detect obstacles and to measure the relative position between the vehicles. Vehicles of Demo 2000 did not communicate with the infrastructure i. e., road-to-vehicle communication except DGPS. These automated vehicles were autonomously driven by exchanging information through inter-vehicle communication [62.11].

Current research initiatives are oriented towards the development of intelligent vehicles in realistic scenarios. However, until recently, primarily due to legal issues, full autonomy had not yet been set as the ultimate goal. The automotive industry had set as its primary goal the need to equip vehicles with supervised systems and – more generally – advanced driving assistance systems (ADAS) instead of automatic pilots. In other words, the driver was still in charge of running the vehicle, but the driver and vehicle were monitored by an electronic system that detects possibly dangerous situations and reacts by either warning the driver in due time, or taking control of the vehicle in order to mitigate the consequences of the driver's inattention. In spite of the aforementioned legal issues, the ultimate goal has now shifted towards developing self-driving systems and a number of automotive manufacturers have announced such cars in the market by model year 2020. Concurrently, departments of transportation worldwide which have been primarily interested in social, economic, or environmental objectives aimed at enhancing fuel efficiency, road network usage, and improving quality of life in terms of mobility have now shifted their attention to the possibility of mixed traffic on the road, consisting of regular vehicles driven by people and autonomous vehicles [62.12].

Different approaches and research goals have evolved internationally. The USA has pushed for fully autonomous vehicles which could be commercialized, Europe has focused on ADAS systems and public transportation applications, clean vehicles, and free-service autonomous vehicles (Sect. 62.7). This contrast is illustrated by the differences in scope and objectives between US funding schemes and competitions (e.g., DARPA challenges, Google cars) and European projects (e.g., Prometheus, Carsense, Prevent, Intersafe).

The good results obtained by ADAS in the automotive arena in recent years has induced the military sector to give a new vigorous push to the original ideas of automating its fleet of ground vehicles. The Defense Advanced Research Projects Agency (DARPA) launched the Grand Challenge in 2003, a race for autonomous vehicles that had to travel for more than 200 km in unstructured environments. This unprecedented challenge attracted a large number of top-level research institutes and helped the scientific community take a considerable step forward.

In 2005 the DARPA Grand Challenge required autonomous driving in a rough terrain desert scenario with no traffic, obstacle types known in advance, and few if any road markers on a course predefined by 2935 GPS points [62.13]. Five cars (maximum speed 40 km/h) completed the 211 km desert course, with Stanford's Stanley winning the challenge in 6 h 54 min. Figure 62.3a–b shows photographs of the top two finishing vehicles. The reader is referred to [62.14] for the technical details of the DARPA Grand Challenge.

Following the Grand Challenges, the 2007 DARPA Urban Challenge was performed in *emulated* city traffic and saw the return to urban vehicles [62.15]. Autonomous ground vehicles competed to win the Urban Challenge 2007 prize by accomplishing an approximately 60 mile urban course area in less than 6 h with moving traffic. The autonomous vehicle had to be built upon a full-size chassis compared to production type vehicles and must have a documented safety report. The autonomy capability demonstrated in forms of a complex set of behaviors by obeying all traffic regulations while negotiating with other traffic and obstacles and merging into traffic. The urban driving environment challenged autonomous driving with some physical limits such as narrow lanes, sharp turns, and also by the daily difficulties in a daily urban driving such as congested intersections, obstacles, blocked streets, parked vehicles, pedestrians, and moving vehicles. The winner of the Urban Challenge was Tartan Racing, a collaborative effort by Carnegie Mellon University and the General Motors Corporation, with their vehicle *Boss*, a heavily modified Chevrolet Tahoe. The

second place finisher was the Stanford Racing Team with their entry *Junior*, a 2006 Volkswagen Passat. Figure 62.3c–d shows photographs of the top two finishing vehicles. In May 2011, The Netherlands Organization for Applied Scientific Research TNO, together with the Dutch High Tech Automotive Systems innovation program (HTAS), organized the Grand Cooperative Driving Challenge (GCDC) in The Netherlands. The 2011 GCDC focused on cooperative adaptive cruise control (CACC). Nine international teams participated in the challenge [62.16].

In 2010, the University of Parma's VisLab Intercontinental Autonomous Challenge [62.17–19] further pushed the boundaries by having four driverless vehicles (Fig. 62.3e) drive autonomously 15 000 km from Parma, Italy to Shanghai, China. A more recent example is the Google driverless car (Fig. 62.3f): a fleet of autonomous vehicles which by 2012 had driven more than 480 000 km autonomously. Google has demonstrated fully autonomous driving in dense urban California traffic, utilizing vision-based systems and Google Street View. This activity, repeated in a number of cities generated extensive public interest and led to the belief that autonomous vehicles will be available for public use in the near future. This has been a major factor leading three US states (California, Nevada, and Florida) to approve laws permitting driverless cars. A number of car manufacturers have announced models to come to market by 2020.

Fig.62.3a–f 2005 Grand Challenge vehicles: **(a)** Stanley (1st), **(b)** Sandstorm (2nd), 2007 Urban Challenge vehicles, **(c)** Boss (1st), **(d)** Junior (2nd), **(e)** VisLab and **(f)** Google vehicles

62.1.2 Benefits of Intelligent Vehicles

Having intelligent vehicles running on our road networks would bring a number of social, environmental, and economical benefits. An intelligent vehicle able to assess the driving scenario and react in case of danger would allow up to 90% of traffic accidents that are caused by human errors to be eliminated, thus saving human lives. According to the World Health Organization an estimated 1.2 million people worldwide are killed each year, and about 40 times this number injured, due to traffic accidents.

At the same time, vehicles able to drive at high speeds and very close to each other would decrease fuel consumption and polluting emissions; furthermore they would also increase road network capacity. Vehicles communicating with a ground station could share their routes and be instructed to reroute in order to maintain a smooth traffic flow. Vehicles that can sense and obey speed limits or traffic rules would reduce the possibility of misinterpretation and antisocial driving behavior.

Fully automatic vehicles would also offer a higher degree and quality of mobility to a larger population, including young, old, or infirm individuals, reducing the need even for a driving licence. Finally, the availability of vehicles that could drive themselves would increase the quality of mobility for everyone, turning personal vehicles into taxis able to pick up people and take them to their final destination in total safety and comfort, enabling people to dedicate their driving time to their preferred activities.

However, this full application of intelligent vehicles is far from being complete, since unmanned vehicle technology is still under development for many other applications. The automation of road vehicles is, perhaps, the most common everyday task that attracts the greatest interest from the industry. However, other domains such as agricultural, mining, construction, search and rescue, and other dangerous applications in general, are looking to autonomous vehicles as a possible solution to the issue of the ever-increasing cost of personnel. If a vehicle could move autonomously on a field to seed, or enter a mined field, or even perform dangerous missions, the number of individuals put at risk would drastically decrease and at the same time the efficiency of the vehicle itself would be increased due to a 24/7 operational schedule. The key challenge for intelligent vehicles is safety; accidents must not occur due to automation errors and there is zero tolerance to human injury and death.

This chapter focuses on road and traffic applications of intelligent vehicles, which are catalyzing the interest of the automotive industry, car makers, and providers of automotive technologies.

62.2 Enabling Technologies

The basic sensing and actuation technologies for intelligent vehicles are readily available on the market. The key challenge to integrating new technologies is heavily dependent on the control strategies associated with the sensor data processing and reasoning. Automated systems must consider all subsystems in a vehicle, including the interaction of the technology with the driver. Key drivers for the new technologies in an intelligent vehicle are the desired applications. Solutions are developed on the premise that the system must satisfy the requirements of the application with the minimum level of technology.

Intelligent vehicle applications require the following:

- The position, and kinematic and dynamic state of the vehicle
- The state of the environment surrounding the vehicle
- Access to digital maps and satellite data
- The state of the driver and occupants
- Communication with roadside infrastructure or other vehicles.

62.2.1 Vehicle State

Position localization is a key technology for intelligent vehicles. The position of the vehicle must be known if the vehicle is to be controlled along a particular trajectory. To control an intelligent vehicle to perform applications such as collision avoidance or automatic lane changing requires knowledge of the kinematic and dynamic states of the vehicle. Standard robotics techniques for position, kinematics, and dynamics are used in intelligent vehicle applications (Chap. 34). Encoders are mounted on the steering column and the tail shaft. Vehicle heading, speed, and acceleration can be computed from the encoders using standard techniques (Chap. 24).

To execute the trajectories in an intelligent vehicle a number of techniques are used. In many current applications of fully automated vehicles, the trajectories are fixed with limited choices at intersections. To execute the trajectories, the vehicles simply follow markers such as an antenna (a wire carrying an alternating current), magnets, or passive transponders such as radio frequency identification (RFID) tags that are imbedded in the road or simple painted lines. In more advanced navigation systems, the vehicle generates trajectories in a map stored in its memory and localizes itself with respect to this map in order to execute them. This localization can be done using absolute positioning through a combination of global navigation satellite systems (GLS), vehicular kinematic states, and inertial navigation. Alternatively, position localization can be done by relative localization using local markers such visual landmarks. (For specific reference to earlier mobile robot landmark navigation see Chap. 46.)

In order for the vehicle to execute trajectories, it needs to control the speed and the steering of the vehicle (Fig. 62.4). Modern vehicles now incorporate electronic control units to control the acceleration (motor torque and possibly gearbox and clutch), the braking (electric or electrohydraulic brake control), and the steering angle (electric power assistance), and it is, therefore, quite easy to implement trajectory control of the vehicles [62.20]. The safety of these critical functions remains a challenge, and is usually solved through redundancies in the sensors, actuators, and control units.

62.2.2 Environment State

Sensing the state of the environment surrounding the vehicle is a critical aspect of intelligent vehicle applications. The most difficult function for intelligent vehicles is road scene understanding. This includes locating key landmarks: the road, other vehicles, pedestrians, traffic signals, road signs, and other unstructured obstacles. A more difficult challenge is speed control following the detection of an event in the road scene. Com-

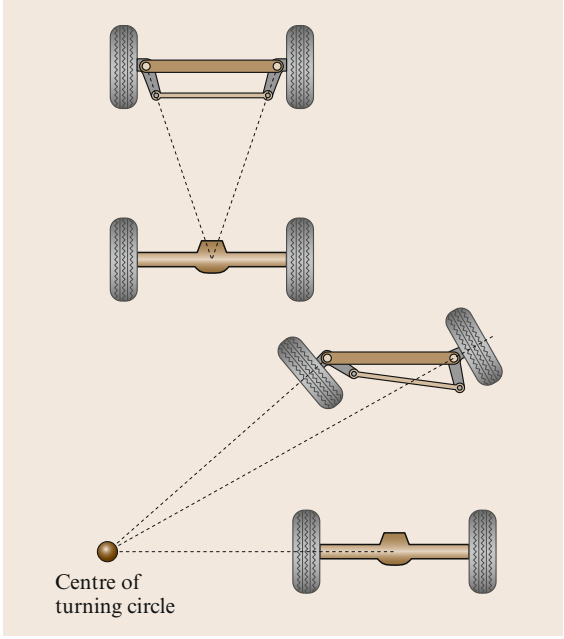


Fig. 62.4 Ackerman steering model

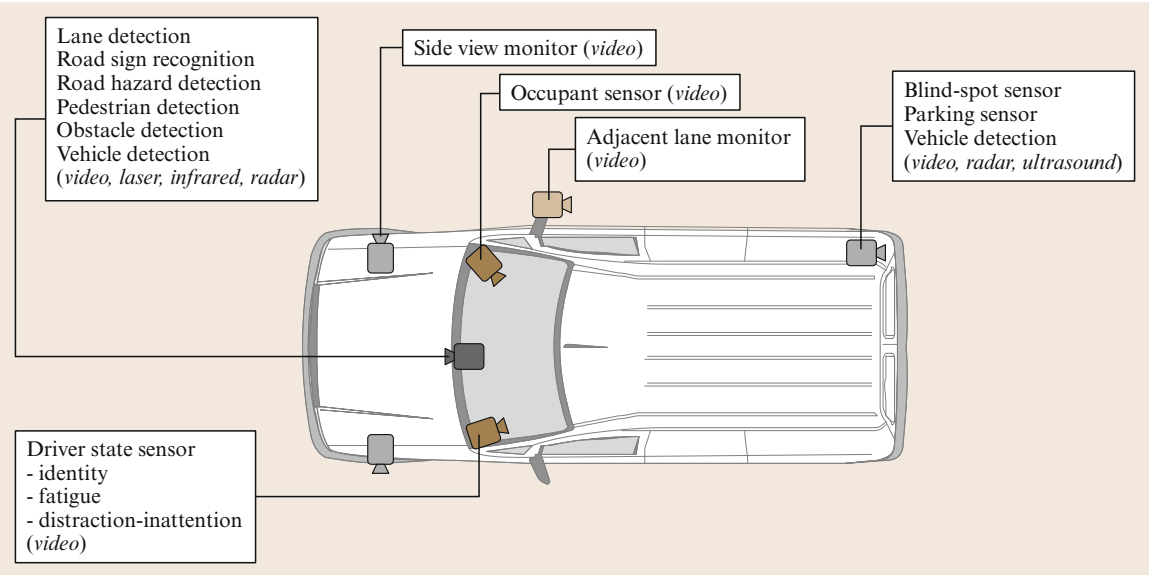


Fig. 62.5 Environment state sensing

mon sensors are infrared [62.21], ultrasound [62.22], radar [62.23], laser range finders [62.24], and computer vision, which continually scan the environment, as shown in Fig. 62.5. Radar is generally used for obstacle detection at a distance, while infrared and ultrasound are used for close proximity obstacle detection. Laser ranging and image processing are used to more robustly recognize the road scene under various weather conditions. Certain road scene conditions such as road signs and traffic lights can only be understood using vision sensing. Sensor fusion is commonly used in intelligent vehicle applications, due to the complementarity of heterogeneous sensor modalities; this can greatly improve perception, for instance, between monocular vision and radar/laser sensors. Sensor fusion (Fig. 62.5) is critical technology for intelligent vehicles and constitutes a wide and active research field for both research teams and automakers (Chap. 35) (Fig. 62.6).

Using such sensors it is possible to map the environment surrounding the vehicle and then, using techniques such as simultaneous localization and mapping (SLAM) (Chap. 46), generate the complex maneuvers needed for parking or for obstacle avoidance.

62.2.3 Digital Maps and Satellite Data

Other additional sensing is generally delivered by global positioning technologies (for example, GPS) to map the vehicle position and refer it to a description of the world known a-priori. This mapping procedure may also be very precise in some areas (like cross country roads) and may be particularly noisy in others (like downtown areas). To fix issues related to multi-path measurements and to fill gaps when the GPS signal is missing, inertial measurements units (IMU) are generally used together with GPS.

Combining GPS with stored digital maps creates a wide variety of intelligent vehicle applications [62.27]. Map data can greatly assist in the problem of road scene interpretation, map data can improve lane detection quality, help deal with the problems when sensors such as cameras do not work, e.g., in sunlight, at dusk, or dawn. Digital maps are widely

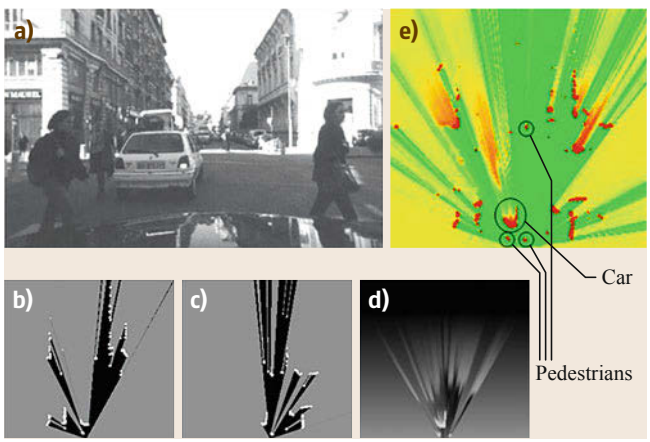


Fig.62.6a–e Multi-sensor fusion on a Toyota/INRIA experimental vehicle (after [62.25, 26]). **(a)** Urban scene as seen from a car, **(c)** occupancy grids, **(d)** lidar, **(e)** occupancy grid from a stereo camera, **(b)** fused data and detected objects

used in commercial navigation systems for route guidance [62.28] and are updated frequently. Such systems will be improved when real-time updating of the map will occur due to the availability of traffic information [62.29]. Additional functionalities to be added include curve approach warnings [62.30], curve speed control [62.31], traffic sign information, and speed limit information [62.32], as well as predetermining an appropriate heading of the headlights in curved and hilly roads.

62.2.4 Driver State

Additionally an intelligent vehicle needs to understand the driver state, if it is to give appropriate warnings or take action. Vision sensors [62.33] can monitor a driver's attentiveness and fatigue by observing the direction of the driver's gaze and eyelid behavior. In an emergency, knowledge of the position of the driver's head as well as all other occupants' position and posture can assist in the safe deployment of airbags. Also, after an accident, observing the state of the driver and other occupants could be useful for the dispatch of emergency services to the accident scene.

62.2.5 Communication

Communication technologies enable interesting applications of intelligent vehicles. Allowing vehicles to communicate with each other and with the highway offers the prospect of vast improvements in the safety and efficiency of the road system. Applications include intersection collision avoidance, emergency braking, and sharing of road and traffic condition information.

The basic communication modes are distinguished as:

- Vehicle to/from roadside infrastructure
- Vehicle to vehicle.

Dedicated short-range communications (DSRC) have been set up to from the vehicle to infrastructure communications to support traveler information, commercial applications (toll/parking fee collection, in-vehicle advertising), and safety applications (intersection collision avoidance, approaching emergency vehicle warning, and rollover warning) [62.34].

DSRC is a wireless protocol specifically designed for intelligent transportation systems. It is a subset of RFID technology, working in the 5.9 GHz band (US) or 5.8 GHz band (Japan and Europe). It is generally implemented with a dedicated protocol, uses short messages, and works in direct line of sight over short ranges (10–1000 m).

Specific applications are being developed to test a broad variety of potential safety and mobility uses of the systems including:

- Warning drivers of unsafe conditions or imminent collisions.
- Warning drivers if they are about to run off the road or speed around a curve too fast.
- Informing system operators of real-time congestion, weather conditions, and incidents.
- Providing operators with information on road capacity for real-time management, planning, and provision of road advisories to drivers.

For intelligent vehicles safety-related applications using DSRC include:

- Stopped vehicle or obstacle avoidance of obstacles on highways.
- Merging assistance from a slow incoming lane into a crowded road.
- Intersection safety and collision warning systems (with or without traffic lights).
- Establishing and maintaining convoys/platoons.

In a vehicle-based intersection collision warning system, each vehicle approaching the intersection generates a message. The message includes the vehicle's location, direction of travel, and the intersection location. This information is based on the GPS position and available map data base. Various strategies to predict driver intention are also being studied, and if available this information is also broadcast. Any other vehicle approaching the intersection can calculate the possibility of a collision and will generate an appropriate warning for its driver.

There numerous other possible applications. These include:

- Notification of the presence or approach of an emergency vehicle
- Notification of construction activity
- Local multi-vehicle sensor fusion to provide better global maps and situation awareness
- Energy saving applications (platooning, energy management for HEVs in stop and go traffic).

The USA, since 1999, and the EU, since 2008, have located spectrum in the 5.9 GHz band for wireless DSRC systems for vehicular safety and intelligent transportation system applications. The standards for DSRC in the US are IEEE 802.11p, finalized in 2010, and wireless access in vehicular environments (WAVE) (IEEE 1609), still in draft status as of July 2010. The 802.11p standard is derived from the 802.11A wireless LAN standard and thus provides a fairly easy and inexpensive route to produce communication electronics

and hardware. These standards provide network and application support, as well as sufficient range and data rates, for both V2V and V2I communication applications.

In addition to wireless communication standards, the development of standardized message sets and data definitions in the US has occurred in the SAE J2735 standards process.

The WAVE standard supports two types of devices, the roadside unit (RSU), and the on-board unit (OBU). The RSU would typically be a service provider deployed on the roadside that would provide map data like the geometric intersection data (GID) message. If it is connected to a traffic light it could also provide signal phase and timing (SPaT) messages. Both message types are defined in SAE Standard J2735 [62.35]. The OBU is responsible for receiving data from RSUs and generating and receiving safety messages that relay information about the vehicle's current state. Many investigations of the utility and capabilities of these standards exist [62.36] and large-scale testing is underway in the US within the *Connected Vehicle* program.

For vehicle-to-infrastructure communications [62.37], a communication protocol architecture is currently being specified by the ISO-SAE Technical Committee 204 Working Group 16 [62.38]. This protocol architecture is known as communication access for land mobiles (CALM) [62.39] and is being implemented in Europe under the cooperative vehicle infrastructure systems (CVIS) integrated project [62.40]. CALM is based on IPv6 protocols developed by the Internet Engineering Task Force (IETF) [62.41], particularly the network mobility (NEMO) protocol, which allows sessions to be maintained between an in-vehicle internet protocol (IP) network and the Internet backbone using any type of available media (3G, general packet radio service (GPRS), Wi-Fi, WiMax, M5, DSRC, satellite, etc.). For vehicle-to-vehicle applications, systems that support ad-hoc networking are still in early stages.

62.3 Road Scene Understanding

Perception plays a key role in any robotic application. In the case of intelligent vehicles, the perception task is referred to as road scene understanding. It involves using different sensors (described in Sect. 62.2) combined with automatic reasoning, in order to create a synthetic representation of the environment around the vehicle. The knowledge base accumulated by this task is then used either to issue warnings to the driver in the case of advanced driving assistance systems (ADAS) or to control vehicle actuators in the case of complete autonomous driving. A complete and precise description of the state of the surrounding environment is the key factor that allows the reduction of the number of false and missed alarms and provides the basis for smooth automatic driving. Needless to say, the perception of an outdoor environment – even if partially structured – is a challenging problem not only due to the intrinsic complexity of the driving environment itself, but also due to the impossibility of controlling many environmental parameters. Figure 62.7 shows examples of day/night, sun/streetlight illumination, temperature, poor visibility, rain/snow, and different meteorological conditions, which in general are impossible to control and have to be faced by sensing devices.

The research community is addressing the issue of providing vehicles with robust and precise perception of the state of the environment from two different perspectives. One approach is to provide vehicles with ever-increasing sensing capabilities and processing power aimed at the provision of powerful onboard intelligent systems; [62.42]. An alternative approach is to use road infrastructure as an active component capable of communicating with all vehicles and sharing information on road conditions in real time [62.43]. Indeed, these

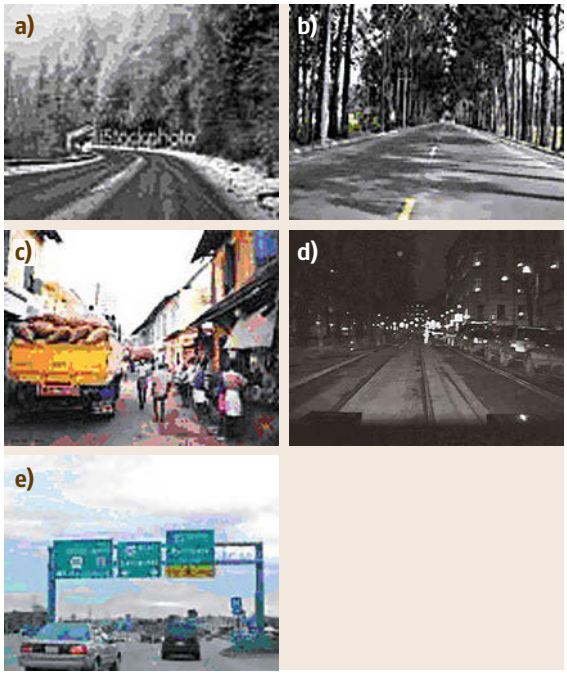


Fig. 62.7 Typical range of road scenes that intelligent vehicles must handle

two perspectives can also be merged to provide a mixed solution to safely control a vehicle and in dynamic environments [62.43].

The task of road scene understanding may be addressed differently, depending on the availability of an intelligent infrastructure and on other players exhibiting cooperative behavior. The task of understanding the state of the environment can be simplified through the availability of information coming from other sources, thereby limiting the need to perform a robust and complete sensing onboard each vehicle. Helpful information could come from the infrastructure itself (for example, road conditions and geometry, number of lanes, visibility, road signs, or even real-time information such as traffic-light status or traffic conditions) or other players (such as the presence of the vehicle with precise position, speed, and direction). The players may also carry real-time information gathered by and shared with other players.

Although research is currently focused on both intelligent vehicles and intelligent infrastructures, the first generation of production intelligent vehicles will have to rely primarily on their own sensing capabilities, since the availability of information coming from other sources such as the infrastructure and other vehicles will take a while to be deployed in real-world situations. In fact, in order to be of practical use, intelligent roads must cover a large proportion of a country, and simultaneously cooperative intelligent vehicles must also be sufficiently widespread. It is important to note that the investment in intelligent infrastructures and intelligent vehicles comes from different sources: mainly from governmental institutions for the former, and vehicle owners for the latter.

The information that is owned by the infrastructure itself and that could be made available to the vehicles includes:

- Precise geometry of the lane/road
- Road signs
- Status of traffic lights.

On the other hand, the infrastructure can also assess and deliver real-time data such as:

- Road conditions
- Visibility
- Traffic conditions.

Another important piece of information that needs to be gathered by intelligent vehicles is the presence of other road players, such as:

- Vehicles
- Vulnerable road users (pedestrians, motorcycles, bicycles).

Although it could be assumed that sometime in the future all vehicles will be equipped with active systems that allow them to be safely avoided by other vehicles, it is quite improbable that pedestrians and bicycles will have similar equipment; their presence will need to be detected using onboard sensors only. The same consideration also applies to obstacles that may unexpectedly be found on the road, or to temporary situations such as roadwork; if a vehicle needs to cope with the unexpected, then it needs to have the capability to assess the situation in real time with its own sensors.

This is why onboard sensing is of paramount importance for future transportation systems; vehicle-to-vehicle and vehicle-to-infrastructure communications may help and improve the sensing, but a complete sensor suite must also be installed on our future vehicles. The main challenges in road environment sensing are examined below.

62.3.1 Road/Lane Tracking

Many vehicle prototypes have been equipped with lane detection and tracking systems, starting from the very first implementations in the early 1980s [62.4–6]. Indeed, in this case computer vision plays a basic role; although generally the road can also be detected with laser scanners [62.24], the only generic technology able to detect lane geometry and lane markings with high precision is computer vision. Most lane tracking approaches have focused on detecting lane markings and exploiting structure in the environment, such as the parallelism of the left and right lane markings, the invariance of road width, or the widely used flat-road assumption. These assumptions were mainly used to overcome the problem of having a single camera (a choice driven by cost). Some systems use stereo vision to detect lane markings and are able to work without such constraints. The problem of lane tracking in highway situations is basically a solved problem – with commercial systems being deployed in passenger and

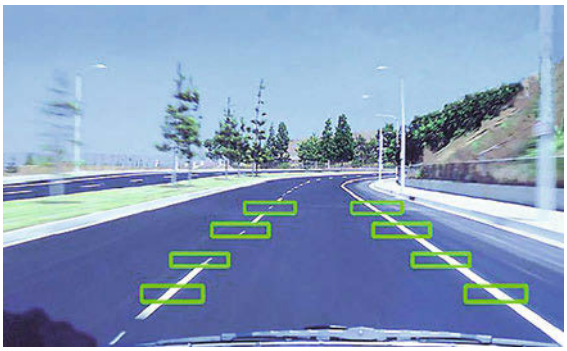


Fig. 62.8 Lane detection

commercial vehicles. A survey and evaluation of the deployed lane tracking technologies is available [62.44]. An example of the typical output of a commercial lane tracker is shown in Fig. 62.8.

However, such systems cannot guarantee that lane detection systems will work with 100% reliability; these systems typically work with 95–99% reliability. Therefore, lane tracking systems are only being used in lane departure warning systems, since no failures can be tolerated for autonomous driving. Efforts are underway to develop algorithms that will tolerate a variety of driving conditions, and push the 100% reliability boundary [62.44, 45], VIDEO 836.

Vision-based lane tracking systems depend on lane markings, which are not always present. In these cases it is often not enough to track the lane or road's limits. Instead it is necessary to build a local representation of the road's shape and traversability. This local map can be built using SLAM-like techniques (Chap. 46), and can classify parts of the map as either static or dynamic [62.46]. Alternative approaches aim to classify 3-D points obtained from laser scanners or stereo cameras as belonging to the road or not [62.47]. The output of the road detection function is a continuously updated representation of the *navigable space* in front of the vehicle (Fig. 62.9).

62.3.2 Road Sign Detection

Another fairly straightforward use of computer vision is road sign detection and understanding. Road signage is deliberately structured to aid human drivers. Road signs use a set of well-defined shapes, colors, and patterns. The signs are placed at consistent heights and positions in relation to the road. Therefore, reading road signs is an achievable task for computer vision. Detection is done using a collection of shape and/or color detection schemes [62.48, 49]. After the detection and localization phase, recognition takes place. Normally this task is performed by pattern-matching techniques such as image cross-correlation, neural networks, or support vector machines, since the possible set of road signs is limited and well defined. Figure 62.10 illustrates the concept of a speeding warning system based on speed sign detection. The challenge for research work in this area lies in the robustness of detection and the reliability of classification of signs. Most automotive companies have systems under development, e.g., Siemens [62.50]. See video on speed sign detection VIDEO 838.

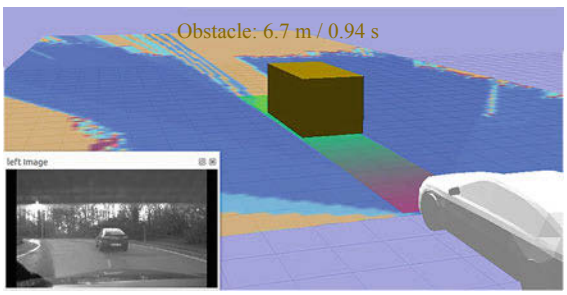


Fig. 62.9 Instantaneous representation of the navigable space in front of the vehicle (after [62.25])

62.3.3 Traffic-Light Detection

Color and pattern matching are also the key techniques used for the detection of traffic lights [62.51]. Although the detection of traffic lights is not overly complex, this application hides one further aspect that makes vehicular applications difficult: besides the correct localization and recognition of a signal, special care has to be taken in checking the signal position and orientation on the road/lane since that signal may not be addressed to the current vehicle. This is particularly true in downtown intersections at which many traffic lights are visible at the same time; in this case the vehicle must have the capability to select the correct traffic signal that must be obeyed. Some experiments have been undertaken with active traffic signals, able to emit the status of the traffic light using radio frequencies [62.52]. This involves additional infrastructure; at this stage vision seems to be the only simple viable solution.

62.3.4 Visibility Assessment

One of the key challenges is the detection of fog. The meteorological visibility distance is defined by the International Commission on Illumination (CIE) as the distance beyond which a black object of an appropri-

Fig. 62.10 Road sign detection for speed warning application

ate size is perceived with a contrast of less than 5%. Different techniques for measuring this parameter – and thus detect foggy conditions – have been implemented [62.53]. Although many of the methods use vision, there are also efficient alternatives – generally used in fixed locations such as airports and traffic-monitoring stations – based on the use of multiple scattering lidar. The main challenge of using vision to estimate visibility is that a moving vehicle generally cannot rely on a specific reference point/object/signal at a specific distance.

62.3.5 Vehicle Detection

The detection of vehicles has been addressed using a large variety of sensor technologies, ranging from vision to lidar, from radar to sonar.

Despite being different in shape and color, vehicles share the same characteristics and feature a large size and reflective material. The position of vehicles is predictable once a rough indication of the road/lane position is available. Vehicles, in fact, can be successfully detected by many different sensors independently [62.55–57]. Figure 62.11 shows a vision-based vehicle detection system.

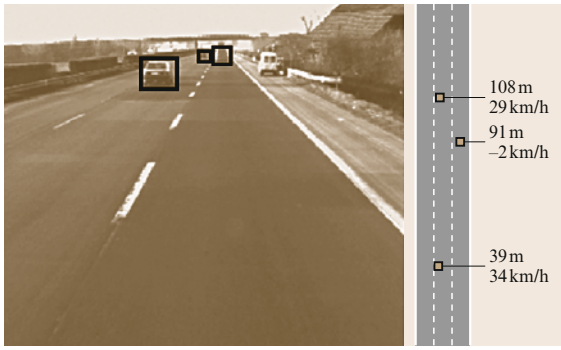


Fig. 62.11 Vehicle detection – lane and range position

Nevertheless, although the solution to this problem seems straightforward, each sensor has its own domain of application and its own challenges. Vision is generally powerful, but may fail in low visibility and bad-illumination scenarios (night or tunnels) or in heavy traffic conditions when vehicles may occlude each other. Vision in the infrared domain (thermal imaging) is able to detect vehicles with a high confidence since vehicle tires and mufflers generally exhibit high temperatures and are, therefore, easily detected in the image. However, parked vehicles, trailers, and even vehicles which have just started to move are colder than running vehicles and, therefore, less visible. Lidar is generally robust, but has decreased sensitivity in adverse weather conditions. Radars, while cheap, can suffer from bias in lateral measurements due to the presence of other nearby reflecting objects. Finally, sonars are applicable only for very short distances. The research challenge is to implement multisensor fusion robustly. A common approach is to fuse vision with radar [62.58].

Vehicle detection is often performed by grouping together data that are believed to belong to a same object, this often implies using some kind of similarity metric based in distance and other relevant features such as color, and texture, etc. An alternate approach is taken by *model based* techniques, which scan scene data trying to match models from a preexisting catalog – e.g., geometric descriptions, part collections. Finally, *grid-based* approaches do not directly model objects themselves, but the evolution of the properties in a regular decomposition of the space (Fig. 62.12) [62.54].

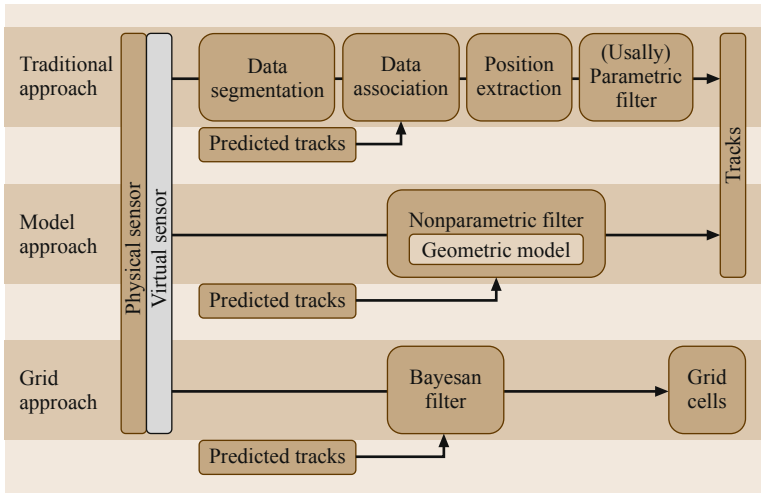


Fig. 62.12 Three alternative approaches to vehicle detection and tracking (after [62.54])

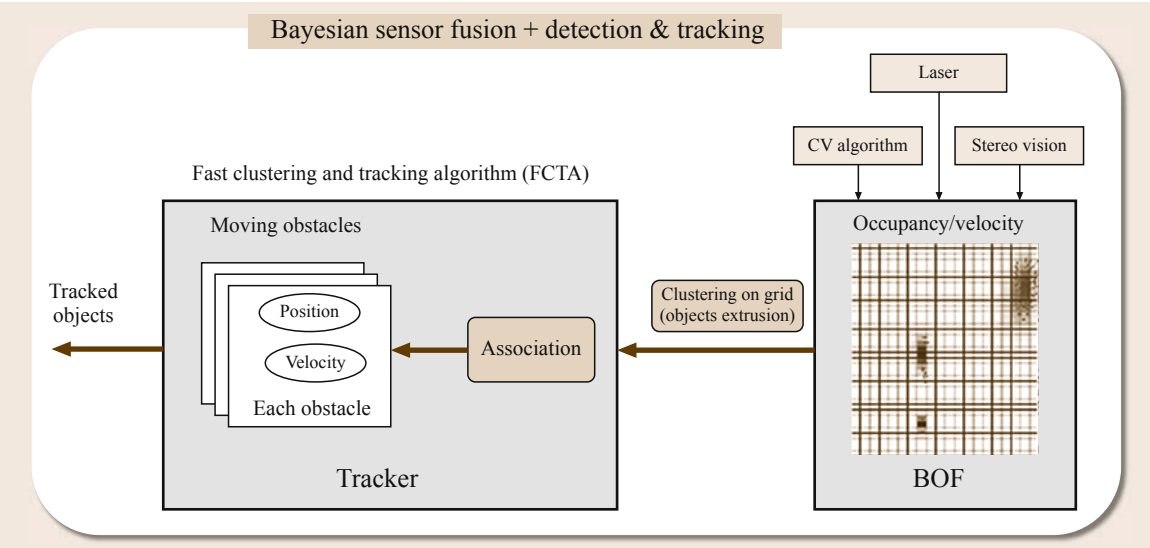


Fig. 62.13 Grid-based detection and tracking (after [62.59])

One of the main problems in tracking is *data association*: finding the correspondence between sensor readings and tracked objects. Tracking approaches differ in the way they solve this problem from the *nearest neighbor*, which just assigns the observation to the closes object to multi-hypothesis tracking (MHT), which maintains a representation of the different association hypothesis for several time steps – pruning is necessary for tractability. An increasingly popular alternative is the use of grid-based approaches, which do not attempt to explicitly represent objects during the iterative grid updates while taking into account all the possible associations with cells in a given neighborhood. This approach leads to postponing as long as possible the object detection and data association steps [62.54]. Refer to Fig. 62.13 for an example of grid-based tracking. VIDEO 420 shows the predicitive capabilities of the Bayesian occupancy approach [62.59].

62.3.6 Pedestrian Detection

The detection of vulnerable road users (pedestrians and bicycles) is one of the most difficult tasks for intelligent vehicles. The appearance of a pedestrian is challenging: a pedestrian shape can change greatly within a few tens of milliseconds; there are no clear invariants in color, texture, or size, and no assumptions can be made about posture, speed, or the visibility of parts of the human body such as the head. Machine learning methods have been successfully applied to this problem [62.24, 42, 48, 60]. The focus is on achieving greater reliability and reduction in false alarms through improved vision algorithms [62.61]. However, the detection of vulnerable road users is one of the most relevant research topics worldwide since a great number of benefits – including insurance reductions – may be achieved once a fully functional pedestrian detector is available on cars. Countermeasures may be activated to reduce the consequences of vehicle–pedestrian accidents, such as the firing of external airbags or the opening of the hood to lessen the impact of a head-on collision. Currently, with all possible technologies under evaluation, no solution seems to offer reliable detection in every scenario: radars are not able to detect pedestrians reliably in crowded scenarios, while vision has the many drawbacks listed above. Even thermal imaging, which – although still very expensive – is generally believed to be one of the most promising technologies, fails in some situations, such as hot summer days and, in general, in high-temperature environments. See video for vision based stereo vision detection VIDEO 839.

62.4 Advanced Driver Assistance

Given the legal liabilities and technical challenges of achieving 100% reliability for autonomous intelligent vehicles, it appears likely that motor vehicles will have pieces of autonomous functionality added progressively and that cars will eventually evolve into autonomous robots. The perception techniques described earlier can

be used in a variety of ways to make driving safer, more efficient, and less demanding. Individual perception techniques, or combinations of sensing modes, are being used to provide warnings to drivers of dangerous situations. These warnings are being used to prevent collisions in a variety of situations, such as when backing up, when leaving a roadway, into rear ends, on lane changing/merging, with pedestrians, and at intersections.

Developing a collision warning system requires many steps beyond building the perception system. For a typical example, roadway departure warning, the steps followed in a program initiated by the US National Highway Traffic Safety Administration include:

1. Statistical studies. In the United States, crashes involving a single vehicle leaving the roadway are relatively rare, but disproportionately dangerous; approximately 40% of the 40 000 fatal crashes per year in the US are single-vehicle crashes where the vehicle leaves the roadway. The first part of the study looked at the prevalence of those crashes, and determined that this would be a good type of crash to prevent.
2. Causal factors. The second step was to determine the causes of those crashes. Most run-off-road crashes are due to driver factors, such as excessive speed, inattention, or loss of control. This is an important observation, because it means that alerting the driver, or warning of difficult situations, could prevent those incidents. For the fraction of crashes caused by mechanical failure a warning system would not be useful; in this type of crash, mechanical failure is involved in less than 5% of the crashes.
3. Opportunities for intervention. This part of the study set out to determine whether a warning system could be effective, and, if so, how far in advance the warning would have to be given. Given typical road departure trajectories, typical widths of roads and shoulders, and the range of potential steering responses, this task generated requirements for how accurately the system would have to track vehicle trajectory in order to predict a roadway departure.
4. Human factors. Since the system being designed is a warning system, rather than an active control, it is crucial to understand what kinds of warning a driver (who may be distracted or sleepy) would respond to, and how quickly and accurately the response will be. Reaction times vary widely across individuals: using 1 s for reaction time is a fairly standard estimate.
5. Simulator studies. A driving simulator is like an aircraft flight simulator for cars or trucks, with a variety of simulated roads and conditions. Simulator studies were used to test driver response to warnings: directional or nondirectional audio warnings, steering wheel shakes, and combinations. A nondirectional audio cue worked best.
6. System specification development. Based on the preceding steps, in order for a system to be useful, it needs to work at day and night, in almost all weather conditions, it needs to measure vehicle speed, lateral position relative to the road, lateral heading, and roadway curvature, and predict future vehicle trajectories long enough in advance to trigger a warning alarm.
7. Perception and system development. Given those specifications, there are several ways that a perception system could be built to sense the road and the vehicle's trajectory relative to the road. For this particular test, a lane keeping system – the Rapidly Adapting Lane Position Handler (RALPH) was developed and tuned [62.62].
8. Limited tests. The entire system – sensors, processing, driver interface – was built and tested with a limited number of volunteers, and the system tuned and validated.
9. Full-scale operational tests. The system was deployed in test fleets, including long-haul trucks and passenger cars.

62.4.1 Collision Avoidance and Mitigation

The complete cycle from the idea of using perception to prevent crashes, to full system development, took over 10 years in this case. The pure *robotics* part of the system is a crucial element, but is only one piece of the development needed to make a useful product. Some active control has already been assumed by today's vehicles. Antilock brakes have been on the market for many years. Traction control systems which control throttle to stop wheel spin are being introduced. Electronic stability control systems take this the next step further, controlling throttle and individual wheel brakes to help in cornering performance. So, gradually, people are willing to cede some control to very reliable automated systems. We can expect this trend to continue.

Each collision warning type has its own list of specific development challenges, as described below.

Backup Collisions

The sensing challenge is to see relatively small objects, such as fence posts or children's toys, while not picking up false alarms from pavement joints or leaves and debris. The sensors used in today's commercial vehicles are piezoelectric ultrasonic sensors, which are inexpensive [62.22]. However, ultrasonic sensors have

well-known limitations. The challenge of developing low-cost, accurate, reliable sensors remains.

Rear-End Collisions

These are among the most difficult collisions to prevent, with the most challenging sensing conditions. Rear-end collisions often happen at high speeds, requiring long-distance sensing of other vehicles (up to 100 m at US highway speeds, much longer at the high speeds found on some European roads or for the longer braking distances needed for heavy trucks). That in itself is not too demanding a challenge: the sensed objects in this case are relatively large and have high metal components, so radar and lidar are both feasible sensing modes. The biggest range-sensing challenge is sorting out true targets (slow or stopped vehicles) from false targets (overhead signs or bridges, and side lobes from strong reflectors on the side of the road). It is also important to determine if the sensed vehicle is in the same lane as the smart car, or a different lane. Sensing lane markings at such a large distance is a very difficult challenge; merging lane sensing (often done by vision) with obstacle sensing (by a different sensor) and registering the two to within the resolution of a lane is a daunting task. This technology remains under development through industry and government programs [62.63].

Lane-Change/Merge Collisions

In the simplest case, the countermeasure to this kind of collision involves short-range sensing to cover the *blind spots* on the rear corners of a vehicle, where it is difficult to see with mirrors. For passenger cars, this area is quite small, and can be covered with a single sonar or radar. Often, the user interface is a warning light placed in the rear-view mirrors [62.64]; this reinforces good driver behavior of checking mirrors before changing lanes. The sensing challenge for heavy trucks or transit buses is the same as for cars, except that the area not visible in planar mirrors can be much larger. Figure 62.14 shows examples of blind-spot detection for cars and heavy vehicles.

The usual solution is a row of sensors along the side of the vehicle, although scanning lidar or panoramic vision have been used in experimental applications [62.65]. The further complication for lane-change warnings is in high-speed driving, where it is important to look not just adjacently to the vehicle but a long way to the rear, to find overtaking vehicles with high relative speeds. The first commercial product was brought to market in 2007 [62.66].

Pedestrian Collisions

Pedestrians are particularly important to detect, because pedestrians are much more vulnerable than people in vehicles; as discussed earlier they are also, unfortunately, relatively difficult to detect and very hard to predict. Just detecting a pedestrian is not sufficient. In transit operations, for instance, a bus operates close to pedestrians much of the time. To do meaningful collision warning, it is important to detect the pedestrian, detect their current path, look for cues such as crosswalks or curb edges that modify the probability of the pedestrian's trajectory, and match all of these factors with the predicted trajectory of the vehicle. It is crucial to tune the warning system to produce few false alarms while not missing real alarms. A particularly dangerous situation is pedestrians slipping and falling underneath a bus: these are very dangerous situations, but very difficult to detect in time to warn the driver. For these reasons, automotive manufacturers have worked on products such as night vision to enhance driver perception [62.67]. Car manufacturers are bringing to market products that use a data-fusion approach combined with radar-vision for emergency breaking in pedestrian collision situations [62.68]

Fig. 62.14 Blind-spot detection

Other Obstacle Collisions

Vehicles have collisions with many things other than other vehicles and pedestrians, for example, animals (deer, dogs, cats), car parts (tire carcasses, rusted-out exhaust systems), cargo that falls off trucks, construction debris, etc. Warning drivers about these kinds of objects on the roadway is a challenging task. A commercial product for safety systems detects large animals

Fig. 62.15 Animal detection

(Fig. 62.15) using vision and radar sensors [62.69]. A piece of construction timber on the roadway may be large enough to do significant damage to a car, but be small enough to be difficult to see, and be invisible to radar. Some interesting work has been done with high-resolution stereo vision [62.70], with polarimetric radar [62.71], and with high-resolution scanning laser range-finders [62.72]. However, in general, this remains a difficult problem.

Other Actions

Besides warning the driver, there are other actions that an intelligent vehicle can take short of assuming control. If a collision is inevitable, particularly from the side of the vehicle where there is limited crush space, the system can brake and deploy airbags even before physical contact. Of course, such a system would have to be nearly 100% reliable. More simply, if the system senses an imminent front collision, it can preload the power brakes, saving fractions of a second in brake reaction time. The driver must still actuate the brakes, but the onset of hard braking can be much quicker. At 100 km/h, a 0.1 s saving in braking actuation saves approximately 3 m of stopping distance, which can be the difference between a severe rear-end collision and a much lighter crash. Such systems are being introduced into the high-end market by all the major automotive manufacturers.

Collision Avoidance

The next step beyond emergency braking is an automated system. Such systems have several advantages over a human driver: they have much quicker reaction times, access to sensors such as individual wheel speeds and slips plus external sensors such as radar or lidar, access to individual brake controls and other controls, and so forth. So, if the system had ideal situational awareness, it could, in many cases, do a better job of avoiding a collision than a human could. This is still a very difficult area for implementation, however. First of all, the human has access to higher-level knowledge: the driver may be watching the behaviors of other cars, may make eye contact with pedestrians or drivers, may be watching a policeman directing traffic, etc. So it may be that a maneuver that, to the system, looks like the best way to prevent the collision, is actually the wrong action to take. Secondly, in most countries, as soon as the vehicle takes control there is a shift in the liability for any resulting crash from the driver to the manufacturer. So there is a great reluctance to take active control. An alternative, recently developed approach is to observe the environment state with lidar, and monitor the dynamic vehicle state to determine whether the accident is unavoidable. If the driver can no longer take corrective action, that is, brake or turn away safely, then emergency braking occurs [62.24, 73]. For now, however, active collision avoidance remains a research area, with significant questions of reliability, human factors, and liability.

Traffic Situation Awareness

Traffic situation awareness is a higher level of understanding that is required for decision making in traffic situations. For instance, knowing that a vehi-

Fig. 62.16 A collision warning system for intersection safety implemented on communicating Renault experimental vehicles using the intention/expectation approach (after [62.26])

cle is traveling in the wrong direction along a street is much more informative for estimating the risk of the current situation than knowing its velocity. Thus, traffic situation awareness is emerging as a new area of research effort to understand and reason about the future motion of pedestrians and vehicles. The obtained predictions can then be fed to a motion planning algorithm, which can use this information to minimize collision probabilities [62.74, 75]. Motion models can be used to infer the maneuver intentions of drivers to change lanes [62.76], overtake another vehicle [62.77], make a turn at an intersection [62.78], or comply with the traffic laws at an intersection [62.79]. VIDEO 566 shows how an embedded Bayesian perception system have been implemented on an experimental vehicle.

Intersection Collisions

Intersection collisions are particularly difficult to prevent because they often involve challenging sensing scenarios. Many of these collisions involve occluded vision, with lines of sight blocked either by large vehicles or by adjacent buildings. They also often involve high closing rates from oblique angles, making it necessary to see a long distance with a very wide field of view. The solution that is usually proposed is to add intelligence to the infrastructure, either in fixed sensing (such as radars looking down each approaching road) or in some kind of radio relay that takes data from approaching smart cars and passes it to other approaching vehicles. All of these solutions are challenging: the large number of intersections makes it difficult to envision any universal solution.

A complementary approach to traffic situation awareness is *risk estimation*, where the objective is to determine how dangerous each potential maneuver is for the vehicle. The classic approach is to use motion models to predict the possible future trajectories for all the dynamic objects in the scene, and then compute the probability of a collision between each possible pair of trajectories for each object. Popular approaches for generating possible future trajectories include rapidly-exploring random trees [62.80], Gaussian processes [62.25, 81], and stochastic reachable states [62.82]. A major limitation of this classic approach to risk estimation is its computational complexity. A recent approach to risk estimation has been proposed, which detects conflicts between the *intentions* of drivers and the *expected* behavior of the vehicle, rather than by detecting collisions between possible future trajectories [62.26, 83] (VIDEO 822). More specifically, dangerous situations are identified by comparing *what drivers intend to do* with *what drivers are expected to do according to the traffic rules*. This formulation of risk reflects the fact that most accidents are caused by driver error [62.84], and matches the intuitive notion that *dangerous* situations are situations where drivers act differently from what is expected of them. This approach can in theory be applied to any traffic situation, but so far has been implemented and tested for scenarios involving cars at road intersections only (Fig. 62.16). Significant research projects are underway in the EU for cooperative vehicle infrastructure projects such as managing intersections [62.85].

Currently, automotive manufacturers are dealing with traffic situation awareness by specifically focusing on discrete driving tasks and typical scenarios. Combining perception with control gives partial automation for specific tasks, such as adaptive cruise control, lane keeping, assisted parking, and slow driving in stop-and-go situations.

62.4.2 Adaptive Cruise Control

Adaptive cruise control (ACC) is the logical extension of standard cruise control to also include keeping a safe distance from the preceding vehicle. If there are no vehicles in front of the smart car, it follows a set speed, as with standard cruise control. If a slower-moving vehicle is in front, an ACC-equipped car will sense the vehicle using radar or lidar, and slow to maintain a safe distance (typically set to a 1.5–2 s following gap). Figure 62.17 shows an illustration of the ACC concept. The sensing challenge for ACC is much easier than the challenge of rear-end collision countermeasures, since ACC systems are only designed to deal with other moving vehicles.

The biggest sensing difficulty for rear-end collision countermeasures is separating stopped vehicles on the road from objects off the road; for ACC this difficulty is bypassed by ignoring all stopped objects. Moving objects are classified as in-lane or out-of-lane based on

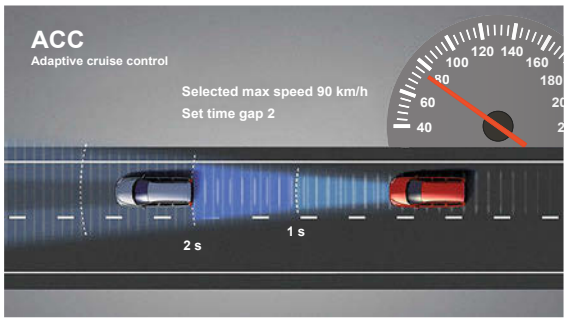


Fig. 62.17 Advanced cruise control (ACC)

a number of heuristics. Often, the smart car's own steering radius is used as an estimate of the road curvature ahead, in order to determine whether vehicles ahead are in the same lane. Since the systems are explicitly sold as *convenience* instead of *safety* systems, they only need to deal with normal situations with relatively low differences in velocity, and they leave the more difficult situations up to the human driver. The human is still alert, controlling the steering, and watching the traffic. These systems are being introduced by all the major car manufacturers.

62.4.3 Stop and Go

Stop-and-go driving assistance (also referred to as low-speed ACC) is at the opposite end of the speed spectrum, when vehicles creep along in dense traffic. At slow speeds, it is easy to track the vehicle ahead, and to move when it moves, steer when it steers, and stop when it stops. If the traffic accelerates to a modest speed, the stop-and-go system disengages, and the human must assume control of the throttle, brake, and steering. Since the speeds are very slow and the distances are short, many different sensing systems will work, such as stereo vision, radar, and lidar [62.86, 87].

62.4.4 Parking Assistance

Parking assistance is also a low-speed aid. In a typical scenario, the driver initiates the system by pushing a button when driving past an empty parking space. The system measures the length of the space using odometry, measures the positions of the cars in front and in back using short-range sensors, and infers the position of the curb by assuming that the surrounding cars are standard-sized cars parked near the curb [62.88]. Figure 62.18 illustrates parking assistance and fully automatic parking [62.88] (VIDEO 567).

Once the system is fully engaged, it takes over steering, planning, and executing the ideal parallel park steering sequence. In some systems, the human is still responsible for throttle and brake, again ensuring that the human is alert, watching for encroaching pedestrians or other obstacles. Such a system has been introduced by Toyota [62.89].

62.4.5 Lane Keeping

Lane keeping assistance is the natural extension to road departure warning systems. Given a lane tracking system, it is straightforward to add control of the steering wheel to keep the vehicle centered in its lane [62.4–6]. This has a number of uses. Some cities would like to run transit buses on narrow roadways, for instance the shoulder of highways or through narrow streets in old cities. Automated lane keeping systems using mechanical guideways are in use in several places [62.90]. It is easier and less expensive if such systems can be electronic rather than relying on specially installed mechanical guides. A specific subcategory is precision docking: for a transit bus to pick up a passenger in a wheelchair, either the bus must deploy a special ramp (which is a slow process), or it must pull up to a level dock and leave a very small gap, so that the wheel chair can safely roll on or off. Short-range precision docking systems use either a downward-looking sensor, looking at painted lines or magnetic markers, or a sideways sensor looking at the curb or dock, in order to guide the bus to its parking spot. Finally lane-keeping assistance is a convenience for driving on highways, especially with gusty winds. Honda has released a vehicle equipped with both lane-keeping assistance and adaptive cruise control (ACC) [62.91]. The danger with such systems is that the driver no longer has an active moment-to-moment role, and may lose concentration or even fall asleep. These systems are not designed for full automation, and still require a driver to handle unusual

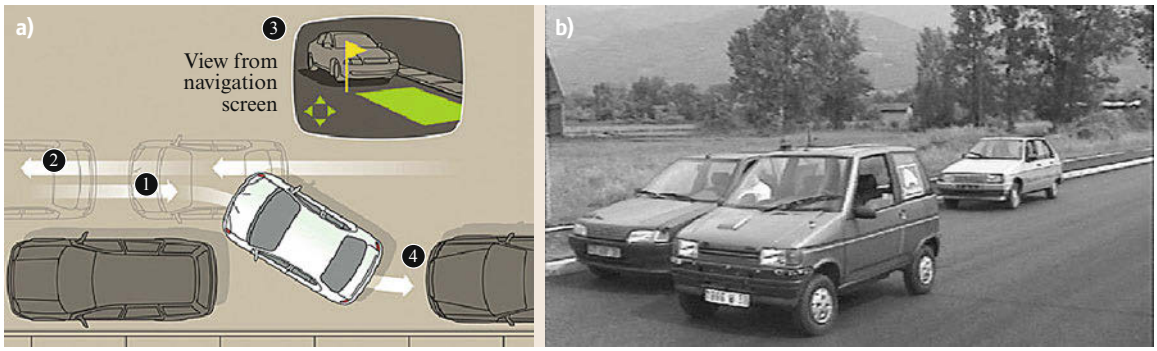


Fig.62.18a,b **(a)** Parking assistance and **(b)** fully automatic parking

circumstances. The next stage is to integrate driver state monitoring. If the driver is inattentive, then all automatic systems are disengaged.

62.4.6 Lane Changing

Lane changing assistance is the next extension in partial autonomy. It combines lane keeping and ACC with blind-spot monitoring. If the car can safely overtake a vehicle, then a lane change is undertaken and speed is unchanged [62.92]. Otherwise, the ACC slows the vehicle down. At its simplest, such a driver assistance system can advise a driver whether a lane change can be safely undertaken [62.93]. In its most advanced form the lane change is undertaken completely automatically by the vehicle [62.4]. Such systems require an additional side-facing sensor, typically radar (shown in Fig. 62.19)

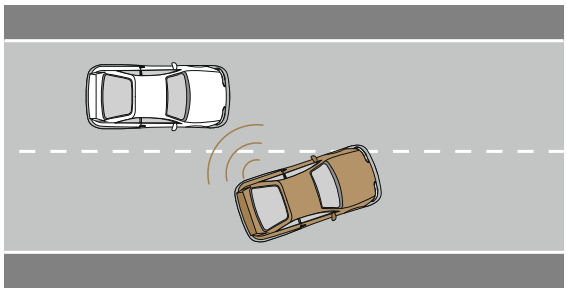


Fig. 62.19 Lane changing – side detection

62.5 Driver Monitoring

There has been an evolution of thought regarding the role of the driver in intelligent vehicles. The grand goal has been to replace the driver with a fully automated system. As discussed earlier, full automation of intelligent vehicles is still some years away due to system reliability and legal liability reasons. The next step in the development of motor vehicles is partial automation – where individual autonomous functions such as ACC, lane keeping, lane changing etc., are developed. Motor vehicle designers have realized that the driver cannot be removed from the vehicle and must instead be supported by systems onboard a motor vehicle. Over 92% of motor vehicle accidents are caused by driver error [62.94]. It is likely that the next generation of intelligent vehicles will work in the following way:

1. The vehicle will monitor the road scene using the advanced driver assistance system technologies discussed earlier to assess the state of the environment and warn the driver of dangerous situations, e.g., lane departure warnings.
2. The vehicle will also monitor the driver using vision and other physiological sensors to assess the state of the driver. If a driver is fatigued, drowsy, inattentive, distracted, under the influence of drugs, or has an emergency health condition such as heart attack, then accidents can occur. The vehicle warns the driver of dangerous circumstances, e.g., drowsiness warning.
3. For legal liability reasons, intelligent vehicles will not take control, rather the driver will be alerted using visual, audio, or tactile warnings. The vehicle will not perform collision avoidance; rather collision mitigation will occur through emergency braking.
4. If an accident is unavoidable, the vehicle can autonomously apply emergency braking. To maximize the safety of the occupants in the vehicle, seat belt restraints are tightened and airbags are safely deployed.
5. After an accident has occurred, knowledge about the state of the driver and the passengers is important. If an occupant has been injured, a call to the emergency services can be dispatched automatically by the vehicle.

In all the steps described above monitoring the driver is critical. For future advanced driver assistance systems (ADAS) to work safely the driver should be put in the loop, for example, in lane departure warning systems, it is not possible to determine whether a vehicle departing from a lane is due to driver intention or error. If the state of the driver is being monitored, and the system can detect that the driver's eyes are closed or the driver is looking away from the road, then it can be inferred that the lane departure was involuntary and that a lane departure warning should be given to the driver. For ADAS to be accepted by drivers, the systems should not give false warnings. If the driver is looking directly at the road then lane departure warnings should not be given (or a different subtle warning should be given). Similarly more sophisticated systems, such as lane keeping, should not be engaged unless the driver is fully attentive and has his/her hands firmly on the steering wheel. The key point is that drivers must be fully engaged with and in control of the driving task.

This is a most important consideration in the design of ADAS for intelligent vehicles. A good overview of the challenges associated with putting drivers into ADAS systems is given in [62.95]

Combining perception with control gives partial automation for specific tasks, such as adaptive cruise control, lane keeping, assisted parking, and slow driving in stop-and-go situations.

62.5.1 Driver Fatigue, Inattention, and Impairment

By directly monitoring the driver using visual sensing opens up the possibility of developing a new class of ADAS applications. It is possible to monitor driver state through monitoring signals such as an electrocardiogram (ECG), temperature, etc. However, market studies by automobile manufacturers have shown that people do not like to have wires or gadgets attached; driver monitoring must be noncontact and noninvasive. The only solution is to use vision as the sensing medium. The technical challenge to develop a computer vision system that can automatically detect a driver of any age, sex, race, with/without eye or sunglasses, and with/without facial hair is enormous. Recently, significant progress was made with systems being developed that can also detect where a person is looking (gaze direction) [62.33, 96].

Once the driver's state (head position, eye gaze, eye blink rate) can be measured, then ADAS applications can be developed. Figure 62.20 shows the output of a commercial driver state detection system. See video for driver distraction, inattention and fatigue detection VIDEO 840.

Driver Impairment

Safety authorities estimate that as many as 50% of all road accident fatalities are caused by driver impairment due to alcohol or drugs [62.97]. Recent research has shown that driver impairment can be detected by sensing abnormal scanning patterns of eye gaze. It promises to open up a new class of ADAS. There have been major education and legislative initiatives in many Organization for Economic Cooperation and Development (OECD) countries, resulting in a significant reduction in road fatalities. However, the difficult cases (fatigue, distraction, and inattention) have become more prominent. ADAS technologies could have a significant impact in further reducing the road toll.

Fig. 62.20 Driver state detection

Driver Fatigue

Safety authorities estimate that 25–30% of all road fatalities are caused by driver fatigue [62.98, 99]. Research has shown that there are four visible factors that indicate the onset of fatigue – prolonged eye closure, uncontrolled head moments, drooping eyelids, and reduced eye-gaze scanning. Systems are under development that focus only on eye closure [62.100]; the challenge of fusing all four factors together into a robust algorithm that works for a wide range of drivers remains an open research problem.

Driver Inattention

Safety authorities estimate that up to 45% of all traffic accidents – from minor dents to serious incidents – are caused by driver inattention or distraction [62.101]. Research has shown that if the drivers consistently keep their eyes on the road, then driving becomes a much safer experience [62.102]. All major automotive manufacturers are developing ADAS applications that warn the driver if they are distracted from the driving task. More sophisticated ADAS under development use the driver's gaze direction to check whether safe driving practices are being followed, for example, did the driver check the side mirror before a lane change was performed? If the driver fails to check the side mirror a warning would be issued. Systems of this type have been demonstrated [62.103].

Driver Workload

The increase in new electronic systems and gadgets that are being installed into today's motor vehicles is also another source of distraction. Questions arise about when and under circumstances should a driver change a compact disc (CD) or answer a phone call. Research is underway into the development of workload systems. These systems take into account the vehicle state (speed, acceleration, braking, gear-change yaw rate, etc.) to determine whether information management tasks such as answering a phone call, sending a short message service (SMS) message are allowed [62.102]. The next stage of research is to include state infor-

mation about the road scene and the driver. If the car is driving on a country road, and the driver is attentive, then distractive tasks could be allowed and managed. Systems for monitoring the scene for monotony have been combined with fatigue detection systems – and can be combined with information management tasks [62.104]

62.5.2 Driver and Passenger Protection

The automotive industry is moving towards the use of active safety systems such as airbags to complement passive systems such as seat belts. ADAS will play a critical role in active safety systems. In emergency braking or impending rear-end collision situations, seat belts can be pretensioned and airbags primed. Drivers and passengers could be further protected through the development of smart airbags [62.105]. Airbags, while considered an essential part of a modern car, can cause fatalities if the occupant is too close to the airbag, is a child, or is not wearing a seat belt. Smart airbags are dependent on the position of the vehicle occupant's head. The driver state monitoring technologies discussed earlier are an essential component of smart airbags.

62.5.3 Emergency Assistance

The speed at which people are given treatment after a serious accident is critically related to the survival rate of victims. While it is expected that ADAS will lead to a reduction in road accidents, it will not eliminate accidents. Therefore, it is important that automated systems for emergency situations are developed. Systems that use GPS, vehicle state information, and mobile communication systems have been developed that send the world coordinates of the vehicle to emergency authorities after an accident [62.106]. This information can be augmented by using driver monitoring technology to assess the condition of the occupants of the vehicle. Such types of system are referred to as Telematics are currently being tested by major automotive suppliers, including the development of aftermarket products [62.107].

62.6 Towards Fully Autonomous Vehicles

Intelligent vehicles that are fully autonomous can be justified for safety, traffic congestion, and environmental considerations.

62.6.1 Operating Safely

The first problem is safety. As stated earlier, automobile accidents are one of the main causes of human fatalities – on a staggering scale. This is a catastrophe of larger magnitude than all armed conflicts since the beginning of mankind. The most economically advanced countries (OECD) have been able to reduce the number of fatalities significantly through improved technology in vehicles – improved handling, braking, and passive safety. Transport infrastructure has been greatly improved; modern freeways are ten times safer than regular roads [62.108]. However, these improvements appear to be reaching a limit in terms of the number of deaths per million passengers-kilometers, particularly in industrialized countries.

As discussed earlier, motor vehicles are inherently dangerous due to their reliance on human control. Slight errors at high speeds can have catastrophic results. As discussed earlier the most common cause is driver distraction, which leads to improper reaction times or driving actions. Human error also frequently occurs while handling emergency situations. A large percentage of drivers will take improper action in such situations and produce an accident that could have been avoided by a skilled and attentive driver [62.108]. The best solution to these problems is to remove the driver from the control loop. As discussed earlier, the interim step is to assist the driver to warn him/her in the case of potential danger (e.g., in the case of excessive speed before a dangerous bend or when a car is present in the blind spot while changing lane), or to take over control in emergency situations (e.g., emergency braking in the case of impending collision). For legal liability reasons, until it can be shown that autonomous systems have high integrity and reliability, people must be kept in the loop – in a supervisory capacity.

62.6.2 Traffic Congestion

The success of the automobile also leads to saturation of the road infrastructure, particularly in cities. Each car needs a certain amount of space in order to operate safely. The usual width of roads is 3.5 m in order to accommodate steering imprecision, while vehicles are about 2 m in width. Spacing between vehicles also has to be kept at a safe minimum to prevent collisions during deceleration (this principally depends on driver reaction time). It is usually recommended that the spacing should correspond to a time gap of at least 1.5 s. This spacing leads to a maximum throughput of about 2200 cars per hour per lane, independent of traf-

fic speed. This is not high if we consider that a suburban train can carry about 60 000 passengers per hour on an infrastructure of similar dimensions. Furthermore, high-density car traffic of greater than 2200 cars leads to a breakdown in traffic flow (stop-and-go traffic) and to an increased likelihood of accidents, which can drastically decrease the overall capacity of the system. The solution to this problem also lies in the removal of the driver from the control loop to improve lateral guidance (reduction of the width of lanes) and longitudinal control (with possible time gaps of around 0.3 s, independent of speed) while maintaining traffic safety. Such techniques of automated driving could multiply the throughput of road infrastructure by a factor of 5. This was demonstrated in particular by the work performed in the advanced highway systems (AHS) project in the USA, which led to the demonstration of seven-car platoons running at speeds up to 130 km/h on a dedicated freeway in San Diego with gaps of about 0.5 s [62.109].

Another congestion problem is associated with parking. Every individual vehicle is only used for a small percentage of its total usable life. Most of the time, motor vehicles occupy space very unproductively. Typically, a car requires about 10 m^2 of space. Usually, parking occurs at the curbside – a space that is very limited in large cities and cannot accommodate all cars of residents and visitors. In parking lots, each car will need four times this amount in order to have access to each individual slot. If a transportation system based on fully automated cars could be developed, people would not need to own cars but could rely on a service such as the one offered by taxis with vehicles that would come on demand and offer a complement to mass transport. This is the concept of the cybercar, which is under development in Europe [62.110].

62.6.3 Environmental Factors

The ever-increasing deployment of passenger and commercial vehicles has led to critical environmental problems of noise and of pollution in the local community. In addition greenhouse-gas emissions have an impact at the global environment level. Recently, automotive manufacturers have been able to reduce emissions of local pollutants drastically; noise in cities is now perceived as the major problem by the inhabitants. At the global level, the generation of CO_2 through the use of fossil fuels is also considered to be a major problem, which will require drastic steps. In the short term, this may lead to limiting the use of vehicles that generate CO_2 above certain levels. In the longer term, this will lead the automotive industry to offer vehicles that run on various forms of energy and to new forms of transportation systems with much higher efficiency. Automated vehicles running in platoon formations on new infrastructures could form such a system.

62.6.4 The Automobile of the Future

All three challenges are now at the heart of new policies being developed in many countries. These policies concern safety and emissions features in vehicles with a strong push towards advanced safety systems, as was recently seen in Europe with the *Intelligent Car Initiative* [62.111]. At the infrastructure level, there is also a strong push to implement regulation schemes to limit and control the use of road transport and promote alternative and more efficient transport means. In the future, this means that the use of a private automobile will be much more regulated and more integrated with other modes of transport.

The automotive industry might move from an industry of products to an industry of service where anyone can have access to mobility in the most cost efficient way. Both mass transportation and individual transportation would be offered by companies in the most cost-efficient way, respecting local regulations. Companies operating in this service industry could be transit operators, taxi companies, car rental companies, car-sharing companies or even new entrants into the transportation business such as mobile phone companies, which are already familiar with large customer bases and mobility services. Car-sharing operators such as those operating in Germany, Switzerland, Japan, and the USA are likely candidates [62.112]. Figure 62.21 shows a commercial car-sharing operation.

In this context, new types of vehicles such as electric ones and automated driving are being developed because of the decrease in the cost of operation and improvements in safety.

Demonstration projects by Honda and by Toyota have already been put in place along these lines. How-

Fig. 62.21 Car sharing at Toyota

ever, this market is still searching for its operators and business model, as well as the right products [62.110]. Figure 62.22 shows the Honda concept automated car.

There are two distinct trends in the future of automated vehicles. One is with advanced driving assistance, which has spread rapidly since the late 1990s with numerous techniques appearing recently in high-end passenger vehicles and commercial vehicles (buses and trucks). This trend was described earlier in this chapter. The other trend is associated with the arrival of people-movers based on automated guided vehicles (small or large) in specific locations and on dedicated tracks (protected or not).

It is forecast that, in the next 10 years, these two trends will merge, with individual vehicles having dual-mode capabilities: manual (with strong control and assistance) driving on regular roads and fully automatic driving on dedicated zones where no (or few) manual vehicles will be allowed, thereby ensuring smooth and safe operation of automated vehicles [62.110]. This type of vehicle will be perfect for the implementation of mobility services with vehicles that can be called on demand (perhaps through a mobile phone) when and where needed. With the development of such zones, new dedicated infrastructures will be built specifically for these vehicles to go automatically and at high speed from one automated zone to the next. This appears to the most realistic path for *automated highways* to be realized, since it is now considered nearly impossible to have a smooth evolution from the actual infrastructure with its manual vehicles to one with a majority of fully automated ones. This is one of the reasons why the AHS project was canceled in the USA despite a very successful demonstration in 1997.

62.6.5 Automated Vehicle Deployment

As discussed earlier some automated functions are being introduced in production vehicles. Honda and Toyota have introduced a combination of lateral control (lane keeping assistance using image processing) and longitudinal control (adaptive cruise control using laser and radar sensors) [62.91]. An intelligent parking assist system was introduced recently by Toyota, offering the ability for the car to be parked without the driver using the steering wheel. However, this system does not use sensors: the driver has to position his/her vehicle on the image of the rear camera [62.89]. However, most car manufacturers and some component manufacturers are actively working on the sensors to remove this task from the driver.

Fully automated vehicles without any human intervention or supervision are now appearing in the commercial domain. Automated bus rapid transit (ABRT) combines the service quality of rail transit with the flexibility and cost of buses (as shown in Fig. 62.23). Nonautomated bus rapid transit (BRT) is already recommended by the World Bank in developing countries as the most efficient mass transport system. By reserving a dedicated lane for bus operation and adding some light infrastructure to facilitate loading and unloading, capacities similar to those of a train (60 000 passengers/h/direction) can be obtained, as has been demonstrated in South America [62.113]. By

Fig. 62.22 Automated electric cars by Honda

Fig. 62.23 Toyota ABRT used in IMTS Phileas

Fig. 62.24 Cybercars at Schipol airport, Amsterdam

adding the driving automation on a BRT, the system can be made more efficient and safer, as is already the case with automated metros.

The intelligent multimode transit system (IMTS) consists of vehicles guided by magnetic markers imbedded in the middle of their dedicated roads. The platoon running function (three electronically linked vehicles running in file formation at uniform speeds) of IMTS consists of precisely controlling the speed of all the vehicles in the platoon to be the same at all times [62.114]. In the city of Eindhoven in The Netherlands, multiarticulated buses with several steering axes also run in automated mode on a dedicated track using magnetic markers [62.115]. These ABRTs usually keep a driver in the vehicle, since there is the possibility of reverting to manual mode if need be (such as for an unexpected obstacle or in situations with many pedestrians); however, future plans are for full automation.

ABRT technology can also be found with smaller vehicles, now called cybercars, for on-demand door-to-door operation [62.110]. These vehicles were put into operation for the first time at Schipol airport (Amsterdam) in December 1997 to move passengers from long-term parking lots to the airport terminal (Fig. 62.24). Since then these cybercars have been further developed, financed by the Information Society Technologies (IST) program of the European Commission [62.110]. The long-term future of cybercars may lie with the development of dual-mode vehicles, with a particular emphasis on car-sharing operations [62.112]. Cybercars will have an automatic mode for operation in city centers (restricted to this type of vehicle) and a driver-operated (with assistance) mode for regular roads. VIDEO 429 illustrates the cybercar concept. The automobile industry is examining the viability of the development of such vehicles.

In order to match the performance of manually driven vehicles in urban environments, automated vehicles face major technical challenges. In particular, planning a trajectory for a car while avoiding moving and static obstacles remains as an important research challenge. In 2005, the DARPA Grand Challenge described at the beginning of this chapter brought together a large number of these researchers to demonstrate the feasibility of automated driving techniques. Five vehicles succeeded in completing the difficult course of 132 miles in the desert in totally autonomous mode. The 2007 DARPA challenge focused on multiple vehicles operating simultaneously in an urban environment; this challenge and the subsequent Google cars has certainly shown that automotive products based on autonomous vehicle techniques are becoming closer to market [62.15].

62.7 Future Trends and Prospects

Our cars, trucks, and buses are inevitably getting smarter. The future path is not completely known, and is likely to vary in different locations, but it is becoming widely accepted that there will be no need to drive our cars in the remote future. The time frame is still undefined, but it appears that autonomous cars will populate our highways – perhaps using special lanes- in less than a decade. Governments will not only have to allow this change, but will have the great opportunity to influence its deployment, for example by increasing the number of dedicated lanes and the benefits that autonomous vehicles will obtain as opposed to less safe driving of manually driven cars.

Four different states in the USA have paved the way for having autonomous vehicles driven on regular

roadways (albeit for testing purposes), moving towards legislation that will make autonomous cars legal on everyday roads. This trend is not only happening in the USA, but also in Europe. Indeed, differences in culture and in driving habits will bring driverless technology into reality in selected locations earlier. Countries in which particularly fast and disrespectful driving is tolerated will make the introduction of autonomous vehicles challenging. Additionally, the safety issues associated with intermingling of autonomous and human driven cars have become critical and extensive investigations are again underway.

Several trends are clear, however. As sensor-equipped vehicles become more common, there will be increasing opportunities to build sensor-friendly technologies into roadways and into vehicles. It is straightforward to reduce the amount of radar clutter on road sides by replacing large metal signs, which are excellent radar reflectors, with nonreflective signs made of plastic or composites. At the same time, it will be desirable to increase the radar cross-section of small vehicles, such as motorcycles, to make them easier for radar-equipped cars to detect. Radar-reflecting licence plates, for instance, would make smaller vehicles stand out more clearly.

The next steps may be to deploy systems that support the active transmission information from infrastructure to vehicle and vehicle to vehicle. Dedicated short-range communications (DSRC) is already in wide use in smart toll collection stations. It is easy to imagine DSRC-equipped vehicles receiving alerts from roadside sensors of upcoming fog, or ice, or congestion. Intelligent vehicles that detect unusual roadway conditions (via radar or lidar, slip detectors, or driver actions such as hard braking) could broadcast that information to all nearby smart vehicles. As market penetration of equipped vehicles increases, the value of such ad-hoc communications nets will go up dramatically.

Vehicle-to-vehicle communications need not be only via radio. Light-emitting diode (LED) brake lights can be pulsed at kilohertz frequencies, well above the bandwidth at which human eyes can detect flicker, but easily received by following vehicles. This would be a straightforward way for a vehicle to tell other vehicles about the onset of hard braking, or other unusual roadway conditions.

62.8 Conclusions and Further Reading

Advances in both the technologies and the algorithmic aspects of developing autonomous vehicles continue. Presently, impressive new systems have been demonstrated (Bertha [62.116] and PROUD [62.117]). In the first case, a Mercedes-Benz S 500 traveled on the historical route between Mannheim and Pforzheim in Germany in August 2013, while the PROUD experiment involved urban driving in open traffic in the inner city of Parma, Italy, and a part of the trip was made with nobody behind the steering wheel (VIDEO 178).

The general public will become increasingly accustomed to intelligent systems – sensors and communications on vehicles. The only showstopper in the large-scale deployment of robotics technologies in the automobile field is the ability of the industry to deliver the technologies with total safety, which focuses attention on the problems of failsafe systems and their certification. The rail and aerospace industries have solved these problems but in a very different environment. In these industries, the cost of safety for each vehicle can be much higher than in a car or a bus, and the operational environment is also quite different, with professionals operating and maintaining the system. Manufacturers will slowly develop experience in reliability and cost engineering, and governments will gradually work out liability issues. For these reasons, systems that put the driver in the loop of new automation technologies will be the current focus of development.

Many issues will need to be carefully addressed, including user acceptance, liability, and safety; however, it seems quite probable that this technology will lead to an unprecedented revolution in the field of transportation and people mobility.

For further reading on this topic there are a number of sources that can be recommended:

- Intelligent vehicles survey text [62.9, 10]
- Intelligent transport systems journals (IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Vehicular Technology, Transportation Research PART B: Methodology, IEEE Transactions on Intelligent Vehicles)
- Annual conferences (Intelligent Vehicles Symposium, Intelligent Transportation Systems Conference, Vehicular Technology Annual Conference, World Congress on Intelligent Transportation Systems, Intelligent Vehicle Systems Symposium)
- Government resources [62.118–121],
- Magazines (ITS Magazine).

Video-References

VIDEO 178 PROUD2013 – Inside VisLab's driverless car
available from http://handbookofrobotics.org/view-chapter/62/videodetails/178

VIDEO 179 VIAC: The VisLab Intercontinental Autonomous Challenge
available from http://handbookofrobotics.org/view-chapter/62/videodetails/179

VIDEO 420 Motion prediction using the Bayesian occupancy filter approach (Inria)
available from http://handbookofrobotics.org/view-chapter/62/videodetails/420

VIDEO 429 Cybercars and the city of tomorrow
available from http://handbookofrobotics.org/view-chapter/62/videodetails/429

VIDEO 566 Bayesian embedded perception in Inria/Toyota instrumented platform
available from http://handbookofrobotics.org/view-chapter/62/videodetails/566

VIDEO 567 Inria/Ligier automated parallel parking demo in an open parking area
available from http://handbookofrobotics.org/view-chapter/62/videodetails/567

VIDEO 822 Collision avoidance at blind intersections using V2V and intention / expectation approach (Inria & Renault)
available from http://handbookofrobotics.org/view-chapter/62/videodetails/822

VIDEO 836 Lane tracking
available from http://handbookofrobotics.org/view-chapter/62/videodetails/836

VIDEO 838 Speed sign detection
available from http://handbookofrobotics.org/view-chapter/62/videodetails/838

VIDEO 839 Pedestrian detection
available from http://handbookofrobotics.org/view-chapter/62/videodetails/839

VIDEO 840 Driver fatigue and inattention
available from http://handbookofrobotics.org/view-chapter/62/videodetails/839

References

62.1 J. Ibañez-Guzmán, C. Laugier, J.-D. Yoder, S. Thrun: Autonomous Driving: Context and state-of-the Art. In: *Handbook of Intelligent Vehicles*, ed. by A. Eskandarian (Springer, Berlin, Heidelberg 2012)

62.2 R.E. Fenton, R.J. Mayhan: Automated highway studies at the Ohio State University – An overview, IEEE Trans. Veh. Technol. **40**(1), 100–113 (1991)

62.3 Society of Automotive Engineers (SAE): J2735 Dedicated Short Range Communications (DSRC) Message Set Dictionary (2009)

62.4 E.D. Dickmanns: The development of machine vision for road vehicles in the last decade, IEEE Intell. Veh. Symp., Vol. 1 (2002) pp. 268–281

62.5 A. Broggi, M. Bertozzi, A. Fascioli, G. Conte: *Automatic Vehicle Guidance: The Experience of the ARGO Autonomous Vehicle* (World Scientific, Singapore 1999)

62.6 C.E. Thorpe (Ed.): *Vision and Navigation: The Carnegie Mellon Navlab* (Kluwer, Boston 1990)

62.7 E.D. Dickmanns, B.D. Mysliwetz: Recursive 3-D road and relative ego-state recognition, IEEE Trans. Pattern. Anal. Mach. Intell. **14**(2), 199–213 (1992)

62.8 D. Pomerleau, T. Jochem: Rapidly adapting machine vision for automated vehicle steering, IEEE Intell. Syste. **11**(2), 19–27 (1996)

62.9 A. Eskandarian (Ed.): *Handbook of Intelligent Vehicles* (Springer, Berlin, Heidelberg 2012)

62.10 Ü. Özgüner, T. Acarman, K. Redmill: *Autonomous Ground Vehicles* (Artech House, Boston 2011)

62.11 S. Kato, S. Tsugawa, K. Tokuda, T. Matsui, H. Fujii: Vehicle control algorithms for cooperative driving with automated vehicles and intervehicle communications, IEEE Trans. Intell. Transp. Syst. **3**(3), 155–161 (2002)

62.12 US Department of Transport, Research and Innovative Technology Administration: Guide to federal intelligent transportation system (ITS) research, http://ntl.bts.gov/lib/48000/48300/48313/9E878E25.pdf (2013)

62.13 DARPA Grand Challenge Web Site: http://archive.darpa.mil/grandchallenge05/

62.14 K. Iagnemma, M. Buehler (Eds.): *Journal of Field Robotics, Special issues on the DARPA Grand Challenge, Vol. 23, No. 8/9* (Wiley, Hoboken 2006)

62.15 Department of Defense, Washington: Darpa Grand Challenge, http://archive.darpa.mil/grandchallenge/ (2007)

62.16 J. Ploeg, S. Shladover, H. Nijmeijer, N. Van De Wouw: Introduction to the special issue on the 2011 grand cooperative driving challenge, IEEE Trans. Intell. Transp. Syst. **13**(3), 989–993 (2012)

62.17 A. Broggi, P. Cerri, M. Felisa, M.C. Laghi, L. Mazzei, P.P. Porta: The VisLab intercontinental autonomous challenge: An extensive test for a platoon of intelligent vehicles, Intl. J. Veh. Auton. Syst. **10**(3), 147–164 (2012)

62.18 M. Bertozzi, A. Broggi, A. Coati, R.I. Fedriga: A 13 000 km intercontinental trip with driverless vehicles: The VIAC experiment, IEEE Intell. Transp. Syst. Mag. **5**(1), 28–41 (2013)

62.19 A. Broggi, P. Medici, P. Zani, A. Coati, M. Panciroli: Autonomous vehicles control in the VisLab intercontinental autonomous challenge, Annu. Rev. Control **36**(1), 161–171 (2012)

62.20 L. Vlacic, M. Parent, F. Harashima: *EDS Intelligent Vehicle Technologies, Theory and Applications* (Butterworth-Heinemann, Oxford 2001)

62.21 M. Hirota, Y. Nakajima, M. Saito, M. Uchiyama: Low-cost infrared imaging sensors for automotive application. In: *Advanced Microsystems for Automotive Applications*, VDI-Buch, ed. by J. Valldorf, W. Gessner (Springer, Berlin, Heidelberg 2004)

62.22 M. Hikita: An introduction to ultrasonic sensors for vehicle parking, http://www.newelectronics.co.uk/electronics-technology/an-introduction-to-ultrasonic-sensors-for-vehicle-parking/24966/ (2010)

62.23 M. Klotz, H. Rohling: 24 GHz radar sensors for automotive applications, Int. Conf. Microw. Radar Wirel. Commun. (MIKON), Vol. 1 (2000) pp. 359–362

62.24 A. Ewald, V. Willhoeft: Laser scanners for obstacle detection in automotive applications, IEEE Intell. Veh. Symp. (2000) pp. 682–687

62.25 C. Laugier, I. Paromtchik, M. Perrollaz, M. Yong, J. Yoder, C. Tay, K. Mekhnacha, A. Negre: Probabilistic analysis of dynamic scenes and collision risks assessment to improve driving safety, IEEE Intell. Transp. Syst. Mag. **3**(4), 4–19 (2011)

62.26 S. Lefèvre, C. Laugier, J. Ibañez-Guzmán: Risk assessment at road intersections: Comparing intention and expectation, IEEE Intell. Veh. Symp. (2012) pp. 165–171

62.27 I. Skog, P. Handel: In-car positioning and navigation technologies – A survey, IEEE Trans. Intell. Transp. Syst. **10**(1), 4–21 (2009)

62.28 TomTom: Manufacturer of navigation systems and how they work, http://www.tomtom.com/howdoesitwork/

62.29 Ertico: ActMAP an EU Project for Dynamic Map Updating, http://www.ertico.com/actmap (2007)

62.30 S. Rogers, W. Zhang: Development and evaluation of a curve rollover warning system for trucks, IEEE Intell. Veh. Symp. (2003) pp. 294–297

62.31 D. Vaughan: Vehicle speed control based on GPS/MAP matching of posted speeds, Patent US 5 485 161 (1996)

62.32 H. Sabel, M.R. Herbst: The map as a component in advanced driver assistance systems, Proc. 7th World Cong. Intell. Syst. (2000)

62.33 Q. Ji, X. Yang: Real-time eye, gaze, and face pose tracking for monitoring driver vigilance, Real-Time Imaging **8**(5), 357–377 (2002)

62.34 J.B. Kenney: Dedicated short-range communications (DSRC) standards in the United States, Proceedings IEEE **99**(7), 1162–1182 (2011)

62.35 Y. Ohshima: Control system for automatic driving, Proc. IFAC Tokyo Symp. Syst. Eng. Control Syst. Des. (1965)

62.36 S. Biddlestone, K. Redmill, R. Miucic, Ü. Özgüner: An integrated 802.11p WAVE DSRC and vehicle traffic simulator with experimentally validated urban (LOS and NLOS) propagation models, IEEE Trans. Intell. Transp. Syst. **13**(4), 1792–1802 (2012)

62.37 T. Schaffnit: Automotive standardization of vehicle networks. In: *Vehicular Networking: Automotive Applications and Beyond*, ed. by M. Emmelmann, B. Bochow, C.C. Kellum (Wiley, Chichester 2010)

62.38 L. Le, A. Festag, R. Baldessari, W. Zhang: CAR-2-X Communication in Europe. In: *Vehicular Networks: From Theory to Practice*, ed. by S. Olariu, M.C. Weigle (CRC, Boca Raton 2008)

62.39 T. Ernst: The information technology era of the vehicular industry, ACM SIGCOMM Comput. Commun. Rev. **36**(2), 49–52 (2006)

62.40 CALM working group producing standards and specifications: http://calm.its-standards.eu/

62.41 CVIS: Cooperative Vehicle-Infrastructure Systems, http://www.cvisproject.org/

62.42 D.M. Gavrila, U. Franke, C. Wöhler, S. Görzig: Real-time vision for intelligent vehicles, IEEE Instr. Meas. Mag. **4**(2), 22–27 (2001)

62.43 Japanese advanced safety vehicle project: http://www.mlit.go.jp/road/ITS/pdf/ITSinitiativesJapan.pdf

62.44 J.C. McCall, M.M. Trivedi: Video-based lane estimation and tracking for driver assistance: Survey, system, and evaluation, IEEE Trans. Intell. Transp. Syst. **7**(1), 20–37 (2006)

62.45 N.E. Apostoloff, A. Zelinsky: Robust vision based lane tracking using multiple cues and particle filtering, IEEE Intell. Veh. Symp. (2003) pp. 558–563

62.46 Q. Baig, M. Perrollaz, C. Laugier: A robust motion detection technique for dynamic environment monitoring. A framework for grid-based monitoring of the dynamic environment, IEEE Robotics Autom. Mag. **21**(1), 40–48 (2014)

62.47 M.R. Blas, M. Agrawal, A. Sundaresan, K. Konolige: Fast color/texture segmentation for outdoor robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008)

62.48 D.M. Gavrila, V. Philomin: Real-time object detection for *smart* vehicles, Proc. 7th IEEE Int. Conf. Comp. Vis., Vol. 1 (1999) pp. 87–93

62.49 G. Piccioli, E. De Micheli, P. Parodi, M. Campani: Robust method for road sign detection and recognition, Image Vis. Comput. **14**(3), 209–223 (1996)

62.50 C. Bahlmann, Y. Zhu, R. Visvanathan, M. Pellkofer, T. Koehler: A system for traffic sign detection, tracking, and recognition using color, shape, and motion information, IEEE Intell. Veh. Symp. (2005) pp. 255–260

62.51 U. Franke, A. Joos: Real-time stereo vision for urban traffic scene understanding, Proc. IEEE Intell. Veh. Symp. (2000) pp. 273–278

62.52 Nissan safety vehicle that interacts with infrastructure: http://www.nissan-global.com/EN/TECHNOLOGY/OVERVIEW/vii.html

62.53 N. Hautiere, R. Labayrade, D. Aubert: Estimation of the visibility distance by stereovision: A Generic Approach, IEICE Trans. Inf. Syst. **E89-D**(7), 2084–2091 (2006)

62.54 A. Petrovskaya, M. Perrollaz, L. Oliveira, L. Spinello, R. Triebel, A. Makris, J.D. Yoder, C. Laugier, U. Nunes, P. Bessiere: Awareness of road scene participants for autonomous driving. In: *Hand-*

book of Intelligent Vehicles, ed. by A. Eskandarian (Springer, Berlin, Heidelberg 2012)
62.55 R. Labayrade, D. Aubert, J.P. Tarel: Real time obstacle detection in stereo vision on non flat road geometry through *V-disparity* representation, Proc. IEEE Intell. Veh. Symp., Vol. 2 (2002) pp. 646–651
62.56 Z. Sun, G. Bebis, R. Miller: On-road vehicle detection: A review, IEEE Trans. Pattern Anal. Mach. Intell. **28**(5), 694–711 (2006)
62.57 E. Segawa, M. Shiohara, S. Sasaki, N. Hashiguchi, T. Takashima, M. Tohno: Preceding vehicle detection using stereo images and non-scanning millimeter-wave radar, IEICE Trans. Inf. Syst. **E89-D**(7), 2101–2108 (2006)
62.58 T. Kato, Y. Ninomiya, I. Masaki: An obstacle detection method by fusion of radar and motion stereo, IEEE Trans. Intell. Transp. Syst. **3**(3), 182–188 (2002)
62.59 C. Coué, C. Pradalier, C. Laugier, T. Fraichard, P. Bessiere: Bayesian occupancy filtering for multitarget tracking: An automotive application, Int. J. Robotics Res. **25**(1), 19–30 (2006)
62.60 M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, T. Poggio: Pedestrian detection using wavelet templates, IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (1997) pp. 193–199
62.61 L. Zhao, C.E. Thorpe: Stereo- and neural network-based pedestrian detection, IEEE Trans. Intell. Transp. Syst. **01**(3), 148–154 (2000)
62.62 D. Pomerleau: Ralph: Rapidly adapting lateral position handler, IEEE Symp. Intell. Veh. (1995) pp. 506–511
62.63 US DOT: Intelligent vehicle initiative program, http://ntl.bts.gov/lib/jpodocs/repts_pr/14153.htm (2005)
62.64 Blind Spot Detection products: http://auto.howstuffworks.com/car-driving-safety/safety-regulatory-devices/cars-making-blind-spot-less-dangerous1.htm
62.65 L. Matuszyk, A. Zelinsky, L. Nilsson, M. Rilbe: Stereo panoramic vision for monitoring vehicle blind-spots, IEEE Intell. Veh. Symp. (2004) pp. 31–36
62.66 Volvo Blind Spot Information System (BLIS): http://www.gizmag.com/go/2937/
62.67 Automotive night vision systems: http://electronics.howstuffworks.com/gadgets/automotive/in-dash-night-vision-system3.htm
62.68 Radar-vision fusion for pedestrian detection: http://www.mobileye.com/technology/applications/radar-vision-fusion/
62.69 Large animal detection and collision avoidance product: http://www.telematics.com/telematics-blog/horse-avoidance-tech-latest-volvo-push/
62.70 T.A. Williamson: *A High-Performance Stereo Vision System for Obstacle Detection*, Tech. Rep., Vol. CMU-RI-TR-98-24 (Carnegie Mellon Univ., Pittsburgh 1998)
62.71 E. Li: Millimeter-Wave Polarimetric Radar System as an Advanced Vehicle Control and Warning Sensor, Ph.D. Thesis (Univ. Michigan, Michigan 1998)
62.72 J.A. Hancock: Laser Intensity Based Obstacle Detection and Tracking, Ph.D. Thesis (Carnegie Mellon Univ., Pittsburgh 1999)
62.73 European Union, 7th Framework Program: Interactive, http://www.interactive-ip.eu
62.74 N.M. Oliver, B. Rosario, A.P. Pentland: A Bayesian computer vision system for modeling human interactions, IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 831–843 (2000)
62.75 D. Vasquez, C. Laugier: Modeling and learning behaviors. In: *Handbook of Intelligent Vehicles*, ed. by A. Eskandarian (Springer, Berlin, Heidelberg 2012)
62.76 J.C. McCall, M.M. Trivedi: Lane change intent analysis using robust operators and sparse Bayesian learning, IEEE Trans. Intell. Transp. Syst. **8**(3), 431–440 (2007)
62.77 D. Meyer-Delius, C. Plagemann, W. Burgard: Probabilistic situation recognition for vehicular traffic scenarios, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 4161–4166
62.78 H. Berndt, J. Emmert, K. Dietmayer: Continuous driver intention recognition with hidden Markov models, Proc. IEEE Intell. Transp. Syst. Conf. (2008) pp. 1189–1194
62.79 G.S. Aoude, V.R. Desaraju, L.H. Stephens, J.P. How: Behavior classification algorithms at intersections and validation using naturalistic data, IEEE Intell. Veh. Symp. (2011) pp. 601–606
62.80 G.S. Aoude, B.D. Luders, D.S. Levine, J.P. How: Threat-aware path planning in uncertain urban environments, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 6058–6063
62.81 C. Tay, K. Mekhnacha, C. Laugier: Probabilistic vehicle motion modeling and risk estimation. In: *Handbook of Intelligent Vehicles*, ed. by A. Eskandarian (Springer, Berlin, Heidelberg 2012)
62.82 M. Althoff, O. Stursberg, M. Buss: Model-based probabilistic collision detection in autonomous driving, IEEE Trans. Intell. Transp. Syst. **10**(2), 299–310 (2009)
62.83 S. Lefèvre, C. Laugier, J. Ibañez-Guzmán: Evaluating risk at road intersections by detecting conflicting intentions, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)
62.84 A. Molinero, H. Evdorides, C. Naing, A. Kirk, J. Tecl, J.M. Barrios, M.C. Simon, V. Phan, T. Hermitte: *Accident causation and pre-accidental driving situations – In-depth accident causation analysis, Deliverable D2.2* (IST, Brussel 2008)
62.85 Intelligent intersections and cooperation – Intersafe 2: http://www.cvisproject.org/
62.86 W.D. Jones: Keeping cars from crashing, IEEE Spectrum **38**(9), 40–45 (2001)
62.87 P. Venhovens, K. Naab, B. Adiprasito: Stop and go cruise control, Proc. FISTA World Automot. Congr. (2000)
62.88 I.E. Paromtchik, C. Laugier: Motion generation and control for parking an autonomous vehi-

cle, Proc. Int. Conf. Robotics Autom. (ICRA), Vol. 4 (1996) pp. 3117–3122
62.89 Toyota: Toyota parking assistance system, http://www.toyota-global.com/innovation/safety_technology/safety_technology/parking/
62.90 V.R. Vuchic: 0-Bahn: Description and evaluation of a new concept, 64th Annual Meet. Transport. Res. Board (Transportation Research Board, Washington DC 1985)
62.91 S. Ishida, J.E. Gayko: Development, evaluation and introduction of a lane keeping assistance system, IEEE Intell. Veh. Symp. (2004) pp. 943–944
62.92 C. Hatipoglu, Ü. Özgüner, K.A. Redmill: Automated lane change controller design, IEEE Trans. Intell. Transp. Syst. **4**(1), 13–22 (2003)
62.93 A. Bartels, M.-M. Meinecke, S. Steinmeyer: Lane change assistance. In: *Handbook of Intelligent Vehicles*, ed. by A. Eskandarian (Springer, Berlin, Heidelberg 2012) pp. 729–757
62.94 J. Treat, N. Tumbas, S. McDonald, D. Shinar, R. Hume, R. Mayer, R. Stansifer, N. Castellan: *Tri-Level study of the causes of traffic accidents: Final report – Executive summary*, Tech. Rep., Vol. DOT-HS-034-3-535-79-TAC(S) (Institute for Research in Public Safety, Bloomington 1979)
62.95 K.A. Brookhuis, D. De Waard, W.H. Janssen: Behavioural impacts of advanced driver assistance systems – an overview, Eur. J. Transp. Infrastruct. Res. **1**(3), 245–253 (2001)
62.96 Q. Ji, Z. Zhu, P. Lan: Real-time nonintrusive monitoring and prediction of driver fatigue, IEEE Trans. Veh. Technol. **53**(4), 1052–1068 (2004)
62.97 M. Carmen del Rio, J. Gomez, M. Sancho, F.J. Alvarez: Alcohol, illicit drugs and medicinal drugs in fatally injured drivers in Spain between 1991 and 2000, Forensic Sci. Int. **127**(1/2), 63–70 (2002)
62.98 N.L. Haworth, T.J. Triggs, E.M. Grey: *Driver Fatigue: Concepts, Measurement and Crash Countermeasures* (Human Factors Group, Department of Psychology, Monash University 1988)
62.99 N.L. Howarth, C.J. Heffernan, E.J. Horne: *Fatigue in truck accidents*, Tech. Rep., Vol. 3 (Monash University, Accident Research Centre 1989)
62.100 W.W. Wierwille, L.A. Ellsworth: Evaluation of driver drowsiness by trained raters, Accid. Analysis Prev. **26**(5), 571–581 (1994)
62.101 J. Stutts, D. Reinfurt, L. Staplin, E. Rodgman: *The role of driver distraction in traffic crashes. Tech. Rep* (AAA Foundation for Traffic Safety, Washington 2001)
62.102 T. Victor: Keeping Eye and Mind on the Road, Ph.D. Thesis (Uppsala Univ., Uppsala 2005)
62.103 L. Fletcher, A. Zelinsky: Driver inattention detection based on eye gaze – road event correlation, Int. J. Robotics Res. **28**(6), 774–801 (2009)
62.104 L. Fletcher, L. Petersson, A. Zelinsky: Road scene monotony detection in a fatigue management driver assistance system, Proc. IEEE Intell. Veh. Symp. (2005) pp. 484–489
62.105 Y. Owechko, N. Srinivasa, S. Medasani, R. Boscolo: Vision-based fusion system for smart airbag applications, IEEE Intell. Veh. Symp., Vol. 1 (2002) pp. 245–250
62.106 F.J. Martinez, T. Chai-Keong, J.-C. Cano, C.T. Calafate, P. Manzoni: Emergency services in future intelligent transportation systems based on vehicular communication networks, IEEE Intell. Transp. Syst. Mag. **2**(2), 6–20 (2010)
62.107 Emergency response and assistance to accidents product: https://www.splitsecnd.com/
62.108 M. Schulze, J. Irion, T. Mäkinen, M. Flament: Accidentology as a basis for requirements and system architecture of preventive safety applications, 10th Int. Forum Adv. Microsyst. Automot. Appl. (AAA) (2006) pp. 407–426
62.109 X.Y. Lu, H.S. Tan, S.E. Shladover, J.K. Heidrick: Implementation and comparison of nonlinear longitudinal controllers for car platooning, 5th Int. Symp. Adv. Veh. Control (AVEC) (2000)
62.110 Cybercar: Cybercar – Automated Bus Rapid Transit vehicle
62.111 European Commission: Intelligent Car Initiative, http://www.ertico.com/the-intelligent-car-initiative
62.112 World Carshare Consortium: http://www.ecoplan.org/carshare/cs_index.htm
62.113 R. Cervero: Creating a Linear City with a Surface Metro: *The Story of Curitiba, Institute of Urban and Regional Development*, IURD Working Paper 643 (Univ. California, Berkeley 1995)
62.114 H.S. Jacob Taso, J.L. Botha: *Definition and Evaluation of Bus and Truck Automation Operations Concepts*, Path Rep. UCB-ITS-PRR-2002-08 (Univ. California, Oakland 2002)
62.115 Phileas advanced public transport: http://www.apts-phileas.com
62.116 J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, C. Stiller: Making Bertha drive: An autonomous journey on a historic route, IEEE Intell. Transp. Syst. Mag. **6**(2), 8–20 (2014)
62.117 A. Broggi, P. Cerri, S. Debattisti, M.C. Laghi, P. Medici, M. Panciroli, A. Prioletti: PROUD – Public ROad Urban Driverless test: Architecture and results, Proc. IEEE Intell. Veh. Symp. (2014) pp. 648–654
62.118 US Department of Transportation, Research and Innovation Technology Administration, Intelligent Transportation Systems, Joint Program Office (DOT, Washington 2014): http://www.its.dot.gov/
62.119 European Union Research on Transportation (European Commission, Brussels 2014) http://ec.europa.eu/research/transport/
62.120 Japan Ministry of Land, Infrastructure and Transport: Road Bureau ITS Program, http://www.its.go.jp/ITS/(2007)
62.121 EU Transport Research and Innovation Portal: http://www.transport-research.info/web/index.cfm

Multimedia Contents

63. Medical Robotics and Computer-Integrated Surgery

Russell H. Taylor, Arianna Menciassi, Gabor Fichtinger, Paolo Fiorini, Paolo Dario

The growth of medical robotics since the mid-1980s has been striking. From a few initial efforts in stereotactic brain surgery, orthopaedics, endoscopic surgery, microsurgery, and other areas, the field has expanded to include commercially marketed, clinically deployed systems, and a robust and exponentially expanding research community. This chapter will discuss some major themes and illustrate them with examples from current and past research. Further reading providing a more comprehensive review of this rapidly expanding field is suggested in Sect. 63.4.

Medical robots may be classified in many ways: by manipulator design (e.g., kinematics, actuation); by level of autonomy (e.g., preprogrammed versus teleoperation versus constrained cooperative control), by targeted anatomy or technique (e.g., cardiac, intravascular, percutaneous, laparoscopic, microsurgical); or intended operating environment (e.g., in-scanner, conventional operating room). In this chapter, we have chosen to focus on the role of medical robots within the context of larger computer-integrated systems including presurgical planning, intraoperative execution, and postoperative assessment and follow-up.

First, we introduce basic concepts of computer-integrated surgery, discuss critical factors affecting the eventual deployment and acceptance of medical robots, and introduce the basic system paradigms of surgical computer-assisted planning, execution, monitoring, and assessment (surgical CAD/CAM) and surgical assistance. In subsequent sections, we provide an overview of the technology of medical robot systems and discuss examples of our basic system paradigms, with brief additional discussion topics of remote telesurgery and robotic surgical simulators. We conclude with some thoughts on future research directions and provide suggested further reading.

63.1 **Core Concepts** 1658
- 63.1.1 Medical Robotics, Computer-Integrated Surgery, and Closed-Loop Interventions ... 1658
- 63.1.2 Factors Affecting the Acceptance of Medical Robots 1658
- 63.1.3 Medical Robotics System Paradigms: Surgical CAD/CAM and Surgical Assistance 1660

63.2 **Technology** 1662
- 63.2.1 Mechanical Design Considerations 1662
- 63.2.2 Control Paradigms 1663
- 63.2.3 Virtual Fixtures and Human–Machine Cooperative Systems 1664
- 63.2.4 Safety and Sterility 1665
- 63.2.5 Imaging and Modelling of Patients 1666
- 63.2.6 Registration 1666

63.3 **Systems, Research Areas, and Applications** 1667
- 63.3.1 Nonrobotic Computer-Assisted Surgery: Navigation and Image Overlay Devices 1667
- 63.3.2 Orthopaedic Systems 1667
- 63.3.3 Percutaneous Needle Placement Systems 1668
- 63.3.4 Telesurgical Systems 1670
- 63.3.5 Microsurgery Systems 1671
- 63.3.6 Endoluminal Robots 1671
- 63.3.7 Sensorized Instruments and Haptic Feedback 1672
- 63.3.8 Surgical Simulators and Telerobotic Systems for Training 1674
- 63.3.9 Other Applications and Research Areas 1675

63.4 **Conclusion and Future Directions** 1675

Video-References 1676

References 1676

63.1 Core Concepts

A fundamental property of robotic systems is their ability to couple complex information to physical action in order to perform a useful task. This ability to replace, supplement, or transcend human performance has had a profound influence on many fields of our society, including industrial production, exploration, quality control, and laboratory processes. Although robots have often been first introduced to automate or improve discrete processes such as welding or test probe placement or to provide access to environments where humans cannot safely go, their greater long-term impact has often come indirectly as essential enablers of computer integration of entire production or service processes.

63.1.1 Medical Robotics, Computer-Integrated Surgery, and Closed-Loop Interventions

Medical robots have a similar potential to fundamentally change surgery and interventional medicine as part of a broader, information-intensive environment that exploits the complementary strengths of humans and computer-based technology. The robots may be thought of as information-driven surgical tools that enable human surgeons to treat individual patients with greater safety, improved efficacy, and reduced morbidity than would otherwise be possible. Further, the consistency and information infrastructure associated with medical robotic and computer-assisted surgery systems have the potential to make *computer-integrated surgery* as important to health care as computer-integrated manufacturing is to industrial production.

Figure 63.1 illustrates this view of computer-integrated surgery (CIS). The process starts with information about the patient, which can include medical images (computed tomography (CT), magnetic resonance imaging (MRI), positron emission tomography (PET), etc.), lab test results, and other information. This patient-specific information is combined with statistical information about human anatomy, physiology, and disease to produce a comprehensive computer representation of the patient, which can then be used to produce an optimized interventional plan. In the operating room, the preoperative patient model and plan must be *registered* to the actual patient. Typically, this is done by identifying corresponding landmarks or structures on the preoperative model and the patient, either by means of additional imaging (x-ray, ultrasound, video), by the use of a tracked pointing device, or by the robot itself. If the patient's anatomy has changed, then the model and plan are updated appropriately, and the planned procedure is carried out with assistance of the robot. As the intervention continues, additional imaging or other sensing is used to monitor the progress of the procedure, to update the patient model, and to verify that the planned procedure has been successfully executed. After the procedure is complete, further imaging, modeling, and computer-assisted assessment is performed for patient follow-up and to plan subsequent interventions, if any should be required. Further, all the patient-specific data generated during the planning, execution, and follow-up phases can be retained. These data can subsequently be analyzed statistically to improve the rules and methods used to plan future procedures.

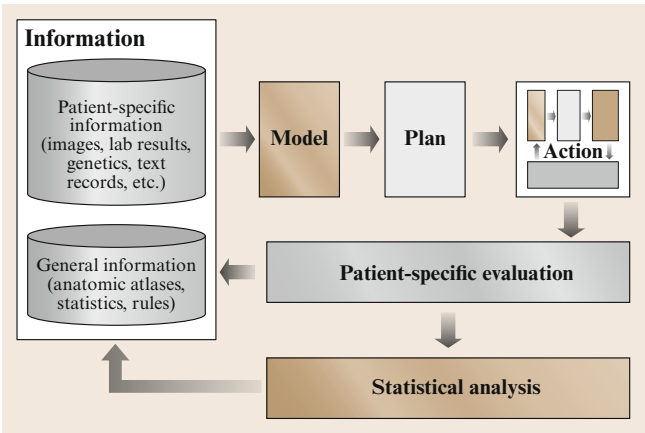


Fig. 63.1 Fundamental information flow in computer-integrated surgery

63.1.2 Factors Affecting the Acceptance of Medical Robots

Medical robotics is ultimately an application-driven research field. Although the development of medical robotic systems requires significant innovation and can lead to very real, fundamental advances in technology, medical robots must provide measurable and significant advantages if they are to be widely accepted and deployed. The situation is complicated by the fact that these advantages are often difficult to measure, can take an extended period to assess, and may be of varying importance to different groups. Table 63.1 lists some of the more important factors that researchers contemplating the development of a new medical robot system should consider in assessing their proposed approach.

Broadly, the advantages offered by medical robots may be grouped into three areas. The first is the potential of a medical robot to *significantly improve surgeons' technical capability* to perform procedures by

Table 63.1 Assessment factors for medical robots or computer-integrated surgery systems (after [63.1])

Assessment factor	Important to whom	Assessment method	Summary of key leverage
New treatment options	Clinical researchers, patients	Clinical and trials preclinical	Transcend human sensory-motor limits (e.g., in microsurgery). Enable less invasive procedures with real-time image feedback (e.g., fluoroscopic or MRI-guided liver or prostate therapy). Speed up clinical research through greater consistency and data gathering
Quality	Surgeons, patients	Clinician judgment; revision rates	Significantly improve the quality of surgical technique (e.g., in microvascular anastomosis), thus improving results and reducing the need for revision surgery
Time and cost	Surgeons, hospitals, insurers	Hours, hospital charges	Speed operating room (OR) time for some interventions. Reduce costs from healing time and revision surgery. Provide effective intervention to treat patient condition
Less invasiveness	Surgeons, patients	Qualitative judgment; recovery times	Provide crucial information and feedback needed to reduce the invasiveness of surgical procedures, thus reducing infection risk, recovery times, and costs (e.g., percutaneous spine surgery)
Safety	Surgeons, patients	Complication and revision surgery rates	Reduce surgical complications and errors, again lowering costs, improving outcomes and shortening hospital stays (e.g., robotic total hip replacement (THR), steady-hand brain surgery)
Real-time feedback	Surgeons	Qualitative assessment, quantitative comparison of plan to observation, revision surgery rates	Integrate preoperative models and intraoperative images to give surgeon timely and accurate information about the patient and intervention (e.g., fluoroscopic x-rays without surgeon exposure, percutaneous therapy in conventional MRI scanners). Assure that the planned intervention has in fact been accomplished
Accuracy or precision	Surgeons	Quantitative comparison of plan to actual	Significantly improve the accuracy of therapy dose pattern delivery and tissue manipulation tasks (e.g., solid organ therapy, microsurgery, robotic bone machining)
Enhanced documentation and follow-up	Surgeons, clinical researchers	Databases, anatomical atlases, images, and clinical observations	Exploit CIS systems' ability to log more varied and detailed information about each surgical case than is practical in conventional manual surgery. Over time, this ability, coupled with CIS systems' consistency, has the potential to significantly improve surgical practice and shorten research trials

exploiting the complementary strengths of humans and robots summarized in Table 63.2. Medical robots can be constructed to be more precise and geometrically accurate than an unaided human. They can operate in hostile radiological environments and can provide great dexterity for minimally invasive procedures inside the patient's body. These capabilities can both enhance the ability of an *average* surgeon to perform procedures that only a few exceptionally gifted surgeons can perform unassisted and can also make it possible to perform interventions that would otherwise be completely infeasible.

Table 63.2 Complementary strengths of human surgeons and robots (after [63.1])

	Strengths	Limitations
Humans	Excellent judgment Excellent hand–eye coordination Excellent dexterity (at natural *human* scale) Able to integrate and act on multiple information sources Easily trained Versatile and able to improvise	Prone to fatigue and inattention Limited fine motion control due to tremor Limited manipulation ability and dexterity outside natural scale Cannot see through tissue Bulky end-effectors (hands) Limited geometric accuracy Hard to keep sterile Affected by radiation, infection
Robots	Excellent geometric accuracy Untiring and stable Immune to ionizing radiation Can be designed to operate at many different scales of motion and payload Able to integrate multiple sources of numerical and sensor data	Poor judgment Hard to adapt to new situations Limited dexterity Limited hand–eye coordination Limited haptic sensing (today) Limited ability to integrate and interpret complex information

A second, closely related capability is the potential of medical robots to *promote surgical safety* both by improving a surgeon's technical performance and by means of active assists such as *no-fly zones* or *virtual fixtures* (Sect. 63.2.3) to prevent surgical instruments from causing unintentional damage to delicate structures. Furthermore, the integration of medical robots within the information infrastructure of a larger CIS system can provide the surgeon with significantly improved monitoring and online decision supports, thus further improving safety.

A third advantage is the inherent ability of medical robots and CIS systems to *promote consistency* while *capturing detailed online information* for every procedure. Consistent execution (e.g., in spacing and tensioning of sutures or in placing of components in joint reconstructions) is itself an important quality factor. If saved and routinely analyzed, the *flight data recorder* information inherently available with a medical robot can be used both in *morbidity and mortality* assessments of serious surgical incidents and, potentially, in statistical analyses examining many cases to develop better surgical plans. Furthermore, such data can provide valuable input for surgical simulators, as well as a database for developing skill assessment and certification tools for surgeons.

63.1.3 Medical Robotics System Paradigms: Surgical CAD/CAM and Surgical Assistance

We call the process of computer-assisted planning, registration, execution, monitoring, and assessment *surgical CAD/CAM*, emphasizing the analogy to manufacturing CAD/CAM. Just as in manufacturing, robots can be critical in this CAD/CAM process by enhancing the surgeon's ability to execute surgical plans. The specific role played by the robot depends somewhat on the application, but current systems tend to exploit the geometric accuracy of the robot and/or its ability to function concurrently with x-ray or other imaging devices. Typical examples include radiation therapy delivery robots such as Accuray's CyberKnife [63.2] (Accuray, Inc., Sunnyvale, CA), shaping of bone in orthopaedic joint reconstructions (discussed further in Sect. 63.3.2) and image-guided placement of therapy needles (Sect. 63.3.3).

Surgery is often highly interactive; many decisions are made by the surgeon in the operating room and executed immediately, usually with direct visual or haptic feedback. Generally, the goal of surgical robotics is not to replace the surgeon so much as to improve his or her ability to treat the patient. The robot is thus a computer-controlled surgical tool in which control of the robot is often shared in one way or another between the human surgeon and a computer. We thus often speak of medical robots as *surgical assistants*.

Broadly, robotic surgical assistants may be broken into two subcategories. The first category, *surgeon extender robots*, manipulate surgical instruments under the direct control of the surgeon, usually through a teleoperation or *hands-on* cooperative control interface. The primary value of these systems is that they can overcome some of the perception and manipulation limitations of the surgeon. Examples include the ability to manipulate surgical instruments with superhuman precision by eliminating hand tremor, the ability to perform highly dexterous tasks inside the patient's body, or the ability to perform surgery on a patient who is physically remote from the surgeon. Although setup time is still a serious concern with most surgeon extender

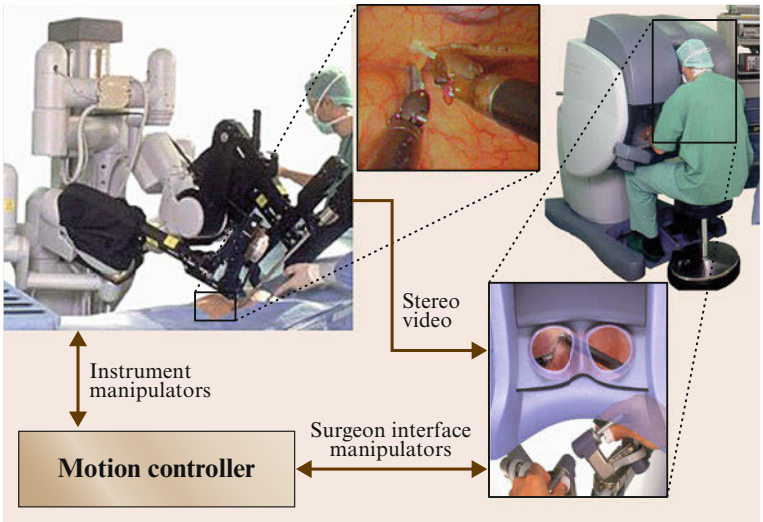


Fig. 63.2 The da Vinci telesurgical robot (after [63.3]) extends a surgeon's capabilities by providing the immediacy and dexterity of *open* surgery in a minimally invasive surgical environment (photos courtesy of Intuitive Surgical, Sunnyvale)

systems, the greater ease of manipulation that such systems offer has the potential to reduce operative times. One widely deployed example of a surgeon extender is the da Vinci system [63.3] (Intuitive Surgical Systems, Sunnyvale, CA) shown in Fig. 63.2. Other examples (among many) incude the Sensei catheter system [63.7] (Hansen Medical Systems, Mountain View, CA.). the Johns Hopkins University (JHU) *Steady Hand* microsurgery robot [63.4–6] shown in Fig. 63.3 and discussed in Sect. 63.3, the Rio orthopaedic robot [63.8] (Mako Surgical Systems, Ft. Lauderdale, Florida), the DLR *Miro* system [63.9], the Surgica Robotica's *Sergenius* system (Surgica Robotica, Udine), and Titan Medical's *Amadeus* System (Titan Medical, Toronto, Canada).

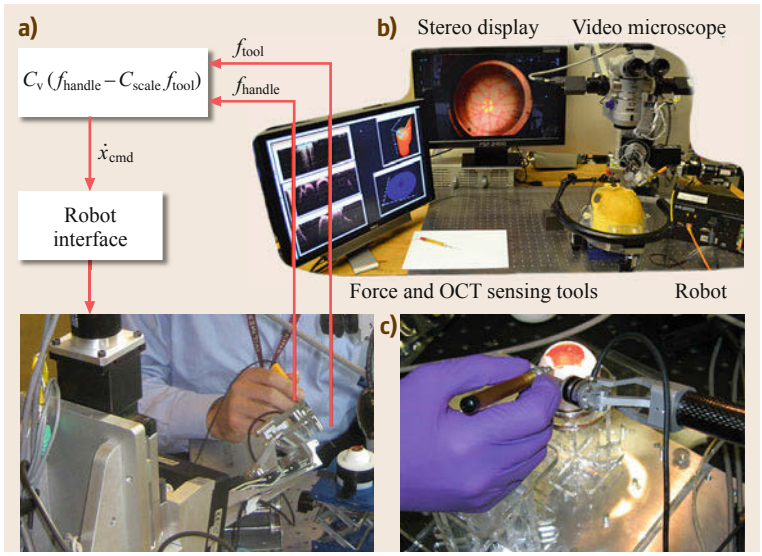


Fig.63.3a–c The Johns Hopkins *Steady Hand* microsurgical robot (after [63.4–6]) extends a surgeon's capabilities by providing the ability to manipulate surgical instruments with very high precision while still exploiting the surgeon's natural hand–eye coordination. **(a)** The basic paradigm of hands-on compliant guiding. The commanded velocity of the robot is proportional to a scaled difference between the forces exerted by the surgeon on the tool handle and (optionally) sensed tool-to-tissue forces. **(b)** Current laboratory setup, showing the robot, stereo video microscope, stereo display with information overlays, display console for optical coherence tomography (OCT) system, and a sensorized tool. **(c)** An earlier recent version of the Steady Hand robot currently being used for experiments in microcannulation of 100 μm blood vessels

A second category, *auxiliary surgical support robots*, generally work alongside the surgeon and perform such routine tasks as tissue retraction, limb positioning, or endoscope holding. One primary advantage of such systems is their potential to reduce the number of people required in the operating room, although that advantage can only be achieved if *all* the tasks routinely performed by an assisting individual can be automated. Other advantages can include improved task performance (e.g., a steadier endoscopic view), safety (e.g., elimination of excessive retraction forces), or simply giving the surgeon a greater feeling of control over the procedure. One of the key challenges in these systems is providing the required assistance without posing an undue burden on the surgeon's attention. A variety of control interfaces are common, including joysticks, head tracking, voice recognition systems, and visual tracking of the surgeon and surgical instruments, for example, the Aesop endoscope positioner [63.10] used both a foot-actuated joystick and a very effective voice recognition system. Again, further examples are discussed in Sect. 63.3.

It is important to realize that *surgical CAD/CAM* and *surgical assistance* are complementary concepts. They are not at all incompatible, and many systems have aspects of both.

63.2 Technology

The mechanical design of a surgical robot depends crucially on its intended application. For example, robots with high precision, stiffness and (possibly) limited dexterity are often very suitable for orthopaedic bone shaping or stereotactic needle placement, and medical robots for these applications [63.17–20] frequently have high gear ratios and consequently, low back-drivability, high stiffness, and low speed. On the other hand, robots for complex, minimally invasive surgery (MIS) on soft tissues require compactness, dexterity, and responsiveness. These systems [63.3, 21] frequently have relatively high speed, low stiffness, and highly backdrivable mechanisms.

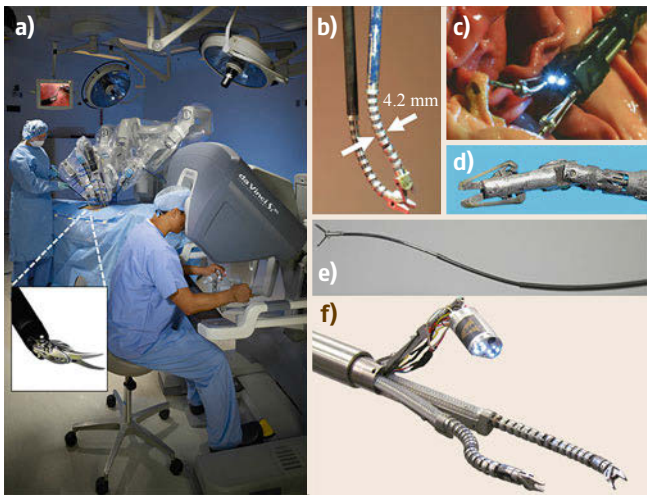


Fig.63.4a–f Dexterity enhancement inside a patient's body: **(a)** The Intuitive da Vinci Si system and wrist with a typical surgical instrument (Photos courtesy of Intuitive Surgical) (after [63.3]); **(b)** The end-effectors of the JHU/Columbia *snake* telesurgical system (after [63.11]); **(c)** Two-handed manipulation system for use in endogastric surgery (after [63.12]); **(d)** five-degree-of-freedom 3 mm wrist and gripper (after [63.13]) for microsurgery in deep and narrow spaces; **(e)** concentric tube robot (after [63.14, 15]); **(f)** Columbia/Vanderbilt high dexterity system for single port access surgery (after [63.16])

63.2.1 Mechanical Design Considerations

Many early medical robots [63.17, 20, 22] were essentially modified industrial robots. This approach has many advantages, including low cost, high reliability, and shortened development times. If suitable modifications are made to ensure safety and sterility, such systems can be very successful clinically [63.18], and they can also be invaluable for rapid prototyping and research use.

However, the specialized requirements of surgical applications have tended to encourage more specialized designs. For example, laparoscopic surgery and percutaneous needle placement procedures typically involve the passage or manipulation of instruments about a common entry point into the patient's body. There are three basic design approaches. The first approach uses a passive wrist to allow the instrument to pivot about the insertion point and has been used in the commercial Aesop and Zeus robots [63.21, 23] as well as several research systems. The second approach mechanically constrains the motion of the surgical tool to rotate about a *remote center of motion* (RCM) distal to the robot's structure. In surgery, the robot is positioned so that the RCM point coincides with the entry point into the patient's body. This approach has been used by the commercially developed da Vinci robot [63.3], as well as by numerous research groups, using a variety of kinematic designs [63.24–26]. Finally, a third approach uses an active external wrist [63.9, 17]and thus supports robot-assisted interventions that do not require a pivot point,

potentially extending robotic surgery to other field of surgery.

The emergence of minimally invasive surgery has created a need for robotic systems that can provide high degrees of dexterity in very constrained spaces inside the patient's body, and at smaller and smaller scales. Figure 63.4 shows several typical examples of current approaches. One common response has been to develop cable-actuated wrists [63.3]. However, a number of investigators have investigated other approaches, including bending structural elements [63.11], shape-memory alloy actuators [63.27, 28], microhydraulic systems [63.29], and electroactive polymers [63.30]. Similarly, the problem of providing access to surgical sites inside the body has led several groups to develop semiautonomously moving robots for epicardial [63.31] or endoluminal applications [63.32, 33].

Two growing trends MIS are natural orifice transluminal surgery (NOTES) [63.34, 35] and single port laparoscopy (SPL) [63.36]: the idea is to get access to the abdominal cavity by using natural orifices and internal incisions (in NOTES) or existing human scars (e.g., the navel, in SPL). From the mechanical design viewpoint, there is the need to develop deployable surgical instruments or accessorised endoscopes and to combine flexibility (to reach the target) and stability of the platform (to achieve precision). Clashing of instruments and difficulty in triangulation are the main limitations which companies and research groups try to approach [63.37, 38].

The problem of distal operation, already present in MIS, is becoming more dramatic in NOTES and SPL and several solutions for helping surgical tasks requiring triangulation have been developed by different research groups [63.39]. They are based on magnetic fields which can generate an internal force without constraining the internal tool to the access port [63.40–42].

Another significant development in recent years has been the emergence and widespread deployment of three-dimensional (3-D) printing and other rapid prototyping technologies for clinically usable medical devices and medical robot components, as well as for construction of realistic patient-specific models [63.43, 44]. This trend has promoted very rapid progress in medical robot design and will be increasingly important in coming years.

Although most surgical robots are mounted to the surgical table, to the operating room ceiling, or to the floor, there has been growing interest in developing systems that directly attach to the patient [63.45, 46], and clinically deployed examples exist [63.47]. The main advantage of this approach is that the relative position of the robot and patient is unaffected if the patient moves. The challenges are that the robot must be smaller and that relatively nonintrusive means for mounting it must be developed.

Finally, robotic systems intended for use in specific imaging environments pose additional design challenges. First, there is the geometric constraint that the robot (or at least its end-effector) must fit within the scanner along with the patient. Second, the robot's mechanical structure and actuators must not interfere with the image formation process. In the case of x-ray and CT, satisfying these constraints is relatively straightforward. The constraints for MRI are more challenging [63.48].

63.2.2 Control Paradigms

Surgical robots assist surgeons in treating patients by moving surgical instruments, sensors, or other devices in relation to the patient. Generally, these motions are controlled by the surgeon in one of three ways:

- *Preprogrammed, semi-autonomous motion*: The desired behavior of the robot's tools is specified interactively by the surgeon, usually based on medical images. The computer fills in the details and obtains the surgeon's concurrence before the robot is moved. Examples include the selection of needle target and insertion points for percutaneous therapy and tool cutter paths for orthopaedic bone machining.
- *Teleoperator control*: The surgeon specifies the desired motions directly through a separate human interface device and the robot moves immediately. Examples include common telesurgery systems such as the da Vinci [63.3]. Although physical *master* manipulators are the most common input devices, other human interfaces are also used, notably voice control [63.21].
- *Hands-on compliant control*: The surgeon grasps the surgical tool held by the robot or a control handle on the robot's end-effector. A force sensor senses the direction that the surgeon wishes to move the tool and the computer moves the robot to comply. Early experiences with Robodoc [63.17] and other surgical robots [63.25] showed that surgeons found this form of control to be very convenient and natural for surgical tasks. Subsequently, a number of groups have exploited this idea for precise surgical tasks, notably the JHU *Steady Hand* microsurgical robot [63.4] shown in Fig. 63.3, the Rio orthopaedic robot [63.8] (Mako Surgical Systems, Ft. Lauderdale, Florida) and the Imperial College Acrobot orthopaedic system [63.49] shown in Fig. 63.5c,d.

These control modes are not mutually exclusive and are frequently mixed. For example, the Robodoc sys-

Fig.63.5a–d Clinically deployed robots for orthopaedic surgery. **(a,b)** The Robodoc system (after [63.17, 18]) represents the first clinically applied robot for joint reconstruction surgery and has been used for both primary and revision hip replacement surgery as well as knee replacement surgery. **(c,d)** The Acrobot system of *Davies* et al. (after [63.49]) uses hands-on compliant guiding together with a form of virtual fixtures to prepare the femur and tibia for knee replacement surgery

tem [63.17, 18] uses hands-on control to position the robot close to the patient's femur or knee and preprogrammed motions for bone machining. Similarly, the IBM/JHU LARS robot. [63.25] used both cooperative and telerobotic control modes. The cooperatively controlled Acrobot [63.49] uses preprogrammed virtual fixtures (Sect. 63.1.3) derived from the implant shape and its planned position relative to medical images.

Each mode has advantages and limitations, depending on the task. Preprogrammed motions permit complex paths to be generated from relatively simple specifications of the specific task to be performed. They are most often encountered in surgical CAD/CAM applications where the planning uses two- (2-D) or three-dimensional (3-D) medical images. However, they can also provide useful complex motions combining sensory feedback in teleoperated or hands-on systems. Examples might include passing a suture or inserting a needle into a vessel after the surgeon has prepositioned the tip. On the other hand, interactive specification of motions based on real-time visual appreciation of deforming anatomy would be very difficult.

Teleoperated control provides the greatest versatility for interactive surgery applications, such as dexterous MIS [63.3, 21, 26, 50] or remote surgery [63.51, 52]. It permits motions to be scaled, and (in some research systems) facilitates haptic feedback between master and slave systems. The main drawbacks are complexity, cost, and disruption to standard operating room work flow associated with having separate master and slave robots.

Hands-on control combines the precision, strength, and tremor-free motion of robotic devices with some of the immediacy of freehand surgical manipulation. These systems tend to be less expensive than telesurgical systems, since there is less hardware, and they can be easier to introduce into existing surgical settings. They exploit a surgeon's natural eye–hand coordination in an intuitively appealing way, and they can be adapted to provide force scaling [63.4, 5]. Although direct motion scaling is not possible, the fact that the tool moves in the direction that the surgeon pulls it makes this limitation relatively unimportant when working with a surgical microscope. The biggest drawbacks are that hands-on control is inherently incompatible with any degree of remoteness between the surgeon and the surgical tool and that it is not practical to provide hands-on control of instruments with distal dexterity.

Teleoperation and hands-on control are both compatible with *shared control modes* in which the robot controller constrains or augments the motions specified by the surgeon, as discussed in Sect. 63.2.3.

63.2.3 Virtual Fixtures and Human–Machine Cooperative Systems

Although one goal of both teleoperation and hands-on control is often *transparency*, i. e., the ability to move an instrument with the freedom and dexterity he/she might expect with a handheld tool, the fact that a computer is actually controlling the robot's motion creates many more possibilities. The simplest is a safety barrier or *no-fly zone*, in which the robot's tool is constrained from entering certain portions of its workspace. More sophisticated versions include virtual springs, dampers, or complex kinematic constraints that help a surgeon align a tool, maintain a desired force, or maintain a desired anatomical relationship. The Acrobot system shown in Fig. 63.5c,d represents a successful clinical applica-

tion of the concept, which has many names, of which *virtual fixtures* seems to be the most popular [63.53, 54]. A number of groups are exploring extensions of the concept to active cooperative control, in which the surgeon and robot share or trade off control of the robot during a surgical task or subtask. As the ability of computers to model and follow along surgical tasks improves, these modes will become more and more important in surgical assistant applications. Figure 63.6 illustrates the overall concept of human–machine cooperative systems in surgery, and Fig. 63.7 illustrates the use of registered anatomical models to generate constraint-based virtual fixtures. These approaches are equally valid whether the surgeon interacts with the system through classical teleoperation or through hands-on compliant control. See also Chap. 43.

Both teleoperation and hands-on control are likewise used in human–machine cooperative systems for rehabilitation and disability assistance systems. Constrained hands-on systems offer special importance for rehabilitation applications and for helping people with movement disorders. Similarly, teleoperation and intelligent task following and control are likely to be vital for further advances in assistive systems for people with severe physical disabilities. See Chap. 64 for a further discussion of human–machine cooperation in assistive systems.

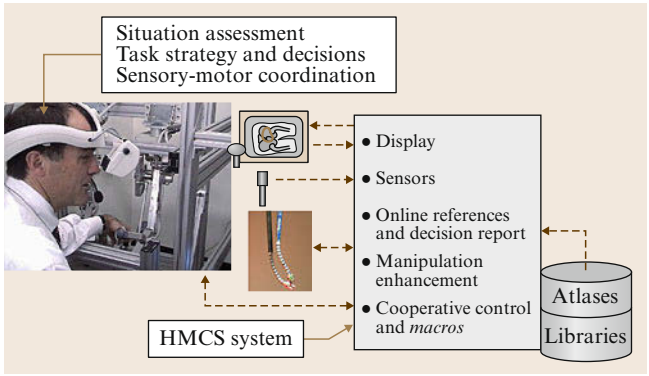


Fig. 63.6 Human–machine cooperative systems (HMCS) in surgery

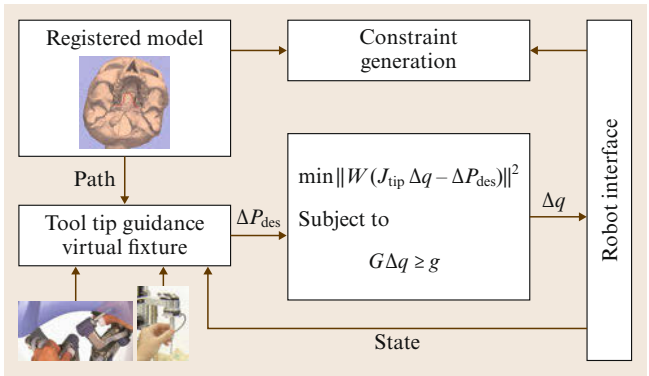


Fig. 63.7 Human–machine cooperative manipulation using constraint-based virtual fixtures, in which patient-specific constraints are derived from registered anatomical models (after [63.53])

63.2.4 Safety and Sterility

Medical robots are safety-critical systems, and safety should be considered from the very beginning of the design process [63.55, 56]. Although there is some difference in detail, government regulatory bodies require a careful and rigorous development process with extensive documentation at all stages of design, implementation, testing, manufacturing, and field support. Generally, systems should have extensive redundancy built into hardware and control software, with multiple consistency conditions constantly enforced. The basic consideration is that no single point of failure should cause the robot to go out of control or to injure a patient. Although there is some difference of opinion as to the best way to make trade-offs, medical manipulators are usually equipped with redundant position encoders and ways to mechanically limit the speed and/or force that the robot can exert. If a consistency check failure is detected, two common approaches are to *freeze* robot motion or to cause the manipulator to go limp. Which is better depends strongly on the particular application.

Sterilizability and biocompatibility are also crucial considerations. Again, the details are application dependent. Common sterilization methods include gamma rays (for disposable tools), autoclaving, soaking or gas sterilization, and the use of sterile drapes to cover unsterile components. Soaking or gas sterilization are less likely to damage robot components, but very rigorous cleaning is required to prevent extraneous foreign matter from shielding microbes from the sterilizing agent.

Careful attention to higher levels of application protocols is also essential. Just like any other tool, surgical robots must be used correctly by surgeons, and careful training is essential for safe practice. Surgeons must understand both the capabilities and limitations of the robot and of the surgical process as well, since safety is a systemic property. This adds new requirements to training programs, which must include robotic capabilities and nontechnical skills (Sect. 63.3.8). In surgical CAD/CAM applications, the surgeon must understand how the robot will execute the plan and be able to verify that the plan is being followed. If the surgeon is interactively commanding the robot, it is essential that the robot interpret these commands cor-

rectly. Similarly, it is essential that the robot's model of its task environment correspond correctly to the actual environment. The availability of task models is necessary to the development of autonomous execution of robotic gestures as well as the formal verification of task correctness [63.57]. Although careful design and implementation can practically eliminate the likelihood of a runaway condition by the manipulator, this will do little good if the robot is badly registered to the patient images used to control the procedure. If the robot fails for any reason, there must be well-documented and planned procedures for recovery (and possibly continuing the procedure manually).

Finally, it is important to remember that a well-designed robot system can actually *enhance* patient safety. The robot is not subject to fatigue or momentary lapses of attention. Its motions can be more precise and there is less chance that a slip of the scalpel may damage some delicate structure. In fact, the system can be programmed to provide *virtual fixtures* (Sect. 63.2.3) preventing a tool from entering a forbidden region unless the surgeon explicitly overrides the system.

63.2.5 Imaging and Modelling of Patients

As the capabilities of medical robots continue to evolve, the use of computer systems to model dynamically changing patient-specific anatomy will become increasingly important. There is a robust and diverse research community addressing a very broad range of research topics, including the creation of patient-specific models from medical images, techniques for updating these models based upon real-time image and other sensor data, and the use of these models for planning and monitoring of surgical procedures. Some of the pertinent research topics include the following:

- Medical image segmentation and image fusion to construct and update patient-specific anatomic models
- Biomechanical property measurement and modelling for analyzing and predicting tissue deformations and functional factors affecting surgical planning, control, and rehabilitation
- Optimization methods for treatment planning and interactive control of systems
- Methods for registering the *virtual reality* of images and computational models to the *physical reality* of an actual patient
- Methods for characterizing treatment plans and individual task steps such as suturing, needle insertion, or limb manipulation for purposes of planning, monitoring, control, and intelligent assistance
- Real-time data fusion for such purposes as updating models from intraoperative images
- Methods for human–machine communication, including real-time visualization of data models, natural language understanding, gesture recognition, etc.
- Methods for characterizing uncertainties in data, models, and systems and for using this information in developing robust planning and control methods.

An in-depth examination of this research is beyond the scope of this article. A more complete discussion of these topics may be found in the suggested further reading in Sect. 63.4.

63.2.6 Registration

Geometric relationships are fundamental in medical robotics, especially in surgical CAD/CAM. There is an extensive literature on techniques for coregistering coordinate systems associated with robots, sensors, images, and the patient [63.58, 59]. Following [63.59], we briefly summarize the main concepts here. Suppose that we have coordinates

$$\boldsymbol{v}_A^r = (x_A, y_A, z_A)$$
$$\boldsymbol{v}_B^r = (x_B, y_B, z_B) ,$$

corresponding to comparable locations in two coordinate systems $\mathrm{Ref_A}$ and $\mathrm{Ref_B}$. Then the process of registration is simply that of finding a function $\boldsymbol{T}_{\mathrm{AB}}(\cdots)$ such that

$$\boldsymbol{v}_B = \boldsymbol{T}_{AB}(\boldsymbol{v}_A).$$

Generally, $\boldsymbol{T}_{\mathrm{AB}}(\cdots)$ is assumed to be a rigid transformation of the form

$$\boldsymbol{T}_{AB}(\boldsymbol{v}_A^r) = \boldsymbol{R}_{AB}\boldsymbol{v}_A^r + \boldsymbol{p}_{AB}^r ,$$

where $\boldsymbol{R}_{\mathrm{AB}}$ represents a rotation and $\boldsymbol{p}_{\mathrm{AB}}$ represents a translation, but nonrigid transformations are becoming increasingly common. There are hundreds of methods for computing $\boldsymbol{T}_{\mathrm{AB}}(\cdots)$. The most common for medical robotics involve finding a set of corresponding geometric features Γ_{A} and Γ_{B} whose coordinates can be determined in both coordinate systems and then finding a transformation that minimizes some distance function $d_{\mathrm{AB}} = \mathrm{distance}[\Gamma_{\mathrm{B}}, \boldsymbol{T}_{\mathrm{AB}}(\Gamma_{\mathrm{A}})]$. Typical features can include artificial fiducial objects (pins, implanted spheres, rods, etc.) or anatomical features such as point landmarks, ridge curves, or surfaces.

One common class of methods is based on the iterated closest-point algorithm of *Besl* and *McKay* [63.60],

for example, 3-D robot coordinates $\boldsymbol{a}_j$ may be found for a collection of points known to be on the surface of an anatomical structure that can also be found in a segmented 3-D image. Given an estimate $\boldsymbol{T}_k$ of the transformation between image and robot coordinates, the method iteratively finds corresponding points $\boldsymbol{b}_j$ on the surface that are closest to $\boldsymbol{T}_k\boldsymbol{a}_j$ and then finds a new transformation

$$\boldsymbol{T}_{k+1} = \arg\min_{T} \sum_j (\boldsymbol{b}_j - \boldsymbol{T}\boldsymbol{a}_j)^2 .$$

The process is repeated until some suitable termination condition is reached.

63.3 Systems, Research Areas, and Applications

Medical robots are not ends in themselves. As the late Hap Paul often remarked, *the robot is a surgical tool designed to improve the efficacy of a procedure*. (Paul was the founder of Integrated Surgical Systems. Along with William Bargar, he was one of the first people to recognize the potential of robots to fundamentally improve the precision of orthopaedic surgery.)

63.3.1 Nonrobotic Computer-Assisted Surgery: Navigation and Image Overlay Devices

In cases where the role of the robot is placing instruments on targets determined from medical images, *surgical navigation* is often a superior alternative. In surgical navigation [63.61], the positions of instruments relative to the reference markers on the patient are tracked using specialized electromechanical, optical, electromagnetic, or sonic digitizers or by more general computer vision techniques. After the relationships between key coordinate systems (patient anatomy, images, surgical tools, etc.) are determined through a registration process (Sect. 63.2.6), a computer workstation provides graphical feedback to the surgeon to assist in performing the planned task, usually by displaying instrument positions relative to medical images, as shown in Fig. 63.8a. Although the registration is usually performed computationally, a simple mechanical alignment of an image display with an imaging device can be surprisingly effective in some cases. One example [63.62] is shown in Fig. 63.8b.

The main advantages of surgical navigation systems are their versatility, their relative simplicity, and their ability to exploit the surgeon's natural dexterity and haptic sensitivity. They are readily combined with *passive* fixtures and manipulation aids [63.65, 66]. The main drawbacks, compared to active robots, are those associated with human limitations in accuracy, strength, ability to work in certain imaging environments, and dexterity inside the patient's body (Table 63.2).

Because these advantages often outweigh the limitations, surgical navigation systems are achieving widespread and increasing acceptance in such fields as neurosurgery, otolaryngology, and orthopaedics. Since much of the technology of these systems is compatible with surgical robots and since technical problems such as registration are common among all these systems, we may expect to see a growing number of hybrid applications combining medical robots and navigation.

63.3.2 Orthopaedic Systems

Orthopaedic surgery represents a natural surgical CAD/CAM application, and both surgical navigation systems and medical robots have been applied to orthopaedics. Bone is rigid and is easily imaged in CT and intraoperative x-rays, and surgeons are accustomed to doing at least some preplanning based on these images. Geometric accuracy in executing surgical plans is very important, for example, bones must be shaped accurately to ensure proper fit and positioning of components in joint replacement surgery. Similarly, osteotomies require both accurate cutting and placement of bone fragments. Spine surgery often requires screws and other hardware to be placed into vertebrae without damage to the spinal cord, nerves, and nearby blood vessels.

The Robodoc system shown in Fig. 63.5a,b represents the first clinically applied robot for joint reconstruction surgery [63.17, 18]. Since 1992, it has been applied successfully to both primary and revision hip replacement surgery, as well as knee surgery. Since this system exhibits many of the characteristics of surgical CAD/CAM, we will discuss it is some detail. In the *surgical CAD* phase, the surgeon selects the desired based on preoperative CT images and interactively specifies the desired position of the implant components. In the *surgical CAM* phase, surgery proceeds normally up to the point where the patient's bones are to be prepared to receive the implant. The robot is moved up to the operating table, the patient's bones are attached rigidly to the robot's base, and the robot is registered to the CT images either by use of implanted fiducial pins or by use of a 3-D digitizer to match bone

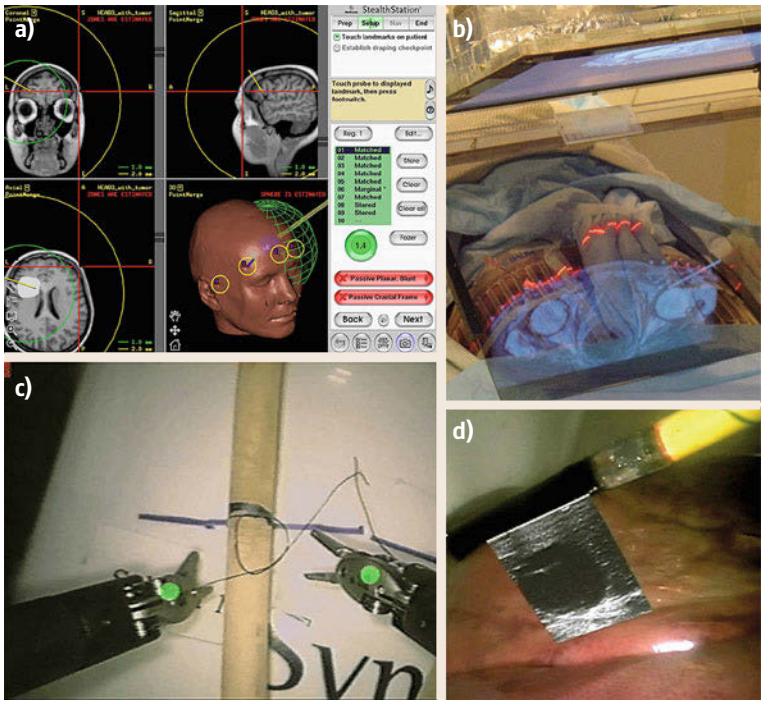


Fig.63.8a–d Information enhancement for surgical assistance. **(a)** Display from a typical surgical navigation system, here the Medtronic StealthStation; **(b)** the JHU image overlay system (after [63.62]) uses a mirror to align the virtual image of a cross-sectional image with the corresponding physical position in the patient's body; **(c)** Sensory substitution display of surgical force information onto da Vinci surgical robot video monitor (after [63.63]); **(d)** Overlay of laparoscopic ultrasound onto the da Vinci surgical robot video monitor (after [63.64])

surfaces to the CT images. After registration, the surgeon's hand guides the robot to an approximate initial starting position. Then, the robot autonomously machines the desired shape with a high-speed rotary cutter while the surgeon monitors progress. During cutting, the robot monitors cutting forces, bone motion, and other safety sensor, and either the robot controller or the surgeon can pause execution at any time. If the procedure is paused for any reason, there are a number of error recovery procedures available to permit the procedure to be resumed or restarted at one of several defined checkpoints. Once the desired shape has been machined, surgery proceeds manually in the normal manner.

Subsequently, several other robotic systems for joint replacement surgery have been introduced or proposed. The references in Sect. 63.4 provide numerous examples. Notable hands-on guided systems include Rio surgical robot [63.8] (Mako Surgical, Ft. Lauderdale, Florida) and the Acrobot [63.49] system for knee surgery shown in Figs. 63.5c,d. Similarly, several groups have recently proposed small orthopaedic robots attaching directly to the patient's bones [63.45] or completely freehand systems such as the NavioPFS surgical system (Blue Belt Technlogies, Pittsburgh, Pa.) which combine surgical navigation with very fast on-off control of a surgical cutter [63.67, 68]. A recent example of a hybrid passive-active robot is [63.69].

63.3.3 Percutaneous Needle Placement Systems

Needle placement procedures have become ubiquitous in image-guided intervention, typically performed through the skin but also through cavities. These procedures fit within the broader paradigm of surgical CAD/CAM, where the process involves use of patient images to identify targets within the patient and planning needle trajectories; inserting the needles and verifying their placement; performing some action such as an injection or taking a biopsy sample; and assessing the results. In most cases, an accuracy of 1–2 mm is acceptable, which is not easy to achieve freehand, because the target is not directly visible, soft tissues tend to deform and deflect, and needles often bend. The procedures typically rely on some form of intraoperative imaging (x-ray, CT, MRI, and ultrasound) for both guidance and verification. The surgical motion sequence typically has three decoupled phases: place the needle tip at the entry point, orient the needle by pivoting around the needle tip, and insert the needle into the body along a straight trajectory. These motions are often performed freehand, with varying degrees of information feedback for the physician. However, passive, semiautonomous, and active robotic systems have all been introduced. Figure 63.9 shows several clinically deployed systems for needle placement.

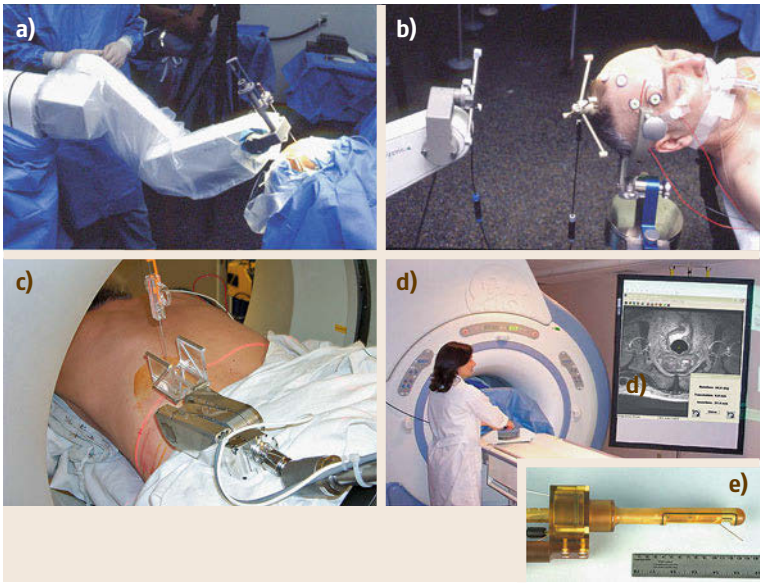


Fig.63.9a–e Clinically deployed systems for in-scanner needle placement. **(a,b)** The Neuromate system (after [63.19]) for stereotactic procedures in the brain uses a novel noncontact sensing system for robot-to-image registration; **(c)** Johns Hopkins system for in-CT needle placement (after [63.70, 71]); **(d,e)** Manually activated device for in-MRI transrectal needle placement into the prostate (after [63.72])

Freehand Needle Placement Systems

Freehand needle placement with CT and MRI guidance uses skin markers to locate the entry point [63.62], reference to the scanner's alignment laser to control needle direction, and markers on the needle to control depth. With ultrasound, the primary reliance is on surgeon experience or the use of some sort of needle guide to drive the needle to target while it passes in the ultrasound plane. Tracked ultrasound snapshots guidance combines orthogonal viewpoints with frozen ultrasound image frames [63.73]. Mechanical needle guides [63.61], hand-held navigation guidance [63.74], and optical guides have been combined with most imaging modalities. These include laser guidance devices [63.75], augmented reality systems [63.76]. Augmented reality with 2-D ultrasound [63.77] and CT/MRI *slices* [63.62] (Fig. 63.8b) was developed, where a semi-transparent mirror is used together with a flat-panel display to create the appearance of a virtual image floating inside the body in the correct position and size.

Passive and Semiautonomous Devices for Needle Placement

Passive, encoded manipulator arms were proposed for image-guided needle placement [63.78], where following a registration step, the position and orientation of a passive needle guide is tracked and the corresponding needle path is displayed on CT or MRI images. Semiautonomous systems allow remote, interactive image-guided placement of needles, such as transrectal prostate needle placement in MRI environment [63.72] with an actuated manipulator from outside the scanner bore, while the needle driver is tracked in MRI with active coils.

Active Robots for Needle Placement

Neurosurgery was one of the first clinical applications of active robots [63.19, 20, 22], a natural application for surgical CAD/CAM. The entry and target points are planned on CT/MRI images, the robot coordinate system is registered to the image coordinate system (typically with markers affixed to the patient's head), and then the robot positions a needle or drill guide. The marker structure may be a conventional stereotactic head frame or, as in the Neuromate system [63.19], registration is achieved by simultaneous tracking of the robot and markers attached to the patient's skull. Spatial constraints in needle placement led to the development of structures achieving *remote center of motion* (RCM) or fulcrum motion [63.24, 25]. In these systems, the RCM is positioned at the entry point, typically with an active Cartesian stage or a passive adjustable arm, and the robot sets the needle direction and (sometimes) the depth. To speed up imaging, planning, registration, and execution, the robot can work concurrently with imaging devices, such as variants of an RCM-based system that was deployed with x-ray [63.24] and CT guidance [63.70, 71]. In [63.71], a marker structure was incorporated in the needle driver to register the robot with a single image slice. MRI has an excellent potential for guiding, monitoring, and controlling therapy, invoking intensive research on MRI-compatible robotic systems for needle placement [63.79] and other more interactive procedures [63.50]. Ultrasound guid-

ance offers many unique advantages: is relatively inexpensive and compact, provides real-time images, does not involve any ionizing radiation, and does not impose significant materials constraints on the robot design. Several robotic systems have been proposed for prostate interventions [63.80] using transrectal ultrasound guidance. For other ultrasound-guided needle placement applications, examples include experimental systems for liver [63.64, 81], gallbladder [63.82], and breast [63.83]. Figure 63.8d shows one example of the use of information overlay to assist in needle placement in a telesurgical application [63.64]. Whatever form of image feedback is available, steering flexible needles to hit desired targets while avoiding obstacles is a ubiquitous problem, having led to several novel approaches [63.84, 85] and reviewed most recently in [63.86]. Concentric tube robots (also interchangeably called *active cannulas* due to their usefulness in medicine), consist of several precurved elastic tubes nested within one another. Actuators grasp these tubes at their bases and extend them telescopically while also applying axial rotations. These movements cause the overall device to elongate and bend, providing a needle-sized device that moves in a manner analogous to a tentacle. Chronologically, a precursor to current concentric tube robots was an early steerable needle design where a curved stylet was used at the tip of a needle to induce steering [63.87]. These robots fall within the broader class of continuously flexible robots called *continuum robots* (see [63.15] for a review.) Over the past few years, mechanics-based models of active cannulas have rapidly matured due to the simultaneous parallel efforts of several research groups [63.14, 88]. Today, these models provide the basis for an active subfield of research on teleoperative control, surgery-specific design, and motion planning research. Surgical applications suggested for concentric tube robots and are in various stages of development, ranging from purely conceptual to animal studies, which will make the next several years exciting for translational clinical research.

63.3.4 Telesurgical Systems

The concepts of telemedicine, telesurgery, and telepresence in surgery date from the 1970s. Since then, the potential for telesurgical systems to facilitate effective interventions in remote or hostile environments such as the battlefield, space, or thinly populated areas has continued to be recognized [63.89], and there have been some spectacular demonstrations including a transatlantic cholecystectomy [63.51] in 2001, experiments in Italy [63.90] and Japan [63.91] as well as more nearly routine use in Canada [63.52]. The operational difficulties due to the intrinsic communication delay of long distance tele-surgery affect usability and safety and make the regular use of tele-surgery quite uncommon.

However, the primary uses of telesurgical systems have been with the surgeon and patient in the same operating room. Teleoperated robots have been used for over 15 years in MIS, both as auxiliary surgical support systems to hold endoscopes or retractors [63.23, 25, 92–94] and as *surgeon extender* systems to manipulate surgical instruments [63.3, 26]. There has also been recent work to develop telesurgical systems for use within imaging environments such as MRI [63.95].

A primary challenge for auxiliary support systems is to permit the surgeon to command the robot while his or her hands are otherwise occupied. Typical approaches have included conventional foot switches [63.23], instrument-mounted joysticks [63.25], voice control [63.10, 25], and computer vision [63.25, 96, 97].

A common goal in surgeon extender systems is to provide a measure of telepresence to the surgeon, specifically, to give the surgeon the sensation of performing open surgery from inside the patient. In early work, *Green* et al. [63.98] developed a successful prototype system for telesurgery combining remote manipulation, force feedback, stereoscopic imaging, ergonomic design, etc. Subsequently, several commercial telesurgical systems have been applied clinically for MIS. Of these, Intuitive Surgical's da Vinci [63.3] has been the most successful, with over 2500 systems deployed as of 2013. Experience with these systems has demonstrated that a high-dexterity wrist is often critical for surgeon acceptance. Although originally targeted at cardiac surgery, as well as more general interventions, the most successful clinical applications to date are radical prostatectomies, where significant improvements in outcomes have been reported [63.99], and hysterectomy, where the clinical benefits compared to conventional laparoscopy are still being studied [63.100].

One emerging area for research exploits the inherent ability of telesurgical systems to act as *flight data recorders* during surgical procedures. Several authors [63.101–104] have begun analyzing such data for such purposes as measuring surgical skill, learning surgical gestures and motions, and providing data for surgical simulators. Another emerging area for research [63.105] focuses on semi-automation of surgical gestures between the surgeon and the robot, often based on learned models. Other research [63.106–108] exploits augmented reality methods to enhance the information available to the surgeon during telesurgical procedures.

63.3.5 Microsurgery Systems

Although *microsurgery* is not a consistently defined term, it generally indicates procedures performed on very small, delicate structures, such as those found in the eye, brain, spinal cord, small blood vessels, nerves, or the like. Microsurgical procedures are commonly performed under direct or video visualization, using some form of magnification (e.g., microscope, surgical loupes, high-magnification endoscope). The surgeon typically has little or no tactile appreciation of the forces being exerted by the surgical instruments and physiological hand tremor can be a significant factor limiting surgical performance. Robotic systems can help overcome these human sensory-motor limitations, and efforts to develop specialized systems for applications such as ophthalmology [63.6, 109–115]) and otology [63.116–118], Several groups have also experimented with magnetic manipulation for these applications [63.113, 119]. There have been several efforts to compare microsurgical anastamosis procedures using laparoscopic telesurgical systems to conventional microsurgery. *Schiff* et al. [63.120] among others reported significant reductions in tremor with either robot and significantly improved technical quality and operative times compared to conventional microsurgery. As the nubber of da Vinci systems has proliferated, such applications are increasingly common [63.121]. A number of groups have implemented telesurgery systems specifically for microsurgery [63.13, 122–124] [63.95, 109]. These systems are in various stages of development, from laboratory prototype to preliminary clinical experimentation.

Not all microsurgical robots are teleoperated. For example, the cooperatively controlled JHU Steady Hand robots [63.4, 5] [63.6, 115] shown in Fig. 63.3 are being developed for retinal, head-and-neck, neurosurgery, and other microsurgical applications. A modified version of this system has also been used for microinjections into single mouse embryos [63.125].

There have also been efforts to develop completely hand-held instruments that actively cancel physiological tremor, for example, *Riviere* et al. [63.126–128] have developed an ophthalmic instrument using optical sensors to sense handle motion and adaptive filtering to estimate the tremulous component of instrument motion. A micromanipulator built into the instrument deflects the tip with an equal but opposite motion, compensating the tremor. Simple mechanical devices [63.129] for reducing tremor in specific tasks have also been developed.

An additional type of hand-held microsurgical and micro-therapeutic devices is reported in [63.130], which describes an active microendoscope for neuroendoscopy and therapy of the spinal cord able to safely navigate in the subarachnoid space and to avoid dangerous contact with the internal delicate structures thanks to a system based on hydrojets. Hydrojets come from the lateral surface of the catheter and, appropriately tuned and oriented, allow the tip of the endoscope to proceed without touching the spinal cord internal walls. The shared control system of the neuroendoscope, based on processing, segmentation, and analysis of the endoscopic images, assists the safe advancement of the tool in real time [63.131].

63.3.6 Endoluminal Robots

The term *endoluminal surgery* was first coined by *Cuschieri* et al. [63.132] as a major component of endoscopic surgery. Endoluminal procedures consist of bringing a set of advanced therapeutic and surgical tools to the area of interest by navigating in the lumina (i. e., the tube-like structures) of the human body, such as the gastrointestinal (GI) tract, the urinary tract, the circulatory system, etc. Currently, most endoluminal robots are designed for gastrointestinal applications, although there has been some initial work for other areas. There are several advantages (from a robotics research perspective) of working in the GI tract. The GI tract is not sterile and relatively large in diameter. Further it can be punctured intentionally to reach other abdominal cavities in NOTES approaches.

Traditionally, catheters and flexible endoscopes for endoluminal procedures have been inserted and manipulated manually from outside the body with the assistance of one or more visualization systems (e.g., direct endoscopic video, x-ray fluoroscopy, ultrasound). One major challenge is limited dexterity making it difficult to reach the desired target. Typically, flexible endoscopes have a bendable tip that can be steered by means of cable drives, and catheters may have only a fixed bend on a guide wire. There is also the inherent difficulty of *pushing a rope*, which some companies are trying to address by using external magnetic field (e.g., [63.133]). Once the target site is reached, these limitations become even more significant. Very simple instruments can be inserted through working channels or slid over guide wires, but dexterity is severely limited and there is no force feedback beyond what can be felt through the long, flexible instrument shaft.

These limitations have led a number of researchers to explore integration of more degrees of freedom in the catheter/endoscope body, as well as the design of intelligent tips with higher dexterity and sensing capabilities. Early work by Ikuta et al. led to the development of a five-segment, 13 mm-diameter sigmoidscope using shape-memory alloy (SMA) actuators.

Subsequently, *Ikuta* et al. developed 3 mm-diameter active endovascular devices using hydraulic actuators incorporating a novel band pass valve fabricated using micro-stereolithographic techniques [63.29].

Several examples exist of instrumented catheter tips with force sensors [63.134] that allow the right branch of the circulatory systems to be found by estimating the force generated between the tip and the vessel walls. Basically, these sensorized endoluminal devices belong to the larger group of micro-electromechanical systems (MEMS)-instrumented surgical devices and the same sensing technologies can be also exploited for microsurgery. A survey article by *Rebello* [63.135] provides an excellent overview of sensorized catheters and other MEMS-based devices in endoluminal and microsurgical applications.

Another approach to endoluminal robots is represented by systems that move under their own power through the body, rather than being pushed. Early work on such systems is well summarized in [63.136]. In 1995 Burdick et al. developed an inchworm-like mechanism for use in the colon. This device combined a central extensor for propulsion and inflatable balloons for friction enhancement with the slippery colon tissue. A more advanced inchworm design for a semi-autonomous robotic colonoscope was developed by *Dario* et al. [63.33] (Fig. 63.10). This device consists of a central silicone elongator, two clamping systems based on suction and gentle mechanical grasping of the colon tissue, and a silicone steerable tip integrating a complementary metal-oxide-semiconductor (CMOS) camera and a light-emitting diode (LED)-based illumination system. Thanks to its intrinsic flexibility, the robotic colonoscope applies forces on colon tissues that are ten times lower than those produced by traditional colonoscopies. This system is now clinically tested [63.137], and similar devices combining flexibility and painless operation have been proposed by some companies [63.138, 139]. Although the application is not endoluminal, the HeartLander system of *Riviere* et al. [63.31] shown in Fig. 63.11a,b shares many of the characteristics of these systems. It uses an inchworm-like gait to traverse the surface of the heart and to perform simple operation. An instructive paper on the combination between flexibility and stiffness of endoscopic devices for allowing painless manoeuvrability and stable anchoring when performing surgical tasks has been recently published [63.140]. It presents a number of design criteria and solutions for transforming endoscopic devices from diagnostic tools to stable surgical paltforms.

The natural evolution of GI endoscopic devices is represented by endoscopic capsules [63.141] which keep the promise to make endoscopy of the GI tract

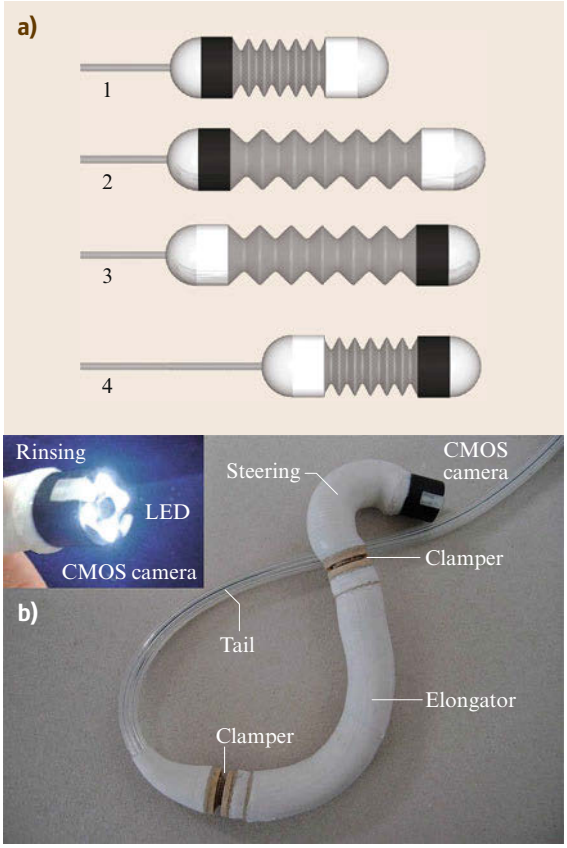


Fig.63.10a,b Medical robot for colonoscopy (after [63.33]): **(a)** the *gait* cycle of the robot, consisting of: (1) proximal clamping, (2) elongation, (3) distal clamping, and (4) retraction; **(b)** a recent working prototype used for clinical trials

a screening method highly tolerated by patients. For transforming simple CMOS swallowable cameras with illumination and transmission functionalities into useful diagnostic devices, several research groups have explored variegated solutions for active capsule locomotions. An example of legged locomotion capsules for GI application is illustrated in Fig. 63.11c and described in [63.28, 142]. In order to overcome dramatic powering problems, magnetic capsules propelled by permanent magnets [63.143] or by modified MRI fields [63.144] have been proposed. An earlier application of electromagnetic manipulation of an object within the body was the *video tumor fighter* of *Ritter* et al. [63.145].

63.3.7 Sensorized Instruments and Haptic Feedback

Surgical procedures almost always involve some form of physical interaction between surgical instruments

and patient anatomy, and surgeons are accustomed to use their haptic appreciation of tool-to-tissue interaction forces in performing surgical procedures. In situations such as where the clumsiness of instrumentation or human sensory-motor limitations limit the surgeon's ability to feel these forces, surgeons have been trained to rely on visual or other cues to compensate. Apart from the need for haptic feedback for diagnostic procedures and tissue identification, it has been demonstrated that reduced tactile and force information can result in the application of unnecessarily large forces on the patient during surgery, with possible harm to the patient [63.146]. The quality of haptic feedback available in currently deployed surgical robots is poor or nonexistent. Current research addresses the limitations firstly by integrating force and other sensors into surgical end-effectors and secondly by developing improved methods for processing and conveying the sensed information to the surgeon. However, the debate of the usefulness of force feedback in robot-assisted surgery is still open and is made more complex by the many scientific and technical difficulties due to the intrinsic presence of (possibly variable) communication time delay in the robotic system [63.147].

Although the most obvious way to *display* haptic information in a telesurgical system is directly through the master hand controllers, this method has several limitations, including friction and limited bandwidth in the hand controllers. Although these issues may be addressed through specialized design (which may raise costs and require other compromises) and improved control, there has been considerable interest in *sensory substitution* schemes [63.149–151] in which force or other sensor information is displayed visually, aurally, or by other means. Figure 63.8c shows one example of sensory substitution for warning when a da Vinci robot is about to break a suture [63.63].

Starting in the 1990s, several groups [63.151–153] have *sensorized* surgical instruments for microsurgery and MIS by equipping them with force sensors. Generally, these efforts relied on sensory substitution to display force data, either for freehand or telesurgical application. For example, *Poulose* et al. [63.152] demonstrated that a force sensing instrument used together with an IBM/JHU LARS robot [63.25] could significantly reduce both average retraction force and variability of retraction force during Nissen fundoplication. The first surgical robot with force feedback used in in vivo experiments was the JPL-NASA RAMS system, which was tested on animal models for vascular anastomosis [63.154] and carotid arteriotomies [63.155]. *Rosen* et al. [63.153] developed a force-controlled teleoperated endoscopic grasper equipped with position

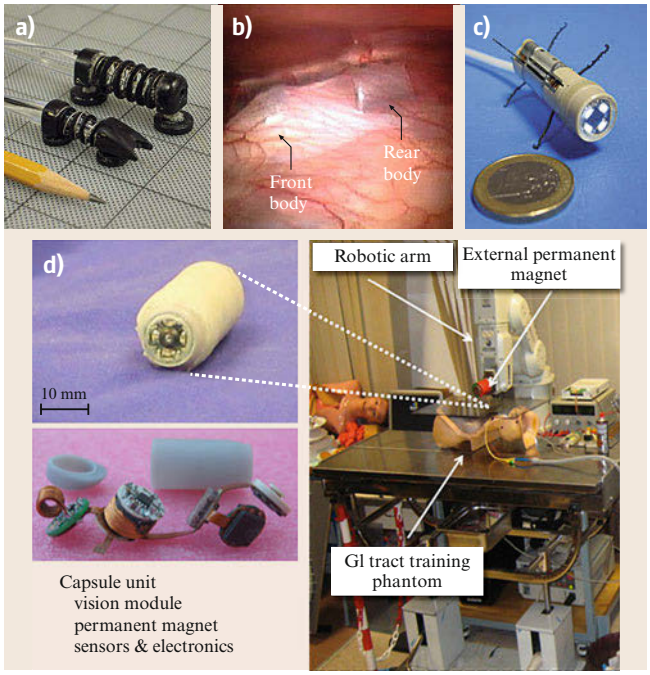


Fig.63.11a–d Mobility inside the body. **(a,b)** *HeartLander* device for crawling across the surface of the heart (after [63.31]). **(c)** Legged capsule for gastrointestinal diagnosis and therapy (after [63.28, 148]), **(d)** magnetic capsule for exploration of the GI tract, showing *left* capsule and components and *right* magnetic dragging platform based on a permanent magnet driven by a robotic manipulator (after [63.143])

sensors and actuated by direct-current (DC) motors whose output torque is sensed and fed back through motors integrated in a grasping handle. A similar approach was used by *Menciassi* et al. [63.156] for a microsurgery gripper equipped with semiconductor strain gauges and a PHANTOM (SensAble Technologies, Inc.) haptic interface. More recent work at Johns Hopkins has focused on incorporation of optical fiber force [63.157–160] and OCT [63.161–163] sensors into microsurgical tools. Both video and auditory sensory substitution [63.164], as well as direct feedback to robotic devices have been used to help surgeons control tool-tissue interaction forces and distance.

Several researchers [63.165] have focused on specialized *fingers* and display devices for palpation tasks requiring delicate tactile feedback (e.g., for detecting hidden blood vessels or cancerous tissues beneath normal tissues). There has also been work to integrate nonhaptic sensors into surgical instruments, for example, *Fischer* et al. have developed instruments measuring both force and tissue oxygenation levels [63.166]. This information can be used for such purposes as assessing tissue viability, distinguishing between different tissue

types, and controlling retraction so as not to cause ischemic tissue damage.

Finally, it is important to note that sensorized surgical tools have important application beyond their direct use in surgical procedures, for example, one use is in biomechanical studies to measure organ and tissue mechanical properties to improve surgical simulators.

63.3.8 Surgical Simulators and Telerobotic Systems for Training

Medical education is undergoing significant changes. The traditional paradigm for surgical technical training is often summarized as *see one, do one, teach one*. This method can be effective in open surgery, where the surgical trainee directly observes the expert surgeon hands, sees the instrument motion, and follows the organ manipulation. However, in endoscopic surgery it is difficult to observe the surgeon's hand movements (outside the body) and the surgical tool actions (inside the body and only visible on a video monitor). In addition, endoscopic surgery requires different skills than open surgery, such as spatial orientation, interpretation of 2-D images in 3-D, and manipulating instruments through entry portals. These considerations led to introduction of surgical simulation systems of varying degrees of complexity and realism for endoscopic and other minimally invasive procedures. Nowadays, training simulators have achieved widespread acceptance in the field of anaesthesia, intensive care, flexible endoscopy, surgery, interventional radiology, and other fields. The use of simulators for training is so common that working groups have been set up in order to evaluate these training systems based on shared guidelines [63.167] and many teaching hospitals have extensive simulation training centers. Simulators are being validated on their basic parameters (face, content and construct) [63.168], whereas results on concurrent and predictive validity of simulation training are still not available.

Survey [63.169] divides training simulators into three groups, depending on the type of technology used: mechanical, hybrid, and virtual reality.

Mechanical simulators are boxes where objects or organs are placed and manipulated using surgical instruments. The manipulation is observed through a laparoscope and a screen. The simulated surgical task is observed by an experienced surgeon, who gives feedback to the trainee. Generally, there are no registration processes and the simulator must be reassembled after any training session. The *LapTrainer with SimuVision* (Simulab Inc., Seattle, USA) is a training system with a simulated laparoscope that consists of a boom-mounted digital camera in an open box trainer.

A hybrid simulator uses a box with objects or organs as a mechanical simulator, but in addition the performance of the trainee are monitored by a computer which gives guidance for the task execution and an objective feedback based on preplanned metrics. Thanks to this feedback, the assistance and judgement of an experienced surgeon are not strictly necessary, for example, the ProMIS (Haptica Inc., Boston, USA) is a typical hybrid simulator for training basic skills such as suturing and knot tying. Surgical instruments are passed through dedicated ports and the trainee receives the same haptic feedback as in real surgery during manipulation in the box. In addition, ProMIS analyzes the trainee's performance by tracking the instrument position in 3-D and by measuring the execution time, path length, and smoothness of task execution. Another recent hybrid simulator is the Perk Tutor open-source platform for ultrasound-guided percutaneous needle insertion training [63.170], which has been successfully applied in teaching of ultrasound-guided facet joint injections [63.171]. Specific to robotic surgery is the RoSS system [63.172], which simulates the operator console of the da Vinci robotic system. Intuitive Surgical currently markets the dv-Trainer [63.173] simulator for the da Vinci system, comprising the surgeon console from a da Vinci *Si* system and a computer back end to simulate the patient-side manipulators, and its efficacy is being evaluated [63.174].

Finally, *virtual-reality* training systems combine visualization and haptic interfaces to enable surgeons to interact efficiently and naturally with real-time computational models of patient anatomy [63.175]. The development of these systems is inherently a multidisciplinary effort, including real-time computer graphics, the development of high-bandwidth haptic devices, real-time biomechanical modeling of organs, tool–tissue interactions, expertise in training assessment, and human–machine information exchange, etc. [63.176]. Research in these areas is closely related to and synergistic with comparable developments in technology and systems for performing real interventions, for example, modeling of organ motion in response to forces is necessary to improve the accuracy of needle placement procedures. Haptic feedback devices must meet similar requirements whether the forces displayed are simulated or measured directly in telesurgery, and so on. Finally, as noted earlier, *sensorized* instruments and real-time imaging are critical sources of data needed to create realistic biomechanical models. The use of the computer graphical processor (GPU) allows the simulation of organs at rates in excess of 10 kHz [63.177], which enables good haptic feedback. Similarly, new physics-based graphical libraries support the development of low cost virtual reality simu-

lators that correct render the interaction of instruments and anatomical parts, and thus can support large training classes [63.178].

The variety of interface devices and virtual reality laparoscopic simulators is quite wide and increasing numbers of systems are becoming commercially available. The Phantom interface is used in conjunction with virtual simulators to provide users with realistic haptic feedback (SensAble Technologies Inc., Woburn, MA, USA). The Xitact LS500 laparoscopy simulator (Xitact S.A., Lausanne, Switzerland) is a modular virtual-reality simulation platform with software for training and assessing performance in laparoscopic tasks. It is an open system including all or some of these subsystems: laparoscopic tools, mechanical abdomen, a personal computer (PC) providing the virtual-reality scenario, a haptic interface, a force feedback system and a tracking system for the tools. Several other systems for virtual reality simulation exist that exploit the hardware from Xitact or Immersion Medical, Inc. (Gaithersburg, MD, USA) and that are dedicated to specific surgical tasks: Lapmentor [63.179], the Surgical Education platform [63.180], LapSim [63.181], Procedicus MIST [63.182], EndoTower [63.183], the Reachin laparoscopic trainer [63.184], Simendo [63.185], and the Vest system [63.186]. Specifically for training eye surgeons, the surgical simulator EYESi [63.187] uses advanced computer technology and virtual reality to simulate the feel of real eye surgery, making it possible for surgeons at all levels to acquire new skills and perfect their techniques in preparation for surgery on the human eye. Training for dexterity skills in robotic surgery is supported by the new virtual simulator *XRON* developed at the University of Verona, which interfaces physics-based simulation with several commercial joysticks and provides evaluation metrics of interactive tasks [63.188].

Making a comparison between these different categories of simulators is not trivial. Basically, box trainers and hybrid simulators require some experienced technicians for the set up and some organizational logistics due to legal and ethical factors related to the storage of freshly explanted organs. The most evident advantage of these simulators is that the tactile response from the manipulated objects is the same as in real surgery and complicated models of organs and tissue–tool interaction are not required. On the other hand, completely virtual-reality trainers are potentially very flexible, can take advantage of powerful graphical engines, but they are limited by the availability of realistic calibration data for the anatomical and biomechanical models. Although demonstrations exist of the ability of simulators to record, objectively score, and hone the psychomotor skills of novice surgeons [63.189], the debate about their value is still open because no multicentre trails has yet been conducted to determine their efficacy. Furthermore, it has been shown that simulation training must be included in structured curricula in robotic surgery, where trainees are also introduced to basic concepts of robotics and, in particular, to the so called *non technical* skills, e.g., organization, leadership and communication, which significantly affect the surgical outcome [63.190].

63.3.9 Other Applications and Research Areas

The research areas described above illustrate major themes within medical robotics, but they are by no means exhaustive. Many important application areas such as otolaryngology [63.191–194] and radiation therapy have necessarily been excluded for space reasons. For a fuller exploration, readers should consult the further reading suggestions in Sect. 63.4.

63.4 Conclusion and Future Directions

Medical robotics (and the larger field of *computer-integrated interventional medicine*) has great potential to revolutionize clinical practice by:

- Exploiting technology to transcend human limitations in treating patients
- Improving the safety, consistency, and overall quality of interventions
- Improving the efficiency and cost-effectiveness of robot-assisted patient care
- Improving training through the use of validated simulators, quantitative data capture and skill assessment methods, and the capture and playback of clinical cases
- Promoting more effective use of information at all levels, both in treating individual patients and in improving treatment processes.

From being the stuff of late-night comedy and science fiction 20 years ago, the field has reached a critical threshold, with clinically useful systems and commercial successes. The scope and number of research programs has grown exponentially in the past 8 years, and this chapter is by no means a comprehensive survey

of the field. In fact, such a survey would be practically impossible in less than 100 pages. Many important emerging systems and applications are necessarily left out. Interested readers are urged to refer to the further reading section for more complete treatments. In particular, the survey articles and books in listed at the end of this section collectively contain somewhat fuller bibliographies than space permits here.

In the future, we can expect to see continued research in all aspects of technology and system development, with increasing emphasis on clinical applications. As this work proceeds, it is important that researchers remember several basic principles. The first, and arguably most important, principle is that medical robotics is fundamentally a *team* activity, involving academic researchers, clinicians, and industry. Each of these groups has unique expertise, and success comes from effective, highly interactive partnerships drawing upon this expertise. Building these teams takes a long-term commitment, and the successes in recent years are largely the pay-off from investments in creating these teams. Second, it is important to work on problems with well-defined clinical and technical goals, in which the criteria for measuring success are ultimately related to real advantages in treating patients. In working toward these goals, it is important to have measurable and meaningful milestones and to emphasize rapid iteration with clinician involvement at all stages. Finally, it is essential that all team members recognize the level of commitment that is required to achieve success and that they enjoy what they are doing.

Video-References

VIDEO 823 Da Vinci Surgery on a grape
available from http://handbookofrobotics.org/view-chapter/63/videodetails/823

VIDEO 824 Da Vinci Xi introduction | Engadget
available from http://handbookofrobotics.org/view-chapter/63/videodetails/824

VIDEO 825 Intuitive surgical Da Vinci single port robotic system
available from http://handbookofrobotics.org/view-chapter/63/videodetails/825

VIDEO 826 SPORT system by Titan Medical
available from http://handbookofrobotics.org/view-chapter/63/videodetails/826

VIDEO 827 Robot for single port surgery by Nebraska University
available from http://handbookofrobotics.org/view-chapter/63/videodetails/827

VIDEO 828 Magnetic and needlescopic instruments for surgical procedures
available from http://handbookofrobotics.org/view-chapter/63/videodetails/828

VIDEO 829 CardioArm
available from http://handbookofrobotics.org/view-chapter/63/videodetails/829

VIDEO 830 Snake robot for surgery in tight spaces
available from http://handbookofrobotics.org/view-chapter/63/videodetails/830

VIDEO 831 IREP robot – Insertable robotic effectors in single port surgery
available from http://handbookofrobotics.org/view-chapter/63/videodetails/831

VIDEO 832 Variable stiffness manipulator based on layer jamming
available from http://handbookofrobotics.org/view-chapter/63/videodetails/832

VIDEO 833 Reconfigurable and modular robot for NOTES applications
available from http://handbookofrobotics.org/view-chapter/63/videodetails/833

VIDEO 834 SPRINT robot for single port surgery
available from http://handbookofrobotics.org/view-chapter/63/videodetails/834

VIDEO 835 A micro-robot operating inside eye
available from http://handbookofrobotics.org/view-chapter/63/videodetails/835

References

63.1 R.H. Taylor, L. Joskowicz: Computer-integrated surgery and medical robotics. In: *Standard Handbook of Biomedical Engineering and Design*, ed. by M. Kutz (McGraw-Hill, New York 2003) pp. 325–353

63.2 J.R. Adler, M.J. Murphy, S.D. Chang, S.L. Hankock: Image guided robotic radiosurgery, Neurosurgery **44**(6), 1299–1306 (1999)

63.3 G.S. Guthart, J.K. Salisbury: The intuitive telesurgery system: Overview and application, IEEE Int. Conf. Robot. Autom. (ICRA), San Francisco (2000) pp. 618–621

63.4 R. Taylor, P. Jensen, L. Whitcomb, A. Barnes, R. Kumar, D. Stoianovici, P. Gupta, Z.X. Wang, E. deJuan, L. Kavoussi: Steady-hand robotic system for microsurgical augmentation, Int. J. Robotics Res. **18**, 1201–1210 (1999)

63.5 D. Rothbaum, J. Roy, G. Hager, R. Taylor, L. Whitcomb: Task performance in stapedotomy: comparison between surgeons of different experience

levels, Otolaryngol. Head Neck Surg. **128**, 71–77 (2003)
63.6 A. Uneri, M. Balicki, J. Handa, P. Gehlbach, R. Taylor, I. Iordachita: New steady-hand eye robot with microforce sensing for vitreoretinal surgery research, Proc. Int. Conf. Biomed. Robotics Biomechatronics (BIOROB), Tokyo (2010) pp. 814–819
63.7 A. Amin, J. Grossman, P.J. Wang: Early experience with a computerized robotically controlled catheter system, J. Int. Card. Electrophysiol. **12**, 199–202 (2005)
63.8 B. Hagag, R. Abovitz, H. Kang, B. Schmidtz, M. Conditt: RIO: Robotic-arm interactive orthopedic system MAKOplasty: User interactive haptic orthopedic robotics. In: *Surgical Robotics: Systems Applications and Visions*, ed. by J. Rosen, B. Hannaford, R. Satava (New York, Springer 2011) pp. 219–246
63.9 R. Konietschke, D. Zerbato, R. Richa, A. Tobergte, P. Poignet, F. Froehlich, D. Botturi, P. Fiorini, G. Hirzinger: Of new features for telerobotic surgery into the MiroSurge System, J. Appl. Bionics Biomech. Integr. **8**(2), 116 (2011)
63.10 L. Mettler, M. Ibrahim, W. Jonat: One year of experience working with the aid of a robotic assistant (the voice-controlled optic holder AESOP) in gynaecological endoscopic surgery, Hum. Reprod. **13**, 2748–2750 (1998)
63.11 N. Simaan, R. Taylor, P. Flint: High dexterity snake-like robotic slaves for minimally invasive telesurgery of the throat, Proc. Int. Symp. Med. Image Comput. Comput. Interv. (2004) pp. 17–24
63.12 N. Suzuki, M. Hayashibe, A. Hattori: Development of a downsized master-slave surgical robot system for intragastic surgery, Proc. ICRA Surg. Robotics Workshop, Barcelona (2005)
63.13 K. Ikuta, K. Yamamoto, K. Sasaki: Development of remote microsurgery robot and new surgical procedure for deep and narrow space, Proc. IEEE Conf. Robot. Autom. (ICRA) (2003) pp. 1103–1108
63.14 R.J. Webster, J.M. Romano, N.J. Cowan: Mechanics of precurved-tube continuum robots, IEEE Trans. Robotics **25**(1), 67–78 (2009)
63.15 R.J. Webster, B.A. Jones: Design and kinematic modeling of constant curvature continuum robots: A review, Int. J. Robotics Res. **29**(13), 1661–1683 (2010)
63.16 J. Ding, R.E. Goldman, K. Xu, P.K. Allen, D.L. Fowler, N. Simaan: Design and coordination kinematics of an insertable robotic effectors platform for single-port access surgery, IEEE/ASME Trans. Mechatron. **99**, 1–13 (2012)
63.17 R.H. Taylor, H.A. Paul, P. Kazandzides, B.D. Mittelstadt, W. Hanson, J.F. Zuhars, B. Williamson, B.L. Musits, E. Glassman, W.L. Bargar: An image-directed robotic system for precise orthopaedic surgery, IEEE Trans. Robot. Autom. **10**, 261–275 (1994)
63.18 P. Kazanzides, B.D. Mittelstadt, B.L. Musits, W.L. Bargar, J.F. Zuhars, B. Williamson, P.W. Cain, E.J. Carbone: An integrated system for cementless hip replacement, IEEE Eng. Med. Biol. **14**, 307–313 (1995)
63.19 Q. Li, L. Zamorano, A. Pandya, R. Perez, J. Gong, F. Diaz: The application accuracy of the NeuroMate robot – A quantitative comparison with frameless and frame-based surgical localization systems, Comput. Assist. Surg. **7**(2), 90–98 (2002)
63.20 P. Cinquin, J. Troccaz, J. Demongeot, S. Lavallee, G. Champleboux, L. Brunie, F. Leitner, P. Sautot, B. Mazier, A. Perez, M. Djaid, T. Fortin, M. Chenic, A. Chapel: IGOR: image guided operating robot, Innov. Technonogie Biol. Med. **13**, 374–394 (1992)
63.21 H. Reichenspurner, R. Demaino, M. Mack, D. Boehm, H. Gulbins, C. Detter, B. Meiser, R. Ellgass, B. Reichart: Use of the voice controlled and computer-assisted surgical system Zeus for endoscopic coronary artery surgery bypass grafting, J. Thorac. Cardiovasc. Surg. **118**, 11–16 (1999)
63.22 Y.S. Kwoh, J. Hou, E.A. Jonckheere, S. Hayati: A robot with improved absolute positioning accuracy for CT guided stereotactic brain surgery, IEEE Trans. Biomed. Eng. **35**, 153–161 (1988)
63.23 J.M. Sackier, Y. Wang: Robotically assisted laparoscopic surgery. From concept to development, Surg. Endosc. **8**, 63–66 (1994)
63.24 D. Stoianovici, L. Whitcomb, J. Anderson, R. Taylor, L. Kavoussi: A modular surgical robotic system for image-guided percutaneous procedures, Proc. Med. Image Comput. Comput. Interv. (MICCAI), Cambridge (1998) pp. 404–410
63.25 R.H. Taylor, J. Funda, B. Eldridge, K. Gruben, D. LaRose, S. Gomory, M. Talamini, L.R. Kavoussi, J. Anderson: A telerobotic assistant for laparoscopic surgery, IEEE Eng. Med. Biol. Mag. **14**, 279–287 (1995)
63.26 M. Mitsuishi, T. Watanabe, H. Nakanishi, T. Hori, H. Watanabe, B. Kramer: A telemicrosurgery system with colocated view and operation points and rotational-force-feedback-free master manipulator, Proc. 2nd Int. Symp. Med. Robot. Comput. Assist. Surg., Baltimore (1995) pp. 111–118
63.27 K. Ikuta, M. Tsukamoto, S. Hirose: Shape memory alloy servo actuator system with electric resistance feedback and application for active endoscope. In: *Computer-Integrated Surgery*, ed. by R.H. Taylor, S. Lavallee, G. Burdea, R. Mosges (MIT Press, Cambridge 1996) pp. 277–282
63.28 A. Menciassi, C. Stefanini, S. Gorini, G. Pernorio, P. Dario, B. Kim, J.O. Park: Legged locomotion in the gastrointestinal tract problem analysis and preliminary technological activity, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 937–942
63.29 K. Ikuta, H. Ichikawa, K. Suzuki, D. Yajima: Multi-degree of freedom hydraulic pressure driven safety active catheter, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Orlando (2006) pp. 4161–4166
63.30 S. Guo, J. Sawamoto, Q. Pan: A novel type of microrobot for biomedical application, Proc.

IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005) pp. 1047–1052
63.31 N. Patronik, C. Riviere, S.E. Qarra, M.A. Zenati: The heartlander: A novel epicardial crawling robot for myocardial injections, Proc. 19th Int. Congr. Comput. Assist. Radiol. Surg. (2005) pp. 735–739
63.32 P. Dario, B. Hannaford, A. Menciassi: Smart surgical tools and augmenting devices, IEEE Trans. Robotics Autom. **19**, 782–792 (2003)
63.33 L. Phee, D. Accoto, A. Menciassi, C. Stefanini, M.C. Carrozza, P. Dario: Analysis and development of locomotion devices for the gastrointestinal tract, IEEE Trans. Biomed. Eng. **49**, 613–616 (2002)
63.34 N. Kalloo, V.K. Singh, S.B. Jagannath, H. Niiyama, S.L. Hill, C.A. Vaughn, C.A. Magee, S.V. Kantsevoy: Flexible transgastricperitoneoscopy: A novel approach to diagnostic and therapeutic interventions in the peritoneal cavity, Gastrointest. Endosc. **60**, 114–117 (2004)
63.35 S.N. Shaikh, C.C. Thompson: Natural orifice translumenal surgery: Flexible platform review, World J. Gastrointest. Surg. **2**(6), 210–216 (2010)
63.36 M. Neto, A. Ramos, J. Campos: Single port laparoscopic access surgery, Tech. Gastrointest. Endosc. **11**, 84–93 (2009)
63.37 Karl Storz Endoskope – Anubis: https://www.karlstorz.com/cps/rde/xchg/SID-8BAF6233-DF3FFEB4/karlstorz-en/hs.xsl/8872.htm (2012)
63.38 S.J. Phee, K.Y. Ho, D. Lomanto, S.C. Low, V.A. Huynh, A.P. Kencana, K. Yang, Z.L. Sun, S.C. Chung: Natural orifice transgastric endoscopic wedge hepatic resection in an experimental model using an intuitively controlled master and slave transluminal endoscopic robot (MASTER), Surg. Endosc. **24**, 2293–2298 (2010)
63.39 M. Piccigallo, U. Scarfogliero, C. Quaglia, G. Petroni, P. Valdastri, A. Menciassi, P. Dario: Design of a novel bimanual robotic system for single-port laparoscopy, IEEE/ASME Trans. Mechatron. **15**(6), 871–878 (2012)
63.40 D.J. Scott, S.J. Tang, R. Fernandez, R. Bergs, M.T. Goova, I. Zeltser, F.J. Kehdy, J.A. Cadeddu: Completely transvaginal NOTES cholecystectomy using magnetically anchored instruments, Surg. Endosc. **21**, 2308–2316 (2007)
63.41 S. Tognarelli, M. Salerno, G. Tortora, C. Quaglia, P. Dario, A. Menciassi: An endoluminal robotic platform for Minimally Invasive Surgery, Proc. IEEE Int. Conf. Biomed. Robotics Biomechatronics (BIOROB), Pisa (2012) pp. 7–12
63.42 M. Simi, R. Pickens, A. Menciassi, S.D. Herrell, P. Valdastri: Fine tilt tuning of a laparoscopic camera by local magnetic actuation: Two-Port Nephrectomy Experience on Human Cadavers, Surg. Innov. **20**(4), 385–394 (2013)
63.43 T. Leuth: MIMED: 3D printing – Rapid technologies, http://www.mimed.mw.tum.de/research/3d-printing-rapid-technologies/ (2013)
63.44 S. G. O'Reilly: Medical manufacturing technology: 3D printing and medical-device development, Medical Design, http://medicaldesign.com/Medical-Manufacturing-Technology-3D-printing-medical-device-development/index.html (2012)
63.45 C. Plaskos, P. Cinquin, S. Lavallee, A.J. Hodgson: Praxiteles: A miniature bone-mounted robot for minimal access total knee arthroplasty, Int. J. Med. Robot. Comput. Assist. Surg. **1**, 67–79 (2005)
63.46 P.J. Berkelman, L. Whitcomb, R. Taylor, P. Jensen: A miniature microsurgical instrument tip force sensor for enhanced force feedback during robot-assisted manipulation, IEEE Trans. Robot. Autom. **19**, 917–922 (2003)
63.47 D.P. Devito, L. Kaplan, R. Dietl, M. Pfeiffer, D. Horne, B. Silberstein, M. Hardenbrook, G. Kiriyanthan, Y. Barzilay, A. Bruskin, D. Sackerer, V. Alexandrovsky, C. Stüer, R. Burger, J. Maeurer, D.G. Gordon, R. Schoenmayr, A. Friedlander, N. Knoller, K. Schmieder, I. Pechlivanis, I.-S. Kim, B. Meyer, M. Shoham: Clinical acceptance and accuracy assessment of spinal implants guided with SpineAssist surgical robot – Retrospective study, Spine **35**(24), 2109–2115 (2010)
63.48 K. Chinzei, R. Gassert, E. Burdet: Workshop on MRI/fMRI compatible robot technology – A critical tool for neuroscience and image guided intervention, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Orlando (2006)
63.49 M. Jakopec, S.J. Harris, F.R.Y. Baena, P. Gomes, J. Cobb, B.L. Davies: The first clinical application of a hands-on robotic knee surgery system, Comput. Aided Surg. **6**, 329–339 (2001)
63.50 D.F. Louw, T. Fielding, P.B. McBeth, D. Gregoris, P. Newhook, G.R. Sutherland: Surgical robotics: A review and neurosurgical prototype development, Neurosurgery **54**, 525–537 (2004)
63.51 J. Marescaux, J. Leroy, M. Gagner, F. Rubino, D. Mutter, M. Vix, S.E. Butner, M.K. Smith: Transatlantic robot-assisted telesurgery, Nature **413**, 379–380 (2001)
63.52 M. Anvari, T. Broderick, H. Stein, T. Chapman, M. Ghodoussi, D.W. Birch, C. Mckinley, P. Trudeau, S. Dutta, C.H. Goldsmith: The impact of latency on surgical precision and task completion during robotic-assisted remote telepresence surgery, Comput. Aided Surg. **10**, 93–99 (2005)
63.53 M. Li, M. Ishii, R.H. Taylor: Spatial motion constraints in medical robot using virtual fixtures generated by anatomy, IEEE Trans. Robotics **23**, 4–19 (2007)
63.54 S. Park, R.D. Howe, D.F. Torchiana: Virtual fixtures for robotic cardiac surgery, Proc. 4th Int. Conf. Med. Image Comput. Comput. Interv. (2001)
63.55 B. Davies: A discussion of safety issues for medical robots. In: *Computer-Integrated Surgery*, ed. by R. Taylor, S. Lavallée, G. Burdea, R. Moesges (MIT Press, Cambridge 1996) pp. 287–296
63.56 P. Varley: Techniques of development of safety-related software in surgical robots, IEEE Trans. Inform. Technol. Biomed. **3**, 261–267 (1999)
63.57 R. Muradore, D. Bresolin, L. Geretti, P. Fiorini, T. Villa: Robotic surgery – Formal verification of

plans, IEEE Robotics Autom. Mag. **18**(3), 24–32 (2011)
63.58 J.B. Maintz, M.A. Viergever: A survey of medical image registration, Med. Image Anal. **2**, 1–37 (1998)
63.59 S. Lavallee: Registration for computer-integrated surgery: methodology, state of the Art. In: *Computer-Integrated Surgery*, ed. by R.H. Taylor, S. Lavallee, G. Burdea, R. Mosges (MIT Press, Cambridge 1996) pp. 77–98
63.60 P.J. Besl, N.D. McKay: A method for registration of 3-D shapes, IEEE Trans. Pattern Anal. Mach. Intell. **14**, 239–256 (1992)
63.61 R.J. Maciunas: *Interactive Image-Guided Neurosurgery* (AANS, Park Ridge 1993)
63.62 G. Fichtinger, A. Degeut, K. Masamune, E. Balogh, G. Fischer, H. Mathieu, R.H. Taylor, S. Zinreich, L.M. Fayad: Image overlay guidance for needle insertion on CT scanner, IEEE Trans. Biomed. Eng. **52**, 1415–1424 (2005)
63.63 T. Akinbiyi, C.E. Reiley, S. Saha, D. Burschka, C.J. Hasser, D.D. Yuh, A.M. Okamura: Dynamic augmented reality for sensory substitution in robot-assisted surgical systems, Proc. 28th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (2006) pp. 567–570
63.64 J. Leven, D. Burschka, R. Kumar, G. Zhang, S. Blumenkranz, X. Dai, M. Awad, G. Hager, M. Marohn, M. Choti, C. Hasser, R.H. Taylor: DaVinci canvas: A telerobotic surgical system with integrated, robot-assisted, laparoscopic ultrasound capability, Proc. Med. Image Comput. Comput. Interv. Palm Springs (2005)
63.65 R.H. Taylor, H.A. Paul, C.B. Cutting, B. Mittelstadt, W. Hanson, P. Kazanzides, B. Musits, Y.Y. Kim, A. Kalvin, B. Haddad, D. Khoramabadi, D. Larose: Augmentation of human precision in computer-integrated surgery, Innov. Technol. Biol. Med. **13**, 450–459 (1992)
63.66 J. Troccaz, M. Peshkin, B. Davies: The use of localizers, robots and synergistic devices in CAS, Lect. Notes Comput. Sci. **1205**, 727–736 (1997)
63.67 G. Brisson, T. Kanade, A.M. DiGioia, B. Jaramaz: Precision freehand sculptig of bone, Proc. Med. Image Comput. Comput.-Assist. Interv. (MICCAI) (2004) pp. 105–112
63.68 R.A. Beasley: Medical Robots: Current systems and research directions, J. Robotics **2012**, 401613 (2012)
63.69 S. Kuang, K.S. Leung, T. Wang, L. Hu, E. Chui, W. Liu, Y. Wang: A novel passive/active hybrid robot for orthopaedic trauma surgery, Int. J. Med. Robotics Comput. Assist, Surg. **8**, 458–467 (2012)
63.70 S. Solomon, A. Patriciu, R.H. Taylor, L. Kavoussi, D. Stoianovici: CT guided robotic needle biopsy: A precise sampling method minimizing radiation exposure, Radiology **225**, 277–282 (2002)
63.71 K. Masamune, G. Fichtinger, A. Patriciu, R. Susil, R. Taylor, L. Kavoussi, J. Anderson, I. Sakuma, T. Dohi, D. Stoianovici: System for robotically assisted percutaneous procedures with computed tomography guidance, J. Comput. Assist. Surg. **6**, 370–383 (2001)
63.72 A. Krieger, R.C. Susil, C. Menard, J.A. Coleman, G. Fichtinger, E. Atalar, L.L. Whitcomb: Design of a novel MRI compatible manipulator for image guided prostate intervention, IEEE Trans. Biomed. Eng. **52**, 306–313 (2005)
63.73 T. Ungi, P. Abolmaesumi, R. Jalal, M. Welch, I. Ayukawa, S. Nagpal, A. Lasso, M. Jaeger, D. Borschneck, G. Fichtinger, P. Mousavi: Spinal needle navigation by tracked ultrasound snapshots, IEEE Trans. Biomed. Eng. **59**(10), 2766–2772 (2012)
63.74 D. DallAlba, B. Maris, P. Fiorini: A compact navigation system for free hand needle placement in percutaneous procedures, Proc. IEEE/RSI Int. Conf. Intell. Robots Syst. (IROS), Vilamoura (2012) pp. 2013–2018
63.75 G.A. Krombach, T. Schmitz-Rode, B.B. Wein, J. Meyer, J.E. Wildberger, K. Brabant, R.W. Gunther: Potential of a new laser target system for percutaneous CT-guided nerve blocks: Technical note, Neuroradiology **42**, 838–841 (2000)
63.76 M. Rosenthal, A. State, J. Lee, G. Hirota, J. Ackerman, K. Keller, E.D. Pisano, M. Jiroutek, K. Muller, H. Fuchs: Augmented reality guidance for needle biopsies: A randomized, controlled trial in phantoms, Proc. 4th Int. Conf. Med. Image Comput. Comput. Interv. (2001) pp. 240–248
63.77 G. Stetten, V. Chib: Overlaying ultrasound images on direct vision, J. Ultrasound Med. **20**, 235–240 (2001)
63.78 H.F. Reinhardt: Neuronagivation: A ten years review. In: *Computer-Integrated Surgery*, ed. by R. Taylor, S. Lavallée, G. Burdea, R. Moegses (MIT Press, Cambridge 1996) pp. 329–342
63.79 E. Hempel, H. Fischer, L. Gumb, T. Hohn, H. Krause, U. Voges, H. Breitwieser, B. Gutmann, J. Durke, M. Bock, A. Melzer: An MRI-compatible surgical robot for precise radiological interventions, Comput. Aided Surg. **8**, 180–191 (2003)
63.80 Z. Wei, G. Wan, L. Gardi, G. Mills, D. Downey, A. Fenster: Robot-assisted 3D-TRUS guided prostate brachytherapy: system integration and validation, Med. Phys. **31**, 539–548 (2004)
63.81 E. Boctor, G. Fischer, M. Choti, G. Fichtinger, R.H. Taylor: Dual-armed robotic system for intraoperative ultrasound guided hepatic ablative therapy: A prospective study, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2004) pp. 377–382
63.82 J. Hong, T. Dohi, M. Hashizume, K. Konishi, N. Hata: An ultrasound-driven needle-insertion robot for percutaneous cholecystostomy, Phys. Med. Biol. **49**, 441–455 (2004)
63.83 G. Megali, O. Tonet, C. Stefanini, M. Boccadoro, V. Papaspyropoulis, L. Angelini, P. Dario: A computer-assisted robotic ultrasound-guided biopsy system for video-assisted surgery, Proc. Med. Image Comput. Comput. Interv. (MICCAI), Utrecht (2001) pp. 343–350
63.84 R.J. Webster III, J.S. Kim, N.J. Cowan, G.S. Chirikjian, A.M. Okamura: Nonholonomic modeling of needle steering, Int. J. Robotics Res. **25**(5-6), 509–525 (2006)

63.85 J.A. Engh, D.S. Minhas, D. Kondziolka, C.N. Riviere: Percutaneous intracerebral navigation by duty-cycled spinning of flexible bevel-tipped needles, Neurosurgery **67**(4), 1117–1122 (2010)

63.86 N. Cowan, K. Goldberg, G. Chirikjian, G. Fichtinger, R. Alterovitz, K. Reed, V. Kallem, W. Park, S. Misra, A.M. Okamura: Robotic needle steering: Design, modeling, planning, and image guidance. In: *Surgical Robotics – Systems Applications and Visions*, ed. by J. Rosen, B. Hannaford, R.M. Satava (New York, Springer 2011)

63.87 S. Okazawa, R. Ebrahimi, J. Chuang, S. Salcudean, R. Rohling: Hand-held steerable needle device, IEEE/ASME Trans. Mechatron. **10**, 285–296 (2005)

63.88 P.E. Dupont, J. Lock, B. Itkowitz, E. Butler: Design and control of concentric-tube robots, IEEE Trans. Robotics **26**(2), 209–225 (2010)

63.89 G.H. Ballantyne: Robotic surgery, telerobotic surgery, telepresence and telementoring, Surg, Endosc. **16**, 1389 (2002)

63.90 A. Rovetta, R. Sala, R. Cosmi, X. Wen, S. Milassesi, D. Sabbadini, A. Togno, L. Angelini, A.K. Bejczy: A new telerobotic application: Remote laparscopic surgery using satellites and optical figer networks for data exchange, Int. J. Robotics Res. **15**(3), 267–279 (1996)

63.91 J. Arata, H. Takahashi, S. Yasunaka, K. Onda, K. Tanaka, N. Sugita, K. Tanoue, K. Konishi, S. Ieiri, Y. Fujino, Y. Ueda, H. Fujimoto, M. Mitsuishi, M. Hashizume: Impact of network time-delay and force feedback on tele-surgery, Int. J. Comput. Assist. Radiol. Surg. **3**(3–4), 371–378 (2008)

63.92 J.A. McEwen, C.R. Bussani, G.F. Auchinleck, M.J. Breault: Development and initial clinical evaluation of pre-robotic and robotic retraction systems for surgery, Proc. 2nd Workshop Med. Health Care Robotics, Newcastle (1989) pp. 91–101

63.93 P. Berkelman, P. Cinquin, J. Troccaz, J. Ayoubi, C. Letoublon, F. Bouchard: A compact, compliant laparoscopic endoscope manipulator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 1870–1875

63.94 K. Olds, A. Hillel, J. Kriss, A. Nair, H. Kim, E. Cha, M. Curry, L. Akst, R. Yung, J. Richmon, R. Taylor: A robotic assistant for trans-oral surgery: The robotic endo-laryngeal fexible (Robo-ELF) scope, J. Robotic. Surg. **6**(1), 13–18 (2012)

63.95 G.R. Sutherland, S. Lama, L.S. Gan, S. Wolfsberger, K. Zareinia: Merging machines with microsurgery: Clinical experience with neuroArm, J. Neurosurg. **118**, 521–529 (2013)

63.96 A. Nishikawa, T. Hosoi, K. Koara, T. Dohi: FAce MOUSe: A novel human-machine interface for controlling the position of a laparoscope, IEEE Trans Robot. Autom. **19**, 825–841 (2003)

63.97 A. Krupa, J. Gangloff, C. Doignon, M.F. deMathelin, G. Morel, J. Leroy, L. Soler, J. Marescaux: Autonomous 3-D positioning of surgical instruments in robotized laparoscopic surgery using visual servoing, IEEE Trans. Robotics Autom. **19**, 842–853 (2003)

63.98 P. Green, R. Satava, J. Hill, I. Simon: Telepresence: Advanced teleoperator technology of minimally invasive surgery (abstract), Surg. Endosc. **6**, 90 (1992)

63.99 T. Ahlering, D. Woo, L. Eichel, D. Lee, R. Edwards, D. Skarecky: Robot-assisted versus open radical prostatectomy: A comparison of one surgeon's outcomes, Urology **63**, 819–822 (2004)

63.100 J.D. Wright, C.V. Ananth, S.N. Lewin, W.M. Burke, Y.-S. Lu, A.I. Neugut, T.J. Herzog, D.L. Hershman: Robotically assisted vs laparoscopic hysterectomy among women with benign gynecologic disease, J. Am. Med. Assoc. **309**(7), 689–698 (2013)

63.101 J. Rosen, J.D. Brown, L. Chang, M. Barreca, M. Sinanan, B. Hannaford: The BlueDRAGON – a system for measuring the kinematics and the dynamics of minimally invasive surgical tools in-vivo, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 1876–1881

63.102 H.C. Lin, I. Shafran, T.E. Murphy, A.M. Okamura, D.D. Yuh, G.D. Hager: Automatic detection and segmentation of robot-assisted surgical motions, Proc. Med. Image Comput. Comput.-Assist. Interv. (MICCAI) (2005) pp. 802–810

63.103 H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll, J. Schmidhuber: A system for robotic heart surgery that learns to tie knots using recurrent neural networks, Proc. Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 543–548

63.104 C.E. Reiley, H.C. Lin, D.D. Yuh, G.D. Hager: A review of methods for objective surgical skill evaluation, Surg. Endosc. **25**(2), 356–366 (2011)

63.105 N. Padoy, G. Hager: Human-machine collaborative surgery using learned models, Proc. IEEE Int. Conf Robotics Autom. (ICRA) (2011) pp. 5285–5292

63.106 L.-M. Su, B.P. Vagvolgyi, R. Agarwal, C.E. Reiley, R.H. Taylor, G.D. Hager: Augmented reality during robot-assisted laparoscopic partial nephrectomy: Toward real-time 3D-CT to stereoscopic video registration, Urology **73**(4), 896–900 (2009)

63.107 F. Volonté, N.C. Buchs, F. Pugin, J. Spaltenstein, B. Schiltz, M. Jung, M. Hagen, O. Ratib, P. Morel: Augmented reality to the rescue of the minimally invasive surgeon. The usefulness of the interposition of stereoscopic images in the Da Vinci robotic console, Int. J. Med. Robotics Comput. Assist. Surg. **3**(9), 34–38 (2013)

63.108 D. Cohen, E. Mayer, D. Chen, A. Anstee, J. Vale, G.-Z. Yang, A. Darzi, P.Ä.Ä. Edwards: Augmented reality image guidance in minimally invasive prostatectomy, Lect. Notes Comput. Sci. **6367**, 101–110 (2010)

63.109 Y. Ida, N. Sugita, T. Ueta, Y. Tamaki, K. Tanimoto, M. Mitsuishi: Microsurgical robotic system for vitreoretinal surgery, Int. J. Comput. Assist. Radiol. Surg. **7**(1), 27–34 (2012)

63.110 R. Taylor, J. Kang, I. Iordachita, G. Hager, P. Kazanzides, C. Riviere, E. Gower, R. Richa, M. Balicki, X. He, X. Liu, K. Olds, R. Sznitman, B. Vagvolgyi, P. Gehlbach, J. Handa: Recent work toward a microsurgical assistant for retinal

surgery, Proc. Hamlyn Symp. Med. Robotics, London (2011) pp. 3–4
63.111 B. Becker, S. Yang, R. MacLachlan, C. Riviere: Towards vision-based control of a handheld micromanipulator for retinal cannulation in an eyeball phantom, Proc. IEEE RAS EMBS Int. Conf. Biomed. Robotics Biomechatronics (BIOROB) (2012) pp. 44–49
63.112 M. Balicki, T. Xia, M.Y. Jung, A. Deguet, B. Vagvolgyi, P. Kazanzides, R. Taylor: Prototyping a hybrid cooperative and tele-robotic surgical system for retinal microsurgery, Proc. MICCAI Workshop Syst. Archit. Comput. Assist. Interv. Tor. (2011), Insight, http://www.midasjournal.org/browse/publication/815
63.113 C. Bergeles, B.E. Kratochvil, B.J. Nelson: Visually servoing magnetic intraocular microdevices, IEEE Trans. Robotics **28**(4), 798–809 (2012)
63.114 Patent N. Simaan, J. T. Handa, R. H. Taylor: A system for macro-micro distal dexterity enhancement in microsurgery of the eye, Patent US 20110125165 A1 (2011)
63.115 X. He, D. Roppenecker, D. Gierlach, M. Balicki, K. Olds, J. Handa, P. Gehlbach, R.H. Taylor, I. Iordachita: Toward a clinically applicable steady-hand eye robot for vitreoretinal surgery, Proc. ASME Int. Mech. Eng. Congr., Houston (2012) p. 88384
63.116 J.K. Niparko, W.W. Chien, I. Iordachita, J.U. Kang, R.H. Taylor: Robot-assisted, sensor-guided cochlear implant electrode insertion (Abstract in Proceedings), Proc. 12th Int. Conf. Cochlear Implant. Other Implant. Auditory Technol., Baltimore (2012)
63.117 D. Schurzig, R.F. Labadie, A. Hussong, T.S. Rau, R.J. Webster: Design of a tool integrating force sensing with automated insertion in cochlear implantation, IEEE/ASME Trans. Mechatron. **17**(2), 381–389 (2012)
63.118 B. Bell, C. Stieger, N. Gerber, A. Arnold, C. Nauer, V. Hamacher, M. Kompis, L. Nolte, M. Caversaccio, S. Weber: A self-developed and constructed robot for minimally invasive cochlear implantation, Acta Oto-Laryngol. **132**(4), 355–360 (2012)
63.119 J.R. Clark, L. Leon, F.M. Warren, J.J. Abbott: Magnetic guidance of cochlear implants: Proof-of-concept and initial feasibility study, Trans. ASME-W-J. Medical Devices **6**(4), 35002 (2012)
63.120 J. Schiff, P. Li, M. Goldstein: Robotic microsurgical vasovasostomy and vasoepididymostomy: A prospective randomized study in a rat model, J. Urol. **171**, 1720–1725 (2004)
63.121 S. Parekattil, M. Cohen: Robotic microsurgery. In: *Robotic Urologic Surgery*, ed. by V.R. Patel (Springer, London 2012) pp. 461–470
63.122 P.S. Jensen, K.W. Grace, R. Attariwala, J.E. Colgate, M.R. Glucksberg: Toward robot assisted vascular microsurgery in the retina, Graefes Arch. Clin. Exp. Ophthalmol. **235**, 696–701 (1997)
63.123 H. Das, H. Zak, J. Johnson, J. Crouch, D. Frambaugh: Evaluation of a telerobotic system to assist surgeons in microsurgery, Comput. Aided Surg. **4**, 15–25 (1999)
63.124 K. Hongo, S. Kobayashi, Y. Kakizawa, J.I. Koyama, T. Goto, H. Okudera, K. Kan, M.G. Fujie, H. Iseki, K. Takakura: NeuRobot: Telecontrolled micromanipulator system for minimally invasive microneurosurgery-preliminary results, Neurosurgery **51**, 985–988 (2002)
63.125 A. Kapoor, R. Kumar, R. Taylor: Simple biomanipulation tasks with a steady hand cooperative manipulator, Proc. 6th Int. Conf. Med. Image Comput. Comput. Assist. Interv. (MICCAI), Montreal (2003) pp. 141–148
63.126 R. MacLachlan, B. Becker, J. Cuevas-Tabarés, G. Podnar, L. Lobes, C. Riviere: Micron: An actively stabilized handheld tool for microsurgery, IEEE Trans. Robotics **28**(1), 195–212 (2012)
63.127 B. Becker, R. MacLachlan, L. Lobes, G. Hager, C. Riviere: Vision-based control of a handheld surgical micromanipulator with virtual fixtures, IEEE Trans. Robotics **29**(3), 674–683 (2013)
63.128 C. Riviere, J. Gangloff, M. de Mathelin: Robotic compensation of biological motion to enhance surgical accuracy, Proceedings IEEE **94**(9), 1705–1716 (2006)
63.129 W. Armstrong, A. Karamzadeh, R. Crumley, T. Kelley, R. Jackson, B. Wong: A novel laryngoscope instrument stabilizer for operative microlaryngoscopy, Otolaryngol. Head Neck Surg. **132**, 471–477 (2004)
63.130 L. Ascari, C. Stefanini, A. Menciassi, S. Sahoo, P. Rabischong, P. Dario: A new active microendoscope for exploring the sub-arachnoid space in the spinal cord, Proc. IEEE Conf. Robotics Autom. (ICRA) (2003) pp. 2657–2662
63.131 U. Bertocchi, L. Ascari, C. Stefanini, C. Laschi, P. Dario: Human-robot shared control for robot-assisted endoscopy of the spinal cord, Proc. IEEE/RAS/EMBS Int. Conf. Biomed. Robot. Biomechatronics (2006) pp. 543–548
63.132 A. Cuschieri, G. Buess, J. Perissat: *Operative Manual of Endoscopic Surgery* (Springer, Berlin, Heidleberg 1992)
63.133 Stereotaxis: The epoch (TM) solution: The heart of innovation, http://www.stereotaxis.com/ (2013)
63.134 Endosense, TactiCath Quartz – The First Contact Force Ablation Catheter: http://neomed.net/portfolio/endosense_sa (2013)
63.135 K.J. Rebello: Applications of MEMS in surgery, Proceedings IEEE **92**(1), 43–55 (2004)
63.136 I. Kassim, W.S. Ng, G. Feng, S.J. Phee: Review of locomotion techniques for robotic colonoscopy, Proc. Int. Conf. Robotics Autom. (ICRA), Taipei (2003) pp. 1086–1091
63.137 Endotics: The endotics system, http://www.endotics.com/en/product (2012)
63.138 Gi-View: The AeroScope, http://www.giview.com/ (2013)
63.139 Invendo-Medical: Gentle colonoscopy with invendoscopy, http://www.invendo-medical.com/ (2013)

63.140 A. Loeve, P. Breeveld, J. Dankelman: Scopes too flexible ... and too stiff, IEEE Pulse **1**(3), 26–41 (2010)

63.141 G. Ciuti, A. Menciassi, P. Dario: Capsule endoscopy: From current achievements to open challenges, IEEE Rev. Biomed. Eng. **4**, 59–72 (2011)

63.142 C. Stefanini, A. Menciassi, P. Dario: Modeling and experiments on a legged microrobot locomoting in a tubular, compliant and slippery environment, Int. J. Robotics Res. **25**, 551–560 (2006)

63.143 G. Ciuti, P. Valdastri, A. Menciassi, P. Dario: Robotic magnetic steering and locomotion of capsule endoscope for diagnostic and surgical endoluminal procedures, Robotica **28**, 199–207 (2010)

63.144 J.F. Rey, H. Ogata, N. Hosoe, K. Ohtsuka, N. Ogata, K. Ikeda, H. Aihara, I. Pangtay, T. Hibi, S. Kudo, H. Tajiri: Feasibility of stomach exploration with a guided capsule endoscope, Endoscopy **42**, 541–545 (2010)

63.145 R.C. Ritter, M.S. Grady, M.A. Howard, G.T. Gilles: Magnetic stereotaxis: Computer assisted, image guided remote movement of implants in the brain. In: *Computer Integrated Surgery*, ed. by R.H. Taylor, S. Lavallee, G.C. Burdea, R. Mosges (MIT Press, Cambridge 1995) pp. 363–369

63.146 C.R. Wagner, N. Stylopoulos, P.G. Jackson, R.D. Howe: The benefit of force feedback in surgery: Examination of blunt dissection, Presence: Teleoperators Virtual Environ. **16**(3), 252–262 (2007)

63.147 P.F. Hokayem, M.W. Spong: Bilateral teleoperation: A historical survey, Automatica **42**, 2035–2057 (2006)

63.148 M. Quirini, A. Menciassi, S. Scapellato, C. Stefanini, P. Dario: Design and fabrication of a motor legged capsule for the active exploration of the gastrointestinal tract, IEEE/ASME Trans. Mechatron. **13**(1), 169–179 (2008)

63.149 A.M. Okamura: Methods for haptic feedback in teleoperated robot-assisted surgery, Ind. Robot **31**, 499–508 (2004)

63.150 M.J. Massimino: Improved force perception through sensory substitution, Contr. Eng. Pract. **3**, 215–222 (1995)

63.151 P. Gupta, P. Jensen, E. de Juan: Quantification of tactile sensation during retinal microsurgery, Proc. 2nd Int. Conf. Med. Image Comput. Comput.-Assist. Interv. (MICCAI), Cambridge (1999)

63.152 P.K. Poulose, M.F. Kutka, M. Mendoza-Sagaon, A.C. Barnes, C. Yang, R.H. Taylor, M.A. Talamini: Human versus robotic organ retraction during laparoscopic nissen fundoplication, Surg. Endosc. **13**, 461–465 (1999)

63.153 J. Rosen, B. Hannaford, M. MacFarlane, M. Sinanan: Force controlled and teleoperated endoscopic grasper for minimally invasive surgery – Experimental performance evaluation, IEEE Trans. Biomed. Eng. **46**, 1212–1221 (1999)

63.154 M. Siemionow, K. Ozer, W. Siemionow, G. Lister: Robotic assistance in microsurgery, J. Reconstr. Microsurg. **16**(8), 643–649 (2000)

63.155 P.D.L. Roux, H. Das, S. Esquenazi, P.J. Kelly: Robot-assisted microsurgery: A feasibility study in the rat, Neurosurgery **48**(3), 584–589 (2001)

63.156 A. Menciassi, A. Eisinberg, M.C. Carrozza, P. Dario: Force sensing microinstrument for measuring tissue properties and pulse in microsurgery, IEEE/ASME Trans. Mechatron. **8**, 10–17 (2003)

63.157 S. Sunshine, M. Balicki, X. He, K. Olds, J. Kang, P. Gehlbach, R. Taylor, I. Iordachita, J. Handa: A Force-sensing microsurgical instruments that detects forces below human tactile sensation, Retina **33**(1), 200–206 (2013)

63.158 I. Iordachita, Z. Sun, M. Balicki, J.U. Kang, S.J. Phee, J. Handa, P. Gehlbach, R. Taylor: A sub-millimetric, 0.25 mN resolution fully integrated fiber-optic force-sensing tool for retinal microsurgery, Int. J. Comput. Assist. Radiol. Surg. **4**(4), 383–390 (2009)

63.159 I. Kuru, B. Gonenc, M. Balicki, J. Handa, P. Gehlbach, R.H. Taylor, I. Iordachita: Force sensing micro-forceps for robot assisted retinal surgery, Proc. Eng. Med. Biol., San Diego (2012) pp. 1401–1404

63.160 X. Liu, I.I. Iordachita, X. He, R.H. Taylor, J.U. Kang: Miniature fiber-optic force sensor for vitreoretinal microsurgery based on low-coherence Fabry-Perot interferometry, Biomed. Opt. Express **3**, 1062–1076 (2012)

63.161 M. Balicki, J.-H. Han, I. Iordachita, P. Gehlbach, J. Handa, R.H. Taylor, J. Kang: Single fiber optical coherence tomography microsurgical instruments for computer and robot-assisted retinal surgery, Proc. Med. Image Comput. Comput.-Assist. Interv. (MICCAI), London (2009) pp. 108–115

63.162 M. Balicki, R. Richa, B. Vagvolgyi, J. Handa, P. Gehlbach, J. Kang, P. Kazanzides, R. Taylor: Interactive OCT annotation and visualization system for vitreoretinal surgery, Proc. MICCAI Workshop Augment. Environ. Comput. Interv. (AE-CAI), Nice (2012)

63.163 X. Liu, Y. Huang, J.U. Kang: Distortion-free freehand-scanning OCT implemented with real-time scanning speed variance correction, Opt. Express **20**(15), 16567–16583 (2012)

63.164 N. Cutler, M. Balicki, M. Finkelstein, J. Wang, P. Gehlbach, J. McGready, I. Iordachita, R. Taylor, J. Handa: Auditory force feedback substitution improves surgical precision during simulated ophthalmic surgery, Investig. Ophthalmol. Vis. Sci. **4**(2), 1316–1324 (2013)

63.165 R.D. Howe, W.J. Peine, D.A. Kontarinis, J.S. Son: Remote palpation technology, IEEE Eng. Med. Biol. **14**(3), 318–323 (1995)

63.166 G. Fischer, T. Akinbiyi, S. Saha, J. Zand, M. Talamini, M. Marohn, R. Taylor: Ischemia and force sensing surgical instruments for augmenting available surgeon information, Proc. IEEE Int. Conf. Biomed. Robot. Biomechatronics (BioRob), Pisa (2006)

63.167 F.J. Carter, M.P. Schijven, R. Aggarwal, T. Grantcharov, N.K. Francis, G.B. Hanna, J.J. Jakimowicz: Consensus guidelines for validation of

virtual reality surgical simulators, Surg. Endosc. **19**, 1523–1532 (2005)
63.168 A. Gavazzi, W.V. Haute, K. Ahmed, O. Elhage, P. Jaye, M.S. Khan, P. Dasgupta: Face, content and construct validity of a virtual reality simulator for robotic surgery (SEP Robot), Ann. R. Coll. Surg. Engl. **93**, 146–150 (2011)
63.169 F.H. Halvorsen, O.J. Elle, E. Fosse: Simulators in surgery, Minim. Invasive Ther. **14**, 214–223 (2005)
63.170 T. Ungi, D. Sargent, E. Moult, A. Lasso, C. Pinter, R. McGraw, G. Fichtinger: Perk Tutor: An open-source training platform for ultrasound-guided needle insertions, IEEE Trans. Biomed. Eng. **59**(12), 3475–3481 (2012)
63.171 E. Moult, T. Ungi, M. Welch, J. Lu, R. McGraw, G. Fichtinger: Ultrasound-guided facet joint injection training using Perk Tutor, Int. J. Comput. Assist. Radiol. Surg. **8**(5), 831–836 (2013)
63.172 T.K.A. Kesavadas, G. Srimathveeravalli, S. Karimpuzha, R. Chandrasekhar, G. Wilding, G.K.Z. Butt: Efficacy of robotic surgery simulator (RoSS) for the davinci surgical system, J. Urol. **181**(4), 823 (2009)
63.173 C. Perrenot, M. Perez, N. Tran, J.-P. Jehl, J. Felblinger, L. Bresler, J. Hubert: The virtual reality simulator dV-Trainer is a valid assessment tool for robotic surgical skills, Surg. Endosc. **26**(9), 2587–2593 (2012)
63.174 R. Korets, J. Graversen, M. Gupta, J. Landman, K. Badani: Comparison of robotic surgery skill acquisition between DV-Trainer and da Vinci surgical system: A randomized controlled study, J. Urol. **185**(4), e593 (2011)
63.175 K. Moorthy, Y. Munz, S.K. Sarker, A. Darzi: Objective assessment of technical skills in surgery, Br. Med. J. **327**, 1032–1037 (2003)
63.176 C. Basdogan, S. De, J. Kim, M. Muniyandi, H. Kim, M.A. Srinivasan: Haptics in minimally invasive surgical simulation and training, IEEE Comput. Graph. Appl. **24**(2), 56–64 (2004)
63.177 M. Altomonte, D. Zerbato, S. Galvan, P. Fiorini: Organ modeling and simulation using graphical processing, Poster Sess. Comput. Assist. Radiol. Surg., Berlin (2007)
63.178 D. Zerbato, L. Vezzaro, L. Gasperotti, P. Fiorini: Virtual training for US guided needle insertion, Proc. Comput. Assist Radiol. Surg., Berlin (2012) pp. 27–30
63.179 Lapmentor: http://www.simbionix.com
63.180 Surgical Education platform: http://www.meti.com
63.181 LapSim: http://www.surgical-science.com
63.182 MIST: http://www.mentice.com
63.183 EndoTower: http://www.verifi.com
63.184 Laparoscopic Trainer: http://www.reachin.se
63.185 Simendo: http://www.simendo.nl
63.186 Vest System: http://www.select-it.de
63.187 EYESI: http://www.vrmagic.com/
63.188 U. Verona: Xron – Surgical simulator by ALTAIR lab, http://metropolis.scienze.univr.it/xron/ (2013)
63.189 F. Cavallo, G. Megali, S. Sinigaglia, O. Tonet, P. Dario: A biomechanical analysis of surgeon's gesture in a laparoscopic virtual scenario. In: *Medicine Meets Virtual Reality*, Vol. 14, ed. by J.D. Westwood, R.S. Haluck, H.M. Hoffmann, G.T. Mogel, R. Phillips, R.A. Robb, K.G. Vosburgh (IOS, Amsterdam 2006) pp. 79–84
63.190 S. Yule, R. Flin, S. Paterson-Brown, N. Maran: Non-technical skills for surgeons: A review of the literature, Surgery **139**, 140–149 (2006)
63.191 K.T. Kavanagh: Applications of image-directed robotics in otolaryngologic surgery, Laryngoscope **194**, 283–293 (1994)
63.192 J. Wurm, H. Steinhart, K. Bumm, M. Vogele, C. Nimsky, H. Iro: A novel robot system for fully automated paranasal sinus surgery, Proc. Comput. Assist. Radiol. Surg. (CARS) (2003) pp. 633–638
63.193 M. Li, M. Ishii, R.H. Taylor: Spatial motion constraints in medical robot using virtual fixtures generated by anatomy, IEEE Trans. Robot. **2**, 1270–1275 (2006)
63.194 G. Strauss, K. Koulechov, R. Richter, A. Dietz, C. Trantakis, T. Lueth: Navigated control in functional endoscopic sinus surgery, Int. J. Med. Robot. Comput. Assist. Surg. **1**, 31–41 (2005)

Multimedia Contents

64. Rehabilitation and Health Care Robotics

H.F. Machiel Van der Loos, David J. Reinkensmeyer, Eugenio Guglielmelli

The field of rehabilitation robotics considers robotic systems that 1) provide therapy for persons seeking to recover their physical, social, communication, or cognitive function, and/or that 2) assist persons who have a chronic disability to accomplish activities of daily living. This chapter will discuss these two main domains and provide descriptions of the major achievements of the field over its short history and chart out the challenges to come. Specifically, after providing background information on demographics (Sect. 64.1.2) and history (Sect. 64.1.3) of the field, Sect. 64.2 describes physical therapy and exercise training robots, and Sect. 64.3 describes robotic aids for people with disabilities. Section 64.4 then presents recent advances in smart prostheses and orthoses that are related to rehabilitation robotics. Finally, Sect. 64.5 provides an overview of recent work in diagnosis and monitoring for rehabilitation as well as other health-care issues. The reader is referred to Chap. 73 for cognitive rehabilitation robotics and to Chap. 65 for robotic smart home technologies, which are often considered assistive technologies for persons with disabilities. At the conclusion of the present chapter, the reader will be familiar with the history of rehabilitation robotics and its primary accomplishments, and will understand the challenges the field may face in the future as it seeks to improve health care and the well being of persons with disabilities.

64.1 **Overview** 1686
64.1.1 Taxonomy of Rehabilitation Robotics 1686
64.1.2 World Demographics 1687
64.1.3 Short History of Rehabilitation Robotics 1688
64.2 **Rehabilitation Therapy and Training Robots** 1692
64.2.1 Grand Challenges and Roadblocks 1692
64.2.2 Movement Therapy After Neurologic Injury 1694
64.2.3 Robotic Therapy for the Upper Extremity 1695
64.2.4 Robotic Therapy for Walking 1699
64.3 **Aids for People with Disabilities** 1703
64.3.1 Grand Challenges and Enabling Technologies 1703
64.3.2 Types and Examples of Assistive Rehabilitation Robots 1705
64.4 **Smart Prostheses and Orthoses** 1711
64.4.1 Grand Challenges and Roadblocks 1711
64.4.2 Targeted Re-Innervation 1711
64.4.3 Neural Interfaces for Limb Prosthetic Devices 1712
64.4.4 Advances in Neural Stimulation 1713
64.4.5 Embedded Intelligence 1713
64.5 **Augmentation for Diagnosis and Monitoring** 1713
64.5.1 Grand Challenges and Enabling Technologies 1714
64.5.2 Smart Homes for Health Care Monitoring and Care 1714
64.5.3 Home-Based Rehabilitation Monitoring and Therapy Systems 1714
64.5.4 Wearable Monitoring Devices 1715
64.6 **Safety, Ethics, Access and Economics** 1715
64.7 **Conclusions and Further Readings** 1716
Video-References 1717
References 1718

64.1 Overview

In this chapter, we discuss an application of robotics that will likely touch many of us in an acutely personal way at some point in our lives. When, through injury or disease, we become unable to interact physically with our immediate environment as we desire to achieve our personal goals, or when one of our family members, friends, or neighbors is in this situation, technology-based solutions will likely be a major component in the treatment interventions that therapists prescribe to assist us in re-learning how to complete our activities of daily living (ADL), or to assist us in actually doing them if we are unable to recover lost function. While human therapists and attendants can indeed shoulder the types of assistance required, the projected short-term demographics of many countries, including China, Japan, and the Scandinavian countries, show a growing shortage of working-age adults. Age-related disabilities will soon substantially impact the service sector job market, put many older and disabled people at risk, and increase the need for institutionalization when there is no viable home-based solution. National programs to develop personal robots, robotic therapy, smart prostheses, smart beds, smart homes, and tele-rehabilitation services have accelerated in the past 20 years and will need to continue apace with the ever increasing ability of health care to allow people to live longer through the repression of disease and improvement in surgical and medication interventions. Rehabilitation robotics, although only a 50 year old discipline [64.1–3], is projected to have a fast-growing future in the coming decades. Within the past 10 years, the field has seen significant strides in the commercialization of rehabilitation robotics due to an increasing acceptance of the validity of this approach by clinical care providers, as well as the cost reductions in sensors and actuators. We have also witnessed the expansion of applications from the social robotics research domain into rehabilitation, increasing the range of persons whose impairments can be targeted by robotic technologies in the years to come.

64.1.1 Taxonomy of Rehabilitation Robotics

The field of rehabilitation robotics is generally divided between the categories of *therapy* and *assistance* robots, although some devices can serve both purposes. In addition, rehabilitation robotics includes aspects of artificial limb (prosthetics) development, functional neural stimulation (FNS) and technology for diagnosis and monitoring of people during activities of daily living.

Therapy robots generally have at least two main users simultaneously, the person with a disability who is receiving the therapy and the therapist who sets up and monitors the interaction with the robot. As this type of therapy is moving into the home, a third user group has also become prominent: the disabled person's caregivers and family.

The Types of therapies that have benefited from robotic assistance are upper- and lower-extremity movement therapy, communication-enabling for children with autism and exploration-enabling (education) for children with cerebral palsy (CP) or other developmental disabilities (Chap. 73). A robot may be a good alternative to a physical or occupational therapist for the actual hands-on intervention for several reasons:

1. Once properly set up by a therapist – an essential role – an automated exercise machine can consistently apply therapy over long periods of time without tiring.
2. The robot's sensors can measure the work performed by the patient and quantify, to an extent not detectable by standard clinical scales, any recovery of function that may have occurred, which may be highly motivating for a person to continue in the therapy.
3. The robot may be able to engage the patient in types of therapy exercises that a therapist cannot do, such as computer game-based therapy or magnifying movement errors to provoke adaptation [64.4, 5].

One way to classify robotic solutions for application to rehabilitation therapy is by how they contact the patient [64.6].

Operational Therapy Robots (or End-Effector-Based Therapy Robots)

For these machines, the trajectories of the robot end-effector and that of the human end-effector, e.g., the hand or the foot, are physically coupled in operational space. In joint space, the trajectories of the robot joints and of the human joints can significantly differ. Main advantages of operational therapy robots are that: (a) they can be designed by using off-the-shelf components or robots; (b) they can be easily programmed in Cartesian space by nonexpert users. Main limitations are that: (a) they are not able to assist each single human joint independently; (b) patients using these robots are supposed to feature a minimum level of residual motor synergies in order to coordinate their own multijoint movement, producing the configurations of the affected limb required by the therapy exercise.

Wearable Therapy Robots (or Exoskeleton-Based Therapy Robots)

For these machines, a larger portion of the human body (typically the whole affected limb) is in continuous physical contact with the robot. In most cases, a biomimetic exoskeleton kinematic structure is selected. In this case, not only are the trajectories of the robot and human end-effectors the same in operational space, but the trajectories of the robot joints approximate those of the human joints in joint space. The main advantage of wearable therapy robots is the possibility of sensing the configuration and assisting each human joint independently. The main drawbacks of these systems are additional design considerations needed to avoid misalignments between robot and human joints, and to minimize the invasiveness for the patient in terms of weight, dimensions, and overall wearability (Sect. 64.2.4).

Non-Contact Therapy Robots (or Socially Assistive Robotics)

Robotic devices are also being developed for rehabilitation therapy that do not physically contact the patient, but instead monitor and coach the patient during therapy [64.7]. A key concept of this approach is that humans are wired to respond to embodied agents in ways that are important for motivation in rehabilitation therapy (see Chap. 73 for a detailed discussion of socially assistive robotics). One of the earliest robotic therapy devices, developed by Erlandson, can be classified as a socially assistive robot, as it provided reaching targets for patients rather than physically assisting movement [64.8] (see later). The main advantage of noncontact therapy robots is that of being intrinsically safe, since they are not supposed to physically interact with the patient, although this could limit significantly the scope of their clinical application.

From a design perspective, it is important to understand that therapy robots are tools typically intended for temporary use (i. e., the duration of the therapy at home or at the clinic), and are designed to maximize the objective clinical effectiveness of the therapy as well as the outcome and the efficiency of the entire clinical process.

Assistive robots, instead, are solutions for promoting independent living of disabled and elderly citizens. They need to be designed to be usable in a lifelong perspective in real-life scenarios, and thus, designers need to take into account (at a much more serious level than in the case of therapy robots) the end-user subjective preferences, and human factors, in general, in order to maximize their overall, long-term acceptability [64.6]. Whereas limited usability, some level of discomfort, or bad aesthetics might be tolerated by the patients experiencing the application of a therapeutic tool used in a gym or at home, these same factors are unacceptable for disabled or elderly persons who are supposed to depend upon an assistive robot for activities of daily living in a variety of social contexts for the rest of their lives.

Assistive robots can be grouped on the basis of their focus on manipulation, mobility, or cognition. Manipulation aids can further be classified into fixed-platform and portable-platform and mobile-autonomous types. Fixed-platform robots perform functions in the kitchen, on the desktop, or by the bed. Portable types are manipulator arms attached to an electric wheelchair to hold and move objects and to interact with other devices and equipment, as in opening a door. Mobile-autonomous robots can be controlled by voice or other means to do manipulation and other errands in the home or workplace. Mobility aids are divided into electric wheelchairs with navigation systems and mobile robots that act as smart, motorized walkers, allowing people with mobility impairment to lean on them for fall-prevention and stability. The third main type, cognitive aids, assists people who have dementia, autism, or other disorders that affect communication and physical well-being.

The fields of prosthetics, orthotics, and functional electrical stimulation (FES) are closely allied with rehabilitation robotics. Prostheses are artificial hands, arms, legs and feet that are worn by the user to replace amputated limbs. Prostheses are increasingly incorporating robotic features [64.9, 10]. Robotic orthoses are actuated braces that can assist a person in walking or moving the arms or hands. FES systems seek to reanimate limb movements of people who are weak or paralyzed by electrically stimulating nerve and muscle. FES control systems are analogous to robotic control systems, except that the actuators being controlled are human muscles. Another related field is technology for monitoring and diagnosing health care issues as a person performs activities of daily living.

This chapter is organized according to this taxonomy. After providing background information on demographics (Sect. 64.1.2) and history (Sect. 64.1.3) of the field, Sect. 64.2 describes rehabilitation therapy and training robots, and Sect. 64.3 describes robotic aids for people with disabilities. Section 64.4 then reviews recent advances in smart prostheses and orthoses that are related to rehabilitation robotics. Finally, Sect. 64.5 provides an overview of recent work in diagnosis and monitoring for rehabilitation as well as other healthcare issues.

64.1.2 World Demographics

Different areas of rehabilitation robotics focus on different user populations, but a linking characteristic

of these populations is that they have a disability. Disability is defined in the US with the Disabilities Act as *A physical or mental impairment that substantially limits one or more of the major life activities*. Worldwide disability, based on survey results from the WHO ranges from about 10 to 40% of the population, depending on gender, age, wealth, and residence, with an overall prevalence of about 16% (Table 64.1). Prevalence is the proportion of a population who have (or had) a specific characteristic in a given time period. In medicine, typically this characteristic is an illness or impairment. Prevalence is usually expressed as a percentage (e.g., 5%,), or as the number of cases per 10 000 or 100 000 people, depending on how common the illness or risk factor is in the population. Across all other parameters, there is approximately a 4 : 1 disparity in disability rates in the elderly population (>60 years) over working-class adults. In addition, lower birth rates and life-extending health care are the dominant factors contributing to the aging of the population and concomitant rise in disability overall. Other factors, such as China's population control policies of the 1970s, have created a lack of working-age adults to support the world economy. These and other factors make it clear that rehabilitation robotics developers will be faced with users who, as a demographic group, have generally lower levels of sensory and motor capability, and may have impaired cognition as well. The urgency of making advances in this field is increasing in line with the demographics.

64.1.3 Short History of Rehabilitation Robotics

The history of rehabilitation robotics is almost as old as that of robotics itself, although emanating from very different sources. Several book chapters and papers have been written on the history of rehabilitation robotics in more detail than this section [64.1, 19, 20], and numerous papers in the proceedings of the IEEE International Conference on Rehabilitation Robotics also provide more grounding for historical perspective. The chronology below pays particular attention to early work and to projects with notable clinical and/or commercial impact.

Early robotics, starting in the late 1950s, focused on large manipulators to replace workers in factories for dirty, dangerous, and undesirable tasks. The earliest rehabilitation robots came from the field of prosthetics and orthotics (P&O). Both the Case Western University Arm (1960s) and the Rancho Los Amigos *Golden Arm* (early 1970s) (reviewed in [64.19]) were adaptations of replacement mechanical arms meant as powered orthoses [64.1]. The user drove the Golden Arm with a set of tongue-operated switches, joint-by-joint, an arduous means of endpoint control. In the mid-1970s, the Department of Veterans Affairs began funding a group at the Applied Physics Lab under the guidance of *Seamone* and *Schmeisser* to computerize an orthosis mounted on a workstation to perform activities of daily living (ADL) tasks such as feeding a person and turning pages [64.21]. For the first time, a rehabilitation robot had a command-type interface, not just a joint-by-joint motion controller.

The 1970s also saw the French Spartacus system being developed, guided by the vision of Vertut, for use by people with high-level spinal cord injury (SCI) as well as children with CP [64.22]. This system did not emerge from the P&O field but was developed by the French Atomic Energy Commission (CEA), which used large tele-manipulators for nuclear fuel rod handling. One of these was adapted so that people with movement impairment could control it using a joystick for pick-and-place tasks. A decade later, one of the researchers on the Spartacus project, *Kwee*, began the Dutch MANUS Project [64.23] a dedicated effort to develop the first wheelchair-mounted manipulator de-

Table 64.1 Prevalence of disability and aging in selected countries (after [64.18])

Country	Number people with disabilities	Percentage of population with disabilities	Number of elderly people	Percentage of population that is elderly
France	5 146 000	8.3	12 151 000	19.6
USA	52 591 000	20.0	35 000 000	12.4
Great Britain	4 453 000	7.3	12 200 000	29.5
Netherlands	1 432 000	9.5	2 118 808	13.4
Spain	3 528 220	8.9	6 936 000	17.6
Japan	5 136 000	4.3	44 982 000	35.7
Korea	3 195 000	7.1	16 300 000	36.0
Italy	2 609 000	4.8 [64.11]	12 302 000	20.3 [64.12]
Germany	7 101 682	8.7	16 844 300	21.0 [64.13]
China	84 600 000	6.5 [64.14]	122 880 000	9.1 [64.15]
India	21 000 000	1.8 [64.16]	67 117 826	5.6 [64.17]

signed expressly as a rehabilitation robot, not adapted from a design from another field.

In between, several other major programs were begun. In 1978, at Stanford University, and then with decades-long funding from the US Department of Veterans Affairs, Leifer started the Vocational Assistant Robot program, culminating in several clinically tested versions of the Desktop robot, DeVAR [64.3, 24, 25], the mobile version, MoVAR [64.26], and finally a Professional version, ProVAR, developed by *Van der Loos* et al., which had the advanced ability for the user to program tasks in an easy-to-use browser-type environment [64.27]. Although DeVAR briefly made it onto the market in the early 1990s, multisite user testing revealed this industrial-arm-based approach to assistive robotics was still too costly for the low level of functionality achieved, even with ProVAR's advanced interface.

In the mid-1980s, from observations on the unsuitability of existing industrial, educational and orthosis-derived manipulators for rehabilitation applications, Tim Jones at Universal Machine Intelligence (later Oxford Intelligent Machines: OxIM) in the UK began an intensive effort to provide the rehabilitation robotics community with its first workhorse system especially designed from the ground up for human service tasks. Over 10 years, a series of systems, starting with the RTX model, were used in numerous research labs and clinics around the world. The most extensive effort to use the OxIM arm was in France, and a suite of research projects, funded by the French government and the European Research Commission, starting as RAID, then as MASTER [64.28], developed and clinically tested workstation-based assistive systems based on the RTX and subsequent OxIM arms. When OxIM ceased building its arms, the French company Afma Robotics [64.29] took over efforts to commercialize the MASTER system, although it has also stopped production.

The UK witnessed the first commercially available feeding robot, Handy-I, an inexpensive and well-received device first designed by *Topping* and then commercialized by Rehabilitation Robotics, Ltd. in the 1990s [64.30]. Primarily for people with CP to achieve a measure of independence in feeding themselves, task environments later also included face washing and applying cosmetics, areas of high demand identified by its users.

The history of mobile manipulator applications began in the 1980s with adaptations of educational and industrial robots, and achieved a boost with the funding of the US National Institute on Disability and Rehabilitation Research (NIDRR) for a Rehabilitation Engineering Research Center on Rehabilitation Robotics (RERC) at the AI duPont Hospital in Delaware from 1993 to 1997. With its ability to fund dozens of research projects in parallel, it also formed a partnership with a local company, Applied Resources, Corp. (ARC), which developed and marketed several rehabilitation technology products. One of the RERC researchers, *Mahoney*, moved to ARC and was instrumental in extending the company's repertoire to the RAPTOR wheelchair-mounted arm [64.31].

In Europe, the most significant mobile manipulator project was the MANUS Project [64.23] mentioned earlier. With much of the work done under the direction of Kwee at the iRV (Rehabilitation R&D Center) in the Netherlands, the project culminated in a robot specifically designed for wheelchair mounting, with user control through a joystick and feedback by a small display on the arm itself. This project led to numerous follow-on research projects, and, most significantly, to the commercialization of the system by Exact Dynamics, BV, in the Netherlands. The company's current product, called the iARM, is provided on physician prescription by the Dutch Government to qualified people with a disability, such as CP or tetraplegia from an SCI.

Realizing a potential growth in this market niche, the Canadian company, Kinova Robotics, in 2009 began commercializing a competing product, called the Jaco Arm [64.32], with a different design approach. Using carbon fiber segments and lightweight actuator and control components, the payload specifications and control options are approximately the same as for the iARM, but achieved at a lower arm weight.

Autonomous navigation systems on electric wheelchairs began in the 1980s also, benefiting initially from the development by Polaroid Corp. of range finders for its cameras using ultrasonic sensors. They were inexpensive, and small enough, at 30 mm in diameter, that dozens of them could be placed around the periphery of a wheelchair to aid medium-range navigation ($\approx$10–500 cm). In the 1990s and early 2000s, with the advent of vision-based servoing and laser-range scanners, algorithms for faster, smarter, less-error prone navigation and obstacle avoidance dominated research advances in this sector. In Korea, for example, *Bien* et al., at the KAIST Human Welfare Robotics Center began developing the KARES line of wheelchair-based navigation systems in the late 1990s [64.33] and the NavChair project at the University of Michigan was the start of a development line that led to the commercialized Hephaestus system at the University of Pittsburgh [64.34, 35].

Therapy robots had a later start than assistive robots. Research on upper extremity therapy robots started in the mid-1980s with early exercise devices, such as the BioDex [64.36] a first step in programmable,

force-controlled, albeit single-axis devices. The first multiaxis concept was published by *Khalili* and *Zomlefer* [64.37], and the first tested system by *Erlandson* et al. at Wayne State University emerged in the mid-1980s as well [64.8]. The RTX manipulator had a touch-sensitive pad as an end-effector, presenting targets at different locations for patients with upper extremity weakness (e.g., following a stroke) to hit after the screen gave a visual signal. Software logged response times, thereby providing a score that was tallied and compared to previous sessions. Later robots used advanced force-based control, which required significantly more computer power. The early-1990s saw the start of the MIT-MANUS Project with Hogan and Krebs, simple robotic devices designed for bimanual therapy after stroke designed by *Lum* et al. at U.C. Berkeley [64.38, 39], followed a few years later by the Palo Alto VA MIME (mirror image motion enabler) Project [64.40] and its derivative, Driver's SEAT [64.41], with Burgar, Van der Loos and Lum, as well as the Rehabilitation Institute of Chicago ARM Guide Project with *Reinkensmeyer* et al. [64.42]. Each had a different philosophy on upper extremity stroke therapy and each was able to demonstrate clinical effectiveness in a different way. Several of these programs, now two decades later, have made significant technical advances, and all the investigators are still active in the field. Current work is described in subsequent sections of this chapter.

Research on lower extremity therapy robots began with the work of Scherb in 1919 who proposed an approach called *mechanotherapy* [64.43]. He also developed a prototype, depicted in Fig. 64.1, of a cable-driven machine for assisting the motion of the lower limbs for bedridden patients. In 1976, the use of a master/slave exoskeleton system for lower limb therapy was introduced by the French physicians *Bel* and *Rabischong* [64.44], who developed a pneumatically actuated wearable robot to be used for assisted therapy of paraplegic patients (Fig. 64.2). The robot movements were remotely controlled by a therapist who wore a master device capable of sensing and recording the configuration of her/his lower limbs.

About 20 years later, the Lokomat project for the development of a stand-alone, programmable wearable robot for lower extremity was launched and originated the establishment of the Hocoma company by Colombo and Hostettler. The first prototype of the Lokomat system [64.45] was developed in close cooperation with the Balgrist University Hospital in Zurich and initially applied to patients who had had spinal cord injuries (Fig. 64.3b). During the same time, *Hesse* and *Uhlenbroch* developed the Gait Trainer [64.46], (Fig. 64.3a), which allowed training without the use of a treadmill by controlling the center of mass (COM) in both in the vertical and horizontal directions.

In the last decade the growth in rehabilitation robotics has been characterized by the development of new devices (as explained above) as well as by new investigation methods aimed at understanding brain changes induced by rehabilitation. A recent clinical study [64.50] confirms that high-intensity, repetitive, and task-oriented training yields a faster learning and recovery time. Robots are also a key-enabling technology for labor-intensive and patient-tailored training as well as for accurately imposing spatial and temporal constraints within interactive scenarios so as to augment patient involvement [64.51]. A recent international online survey, distributed through professional organizations to therapists, has set out a set of important robotic rehabilitation device requirements [64.52], such as variety of arm movements, biofeedback to patients, adjustable resistance, and need for virtual activities. Robots offer the advantage of providing ob-

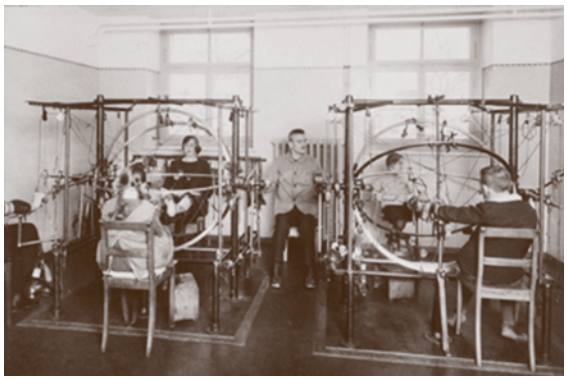


Fig. 64.1 The meridian. A universal machine for applying passive motion to any joint (after [64.43])

Fig. 64.2 This machine developed by Professor Rabischong allows the patient in rehabilitation to maintain her balance while inciting her muscles to move (after [64.47])

Fig.64.3a–e Gait-training robotic systems in use in clinics; **(a)** the gait trainer GT-I, developed by Hesse's group and commercialized by Reha-Stim (Germany), **(b)** the Lokomat, developed by Colombo and colleagues and commercialized by Hocoma AG (Switzerland), **(c)** AutoAmbulator, developed by the HealthSouth Corp. (USA), **(d)** LokoStation with Lokohelp (Darkov Spa, Czech Republic; after [64.48]), **(e)** G-EO Evolution (Panos Th. Skoutas S.A., Greece; after [64.49])

jective measures of user performance, thus enabling the development of quantitative kinematic and dynamic metrics [64.53–56]. The analysis of brain reorganization and associated clinical outcomes induced by robot-aided therapy in chronic stroke patients have shown that motor performance improvement is correlated with changes in inter-hemispheric connectivity between primary somatosensory areas: for those patients the connectivity returns to nearly *physiological levels* [64.57].

Protocols based on monitoring patient performance, strength, endurance, and emotional state are being investigated. Anticipating patient movement intentions and modeling internal states lead to adaptive patient–robot interaction control. This approach paves the way to a new generation of robotic therapy devices using *bio-cooperative* controllers, where *psychophysiological* and biomechanical information is used as feedback for updating robot control [64.58–60]. First examples of biocooperative controllers can be found in the robotic platforms developed within the European projects ECHORD/MAAT [64.56] (multimodal interfaces to improve therapeutic outcomes in robot-assisted rehabilitation) (Fig. 64.4) and MIMICS (multimodal immersive motion rehabilitation with interactive cognitive systems). Other bio-cooperative controllers have been developed by *Riener* and *Munih* [64.58], who developed new control strategies applied to lower limb

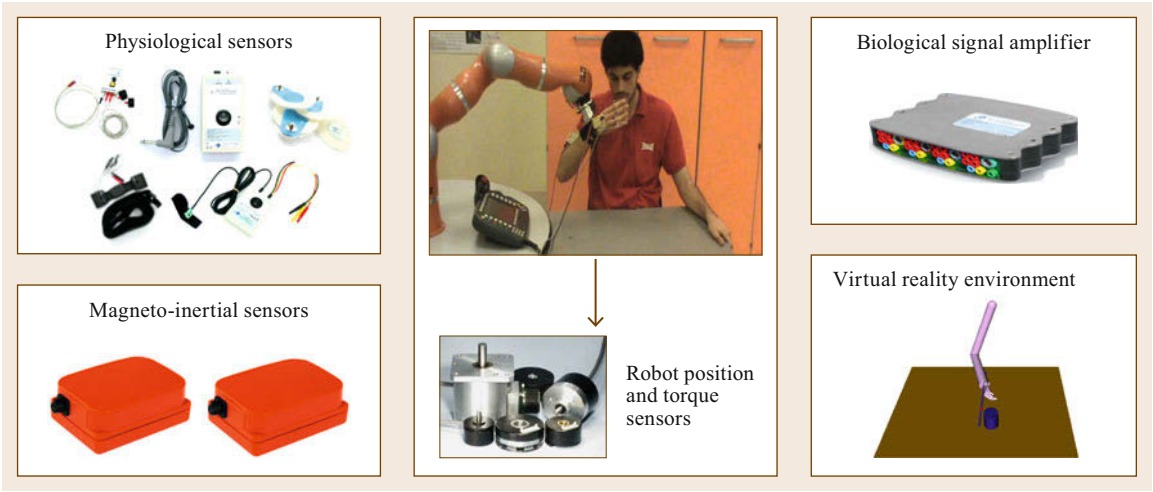


Fig. 64.4 Set up of the MAAT platform (after [64.61])

Fig. 64.5 PARO, an advanced interactive robot developed by AIST (after [64.62])

rehabilitation protocols that promote active participation of patients during training.

Cognitive robotics had started in the early 1980s to aid children with communication disorders and physical impairment to achieve a measure of control of their physical space. Using mostly educational manipulators, several demonstration systems were developed. In the early 2000s, Latham of Anthrotronix, Inc. commercialized small robot systems to enable children with physical disabilities to play games with simple interfaces. A bit later, small mobile robots were used in clinics by *Dautenhahn* and *Werry* [64.63] with children who had autism, since robots have such simple interfaces that they appear not to be that challenging as communication with other humans. The early 2000s also saw the advent of pet robots, such as the Paro seal robot developed by *Shibata* et al. [64.64] (Fig. 64.5), as companions for both children and the elderly who are confined to clinics and have limited real companionship. This topic is further elaborated in Chap. 73.

The applications for robotics continues to increase in number as advances in materials, control software, higher robustness, and the diminishing size of sensors and actuators allow designers to attempt new ways of using mechatronics technology to further the well-being of people with disabilities.

64.2 Rehabilitation Therapy and Training Robots

The human neuromuscular system exhibits use-dependent plasticity, which is to say that use alters the properties of muscles and neurons, including the pattern of their connectivity, and thus their function [64.65–67]. For a detailed description of patient populations expected to benefit from robot-assisted therapy and the most common impairments to be treated, which include incoordination, abnormal synergies, spasticity and weakness, please see [64.50, 68–70].

64.2.1 Grand Challenges and Roadblocks

The process of neurorehabilitation seeks to exploit use-dependent plasticity in order to help people to relearn how to move following neuromuscular injuries or diseases. Neurorehabilitation is typically provided by skilled therapists, including physical, occupational, and speech therapists. This process is time-consuming, involving daily, intensive movement practice over many

weeks. It is also labor-intensive, requiring hands-on assistance from therapists. For some tasks, such as teaching a person with poor balance and weak legs to walk, this hands-on assistance requires that the therapists have substantial strength and agility to provide safe and effective interventions.

Because neurorehabilitation is time and labor intensive, in recent years healthcare players have put limits on the amount of therapy that they will reimburse in an effort to contain spiraling healthcare costs. Ironically, at the same time, there has been increasing scientific evidence that more therapy can in many cases increase movement recovery via use-dependent plasticity [64.71]. As robotics and rehabilitation researchers began to recognize beginning in the late 1980s, neurorehabilitation is a logical target for automation because of its labor-intensive, mechanical nature, and because the amount of recovery is linked with the amount of repetition. Robots could deliver at least the repetitive parts of movement therapy at lower cost than human therapists, allowing patients to receive more therapy, recover more function, remain independent longer, and reduce downstream healthcare expenses.

The grand challenge for automating movement therapy is determining how to optimize use-dependent plasticity. That is, researchers in this field must determine what the robot should do in cooperation with the patient's own movement attempts in order to maximally improve movement ability, all the while engaging the patient. Meeting this challenge involves solving two key problems: determining appropriate movement tasks (what movements should patients practice and what feedback should they receive about their performance), and determining an appropriate pattern of mechanical input to the patient during these movement tasks (in other words, at what magnitude, how often, and in which directions, should the robot apply forces to the patient's limb to provoke increased plasticity). The prescription of movement tasks and mechanical input fundamentally constrains the mechanical and control design of the robotic therapy device.

There are three main roadblocks to achieving the grand challenge. The first is a scientific roadblock: neither the optimal movement tasks nor the optimal mechanical inputs are known. The scientific basis for neurorehabilitation remains ill-defined, with competing schools of thought. The number of large, randomized, controlled trials that have rigorously compared different therapy techniques is still small, in part because these trials are expensive and difficult to control well. Therefore, the first problem that a robotics engineer will encounter when setting out to build a robotic therapy device is that there is still substantial uncertainty as to what exactly the device should do.

This uncertainty corresponds to an opportunity to use robotic therapy devices as scientific tools themselves. Robotic therapy devices have the potential to help identify what exactly provokes plasticity during movement rehabilitation, because they can provide well-controlled patterns of therapy. They can also simultaneously measure the results of that therapy. Better control over therapy delivery and improved quantitative assessment of patient improvement are two desirable features for clinical trials that have often been lacking in the past. Recent work with robotic movement training devices is leading, for example, to the characterization of computations that underlie motor adaptation, and then to strategies for enhancing adaptation based on optimization approaches [64.5, 72, 73]. Early work at UBC on the characterization of human balance in quiet standing will lead to the ability to use robotics clinically to develop new therapies to prevent falls for stroke survivors and others with balance impairment [64.74, 75].

The second roadblock is a technological one: robotic therapy devices often have as their goal to assist in therapy of many-body degrees of freedom (e.g., the arms and torso for reaching, or the pelvis and legs for walking). The devices also require a wide dynamic bandwidth such that they can, for example, impose a desired movement on a patient who is paralyzed, but also *fade-to-nothing* as the patient recovers. Further, making the devices lightweight enough to be wearable is desirable, so that the patient can participate in rehabilitation in a natural setting (for example, by walking over ground or working at a counter in a kitchen), or even throughout the course of normal activities of daily living. The development of high degree-of-freedom, wearable, high-bandwidth robotic exoskeletons is an unsolved problem in robotics, although much progress has been made toward this goal in the last 10 years, as described below. Still, no device at present comes close to matching the flexibility of a human therapist, in terms of assisting in moving different body degrees of freedom in a variety of settings (e.g., walking, reaching, grasping, neck movement), or the intelligence of a human therapist, in terms of providing different forms of mechanical input (e.g., stretching, assisting, resisting, perturbing) based on a real-time assessment of the patient's response.

The third roadblock is maintaining the motivation and engagement of the patient through the tens of thousands of repetitions required to achieve meaningful increases in function [64.50] that will carry over to performing actual ADLs. Rehabilitation can be compared to the exercise required by an athlete to compete at an elite competition level. By necessity, therapy at this intensity will have to move largely to the home setting to be economically viable, and therefore motivation

becomes the key ingredient of success. Current work focuses on developing adaptive therapy through robotics and feedback, embedding therapy in computer games, music, and sport, and applying motor-learning theories, for example, the *Challenge Point* [64.76] model of keeping people at a desirable difficulty level, to maximize effectiveness and minimize frustration over the long term [64.76]. Meeting the grand challenge of robotic therapy therefore will require substantial, inter-related advances in clinical neuroscience, robot engineering, and kinesiology.

64.2.2 Movement Therapy After Neurologic Injury

Modern, evidence-based medicine relies on objective evaluation and quantitative comparative analysis of the impact of different therapeutic approaches. Robotics technology has the potential to boost evidence-based neurorehabilitation: therapy robots provide precise and sensitive tools for assessing and modeling human behavior, well beyond the capability of a human observer. This is of paramount importance for enabling appropriate initial diagnosis, early adoption of corrective clinical strategies, and for identifying verifiable milestones as well as prognostic indicators of the recovery process.

Bosecker et al. [64.77] tried to evaluate clinical scores during upper limb therapy by means of robot-based metrics; *Zollo* et al. [64.54, 78] evaluated robotic therapy exploiting robotic outcome; thus providing quantitative measure of biomechanical and motion planning features of arm motor control following upper limb rehabilitation. A recent study carried out by *Krebs* et al. showed that robotic measurements of arm movements after upper limb, robot-aided therapy may establish biomarkers for motor recovery [64.79]. Similarly, *Domingo* and *Lam* [64.80] tried to apply robot-based assessment to the lower limb by using the Lokomat system and customized software.

Another important advantage of therapy robots is the possibility that a single operator can effectively supervise multiple patients, locally or even remotely, i. e., in a telerehabilitation scenario [64.81]. Therapy robots can potentially improve patients' access to rehabilitation by providing the opportunity to increase the duration and the frequency of their therapy experience, with limitations mainly depending on clinical considerations rather than on other organizational or economic constraints.

At present, much of the activity in physical therapy and training robots has been focused on retraining movement ability for individuals who have had a stroke or SCI, or are affected by CP. The main reasons for this emphasis are that there are a relatively large number of patients with these conditions, the rehabilitation costs associated with them are high, and because these patients can sometimes experience large gains with intensive rehabilitation because of use-dependent plasticity.

A stroke refers to an obstruction or breakage of a blood vessel supplying oxygen and nutrients to the brain. Approximately 800 000 people suffer a stroke each year in the United States, and about 80% of these people experience acute movement deficits [64.82]. In Europe, 1 100 000 people suffered a stroke in 2000, and they are expected to increase up to 1 500 000 in 2025. There are over 3 000 000 survivors of stroke in the United States [64.83], with over half of these individuals experiencing persistent, disabling, movement impairments. Data are similar to those of Europe, where it has been estimated that between half and two-thirds of people survive stroke. Of these, half do not recover fully, and a quarter need assistance in daily living. While most of the survivors of stroke are elderly persons, motor impairment from a stroke affects 3000 of the 10 000 children born with CP each year [64.83]. While this is a small proportion of all stroke survivors, the motor function impairment will persist throughout the person's entire life, so the impact on independence is disproportionately large.

The number of people who have experienced and survived a stroke is expected to increase substantially in the United States and other industrialized countries in the next two decades, mostly because age is a risk factor for stroke and the mean age of people in industrialized countries is rapidly increasing due to the baby boom of the 1950s, but also because improved acute treatments will allow more people to survive a stroke, albeit perhaps with impairment.

The number of people who experience an SCI in the United States each year is relatively smaller – about 15 000, with about 200 000 people alive who have survived an SCI – but the consequences can be even more costly than stroke [64.82]. In Europe, there are about 11 000 new cases of SCI per year, and about 330 000 have survived an SCI [64.84]. The most common causes of SCI are automobile accidents and falls. These accidents crush the spinal column and contuse the spinal cord, damaging, or destroying neurons within the spinal cord. Lower limb robotic therapy has demonstrated promising results when applied to SCI patients with incomplete lesions [64.85] although large, systematic, and randomized controlled trials are still missing. It is important to note that SCI patients are typically younger than stroke patients, which may affect their familiarity with and acceptance of technical aids.

64.2.3 Robotic Therapy for the Upper Extremity

This section first describes early, clinically tested upper limb therapy robot systems (Fig. 64.6a–e), then more recent systems.

MIT-MANUS

The first robotic therapy device to undergo extensive clinical testing, and, now, to achieve some commercial success, is the MIT-MANUS, sold as the InMotion2 by Interactive Motion, Inc. [64.86]. MIT-MANUS is a planar two-joint arm that makes use of the SCARA configuration, allowing two large, mechanically grounded motors to drive a lightweight linkage. The patient sits across from the device, with the weaker hand attached to the end-effector, and the arm supported on a table with a low-friction support. By virtue of the use of the SCARA configuration, the MIT-MANUS is perhaps the simplest possible mechanical design that allows planar movements while also allowing a large range of forces to be applied to the arm without requiring force feedback control.

MIT-MANUS assists the patient in moving the arm across the table-top as the patient plays simple video games, such as moving a cursor into a target that changes locations on a computer screen. Assistance is achieved using a position controller with an adjustable impedance. Additional modules have been developed for the device for allowing vertical motion [64.87], wrist motion [64.88], and hand grasp [64.89]. Software has been developed for providing graded resistance as well as assistance to movement [64.90], and for varying the firmness and timing of assistance based on real-time measurements of the patient's performance on the video games [64.91].

MIT-MANUS has undergone extensive clinical testing in several studies, summarized as follows. The first clinical test of the device compared the motor recovery of acute stroke patients who received an additional dose of robot therapy on top of their conventional therapy, to that of a control group, who received con-

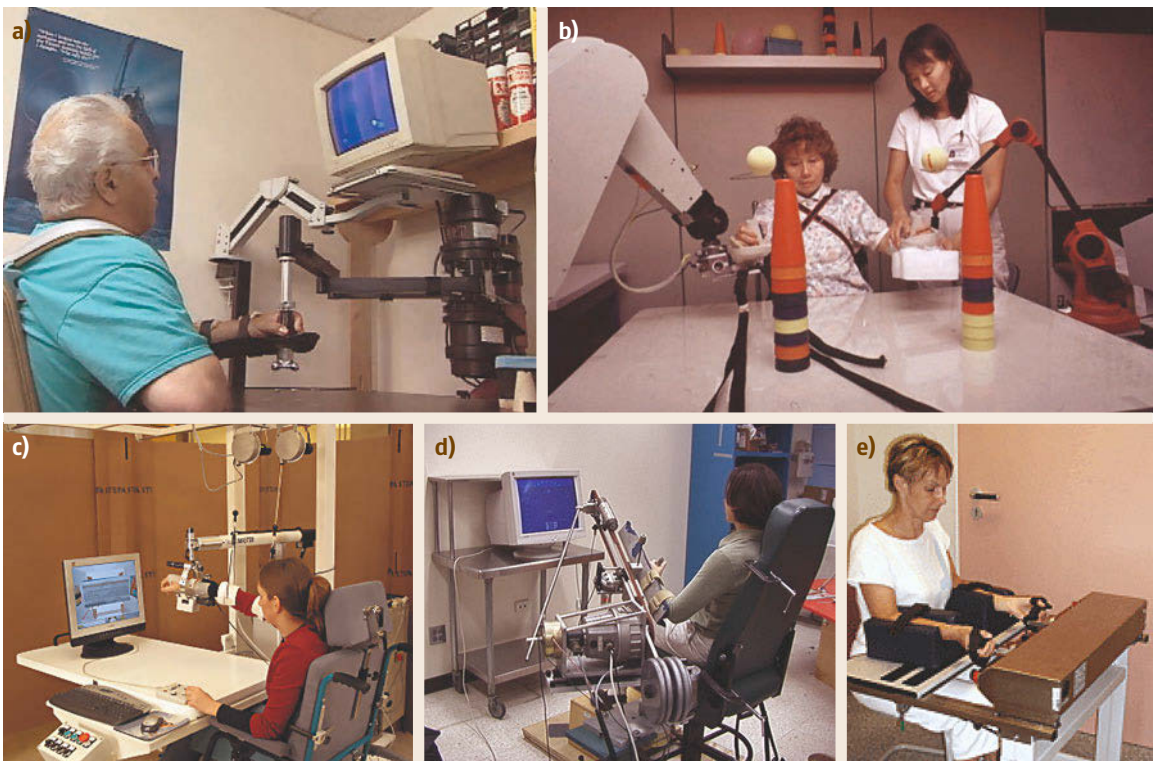


Fig.64.6a–e Arm-therapy robotic systems that have undergone extensive clinical testing; **(a)** MIT-MANUS, developed by Hogan, Krebs, and colleagues at the Massachusetts Institute of Technology (USA), **(b)** MIME, developed at the Department of Veterans Affairs in Palo Alto in collaboration with Stanford University (USA), **(c)** GENTLE/s developed in the EU, **(d)** ARM-Guide, developed at the Rehabilitation Institute of Chicago and the University of California, Irvine (USA), and **(e)** Bi-Manu-Track, developed by Reha-Stim (Germany)

ventional therapy and a brief, sham exposure to the robot [64.92]. The robot group patients received the additional robotic therapy for an hour each day, five days per week, for several weeks. The robot group recovered more arm movement ability than the control group according to clinical scales, without any increase in adverse effects such as shoulder pain. The improvements might subjectively be characterized as *small but somewhat meaningful to the patient*. The improvements were sustained at a three-year follow-up.

This first study with MIT-MANUS demonstrated that acute stroke patients who received more therapy recover better, and that this extra therapy can be delivered by a robotic device. It did not answer the question as to whether the robotic features of the robotic device were necessary. In other words, it may have been that patients would have also improved their movement ability if they had practiced additional movements with MIT-MANUS with the motor's off (thus making it equivalent to a computer mouse), simply by virtue of the increased dose of movement practice stimulating use-dependent plasticity. Thus, while this study indicated the promise of robots for rehabilitation therapy, it did not close the gap of knowledge as to how external mechanical forces provoke use-dependent plasticity.

Subsequent studies with MIT-MANUS have confirmed that robotic therapy can also benefit chronic stroke patients [64.93]. The device has been used to analyze different types of therapies, for example, to compare assisting movement versus resisting movement in chronic stroke subjects, but with inconclusive results: both types of therapies produced benefits [64.90]. The device has also been used to compare assistive robot therapy with another technological approach to rehabilitation – electrical stimulation of finger and wrist muscles [64.94]. Again, significant benefits were found for both therapies, and those benefits were specific to the movements practiced, but the benefits were not significantly different between therapies. We note that the lack of a significant difference in these studies may simply be due to the limited number of patients who participated in these studies (i. e., inadequate study power), rather than a close similarity of the effectiveness of the therapies.

As mentioned above, the MIT-MANUS device was recently tested in a multisite clinical trial funded by the Department of Veterans Affairs in the United States [64.50]. This study compared clinical outcomes in chronic stroke survivors who were randomized to three groups: usual care, robot-assisted therapy with arm, wrist, and hand modules, and one-on-one therapy with a rehabilitation therapist that was dose matched to the robotic therapy in terms of the number of movements achieved per therapy session. The robot-assisted therapy group improved their movement ability more than the usual care, and about the same as the dose-matched group. This is an important finding for the field, as it demonstrated with the highest scientific rigor that robotic therapy was about as effective as an intense, therapist-delivered therapy.

A cost-benefit analysis of this study suggested that although both robotic and intense, therapist-based therapies were more expensive to deliver than usual care, it reduced long-term follow-up costs, so that the total cost of care of the patients was the same over the duration of the study [64.95]. Thus, as the cost of robotic therapy devices decreases, it should be possible to provide patients with improved outcomes while reducing the cost of therapy – another important finding.

MIME

Another early system to undergo clinical testing was the MIME (mirror-image movement enhancer) system, which used a Puma-560 robot arm to assist in movement of the patient's arm [64.96]. The device is attached to the hand through a customized splint and a connector that is designed to break away if interaction forces become too large. Compared to MIT-MANUS, the device allows more naturalistic motion of the arm because of its six degrees of freedom (DOF), but must rely on force feedback so that the patient can drive the robot arm. Four control modes were developed for MIME. In the passive mode, the patient relaxes and the robot moves the arm through a desired pattern. In the active assist mode, the patient initiates a reach toward a target, indicated by physical cones on a table top, which then triggers a smooth movement of the robot toward the target. In the active-constrained mode, the device acts as a sort of virtual ratchet, allowing movement toward the target, but preventing the patient from moving away from the target. Finally, in the mirror-image mode, the motion of the patient's less impaired arm is measured with a digitizing linkage, and the impaired arm is controlled to follow along in a mirror-symmetric path. The initial clinical test of MIME found that chronic stroke patients who received therapy with the device improved their movement ability about as much as patients who received conventional table-top exercises with an occupational therapist [64.96]. The robot group even surpassed the gains from human-delivered therapy for the outcome measures of reaching the range of motion and strength at key joints of the arm. A follow-on study attempted to elucidate which of the control modes or what combination of MIME exercises caused the gains, but was inconclusive [64.97]. A multisite randomized control of MIME again funded by the Department of Veterans Affairs compared the

effect of a high-dose (30 h) of additional therapy with MIME, to a lower dose (15 h), to 15 h of additional conventional therapy in 54 acute stroke patients [64.98]. Gains in the primary outcome measure, the Fugl-Meyer assessment, were not significantly different between the groups at the 6-month follow-up, although there the actual dose of robotic therapy patients received predicted their recovery.

ARM Guide

The question of the effect of robot forces on movement recovery was also left unresolved by a study with another device, the ARM Guide, which is a trombone-like device that can be oriented then locked in different directions, and can assist people in reaching in a straight line. Chronic stroke patients who received assistance during reaching with the robot improved their movement ability [64.99]. However, they improved about as much as a control group that simply practiced a matched number of reaches without assistance from the robot. This suggests that movement effort by the patient is a key factor for recovery, although the small sample size of this study limited the ability to resolve the size of the difference between guided and unguided therapies.

Bi-Manu-Track

Perhaps the most striking clinical results generated so far have come from one of the simplest devices built. Similar to a design proposed previously by *Lum* et al. [64.39], the Bi-Manu-Track uses two motors, one for each hand, to allow bimanual wrist-flexion extension [64.100]. The device can also assist in forearm pronation/supination if it is tilted downward and the handles are changed. In an extensive clinical test of the device, 22 subacute patients (i. e., 4–6 weeks after stroke) practiced 800 movements with the device for 20 minutes per day, five days per week for six weeks [64.100]. For half the movements, the device drove both arms, and for the other half, the patient's stronger arm drove the motion of the more impaired arm. A control group received a matched duration of electrical stimulation (ES) of their wrist extensor muscles, with the stimulation triggered by voluntary activation of their muscles when possible, as measured by electromyography (EMG). The number of movements performed with EMG-triggered ES was 60–80 per session. The robot-trained group improved by 15 points more on the Fugl-Meyer scale, a standard clinical scale of movement ability with a range from 0 to 66 points in upper extremity function. It assigns a score of 0 (cannot complete), 1 (completes partially), or 2 (completes normally) for 33 test movements, such as lifting the arm without flexing the elbow. For comparison, reported gains in Fugl-Meyer score after therapy with the MIT-MANUS and MIME devices ranged from 0 to 5points [64.101].

Other Early Devices to Undergo Clinical Testing

Other early devices to undergo clinical testing are as follows. The GENTLE/s system uses a commercial robot, the HapticMaster, to assist in patient movement as the patient plays video games. The HapticMaster allows four degrees of freedom of movement and achieves a high bandwidth of force control using force feedback. Chronic stroke patients who exercised with GENTLE/s improved their movement ability [64.102, 103]. The Rutgers ARM robotic device uses low-friction pneumatic cylinders to help extend or flex the fingers, and has been shown to improve hand movement ability of chronic stroke subjects [64.104]. Simple force-feedback controlled devices, including a 1-DOF wrist manipulator and a 2-DOF elbow–shoulder manipulator, were also recently shown to improve movement ability of chronic stroke subjects who exercised with the devices [64.105]. A passive exoskeleton, the T-WREX arm orthosis, provides support to the arm against gravity using elastic bands, while still allowing a large range of motion of the arm [64.106]. By incorporating a simple hand-grasp sensor, this device allows substantially weakened patients to practice simple virtual reality exercises that simulate functional tasks such as shopping and cooking. Chronic stroke patients who practiced exercising with this nonrobotic device recovered significant amounts of movement ability, comparable with the Fugl-Meyer gains seen with MIT-MANUS and MIME. A randomized controlled trial of T-WREX in 28 chronic stroke subjects compared therapy with the device to conventional, table-top therapy [64.107] and found that the device was slightly more effective in improving patient's motor ability at the six-month follow-up according to the primary outcome measure, the Fugl-Meyer score, and that patients strongly preferred exercising with the device. T-WREX has been commercialized as the Armeo Spring upper extremity training device by Hocoma, and was in use in over 500 clinics as of early 2014, with published reports studying its use for individuals with multiple sclerosis [64.108], SCI [64.109], and proximal humeral fracture [64.110]. NeReBot is a 3-DOF wire-based robot that can slowly move a stroke patient's arm in spatial paths. Acute stroke patients who received additional movement therapy beyond their conventional rehabilitation therapy with NeReBot recovered significantly more movement ability than patients who received just conventional rehabilitation therapy [64.111]. RehaRob uses an industrial robot arm to mobilize patients' arms along arbitrary trajectories following stroke [64.112].

Further Research and Developments on Robotic Therapy for the Upper Extremity

One recent trend in the field is to develop devices that can provide therapy for a larger number of degrees of freedom of the arm. For example, at the high end of cost and complexity are the ARMin [64.113], Pneu-WREX [64.114], and BONES [64.115], which are exoskeletons that accommodate nearly naturalistic movement of the arm while still achieving a wide range of force control. A system that couples a immersive virtual reality display with a haptic robot arm is described in [64.116]. A wearable exoskeleton driven by pneumatic muscles is described in [64.117]. Several other exoskeletal machines have been proposed to jointly train arm and wrist, such as the CADEN-7, an anthropometric 7-DOF powered exoskeleton system [64.118]; the SUEFEL-7, which exploits EMG signals to adjust impedance parameters [64.119]; a lightweight exoskeleton described in [64.120], and the highly redundant 9-DOFs machine introduced in [64.121].

Another area of active development in robotic therapy devices for the upper extremity is to devices that can be used at home or in rank-and-filed clinics. For example, at the lower end of cost/complexity are devices that use force feedback joysticks and steering wheels with a view toward implementation in the home [64.114–125]. In 2008, a planar machine for upper limb rehabilitation designed for delivering at-home neurorehabilitation, namely CBM MOTUS, was developed and patented [64.126] (Fig. 64.7). Such a device has low inertia and highly isotropic behavior. Recently, a passive module has been developed to be installed on the CBM-MOTUS, able to further reduce robot perceived inertia when the machine is moved by the patient, while being highly rigid when the machine assists the patient's movements.

As far as wearable robotic systems for the upper limb are concerned, Hocoma engineered and commercialized the ARMEO power system, and also the ArmeoSpring and ArmeoBoom machines [64.108].

A third area in robotic therapy devices for the upper extremity that has received increased attention is to develop devices for hand rehabilitation, since hand movement ability is essential for functional recovery. Examples of recent, novel robotic devices for the hand are the Haptic Knob for grasping and wrist pronation/supination [64.128]; the HandCARE, a cable-actuated rehabilitation system [64.129]; a device for repeating controlled passive movements of paralyzed fingers given in [64.130]; the exoskeletal machines HWARD [64.131], and HEXORR [64.132] to help open and close the hand; the 18-DOFs highly redundant Gifu Haptic Interface [64.133], able to produce adduction–abduction and flexion–extension finger movements. Other devices are presented in [64.134–138], and a review of robotic therapy of the hand is given in [64.139]. One robotic therapy system for the hand incorporates the idea of using visual feedback distortion to enhance motivation of patients during movement therapy [64.140]. A few commercial machines are available, such as the Reha-Digit, Amadeo, and ManovoSpring.

Based on the availability of modular hand, wrist, and shoulder–elbow therapeutic robotic devices, some recent studies tried to tackle the fundamental question of whether proximal or distal treatment differentially affect the recovery of arm/hand function. For more information of these preliminary studies, see references [64.100, 141, 142].

An important recent development in robotic therapy devices is the development of devices that can be used in conjunction with the instrumentation needed to measure neurophysiological signals, such as functional magnetic resonance imaging (fMRI)-compatible systems [64.143, 144] or, more generally, brain-imaging (BI)-compatible robotic systems that can be used in conjunction not only with fMRI but also with magne-

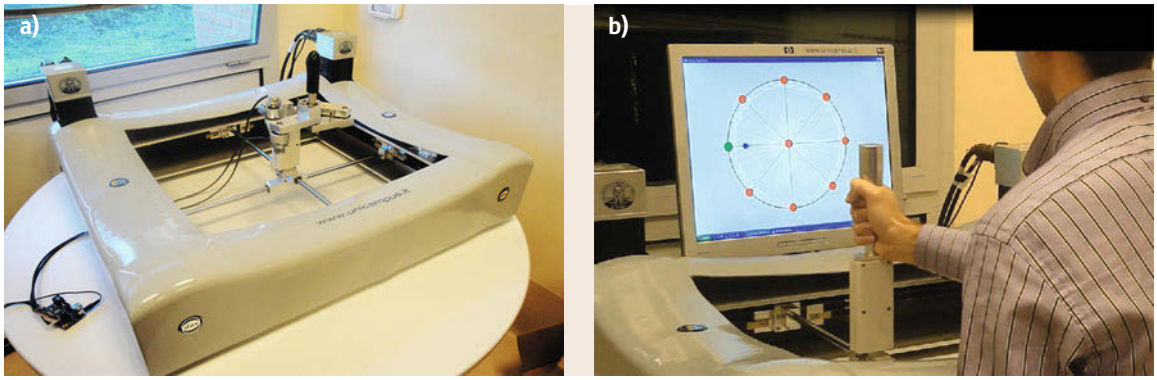


Fig.64.7a,b CBM Motus robot for upper limb rehabilitation (after [64.127])

toencephalography (MEG), transcranial magnetic stimulation (TMS), repetitive TMS (rTMS), transcranial direct current stimulation (tDCS), near infrared spectroscopy (NIRS) and other brain imaging and stimulation equipment. Such devices are notable because they will allow a systematic scientific study of neural recovery during robotic therapy. Developing these devices requires dealing with the problems of electromagnetic compatibility and interaction.

Recently, a few fMRI-compatible devices for hand rehabilitation have been designed to acquire functional imaging while performing therapeutic exercises, such as the exoskeletal machine in [64.145]; the pneumatic actuated 2-DOF device to help wrist pronation/supination and hand open/closure in [64.146]; machines driven by electro-rheological fluids, able to provide a variable resistance to patient motion [64.147]; the planar device to help move fingers described in [64.148]. A passive fMRI-compatible manipulandum morphologically similar to MIT-MANUS has been developed and tested to acquire functional imaging in [64.149]. Another study proposed the applications of a brain–computer interface (BCI) therapy approach on a cohort of stroke patients, [64.150]; a magnetoencephalography-based BCI system is used which in turn raised or lowered a screen cursor in the direction of a target. Results suggest that volitional control of neuromagnetic activity features recorded over central scalp regions can be achieved with BCI training after stroke.

Finally, the primary paradigm that has been tested so far with upper extremity robotics is to assist patients in moving, a strategy which may in some cases have the unintended affect of causing patients to *slack* [64.151]. In general, the field is still relatively undeveloped in its ability to identifying the most appropriate forms of robotic intervention given the nature of the impairment and the patient. For example, an approach opposite to physical assistance, which is using robotic force fields to amplify the kinematic errors of stroke patients during reaching, may provoke novel forms of the adaptation of those patterns [64.4, 152]. A major emphasis of the field in the next 10 years will be improving the mechanistic understanding of how robotic interaction influences brain plasticity.

64.2.4 Robotic Therapy for Walking

Scientific evidence that gait training improves the recovery of mobility after neurologic injury started to accumulate in the 1980s through studies with cats. Cats with SCI can be trained to step with their hind limbs on a treadmill with the partial support of body weight and assistance of leg movements [64.153, 154]. Following the animal studies, various laboratories developed a rehabilitation approach in which the patient steps on a treadmill with the body weight partially supported by an overhead harness and assistance from therapists [64.155–158]. Depending on the patient's impairment level, from one to three therapists are needed for BWSTT (body-weight supported treadmill training), with one therapist assisting in stabilizing and moving the pelvis, while two additional therapists sit next to the treadmill and assist the patient's legs in swing and stance. This type of training is based on the principle of generating normative, locomotor-like sensory input that promotes the functional reorganization and recovery of the injured neural circuitry [64.159]. In the 1990s, several independent studies indicated that BWSTT improves stepping in people with SCI or hemiplegia after stroke [64.155–157].

Gait training is particularly labor-intensive and strenuous for therapists, so it is an important target for automation. The efforts of roboticists have been especially focused on BWSTT rather than over-ground gait training because BWSTT is done on a stationary setup in a well-defined manner and thus can be more easily automated than over-ground gait training. Randomized, controlled clinical trials have shown that BWSTT is comparable in effectiveness to conventional physical therapy for various gait-impairing diseases [64.160–166]. These trials support the efforts toward the automation of BWSTT, as the working conditions of physical therapists will improve if the robots do much of the physical work, which, in the case of BWSTT, actually leads to occasional back injuries to therapists. Usually, only one therapist is needed in robot-assisted training, for the tasks of helping the patient into and out of the robot and monitoring the therapy. In the case of SCI patients, a small randomized, controlled trial [64.161] reported that robotic-assisted BWSTT with a first-generation robot required significantly less labor than both conventional overground training and therapist-assisted BWSTT, with no significant difference found in effectiveness.

Gait-Training Robots in Current Clinical Use

Some gait-training robot systems are commercially available and are used for therapy in several clinics worldwide, such as the gait trainer GT-I [64.46], the Lokomat [64.45] the ReoAmbulator, the Loko-Help [64.167], and the G-EO System [64.168] (Fig. 64.3a–e).

Of these robots, the GT-I (commercialized by Reha-Stim) is the one that departs most from therapist-assisted BWSTT, since it interacts with the patient's lower limbs through two footplates rather than acting on the shank as human therapists do. It also ap-

pears to depart more from natural walking because the footplate principle substantially alters the sensory cues of the foot impact with the ground or treadmill band. The GT-I footplates are driven by a singly actuated mechanism that moves the foot along a fixed gait-like trajectory with a doubled crank and rocker system [64.46]. The stride length can be adjusted between sessions by changing gears. The body weight is unloaded as needed by an overhead harness. The torso is moved sagittally in a phase-dependent manner by ropes attached to the harness and connected by another crank to the foot crank. The GT-I is currently installed in dozens of clinics, mainly in Europe. One randomized, controlled study has been reported that tested the GT-I with 30 subacute stroke patients [64.169]. The robot group improved their overground walking ability more than the control group, although differences were not significant at a 6-month follow-up. A total of 80% of the patients said they preferred training with the robot rather than the therapists because training with the robot was less demanding and more comfortable. The other 20% of patients stated that swinging of the paretic limb seemed less natural and thus less effective when training with the robot. Robot-assisted training required an average of one therapist per patient, while therapist-assisted training required two therapists per patient on average. A follow-up, randomized controlled study comparing conventional training plus robotic training with the GTI, to a time-matched amount of conventional training alone with subacute stroke patients, found that the group that received some robotic training recovery walking ability to a great extent [64.170]. More recent clinical tests with the GT I are reported in [64.171].

The Lokomat (commercialized by Hocoma) is a robotic exoskeleton worn by the patients during treadmill walking [64.45]. Four motorized joints (two per leg) move hip and knee. The actuators consist of ball screws connected to dc motors. The legs are driven in a gait-like pattern along a fixed position-controlled trajectory. The device attaches to the thighs and shanks through padded straps. A parallelogram mechanism allows the vertical translation of the patient's torso, restricting lateral translation. The patient's body weight is actively unloaded as needed through an overhead harness. The Lokomat is currently being used in over 100 clinics worldwide. In 2005, *Wirz* et al. [64.85] reported preliminary results of robot-assisted BWSTT with the Lokomat in 20 chronic incomplete SCI patients. The improvements in overground walking speed and endurance were statistically significant: approximately 50% gain on average in the 16 patients who were ambulatory before training. There were no significant changes in the requirement of walking aids, orthoses, or external physical assistance. The improvements appear to be comparable to those achieved by similar SCI patients who received therapist-assisted BWSTT [64.161, 172]). For the case of stroke patients, however, therapy with the Lokomat was beneficial but about half as effective as treadmill-based or therapist-based training in improving overground gait velocity and endurance [64.173, 174].

The ReoAmbulator (commercialized by Motorika and marketed in the United States as AutoAmbulator) consists of two robotic arms that assist patients to step on a treadmill with their body weight supported as needed. The interface to the patient's legs is through straps at the thigh and ankle. The ReoAmbulator is currently being used in at least 57 HealthSouth rehabilitation centers, all of them in the United States, but little data have been published concerning its use.

The LokoHelp (commercialized by LokoHelp Group) assists users' feet motion along physiological trajectories while walking on a treadmill, also providing body weight support. Clinical trials [64.167, 171, 175] show that therapeutic outcomes are similar to manual training with reduced therapist effort.

The G-EO system (commercialized by Reha Technology) is also based on the footplate principle as for the GT I system. Studies with stroke survivors [64.176], patients with SCI [64.177] and with Parkinson's disease [64.178] have recently shown the value of this particular device.

Further Research and Developments on Robotic Therapy for Walking

Several groups worldwide are working toward improving gait-training robotic technologies. A great deal of effort has been made to incorporate and investigate the ability to assist as needed [64.72, 179–183], that is, the ability of the robot to let the patients contribute to the locomotor efforts as much as they are able. This is likely essential for maximizing locomotor plasticity [64.184]. Some effort has also been directed toward adding more active DOFs, particularly for torso manipulation [64.182, 185]. These robotic tools are needed not only for their potential clinical use in therapy, but for studying what aspects of the assistance are important for effective gait training and how best to control and implement them with robotic devices.

The team responsible for the GT-1 has developed the Haptic Walker [64.179], which maintains the permanent foot/machine contact but allows the footplates to move along 3-DOF trajectories. In addition, it incorporates force feedback and compliance control, as well as haptic simulation of ground conditions (e.g., stair climbing).

An advanced version of the Lokomat integrates force sensors and automatic adaptation of gait patterns to allow for a reduction of the interaction effort between patient persons and machine [64.180]. It has been tested on unimpaired persons and SCI patients, who were able to influence the gait trajectories toward a more desired motion by means of their own motor activity [64.180, 186].

PAM is a 5-DOF robot for torso manipulation, and POGO is a leg robot with 2-DOF per leg. PAM's and POGO's actuators are pneumatic, which cost less than electric motors and have higher power-to-weight ratios [64.185]. The robots' ability to control forces and yield to patients and/or therapists has been tested with unimpaired and SCI participants [64.187]. Of particular note here is the development of an adaptive synchronization algorithm that allows these compliant robots to provide assistance at the right time as the participants varies the timing or size of steps.

Based on the string-puppet principle, the String-Man achieves weight bearing and compliant 6-DOF torso manipulation by means of seven wires and a force sensor on each wire [64.182]. In addition, a control scheme has been designed for the String-Man to control both the zero-moment-point location and the ground reaction force with help of foot force sensors.

Veneman et al. [64.189] developed the LOPES exoskeleton (lower extremity powered exoskeleton) visible in Fig. 64.8. In this system, two horizontal pelvis translations are actuated, while the vertical motion is left free and weight is compensated. Furthermore, the LOPES have three actuated rotational joints per leg: two at the hip and one at the knee. The actuation system for hip and knee flexion/extension [64.183] combines Bowden cables with series elastic actuation. The Bowden cables allow the motors to be mounted remotely in a fixed position, thus reducing the mass to be moved on the exoskeleton links. The spring element connecting the Bowden cables with the joint allows the closing of a torque feedback control loop with a position sensor that measures the spring elongation, a concept inspired from the series elastic actuators (SEAs) described by Pratt and coworkers [64.190].

Veneman et al.'s experimental results on LOPES SEAs show that adequate torque control bandwidth was achieved by the prototype of their Bowden-cable-based actuation design [64.183], so that the robot can execute both a stiff, position-dominated robot-in-charge mode and a compliant, low-impedance patient-in-charge mode. Some results on the testing of the LOPES exoskeleton are presented in [64.192–194].

The FET European project *Evolving Morphologies for Human–Robot Symbiotic Interaction* (EVRYON) investigated a novel approach for the design of wearable robots in which robot morphology and control are co-evolved in a physics-based simulation environment to achieve a symbiotic interaction, with useful behaviors (walking patterns) emerging from the dynamic interaction between the robot and the human body. To narrow the search space, an atlas of topologies of robot architectures assisting hip and knee flexion/extension was produced [64.195], showing that only ten topologies are capable of providing independent assistance to hip and knee if the number of robot links is not higher than 4 and only revolute joints are considered. The kinematic structures (morphologies) descending from such topologies do not require the alignment of robot and human joints (nonanthropomorphism), thus possibly shortening calibration time and limiting wearability issues. Among the abovementioned ten topologies, only

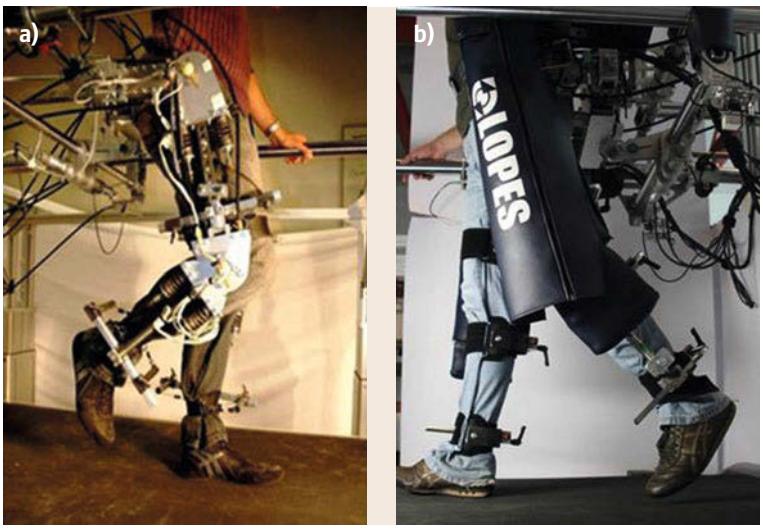


Fig.64.8a,b LOPES (lower-extremity powered exoskeleton) device for gait training and assessment of motor function in stroke survivors (after [64.188])

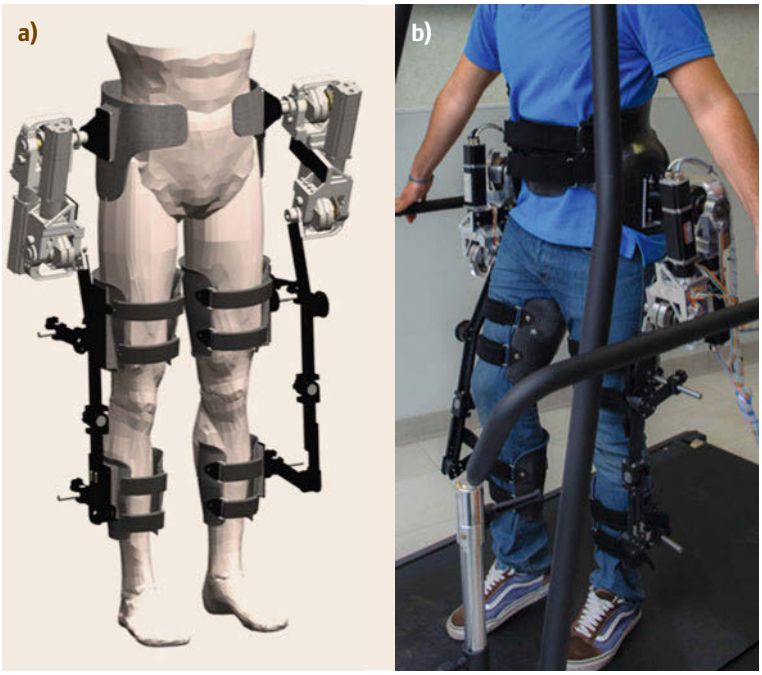


Fig.64.9a,b EVRYON/LENAR. **(a)** Wearable robot to assist hip and **(b)** knee flexion/extension through series elastic actuators (after [64.191])

one allows us to mechanically (intrinsically) minimize unwanted shear forces while keeping encumbrance low. A morphology belonging to this topology class was optimized to minimize reaction forces on the human body [64.196] and to maximize wearability and backdrivability [64.197]. Based on this study, the lower extremity nonanthropomorphic robot (LENAR) [64.191] was developed (Fig. 64.9), incorporating custom-made series-elastic actuators for a robust interaction control [64.198].

A different approach to gait training was taken with the KineAssist device [64.199]. KineAssist (Fig. 64.10) is a motorized mobile platform that follows the patient and therapist as they move overground and incorporates a smart brace that compliantly supports the patient's trunk and pelvis. This smart support is designed to allow the therapist to adjust its stiffness from fully rigid down to fully compliant. Within a safety zone, the fully compliant mode allows patients to challenge the limits of their stability. A compliant virtual wall catches the patients when they lose balance. The location of this virtual wall is also adjustable. The body weight can be unloaded as needed. The main advantage of this system is the possibility for the therapist to work in close contact with the patient while cooperating with the robotic system, which deals with the crucial, basic task to keep the patient stable and safe. From this research platform, HDT Robotics began commercialization of the KineAssist-MX. Actuation and sensors allow interactive force-field environments so that a wide variety of challenging mobility experiences can be delivered to the user.

Other efforts include *Ferris* and coworkers [64.202], who are developing foot, ankle, knee, and hip orthoses actuated by artificial pneumatic muscles that may possibly be used to assist in gait training. The Rutgers Ankle is a 6-DOF pneumatic system based on a Stewart Platform that allows exercise of the ankle [64.203]. Also in the United States, Agrawal's group proposes the use of gravity-balancing leg orthoses for people with gait impairments to practice walking [64.204]. Their designs allow the orthoses to passively support the gravity torque required at the patient's joints. This approach would have the advantage of being safer than powerful robots for clinical use. They have also extended their design

Fig. 64.10 KINE ASSIST for unobtrusive support to patients, allowing them to walk on their own, or with variable levels of support (after [64.200])

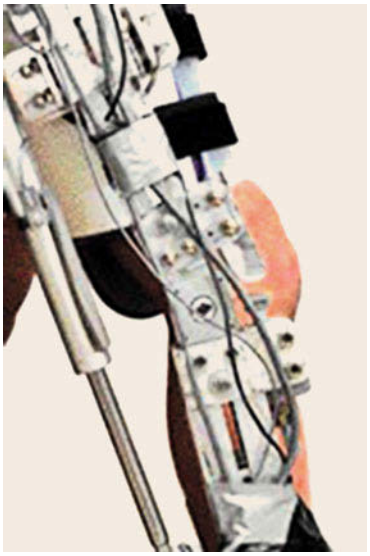


Fig. 64.11 Active leg exoskeleton (ALEX) to supplement traditional rehabilitation therapy (after [64.201])

to include actuators with reduced torque requirements [64.205]. A robot has been used to provide graded body weight support as a patient who cannot bear full body weight because of a medical problem walks in a circle [64.206]. *Banala* et al. developed the treadmill-based rehabilitation robot ALEX [64.207], as seen in Fig. 64.11. In this system, hip and knee joints are actuated in the sagittal plane while hip abduction/adduction and ankle motion are spring-loaded. More details and further examples of robotic systems for walking therapy are reported in [64.208].

Other Robotic Movement Therapy Approaches

As reviewed earlier, most of the work to date in robotic therapy devices has focused on robots that are attached to patients to assist them in practicing reaching or walking exercises. Other early proposals for using robots for movement therapy included using two planar robot arms to carefully control continuous passive motion of the knee following joint surgery [64.37], and using a multiaxis robot arm to place targets for patients doing reaching exercises [64.209]. An emerging approach toward robotic movement therapy is to provide the therapy at a distance, in a form of telerehabilitation, in order to improve accessibility to the therapy [64.81, 123, 210]. Non-contacting, socially assistive robots, as reviewed in Chap. 73, may play an important role in motivating and monitoring therapy.

64.3 Aids for People with Disabilities

Enabling technologies assist people with disabilities to achieve a quality of life on a par with able-bodied individuals through increased functional independence. The main issue with most such technologies is that disability has a highly individualized impact: a solution for one person will not work for someone else, even if their disabilities appear clinically similar.

64.3.1 Grand Challenges and Enabling Technologies

The more a disability impacts function, the more costly the technical intervention tends to be, since the consumer market cannot benefit from economies of scale if each solution must be individualized. As an extreme example, an electric wheelchair with individualized padding, motorized recliner, and customized joystick control costs as much as a mass-produced mid-sized automobile, but has a fraction of the electronics, robustness, and functions. A grand challenge for assistive, enabling technologies is to find a means to make mass-personalization possible, as it has been in the automotive industry, for example. One component is designer focus. If we can re-badge *assistive technology* as *design for well-being products*, the change in focus from *fixing* people to improving their quality of life will have the effect of mainstreaming disability itself so that manufacturers of consumer equipment tend to develop products that can explicitly accommodate a much wider range of functional abilities and therefore provide benefit to a larger, overall less-able, consumer base. As the average age of the baby boomers climbs into retirement years with significant disposable income, this segment will compel the market into providing better solutions to their well-being needs.

Another grand challenge is robotic autonomy. Especially for persons with reduced communication, physical and/or cognitive abilities, a rehabilitation robot will need to have sensory (e.g., vision, auditory) and motor capabilities, combined with its own software processing capabilities (also termed artificial intelligence), that make it a sufficiently safe and capable system to coexist with and benefit humans. This challenge will to some extent be dependent on continuing increases in computer-processing power, and also specifically dependent on the algorithmic developments that issue from the community of robotics researchers.

For instance, several advanced navigation assistive tools for blind and visually impaired persons have

been developed by exploiting knowledge and technologies directly derived from research on autonomous robot navigation; as a sample, *Borenstein* and *Ulrich* in 1997 developed the GuideCane [64.211], an intelligent cane, based on ultrasound proximity sensor technology, which was designed to help blind or visually impaired travellers to navigate safely and quickly among obstacles and other hazards faced by blind pedestrians [64.212].

Research on robotic aids (namely *physically assistive robots* or, also, *contact assistive robots*) has so far primarily targeted persons with mobility and manipulation limitations, rather than children and adults with cognitive impairments [64.213, 214]. However, increases in the prevalence of cognitive impairments related to aging will make the latter focus increasingly important. *Socially assistive robots*, also named *contactless assistive robots*, are emerging assistive systems that focus on helping human users through social rather than physical interaction (see Chap. 73 for more details on socially assistive robots). Research has been limited to the mobility focus due to the difficulty of designing and developing intrinsically safe robots that can coexist with people and exhibit a certain amount of autonomy while performing useful work. Robots, therefore, today rely on user vigilance and explicit control to be safe. If the user does not have the cognitive capacity to evaluate a robot's safety situation or the ability to communicate efficiently, then the positive value of a function-enhancing robot is nullified by the harm that it could inflict on the user or bystanders. Coupled with the fact that the design of interfaces to personal robots is still in its infancy, a challenge for robotic aid developers is a significant improvement in intrinsic safety without a decrease in function (strength, speed, etc.) from what is typical today in industrial robotics.

To address some of these challenges, the US government, through NSF, NIH and other federal agencies, in 2011 issued a call for a $ 50 million per year, 5-year program called the National Robotics Initiative (NRI) [64.215]. The realization of co-robots acting in the direct support of individuals and groups. A substantial amount of this funding is focused on healthcare of the future.

Disabilities and Functional Limitations Served by Robotic Aids

Assistive robots have been designed for people who have become severely disabled as a result of, for example, muscular dystrophy or a high-level SCI, for children who have CP, and more generally for anyone who lacks the ability to manipulate household objects. A market research study conducted 10 years ago, specifically for rehabilitation robotics clients, conservatively projected a US market of 100 000 people [64.216]. With the incidence of disability increasing exponentially, and the niches that robots can fill in rehabilitation applications multiplying with advances in robotics and rehabilitation science, it is clear that the market for rehabilitation robotics can only continue to increase.

Human–Robot Interface Design for Assistive Robots

A fundamental difference between using industrial and assistive robots is the interface required to command, control, and ultimately benefit from them. An industrial robot commonly has a combination of a manual controller and a programming language interface to allow an operator to teach a robot where to go and to enter the specific motion, grasping, tool changing, and error-recovery steps it must follow repeatedly in its factory automation scenario. An assistive robot, on the other hand, typically has three main differences and challenges:

1. The operator is not by definition a roboticist or engineer, so the interface must make accessible all the functions of the robot to allow its user to complete the required tasks.
2. The user of a rehabilitation robot is, by definition, a person with a disability, which means that physical, sensory, communication, and/or cognitive limitations in accessing the commands and controls of a robot need to be handled on a systems level by the designers of robots and their interfaces, with critical attention to universal design principles, and
3. All rehabilitation robots require individualization of the interface to each user by the engineering and therapy professionals in charge of prescription and fitting, since disabilities vary considerably in how they restrict adaptability to standard configurations [64.217].

Interfaces of assistive robots consist of the software and hardware components conceived to enable a person with a disability to interact with an assistive device, thus tapping into residual communication capabilities of each user. For example, many people with tetraplegia retain the ability to move a hand, arm, foot, or the head in a repeatable, even if range-limited way, and possibly even in two axes, such as forward/backward and left/right. With a proper placement of pushbuttons, a joystick, or noncontact position measurement device, a rehabilitation engineer and therapist can develop a custom solution for each of their clients with disabilities to control a wheelchair computer and robot. In addition, adaptive hardware and software for controlling a computer, such as head-position cursor control,

eye-trackers, speech recognition systems, trackballs and special keyboards, can be used to provide access to computer-based robot functionality.

Even more important than for able-bodied computer and robot users, redundancy in input modality is important for persons with disabilities to prevent a system from becoming inoperable due to a simple interface malfunction or calibration problem. Providing two means of creating a mouse-click action (for example, a separate button placed next to a cursor-control trackball, as well as dwell time on a software *button* on-screen), even if one is inherently slower than the other, allows continued and uninterrupted use of the computer without outside assistance even if one of the two fails.

For therapy robots, physical interfaces resemble those for physical and occupational therapy equipment, in general, and have a commonality with sports equipment interfaces as well, with adjustable hook-and-loop-type straps, heat-formable plastic cuffs, soft rubber, foam-based materials, and durable coverings for abrasion resistance and long wear. After a session or two for fitting and adaptation, a person using a therapy robot can often use the same interface for a long period of time.

In summary, the keys to interface design are customizability, individualization, functional redundancy, adaptability, and patience in getting the interface to a comfort and functional level appropriate for the effective use of the robot.

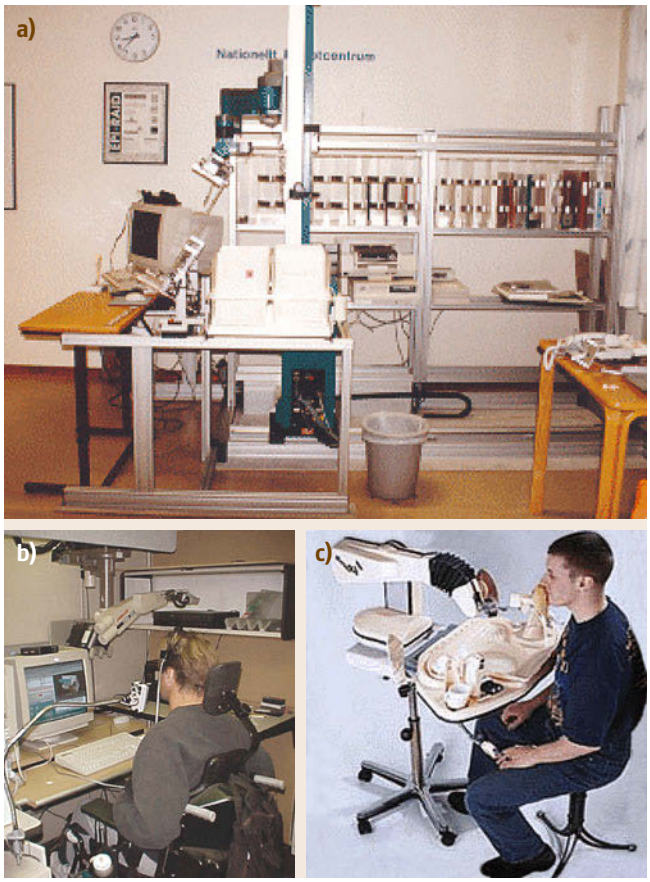


Fig.64.12a–c Workstation-type robots: **(a)** AfMaster, developed by the French Muscular Dystrophy Association, **(b)** ProVAR, developed at the VA Palo Alto Rehabilitation R&D Center, and **(c)** Handy-1, developed by RehabRobotics, Ltd. (UK)

64.3.2 Types and Examples of Assistive Rehabilitation Robots

As mentioned in the Introduction, assistive robots can be divided into three main categories: manipulation aids, mobility aids, and cognitive aids. Each can be subdivided as follows. Manipulation aids are commonly divided into fixed, portable, and mobile subtypes. Mobility aids are divided into electric wheelchairs with autonomous navigation features and smart walkers. Cognitive aids are divided into communication aids such as pet robots and autonomous caretaker robots. These categories are introduced below, and representative systems that have undergone scientific user studies or are commercial products are presented (Fig. 64.12a–c). Other examples are mentioned in Sect. 64.1.3.

Manipulation Aids: Fixed-Base

Common robots of this type are ADL and vocational manipulation aids and kitchen robots. In the United States, the professional vocational assistive robot (ProVAR) is a research prototype based initially on a PUMA-260 robot arm mounted on a 1 m-transverse overhead track that allows the robot to manipulate objects and operate devices on side shelves and the table-top, bringing objects (like a drink of water or throat lozenge) to the robot's operator. The interface is via a JAVA/VRML plug-in to a common Internet browser, delivering high-level control to disabled office workers in a conventional pull-down menu and a control screen interface [64.27, 218]. This system and its predecessor DeVAR have been field tested by over 50 subjects at 5 rehabilitation clinics to assess feasibility and acceptability [64.219, 220]. At a cost of over US $ 100 000 currently, there are currently no plans for eventual product introduction.

In the EU, following a development path parallel to ProVAR's, there is the AfMASTER/RAID workstation, whose concept, instead of being built into a workstation, includes a 2 m × 3 m robot work area in the user's office to store objects and place appliances, next to the user's own office space. The system has been developed

over a 20-year span and was briefly in limited production [64.29] but is no longer offered for sale.

The kitchen robot, Giving-A-Hand, developed at the Scuola Superiore Sant'Anna in Pisa, Italy, is a low-degree-of-freedom device for mounting on the front rail of a kitchen counter and able to move food containers to and from appliances, such as refrigerators and ovens [64.221]. With an integrated control system, it can also make use of the internal controls of the devices to, for example, set cooking times and open doors.

The UK-developed Handy-1 is a domestic robot with 3-DOF designed for one-switch operation by persons with CP [64.30]. Originally designed to allow a person to eat a meal one bite at a time, its application areas have been extended to face hygiene and cosmetics. A commercial product selling for about US$ 6000, it has been a commercial success due to its simplicity and application focus. An even simpler feeding robot, the UK's electric Neater Eater [64.20] is for sale worldwide at about US$ 5000, and is designed for eating only.

In [64.222], an overview of manipulation robotic aids is provided. They are classified through five criteria, based on robotic arm usage scenarios and surveys. In particular, new assistive manipulators have recently been developed that address interaction safety as a priority criterion in their design. Take for instance JACO [64.223], iARM [64.224] and RAPUDA [64.225] as examples of robot arms achieving safety by limiting the performance of the robotic arm in terms of arm-movement speed and acceleration in space, end-effector force and maximum possible payload. Examples of robots addressing safety through backdrivable joints (as in the WAM Arm [64.226]) or through active impedance control (as in KARES II, WAM Arm, Elumotion RT2 [64.227], DLR LWR-III [64.228]] can also be mentioned.

While a robot conventionally connotes a standalone system with some automation features, a smart bed and a smart home can legitimately be termed robots since they sense and act with motors under the shared control of its human users and its real-time software programming. Smart beds, such as SleepSmart measures body position and temperature, as well as trends and anomalies over the course of a night. Restlessness can be measured, and bed geometry (tilt of bed segments) and ambient conditions (light, temperature, sounds) can be adjusted according to presets and preferences [64.229].

Smart homes, such as the AwareHome domotic environment at Georgia Tech, NL-iRV, and the University of Tokyo [64.230], provide integrated climate, security, lighting, entertainment and transport assistance, which is enabling especially to persons with severe functional disabilities. Coupled with health care-related functionality (following section), these robotic homes can allow a person with a cognitive or physical disability to control many ADL functions and live safely through monitoring.

Manipulation Aids: Wheelchair Manipulator Arm Systems

A need for electric wheelchair users is the manipulation of objects while navigating a home or a public place such as a restaurant or grocery store. The assistive robot service manipulator (ARM) (Exact Dynamics, Netherlands) – previously known as MANUS – is a commercial robot arm that can be attached to an existing wheelchair to the side of the lap tray and controlled by the wheelchair's own joystick or a number pad [64.23, 231] (Fig. 64.13). The robot has undergone numerous user studies with persons who have muscular dystrophy, a high-level SCI or CP. Worldwide, this is currently the only commercial rehabilitation robot arm that can be prescribed by a physician and that is reimbursed by a government health care system.

Weston [64.232] and Bridgit [64.233] are two wheelchair manipulator arm systems addressing the issue of interaction safety in their design, as for some of the manipulation aids with a fixed base presented earlier. Weston uses low-power motors in order to statically and intrinsically limiting the arm's acceleration, force and payload. Bridgit is a manipulator arm placed

Fig. 64.13 Wheelchair manipulator robot MANUS developed at the Rehabilitation R&D Center, Hoensbroek, and marketed by Exact Dynamics (The Netherlands)

on a wheelchair on a rail system. The robot moves over the rail system resulting in optimal positioning for each task. The rail system also allows for easy docking of the robot either on the back or front side of the wheelchair when not in use.

Manipulation Aids: Mobile Autonomous Systems

The most commonly thought-of form of a robot is that of an autonomous, mobile system with arms, having sensory-motor functionality similar to that of a human being, while serving people in performing menial physical tasks. This Handbook's chapter on Humanoid Robots explores the domain as well (Chap. 67). Since locomotion is a key requirement for Humanoid robotics, other robots with wheeled bases have been developed before the first walking robots were invented to explore more applied domains with more short-term usefulness. In film, robots such as Star Wars' R2D2 have made this form factor commonly known around the world. More recently, real robots such as the HelpMate [64.234] have been employed in US hospitals as fetch-and-carry robot orderlies, using floor maps and short-range ultrasonic sensors for navigation and obstacle avoidance. The Italian MovAid research robot platform [64.235] (in Fig. 64.14) adds manipulation and vision to these capabilities to navigate in home-like environments to provide object manipulation and device operation to individual users. The European project Robot-Era [64.236] is following up these developments with a specific target on the needs of aging population. The German Care-O-bot [64.237] has explored advanced navigation and sensing in a wheeled robot that can also be used as a physical support to people requiring mobility and stability assistance. It has also doubled as a mobile kiosk, moving around a trade show floor and delivering information to attendees. In [64.238] a case study of a personal robot based on the PR2 Humanoid robot (Willow Garage, Menlo Park) is presented. The approach pursued consists of developing a diverse suite of open-source software tools that blend the capabilities of the user and the robot in order to enable the assistive mobile manipulators to move in real homes and work with people with disabilities.

Mobility Aids: Wheelchair Navigation Systems

A critical function for people who use electric wheelchairs for their mobility impairment and who in addition have communication or cognitive disability is semiautonomous navigation assistance (Fig. 64.15).

Add-ons to commercial wheelchairs have been developed by numerous research groups for this service. Such wheelchairs are typically referred to as smart or intelligent wheelchairs. As proposed by *Simpson* [64.240], smart wheelchairs can be classified by form factor: early smart wheelchairs were mobile robots with added seats. The vast majority of smart wheelchairs developed until 2005 were based on heavily modified, commercially available, powered wheelchairs. Only a smaller fraction of them were

Fig. 64.14 MovAid (after [64.239])

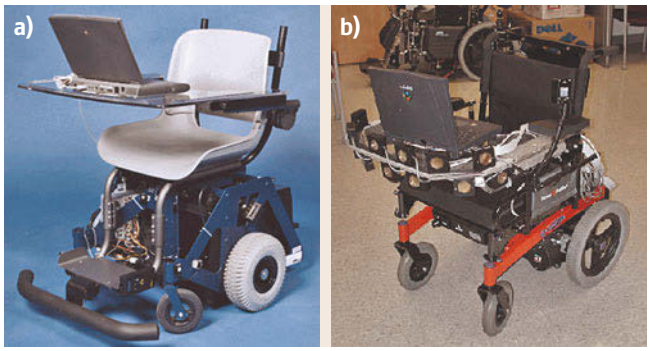


Fig.64.15a,b Wheelchair navigation aids: **(a)** Wheelesley and **(b)** Hepahaestus

equipped with *add-on* units that could be attached to and removed from the underlying powered wheelchair.

The NavChair [64.241] was one of the first to demonstrate robust wall-following, door passage even with narrow doorways, and speed adaptation to people walking in front of the wheelchair, all using only short-range ultrasonic and other sensors, but not vision. The Hephaestus [64.35] is a next-generation system made specifically as a commercial accessory for a variety of wheelchair brands, tapping into the joystick controller and power system. The *Wheelseley* [64.242] and KARES [64.33] robots have explored similar functionality using a vision system for scene analysis and way-finding.

More recently, the CanWheel project team developed an intelligent wheelchair system called NOAH [64.243, 244]. The system has three main capabilities: collision avoidance, infer the user's goal location/activity and provide automated reminders, provide navigation assistance using prompts. The rationale for proposing such solutions is to enhance mobility and to help improving the quality of life of older adults with cognitive impairments, while simultaneously reducing the burden on caregivers.

In 2013, *How* et al. [64.245] proposed a new intelligent wheelchair system (IWS) with anticollision and navigation features. User trials showed the IWS's potential to improve powered wheelchair safety and subjective usability.

The IntellWheels [64.246] project proposes a modular platform based on a multiagent system paradigm for the development of intelligent wheelchairs based on commercial products. Within this project, promising results have been achieved on the development of adapted control methods for CP users of an intelligent wheelchair. Experiments demonstrated that users felt that they had better control over the wheelchair movement when using shared control rather than manual or automatic control modalities [64.247].

The iBOT [64.248–250] is a powered wheelchair for persons with mobility impairment developed by Kamen, in a partnership between DEKA and Johnson & Johnson's Independence Technology division. Research on iBOT was discontinued in 2009. The iBOT features self-balancing technology, which allows users to go up and down staircases, to navigate on uneven terrain and to *stand* at an eye level with people walking nearby.

Mobility Aids: Walking Assistance Systems

A third type of mobile robot for stability assistance has the peculiarity that it is underactuated and has similarity with the *co-bot* concept, in that the wheels are not driven, but are actively steered and braked (Fig. 64.16). The concept of collaborative *co-bots* was originally introduced by *Colgate* et al. for robots operating in di-

Fig.64.16a–c Human assistance robots: **(a)** Care-O-bot, developed by the Fraunhofer Research Institute (Germany). **(b)** Helpmate by Transitions Research Inc., USA. **(c)** Pam-Aid (aka Guido), developed in the UK

rect physical interaction with a human factory worker, handling a shared payload. They are a marked departure from autonomous industrial robots, which must be isolated from people for safety reasons [64.251, 252]. Co-bots interact with people by producing software-defined *virtual surfaces* which constrain and guide the motion of the shared payload, but add little or no power [64.253]. Today, cobots are being prototyped in the rehabilitative and assistive context, e.g., for bed to chair/wheelchair transfer or table-top upper limb, stroke rehabilitation at home. For example, the Pam-Aid [64.254] looks like a closed-front walker on wheels and has bicycle-type handlebars. The person walking behind the device turns the handlebars, causing the wheels to turn in the correct direction. If the ultrasonic sensors detect an obstacle in front of it, the brakes prevent the user and device from colliding with it. The Care-O-bot (see earlier) designed originally as a mobile autonomous robot approximately human size, has a similar set of handlebars, similar to the Pam-Aid it, so it can be used as a smart walker. The larger mass of the Care-O-bot, however, requires it to be motorized.

Mobility Aids: Exoskeletons

Exoskeletons for walking assistance are similar to robots used in treadmill-based environments for rehabilitation. These systems are portable and autonomous and intended to be used in daily life scenarios. Several review papers have been recently published [64.255–257] on this topic.

In the framework of NISTs Advanced Technology Program, Ekso Bionics (Berkeley, USA) developed the Ekso device [64.258] (Fig. 64.17). This robot has been developed for people with lower extremity weakness or paralysis due to neurological disease or injury (e.g., spinal cord injuries, multiple sclerosis, Guillain–Barré syndrome) and it has an almost anthropomorphic structure with hip and knee joints actuated in the sagittal plane. The ankle joints are not actuated but are compliant in the sagittal plane and locked out in the other DOFs. Testing of the device has included paraplegic persons with complete or incomplete paralysis [64.259] and chronic stroke patients [64.260].

The ReWalk was developed by Argo Medical Technologies [64.261]. It is actuated by DC (direct current) motors at the hip and knee joints in the sagittal plane, while the ankle joint is not actuated. The system is designed with a remote controller that can be used to change the motion mode (e.g., ground walking, climbing stairs). A posture sensor on the torso detects the upper body movement of the user and estimates motion intention. The wearer also has to use crutches for stability and safety reasons. The system is undergoing

Fig. 64.17 Ekso Bionics exoskeleton for paraplegics (after [64.258])

clinical trials among other research centers at Moss-Rehab (Philadelphia, USA) and at the Centro Protesi INAIL di Vigorso di Budrio (Bologna, Italy) on paraplegic subjects. The device is now available on the market in two versions: the ReWalk-Rehabilitation for institutional use and the ReWalk-Personal developed for daily use (Fig. 64.18).

Sankai's group at the University of Tsukuba (Japan) developed an exoskeleton both for performance augmentation and for rehabilitation and assistance [64.263, 264]. The current version, HAL-5, powers the flexion/extension of hip and knee via DC motors while ankle dorsi/plantar flexion DOF is passive. The HAL-5 system (Fig. 64.19) integrates a number of sensors: skin-surface EMG electrodes placed below the hip and above the knee on both the anterior (front) and posterior (back) sides of the wearer's body, potentiometers for joints angles measurement, ground reaction force sensors, a gyroscope and accelerometer mounted on the backpack for torso posture estimation. HAL-5 is currently commercialized by the spinoff company Cyberdyne (Tsukuba, Japan). To date, it appears that no peer-reviewed, quantitative results have been published

Fig. 64.18 ReWalk computerized exoskeleton (ReWalk Robotics, Inc., USA; after [64.262])

Fig. 64.19 Robot suit HAL-5 designed by Japanese robotics firm Cyberdene

on the effectiveness of the exoskeleton for the improvement of walking functions.

The Vanderbilt powered orthosis [64.265] is a powered lower limb exoskeleton intended to provide gait assistance to individuals suffering from spinal cord injury by providing assistive torques in the sagittal plane for both the hip and knee joints. It includes neither a portion that is worn over the shoulders, nor a portion that is worn under the shoes. Each joint is powered by a brushless DC motor. The orthotic is intended to be worn together with a standard ankle–foot orthosis, which provides support at the ankle and prevents foot drop during swing. In order to demonstrate its ability to assist walking, the orthosis was experimentally tested on paraplegic subjects. Experimental results indicate that the orthosis is capable of providing a repeatable gait with knee and hip joint amplitudes that are similar to those observed during non-SCI walking.

REX, produced by REX Bionics (Auckland, New Zealand), is an anthropomorphic lower body robot designed for sit-to-stand, stair climbing and overground walking, without the use of crutches. The system does not use sensors to detect the intention of the user but rather it uses a joystick as its interface. The system has been tested with healthy subjects, and for sit-to-stand of wheelchair users [64.266].

Cognitive Aids

There has recently been increased interest in using robots as motivational and educational agents during rehabilitation therapy. This approach typically involves small, pet-like, toy-like, approachable devices that do not physically interact with the patient, but exist primarily to engage the patient in an affective way that promotes personal health, growth, and interaction. For more information, please see Chap. 73.

64.4 Smart Prostheses and Orthoses

In 2005, the Defense Sciences Office (DSO) of the US governmental research agency DARPA launched a program to revolutionize prosthetics in a four-year timeframe. According to the agency website, this program will:

> *deliver a prosthetic arm for clinical trials that is far more advanced than any currently available. This device will enable many degrees of freedom for grasping and other hand functions, and will be rugged and resilient to environmental factors. In 4 years, DSO will deliver a prosthetic for clinical trials that has function almost identical to a natural limb in terms of motor control and dexterity, sensory feedback (including proprioception), weight and environmental resilience. The four-year device will be directly controlled by neural signals. The results of this program will allow upper limb amputees to have as normal a life as possible despite their severe injuries.*

64.4.1 Grand Challenges and Roadblocks

This program announcement lays down the grand challenge for prosthetics research in an ambitious timeframe: develop an artificial limb that has function and durability at least as good as a natural limb. There are several roadblocks to meeting this challenge. First, providing an intuitive way for individuals to control and coordinate multiple joints of a robotic limb is challenging. Second, robots do not yet match the human arm in terms of the combination of range of force, weight, and duration of use with a portable power source. Third, human limbs are rich with tactile and movement sensors. Installing artificial sensors on a robotic limb, and then returning information from those sensors in a way that is usable by the user is challenging. Thus, solving the grand challenge will require better sensory–motor interfaces for prosthetic limbs, as well as lighter, stronger actuators, and better power sources.

Substantial progress has recently been made in improving sensory-motor interfaces for prosthetic limbs, and this progress is the focus of this section. For the current state of robotic actuators that could be used in prosthetic devices, the reader is referred to Chap. 77 on Neurorobotics. For an overview of the design of conventional prosthetic hands and arms, the reader is referred to [64.267].

64.4.2 Targeted Re-Innervation

Standard prosthetic arms and hands are commonly controlled with a cable drive or by electromyogram (EMG) signals from residual muscles. For example, to open and close an artificial hand, one common technique is to place a Bowden cable around the shoulders in a harness, and connect the cable directly to the artificial hand. The user can then shrug the shoulders to move the cable and open and close the hand. Alternately, electrodes can be placed on a muscle in the residual limb or on the user's back, for example, and then used to control a motor on the artificial hand. The cable technique has the advantages of simplicity, and of having the property of extended physiological proprioception (EPP), which refers to the fact that the grip force is mechanically transmitted back to the user's shoulder muscle force sensors so that the user can gauge the strength of the grasp. Because of their simplicity and EPP, cable drives (or body powered prostheses) have been more popular than myoelectric (or externally powered) prostheses. However, the body-powered technique is amenable to controlling only one degree of freedom at a time, although chin switches, for example, can be used to switch between degrees of freedom in a somewhat cumbersome way. The myoelectric approach can be used to control multiple degrees of freedom, but such control is nonintuitive and cumbersome. Also, multiple control sites for reading out EMG are not available for people who have lost their entire arm. Thus, prosthetic control systems are typically limited to one or two degrees of freedom, while functional arm and hand movement benefits from at least four degrees of freedom (three to position the hand, and one to open and close it).

Kuiken et al. [64.268] recently developed a novel approach to improving control of a multijointed prosthetic arm. In this targeted re-innervation technique, they re-routed the nerves that previously innervated the lost limb to a spared muscle, and then read out the user's intent to move the limb using electromyography at the spared muscle. They demonstrated this technique in a bilateral shoulder disarticulation amputee who had lost both of his arms in an electrical power accident. They took the residual brachial plexus nerve for the left arm, which normally innervates the left elbow, wrist, and hand, and moved it to the pectoralis muscle. The subject could still contract his pectoralis muscle, but this muscle was no longer useful to him since it used to attach to his now-missing humerus. A surgeon dissected portions of the nerve associated with different muscles in the elbow, wrist, and hand, and innervated three bundles of the pectoralis muscle. After three months, the nerve re-innervated the bundles so that the patient could cause the bundles to twitch by trying to bend his missing elbow, for example. Surface EMG electrodes were

placed over the bundles. Then, when the user willed to open his hand, for example, a pectoralis muscle bundle contracted, and this contraction was detectable with the EMG electrodes. The EMG signal was in turn used to control the hand motor of the prosthetic arm. The net result was that the user could will his different (missing) anatomical joints to move, and the corresponding joints on the robotic arm would move. He could simultaneously operate two joints, such as the elbow and the hand. The user became able to do tasks that he was not able to do before with his conventional myoelectric controlled arm, such as feeding himself, shaving and throwing a ball. A secondary remarkable finding was that the sensory neurons in the re-routed nerves re-innervated sensors, so that now when the person's chest is touched, the person perceives it as a touch to his missing limb. This sensory re-innervation could possibly be made into an interface to provide tactile sensation from the artificial limb. These findings were recently confirmed in another person who received targeted re-innervation [64.269].

64.4.3 Neural Interfaces for Limb Prosthetic Devices

Neural interfaces provide an interesting and challenging solution to retrieve the natural way of interfacing the human nervous system to prosthetic artifacts. They are systems capable of recording either invasively or noninvasively the electrical activity of peripheral nerves as well as of the brain cortex. It has also recently been shown that the direct electrical stimulation of a residual peripheral nerve can provide usable information regarding force to a person with an amputation [64.270], thus paving the way for bidirectional neural interfaces capable of restoring both efferent and afferent information flow to/from the prosthesis. Recently, thin film intrafascicular electrodes implantable in peripheral nerves have been developed [64.271], and successfully validated in 2008 in Italy at Campus Bio-Medico University of Rome on a human amputee [64.272] within the LifeHand project, a cluster of European and Italian research actions focused on neural interfaces for prosthetics (Fig. 64.20). In 2013, a second round of experiments in Italy demonstrated the possibility of delivering physiologically appropriate (near-natural) sensory information to an amputee during the real-time decoding of different grasping tasks to control a dexterous hand prosthesis by stimulating the median and ulnar nerve fascicles using transversal multichannel intrafascicular electrodes, according to the information provided by the artificial sensors from a hand prosthesis [64.273]. The results also demonstrate that the subject was able to identify the stiffness and shape of

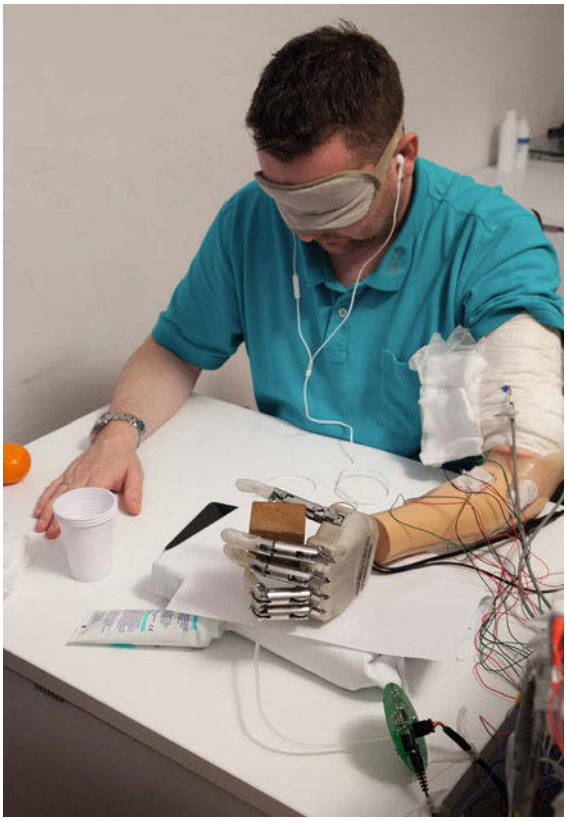


Fig. 64.20 LifeHand aims to create a completely implantable prosthesis system, richly sensorized and cronrolled through the patient's nervous system, with a dexterity comparable to a natural limb (courtesy of Campus Bio-Medico University Rome, Italy)

three different objects by exploiting different characteristics of the elicited tactile sensations. These results are in line with earlier studies which outlined the importance of restoring tactile feedback on a prosthetic device, such as the one proposed by *Meek* et al. in 1989 [64.274].

Today the above mentioned approach is trying to be extended on lower limb prosthetics as well. *Herr* et al. a MIT pioneered a new class of bio-hybrid smart prostheses and exoskeletons [64.275] aiming at improving the quality of life of people with physical challenges. Some of these devices are now commercialized by a spin-off company, BiOM. For instance, a computer-controlled prosthesis called the Rheo Knee [64.275] is outfitted with a microprocessor that continually senses the joint's position and the loads applied to the limb. A powered ankle–foot prosthesis emulates the action of a biological leg to create a natural gait, allowing amputees to walk with normal levels of speed and metabolism as if their legs were biological [64.275].

There has also recently been progress in decoding movement-related signals in real-time directly from the brain (see the cover story and related articles in *Nature* of 13 July 2006 [64.276]. The first of several recent human volunteers, a person with tetraplegia due to SCI, has received a BrainGate electrode array implant, and has been able to control the movement of a cursor on a computer screen [64.277]. Noninvasive systems operate by recording brain activity from *the outside* of the skull via well-known clinical noninvasive diagnostic devices [64.278] such as electro-encephalogram (EEG). Individuals have been demonstrated to be able to learn to control the amplitude of the EEG signal as a function of time, or the amplitude of specific frequency components of the EEG signal, with a moderate amount of practice (several hours to several days). The level of control is sufficient to operate a typing program on a computer, or to control the movement of a cursor to multiple targets.

In summary, given the significant progress observed in the last decade, it appears that future control systems for smart prosthetics and orthoses will have the option to rely on direct interfaces to the brain, which should allow control of multiple joints through thought alone. The initial work on both targeted re-innervation and brain–machine interfaces to the PNS or to the CNS has allowed three to 4-DOF of control in a naturalistic manner and elicitation of some sensory feedback, which is an advance over conventional prosthetic control techniques.

64.4.4 Advances in Neural Stimulation

Functional neural stimulation techniques (FNSs) seek to electrically stimulate the residual nervous system to re-animate the limbs. FNS for standing, walking, reaching, and grasping has been demonstrated, but these techniques have met with limited commercial success because of a combination of factors, including the ease of use of systems that use surface electrodes, duration of use before fatigue, risk from implantation and complexity of the associated control problems.

Two research lines are being pursued to help move the FNS field forward. The first one focuses on hardware innovation. A good example is the BION, an injectable stimulator the size of a very large grain of cooked rice [64.279], which can be inserted without surgery (using a large-gauge needle) and is robust and resistant to infections. The second research line aims at stimulating the control circuits in the nervous system rather than individual muscles. For example, it has been shown that locomotor-like movements can be eliciting in multiple muscles of the cat hind limb by stimulating regions of the spinal cord directly [64.280].

64.4.5 Embedded Intelligence

Recent robotics-related advances for prosthetic legs have included embedding microprocessors and passive braking systems into artificial knees, so that the knees can, for example, be made relatively stiff during the stance phase of gait, and free to move during the swing phase of gait [64.281]. The first microprocessor knee introduced was the Ottobock C-Leg (Germany), introduced in 1999. The C-Leg uses a servomotor to adjust valves to hydraulic pistons. The rechargeable battery lasts about 24 hours. The pattern of resistance throughout the gait cycle can be adjusted for each user. A dramatic example of the benefit of the C-Leg is the story of a man who made it down from the 70th floor of the World Trade Center on 9/11/2001 with only minor bruising to his residual stump [64.282]. Other microprocessor-controlled knees are the endolite adaptive prosthesis, which uses pneumatic and hydraulic valves, the Rheo Knee (Ossür Hf, Iceland), which uses magneto-rheologic fluid to vary the knee impedance and the Intelligent Prosthesis.

The first powered knee that can generate power, rather than just dissipate energy, is currently being commercialized by Össur as the Power Knee. The system combines an electromechanical power source that will be controlled with input from sensors on the sound leg shoe. Initial reports suggest that this is the first knee that allows the user to walk up stairs with a step-over-step pattern.

64.5 Augmentation for Diagnosis and Monitoring

A critical aspect of rehabilitation is health maintenance with age-related or degenerative functional decline and after a medical intervention. In-home diagnostic equipment, devices worn on or in the body for vital signs monitoring, tele-health services, and institution-based monitoring automation are all examples of systems being developed to improve the quality of life for both persons at risk and their caregivers. Institutional systems of this nature, more properly part of the field of clinical engineering, are incorporating more robotic, networked and autonomous devices to take more accurate diagnostics, provide better information to physicians and

provide faster alerts. Key enabling technologies in this field are advanced materials and nanotechnologies.

64.5.1 Grand Challenges and Enabling Technologies

For all devices that are worn on the body, the interface must be skin compatible. A grand challenge for this field in the near-term is the better incorporation of active and sensing elements with textiles. Several prototype sensor shirts show promise, but rehabilitation will have a much richer toolset for diagnosis and monitoring with advances in this area. Nanotechnology, an enabling technology for the longer term, has the promise to miniaturize virtually everything mechatronic that is currently macroscopic. Injected devices, such as nano-robotic drug dispensers and clot-busters will aid in rehabilitation.

64.5.2 Smart Homes for Health Care Monitoring and Care

A special class of fixed-station rehabilitation robots is an automation system designed to provide a safe environment to assist and monitor a person with a disability living at home or in an institutionalized setting such as an assisted living facility or nursing home.

Smart Nursing Home Automation

An assisted living, hospice, or nursing care institutional facility will include residents who have mild to severe cognitive impairment in addition to physical disabilities. The facility may have zones to separate residents who have different levels of dependency since the architectural, monitoring and personnel needs are different. To better serve residents and guests, to optimize function and to minimize cost, only the areas for persons with high dependency have a 24 h staffed vital signs monitoring and alert capability, for example. Facility care is highly staff-intensive, though automation through diagnostic vital signs monitoring, electronic surveillance, and patient tracking continue to improve safety and efficiency. Robotics and automation are beginning to find applications in the physical tasks associated with patient care, therapy and oversight. Some examples are described below.

Examples of the State of the Art

Wandering, especially at night, is a significant problem for institutional facilities with ambulatory residents who are cognitively impaired. Simple architectural modifications include painting the hallway in front of doors black to make them look like deep holes. An automated voice system triggered by a motion detector to say *Go back to bed* is effective, but not fool-proof, either. Resident detection systems based on ID badges with embedded RFID chips that can be sensed in a hallway work, but only if the resident is wearing it. A robotic sentry system, including mobile platforms to aid in solving this problem, especially at night, has not yet been developed.

A serious rehabilitation issue in institutionalized facilities is the transfer of residents from bed to wheelchair and other surfaces. A number of manual, electric, and robotic devices have been developed to assist the nursing staff to safely transfer residents and patients who may be significantly heaver than they are. This remains an unmet clinical need, though not for lack of innovative attempts [64.283, 284].

64.5.3 Home-Based Rehabilitation Monitoring and Therapy Systems

Numerous smart homes have been developed for non-rehabilitation as well as assistive purposes [64.285]. These systems have as their goal the safety of people with disabilities living in the home and communication with caregivers outside of the home. Caregivers can be live-in family or attendants who, even when they are not home, need to be kept informed on the status of the disabled person, as well as clinicians who need to be sent regular vital signs and other medical/therapy reports. In-home systems typically feature the same principal elements:

1. Sensors to monitor both ambient and as people- and object-specific parameters (e.g., person location, stovetop operation); actuators to modify ambient conditions (heat, lighting, sound system, etc.) and operate devices (doors, refrigerator, etc.).
2. A means to network all the sensors and actuators for uni- or bi-directional communication with the host computer. This network can be wireless (e.g., 802.11 g), wired (e.g., coaxial cable), or dependent on an existing network (e.g., signals superimposed on current carried by the electrical mains wiring or phone wiring).
3. A host computer that allows all sensor states to be displayed and actuators to be operated from one or more locations in the home by the inhabitant(s) using common computer I/O devices. Higher order functions are built upon this basic capability.
4. An external network to allow communication with the Internet via phone, cable, satellite or other means. This capability allows for remote monitoring and operation, sending of alarms and discussions with rehabilitation professionals at medical centers.

The host computer software may also have higher order features, for example timers for repetitive actuation of lights and monitoring for anomalous sensor readings (e.g., call security when the smoke detector activates, alert inhabitants with in-home alarm when stove top power is on and no pot is on the stove). More advanced features that involve multiinput and multioutput control and adaptive, predictive, context-aware operation [64.286] are areas of active research, and especially important to the rehabilitation community.

As previously mentioned in this chapter, devices for home-based robot-assisted therapy have been recently developed, and home-based therapy delivered by robotic devices is expected to have a significant growth in the short-medium term.

CBM Motus [64.126] and T-WREX, [64.287, 288], a passive exoskeleton for upper limb rehabilitation now sold by Hocoma AG as ArmeoSpring, are just two examples of systems for home-based therapy described above (Sect. 64.2.3). Another example is the Hand Mentor, which is a device which uses video games and robotics to cognitively involve the patient in his/her rehabilitation [64.289, 290]. The Hand Mentor can be used in the clinic, or taken at home and incorporated into patients' daily therapy sessions to lengthen shortened tissues, facilitate hand opening and closing, and reduce spasticity. It is mainly conceived for stroke patients' rehabilitation and is commercialized by Kinetic Muscles, Inc.

64.5.4 Wearable Monitoring Devices

One component of an automated rehabilitation environment is the subsystem that a person wears to be able to measure, analyze, and communicate physiological signals to an external computer wirelessly. Systems such as the LifeShirt (VivoMetrics, Inc., Ventura) [64.291] have been and are being developed for front-line soldiers and rescue operation personnel whose health may become imperiled when out of touch with and unable to communicate verbally with their base command. For rehabilitative purposes, for example, the Intel Proactive Health Initiative [64.292] is an example of a system that use on-person position and motion sensing to detect potentially dangerous or undesirable situations.

The use of a robot to deliver therapy already enables measuring information on the patient state by exploiting robot sensors and other wearable units, if needed [64.293]. Such capability can be exploited for patient monitoring. A recent work proposed a method for reconstructing the human arm kinematics by resorting to an inverse kinematics technique for redundant robot arms [64.294].

The most significant obstacles to the widespread adoption of these technologies in the short term are cost, false-positive alarms, inconvenience and encumbrance. Advances in micro-electronics, nanotechnologies, software algorithms, and networking capabilities will continue to drive the research and consumer acceptance of this technology sector.

64.6 Safety, Ethics, Access and Economics

Rehabilitation robots interact closely with humans, often sharing the same workspace and sometimes physically attaching to humans, as in the case of robotic movement training devices and prosthetic limbs. Furthermore, the devices are by necessity powerful enough to manipulate the environment or the user's own limbs, which means that they are also often powerful enough to injure the user or another person nearby by colliding with them or moving their limbs inappropriately. Safety is clearly of paramount importance.

A common strategy for ensuring safety is to incorporate multiple, redundant safety features. A device can be designed to be mechanically incapable of moving itself or the user's limbs in such a way as to cause injury. Limits can be placed on the range, strength and speed of actuators so that they can accomplish the desired task but no more. Break-away attachments can be used to attach to users' limbs. Covers can protect the user from pinch points in the device. Redundant sensors can be included, so that if one sensor malfunctions another sensor can identify the malfunction and help safely shut down the machine. Watchdog timers can monitor the health of the control computer. Software checks can limit forces, motions, speeds, and user adjustments to control parameters, as well as check for sensor health and other dangerous situations. Control strategies can be designed so that the device is mechanically compliant, reducing the risk of forcing a limb into an undesirable configuration, or of a high-impact collision. A manual override switch can be incorporated so that the user can shutdown the system. Finally, the user can be instructed on how to safely operate the device and avoid dangerous situations. Safety ultimately depends, however, on careful and rigorous failure mode analysis and remediation by the system designers.

From a system's perspective, when all else fails actively to protect the user, it must be the design itself

that makes the robot inherently unable to injure its user. Part of the solution is in reducing the weight, rounding the surface characteristics, and making appropriate materials choices. The goal of inherent safety, however, is often at odds with high performance and adequate payload for real-life tasks. Recently, several approaches to designing personal robots – in other words – the class to which assistive rehabilitation robots belong – have sought to address both goals by dividing the two tasks of compensating for gravity (arm plus payload) and moving the payload around in space [64.295]. The solution is to provide two actuators per joint on the joints that support the arm segments and payload against gravity: one slow, gear-reduced motor and energy storing device such as a large spring or compressed air volume, and one small, backdrivable motor that provides the power needed to move objects around quickly and precisely. Most robot manipulators have approximately a 1 : 10 (or worse) payload-to-weight ratio. A system with a dual, parallel actuator system that requires only the small, fast actuator to be carried in the arm, leaving the slow energy-storage system on the base and not contributing to the inertia of the arm itself, can lead to a 1 : 1 payload-to-weight ratio more in line with a human's own arm characteristics, and thereby provides a safe yet high-performance solution. An added benefit from this type of arrangement is that the movements of the arm will tend to be more human-like, providing a measure of confidence to the user that the robot is performing properly and moving in a safe way.

Strategies for improving safety have been proposed and methods to assess safety have been developed and adapted for rehabilitation robotics [64.296, 297] based on accepted risk analysis methods. While industrial robots have benefited from ISO user safety regulations since 1992 (ISO 10218), the fundamental issue of human proximity to robots for the personal, service, and rehabilitation sectors has prevented that standard from being applicable to rehabilitation robots. Currently, a draft of a new *safety requirements for personal care robots* standard, ISO 13482, is under development. Currently, the existing industrial standards, augmented with provisions from medical equipment standards and buttressed by engineering best practices and adherence to professional codes of ethics by designers, have guided rehabilitation robotics designers. Clearly, as products appear on the market and the expected rapid expansion of this sector happens, better regulations and standards must be developed.

Beyond safety, there are other ethical concerns that will emerge as robotics technology becomes more intelligent with advances in cognitive software, more invasive with nanotechnologies and better integrated with human systems through bioengineering advances. Ethicists and roboticists are starting to deal with these issues [64.298, 299], which to date have been the purview of only futurists and science fiction writers. Chap. 80 deals with these issues in detail.

An economic advantage has not yet been demonstrated in a decisive way for most rehabilitation robotics. For example, the therapeutic benefits conferred by robotic therapy devices, and the assistive benefits conferred by wheelchair-mounted robots relative to the devices' cost, have not yet been large enough to cause widespread adoption. Improvements in their efficacy and reductions in their cost will increase their usage. For example, a robotic therapy device that helps people learn to walk after a stroke, in a way that is decisively better than other training techniques, would become widely used very quickly. Likewise, a wheelchair-mounted robot that gives a disabled person a substantial and efficient increase in autonomy at a reasonable cost would also quickly become widely used. An example of a robotics technology that has achieved an attractive cost-benefit ratio and thus is commercially successful is the powered wheelchair.

64.7 Conclusions and Further Readings

Rehabilitation robotics is a dynamic application area because its grand challenges are at the forefront of both robotics and biology research. The ongoing major themes of the field can be summarized as the development of robotic therapy devices, smart prostheses, orthoses, functional aids and nurses that match or exceed the capabilities of their human counterparts. Rehabilitation robotics is also a highly motivating field because the technology developed will directly help people who are limited in major life activities. The field will continue to grow because of the dramatic aging of the populations of industrialized countries that is just beginning.

The grand challenges of rehabilitation robotics are grounded in the distinguishing features of the field: functional involvement with humans, a physical user interface and behavior that is intelligent, adaptive, and safe. These characteristics require high levels of redundancy, sensory-motor capability, adaptability, and multilevel software architecture. The grand challenges therefore span the domains of electromechanical design, software design, and, due to the applied and in-

nately human-focused nature of rehabilitation robotics, all aspects of user interface design, including physical, communication, learning, emotional, and motivational factors. The first products in this field have come on the market in only the last 15 years; worldwide demographic trends will provide the force to accelerate product development in the future.

For further investigation on rehabilitation robotics, there are three major sources of published information:

1. Books on personal, service and rehabilitation robotics, such as: [64.300–302]
2. Review articles in journals and periodicals, such as [64.303–307] and
3. Articles that deal with individual topics, such as those in the reference list below, and conferences such as ICORR, RESNA, RO-MAN, BIOROB, and AAATE, which are also represented in the reference list.

Cutting-edge research will be reported on the websites of investigators at academic, government and corporate research labs, and it is recommended to start at the sites of the researchers cited in this chapter.

Video-References

VIDEO 499 The WREX exoskeleton
available from http://handbookofrobotics.org/view-chapter/64/videodetails/499

VIDEO 496 MIT Manus robotic therapy robot and other robots from the MIT group
available from http://handbookofrobotics.org/view-chapter/64/videodetails/496

VIDEO 500 MANUS assistive robot
available from http://handbookofrobotics.org/view-chapter/64/videodetails/500

VIDEO 502 The ArmeoSpring therapy exoskeleton
available from http://handbookofrobotics.org/view-chapter/64/videodetails/502

VIDEO 503 Lokomat
available from http://handbookofrobotics.org/view-chapter/64/videodetails/503

VIDEO 504 Gait Trainer GT1
available from http://handbookofrobotics.org/view-chapter/64/videodetails/504

VIDEO 505 Kineassist
available from http://handbookofrobotics.org/view-chapter/64/videodetails/505

VIDEO 507 Ekso
available from http://handbookofrobotics.org/view-chapter/64/videodetails/507

VIDEO 508 ReWalk
available from http://handbookofrobotics.org/view-chapter/64/videodetails/508

VIDEO 509 HAL
available from http://handbookofrobotics.org/view-chapter/64/videodetails/509

VIDEO 510 Indego
available from http://handbookofrobotics.org/view-chapter/64/videodetails/510

VIDEO 511 REX
available from http://handbookofrobotics.org/view-chapter/64/videodetails/511

VIDEO 513 Targetted reinnervation and the DEKA arm
available from http://handbookofrobotics.org/view-chapter/64/videodetails/513

VIDEO 515 PAM
available from http://handbookofrobotics.org/view-chapter/64/videodetails/515

VIDEO 494 The Arm Guide
available from http://handbookofrobotics.org/view-chapter/64/videodetails/494

VIDEO 497 ARMin plus HandSOME robotic therapy system
available from http://handbookofrobotics.org/view-chapter/64/videodetails/497

VIDEO 498 BONES and SUE exoskeletons for robotic therapy
available from http://handbookofrobotics.org/view-chapter/64/videodetails/498

VIDEO 495 The MIME rehabilitation therapy robot
available from http://handbookofrobotics.org/view-chapter/64/videodetails/495

VIDEO 568 Handsome exoskeleton
available from http://handbookofrobotics.org/view-chapter/64/videodetails/568

References

64.1 L. Leifer: Rehabilitative robots. In: *Robotics Age, in the Beginning: Selected from Robotics Age Magazine*, ed. by C. Helmers (Hayden Book, Rochelle Park 1983) pp. 227–241

64.2 M. Kassler: Introduction to the special issue on robotics for health care, Robotica **11**, 493–494 (1993)

64.3 H.F.M. Van der Loos: VA/Stanford rehabilitation robotics research and development program: Lessons learned in the application of robotics technology to the field of rehabilitation, IEEE Trans. Rehabil. Eng. **3**, 46–55 (1995)

64.4 J.L. Patton, M.E. Phillips-Stoykov, M. Stojakovich, F.A. Mussa-Ivaldi: Evaluation of robotic training forces that either enhance or reduce error in chronic hemiparetic stroke survivors, Exp. Brain Res. **168**, 368–383 (2006)

64.5 J. Emken, D. Reinkensmeyer: Robot-enhanced motor learning: Accelerating internal model formation during locomotion by transient dynamic amplification, IEEE Trans. Neural Syst. Rehabil. Eng. **99**, 1–7 (2005)

64.6 E. Guglielmelli, M. Johnson, T. Shibata: Guest editorial special issue on rehabilitation robotics, IEEE Trans. Robotics **25**, 477–480 (2009)

64.7 M.J. Matarić, J. Eriksson, D.J. Feil-Seifer, C.J. Winstein: Socially assistive robotics for post-stroke rehabilitation, J. Neuroeng. Rehabil. **19**, 5 (2007)

64.8 M.P. Dijkers, P.C. deBear, R.F. Erlandson, K.A. Kristy, D.M. Geer, A. Nichols: Patient and staff acceptance of robotic technology in occupational therapy: A pilot study, J. Rehabil. Res. **28**, 33–44 (1991)

64.9 R. Jimínez-Fabián, O. Verlinden: Review of control algorithms for robotic ankle systems in lower-limb orthoses, prostheses, and exoskeletons, Med. Eng. Phys. **34**, 397–408 (2012)

64.10 D.G. Smith, J.D. Bigelow: Biomedicine: Revolutionizing prosthetics-guest editors' introduction, Johns Hopkins APL Tech. Dig. **30**, 182–185 (2011)

64.11 G. Baldassarre: *La disabilità in Italia. Il quadro della statistica ufficiale* (Istat, Roma 2010)

64.12 Demo-Geodemo: *Mappe, Popolazione, Statistiche Demografiche dell'ISTAT*, available online at http://demo.istat.it

64.13 Destatis (Statistisches Bundesamt), https://www.destatis.de/EN/

64.14 X. Zheng, G. Chen, X. Song, J. Liu, L. Yan, W. Du, L. Pang, L. Zhang, J. Wu, B. Zhang, J. Zhang: Twenty-year trends in the prevalence of disability in China, Bull. World Health Organ. **89**, 788–797 (2011)

64.15 National-Bureau-of-Statistics-of-China: http://www.stats.gov.cn/english/

64.16 Central-Statistics-Office-Ministry-of-Statistics-&-Programme-Implementation: *Disability in India – A statistical profile 2011* (2011)

64.17 CIA: *The World Factbook*, available online at https://www.cia.gov/library/publications/the-world-factbook/geos/in.html

64.18 H.F.M. Van der Loos, R. Mahoney, C. Ammi: Great expectations for rehabilitation mechatronics in the coming decade. In: *Advances in Rehabilitation Robotics*, ed. by Z. Bien, D. Stefanov (Springer, Berlin, Heidelberg 2004) pp. 427–433

64.19 K. Corker, J.H. Lyman, S. Sheredos: A preliminary evaluation of remote medical manipulators, Bull. Prosthet. Res. **10**, 107–134 (1979)

64.20 M. Hillman: Rehabilitation robotics from past to present – A historical perspective. In: *Advances in Rehabilitation Robotics*, ed. by Z. Bien, D. Stefanov (Springer, Berlin, Heidelberg 2004) pp. 25–44

64.21 W. Seamone, G. Schmeisser: Early clinical evaluation of a robot arm/work table system for spinal cord injured persons, J. Rehabil. Res. Dev. **22**, 38–57 (1985)

64.22 J. Guittet, H.H. Kwee, N. Quetin, J. Yelon: The SPARTACUS telethesis: Manipulator control studies, Bull. Prosthet. Res. **10**, 69–105 (1979)

64.23 H.H. Kwee: Integrated control of MANUS manipulator and wheelchair enhanced by environmental docking, Robotica **16**, 491–498 (2000)

64.24 J. Hammel, K. Hall, D.S. Lees, L.J. Leifer, H.F.M. Van der Loos, I. Perkash, R. Crigler: Clinical evaluation of a desktop robotic assistant, J. Rehabil. Res. Dev. **26**, 1–16 (1989)

64.25 J.M. Hammel, H.F.M. Van der Loos, I. Perkash: Evaluation of a vocational robot with a quadriplegic employee, Arch. Phys. Med. Rehabil. **73**, 683–693 (1992)

64.26 H.F.M. Van der Loos, S.J. Michalowski, L.J. Leifer: Design of an omnidirectional mobile robot as a manipulation aid for the severely disabled. In: *Interactive Robotic Aids – One Option for Independent Living: An International Perspective*, ed. by R. Foulds (World Rehabilitation Fund, New York 1986) pp. 61–63

64.27 J.J. Wagner, M. Wickizer, H.F.M. Van der Loos, L.J. Leifer: User testing and design iteration of the ProVAR user interface, 8th IEEE Int. Workshop Robot Hum. Interact. (RO-MAN) (1999) pp. 18–22

64.28 M. Busnel, R. Cammoun, F. Coulon-Lauture, J.-M. Detriche, G. Le Claire, B. Lesigne: The robotized workstation *MASTER* for users with tetraplegia: Description and evaluation, J. Rehabil. Res. Dev. **36**, 217–229 (1999)

64.29 R. Gelin, B. Lesigne, M. Busnel, J.P. Michel: The first moves of the AFMASTER workstation, Adv. Robotics **14**, 639–649 (2001)

64.30 M. Topping: The development of Handy 1, a rehabilitation robotic system to assist the severely disabled, Ind. Robot **25**, 316–320 (1998)

64.31 R.M. Mahoney: The raptor wheelchair robot system. In: *Integration of Assistive Technology in the Information Age*, ed. by M. Mokhtari (IOS, Amsterdam 2001) pp. 135–141

64.32 V. Maheu, J. Frappier, P.S. Archambault, F. Routhier: Evaluation of the JACO robotic arm: Clinico-economic study for powered wheelchair

users with upper-extremity disabilities, IEEE Int. Conf. Rehabil. Robotics (2011) p. 5975397
64.33 Z. Bien, M.J. Chung, P.H. Chang, D.S. Kwon, D.J. Kim, J.S. Han, J.-H. Kim, D.-H. Kim, H.-S. Park, S.-H. Kang, K. Lee, S.-C. Lim: Integration of a rehabilitation robotic system (KARES II) with human-friendly man-machine interaction units, Auton. Robots **16**, 165–191 (2004)
64.34 R.C. Simpson, S.P. Levine, D.A. Bell, L. Jaros, Y. Koren, J. Borenstein: NavChair: An assistive wheelchair navigation system with automatic adaptation, assistive technology and AI, Lect. Notes Artif. Intell. **1458**, 235–255 (1998)
64.35 R.C. Simpson, D. Poirot, F. Baxter: The Hephaestus smart wheelchair system, IEEE Trans. Neural Syst. Rehabil. Eng. **10**, 118–122 (2002)
64.36 R. Krukowski: Particle brake clutch muscle exercise and rehabilitation apparatus, US Patent Ser 476 5315 A (1986)
64.37 D. Khalili, M. Zomlefer: An intelligent robotic system for rehabilitation of joints and estimation of body segment parameters, IEEE Trans. Biomed. Eng. **35**, 138–146 (1988)
64.38 P.S. Lum, S.L. Lehman, D.J. Reinkensmeyer: The bimanual lifting rehabilitator: A device for rehabilitating bimanual control in stroke patients, IEEE Trans. Rehabil. Eng. **3**, 166–174 (1995)
64.39 P.S. Lum, D.J. Reinkensmeyer, S.L. Lehman: Robotic assist devices for bimanual physical therapy: Preliminary experiments, IEEE Trans. Rehabil. Eng. **1**, 185–191 (1993)
64.40 P.S. Lum, H.F.M. Van der Loos, P. Shor, C.G. Burgar: A robotic system for upper-limb exercises to promote recovery of motor function following stroke, Proc. 6th Int. Conf. Rehabil. Robotics (1999) pp. 235–239
64.41 M.J. Johnson, H.F. Van der Loos, C.G. Burgar, P. Shor, L.J. Leifer: Experimental results using force-feedback cueing in robot-assisted stroke therapy, IEEE Trans. Neural Syst. Rehabil. Eng. **13**, 335–348 (2005)
64.42 D.J. Reinkensmeyer, J.P.A. Dewald, W.Z. Rymer: Guidance-based quantification of arm impairment following brain injury: A pilot study, IEEE Trans. Rehabil. Eng. **7**, 1–11 (1999)
64.43 P. Hilaire, A.C. Jacob, T. Böni, B. Rüttimann: Richard Scherb: Orthopaedic surgeon and muscle physiologist, 28th Int. Soc. Biomech. Congr. (2001)
64.44 J.P.L. Bel, P. Rabischong: Orthopaedic appliances, US Patent Ser 399 3056 A (1976)
64.45 G. Colombo, M. Joerg, R. Schreier, V. Dietz: Treadmill training of paraplegic patients with a robotic orthosis, J. Rehabil. Res. Dev. **37**, 693–700 (2000)
64.46 S. Hesse, D. Uhlenbrock: A mechanized gait trainer for restoration of gait, J. Rehabil. Res. Dev. **37**, 701–708 (2000)
64.47 Cybernetic Zoo: Stock Photo ID 42-17253903, available online at http://cyberneticzoo.com/bionics/1976-pneumatic-exoskeleton-prosthesis-pierre-rabischong-french/
64.48 LokoHelp: http://www.darkov.com/treatment/curative-procedures/lokostation-with-lokohelp.aspx
64.49 GEO System: http://www.skoutasmedical.gr/portal/index.php?page=shop.product_details&product_id=742&flypage=ilvm_fly2_blue.tpl&pop=0&option=com_virtuemart&Itemid=120&lang=en&vmcchk=1&Itemid=120
64.50 A.C. Lo, P.D. Guarino, L.G. Richards, J.K. Haselkorn, G.F. Wittenberg, D.G. Federman, R.J. Ringer, T.H. Wagner, H.I. Krebs, B.T. Volpe, C.T. Bever, D.M. Bratava, P.W. Duncan, B.H. Corn, A.D. Maffucci, S.E. Nadeau, S.S. Conroy, J.M. Powell, G.D. Huang, P. Peduzzi: Robot-assisted therapy for long-term upper-limb impairment after stroke, N. Engl. J. Med. **362**, 1772–1783 (2010)
64.51 G. Kwakkel, B.J. Kollen, H.I. Krebs: Effects of robot-assisted therapy on upper limb recovery after stroke: A systematic review, Neurorehabil. Neural Repair **22**, 111–121 (2008)
64.52 E.C. Lu, R.H. Wang, D. Hebert, J. Boger, M.P. Galea, A. Mihailidis: The development of an upper limb stroke rehabilitation robot: Identification of clinical practices and design requirements through a survey of therapists, Disabil. Rehabil. Assist. Technol. **6**, 420–431 (2011)
64.53 B.T. Volpe, P.T. Huerta, J.L. Zipse, A. Rykman, D. Edwards, L. Dipietro, N. Hogan, H.I. Krebs: Robotic devices as therapeutic and diagnostic tools for stroke recovery, Arch. Neurol. **66**, 1086–1090 (2009)
64.54 L. Zollo, L. Rossini, M. Bravi, G. Magrone, S. Sterzi, E. Guglielmelli: Quantitative evaluation of upper-limb motor control in robot-aided rehabilitation, Med. Biol. Eng. Comput. **49**, 1131–1144 (2011)
64.55 B. Rohrer, S. Fasoli, H.I. Krebs, R. Hughes, B. Volpe, W.R. Frontera, J. Stein, N. Hogan: Movement smoothness changes during stroke recovery, J. Neurosci. **22**, 8297–8304 (2002)
64.56 E. Papaleo, L. Zollo, L. Spedaliere, E. Guglielmelli: Patient-tailored adaptive robotic system for upper-limb rehabilitation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013)
64.57 G. Pellegrino, L. Tomasevic, M. Tombini, G. Assenza, M. Bravi, S. Sterzi, V. Giacobbe, L. Zollo, E. Guglielmelli, G. Cavallo, F. Vernieri, F. Tecchio: Inter-hemispheric coupling changes associate with motor improvements after robotic stroke rehabilitation, Restor. Neurol. Neurosci. **30**, 497–510 (2012)
64.58 R. Riener, M. Munih: Special section on rehabilitation via bio-cooperative control, IEEE Trans. Neural Syst. Rehabil. Eng. **18**, 337–338 (2010)
64.59 R.C. Loureiro, W.S. Harwin, K. Nagai, M. Johnson: Advances in upper limb stroke rehabilitation: A technology push, Med. Biol. Eng. Comput. **49**, 1103–1118 (2011)
64.60 F.J. Badesa, R. Morales-Vidal, N. Garcia, C. Perez, J.M. Sabater-Navarro, E. Papaleo: New concept of multimodal assistive robotic device, IEEE Robotics Autom. Mag. (IEEE/RAM) (2012)

64.61 L. Zollo, E. Papaleo, L. Spedaliere, E. Guglielmelli, F.J. Badesa, R. Morales, N. Garcia-Aracil: Multimodal interfaces to improve therapeutic outcomes in robot-assisted rehabilitation, Springer Tract. Adv. Robotics **94**, 321–343 (2014)

64.62 Intelligent System Co., Ltd., Japan: http://www.parorobots.com/

64.63 K. Dautenhahn, I. Werry: Towards interaction robots in autism therapy: Background, motivation, and challenges, Pragmact. Cogn. **12**, 1–35 (2004)

64.64 T. Shibata, T. Mitsui, K. Wada, K. Tanie: Psychophysiological effects by interaction with mental commit robot, J. Robotics Mechatron. **14**, 13–19 (2002)

64.65 L. Sawaki: Use-dependent plasticity of the human motor cortex in health and disease, IEEE Eng. Med. Biol. Mag. **24**, 36–39 (2005)

64.66 J.R. Wolpaw, A.M. Tennissen: Activity-dependent spinal cord plasticity in health and disease, Annu. Rev. Neurosci. **24**, 807–843 (2001)

64.67 K.M. Baldwin, F. Haddad: Skeletal muscle plasticity: Cellular and molecular responses to altered physical activity paradigms, Am. J. Phys. Med. Rehabil. **81**, S40–S51 (2002)

64.68 V.S. Huang, J.W. Krakauer: Robotic neurorehabilitation: A computational motor learning perspective, J. Neuroeng. Rehabil. **6**(1), 5 (2009)

64.69 J. Mehrholz, M. Pohl: Electromechanical-assisted gait training after stroke: A systematic review comparing end-effector and exoskeleton devices, J. Rehabil. Med. **44**(3), 193–199 (2012)

64.70 V. Klamroth-Marganska, J. Blanco, K. Campen, A. Curt, V. Dietz, T. Ettlin, M. Felder, B. Fellinghauer, M. Guidali, A. Kollmar, A. Luft, T. Nef, C. Schuster-Amft, W. Stahel, R. Riener: Three-dimensional, task-specific robot therapy of the arm after stroke: A multicentre, parallel-group randomised trial, Lancet Neurol. **13**(2), 159–166 (2014)

64.71 P. Langhorne, F. Coupar, A. Pollock: Motor recovery after stroke: A systematic review, Lancet Neurol. **8**, 741–754 (2009)

64.72 J.L. Emken, R. Benitez, D.J. Reinkensmeyer: Human-robot cooperative movement training: Learning a novel sensory motor transformation during walking with robotic assistance-as-needed, J. Neuroeng. Rehabil. **4**, 8 (2007)

64.73 M. Casadio, V. Sanguineti: Learning, retention, and slacking: A model of the dynamics of recovery in robot therapy, IEEE Trans. Neural Syst. Rehabil. Eng. **20**, 286–296 (2012)

64.74 E.R. Pospisil, B.L. Luu, J.S. Blouin, H.F. Van der Loos, E.A. Croft: Independent ankle motion control improves robotic balance simulator, IEEE Annu. Int. Conf. Eng. Med. Biol. Soc. (2012) pp. 6487–6491

64.75 B. Luu, T. Huryn, E. Croft, H. Van der Loos, J. Blouin: Investigating load stiffness in quiet stance using a robotic balance system, IEEE Trans. Neural Syst. Rehabil. Eng. **19**(4), 382–390 (2011)

64.76 M. Guadagnoli, T. Lee: Challenge point: A framework for conceptualizing the effects of various practice conditions in motor learning, J. Mot. Beh. **36**, 212–224 (2004)

64.77 C. Bosecker, L. Dipietro, B. Volpe, H.I. Krebs: Kinematic robot-based evaluation scales and clinical counterparts to measure upper limb motor performance in patients with chronic stroke, Neurorehabil. Neural Repair **24**(1), 62–69 (2009)

64.78 L. Zollo, O.E. Gallotta, E. Guglielmelli, S. Sterzi: Robotic technologies and rehabilitation: New tools for upper-limb therapy and assessment in chronic stroke, Eur. J. Phys. Rehabil. Med. **47.2**, 223–236 (2011)

64.79 H.I. Krebs, M. Krams, D. Agrafiotis, A. DiBernardo, J.C. Chavez, G.S. Littman, E. Yang, G. Byttebier, L. Dipietro, A. Rykman, K. McArthur, K. Hajjar, K.R. Lees, B.T. Volpe: Robotic measurement of arm movements after stroke establishes biomarkers of motorecovery, Stroke **45**(1), 200–204 (2014)

64.80 A. Domingo, T. Lam: Reliability and validity of using the Lokomat to assess lower limb joint position sense in people with incomplete spinal cord injury, J. Neuroeng. Rehabil. **11**(1), 167 (2014)

64.81 C. Carignan, H. Krebs: Telerehabilitation robotics: Bright lights, big future?, J. Rehabil. Res. Dev. **43**, 695–710 (2006)

64.82 B.H. Dobkin: *Neurologic Rehabilitation* (F.A. Davis, Philadelphia 1996)

64.83 T. Truelsen, B. Piechowski-Jóźwiak, R. Bonita, C. Mathers, J. Bogousslavsky, G. Boysen: Stroke incidence and prevalence in Europe: A review of available data, Eur. J. Neurol. **13**, 581–598 (2006)

64.84 ESCIF (European Spinal Cord Injury Federation): *Statement on Spinal Cord Injury Regenerative Research* (ESCIF, Nottwill 2011)

64.85 M. Wirz, D.H. Zemon, R. Rupp, A. Scheel, G. Colombo, V. Dietz, T.G. Hornby: Effectiveness of automated locomotor training in patients with chronic incomplete spinal cord injury: A multicenter trial, Arch. Phys. Med. Rehabil. **86**, 672–680 (2005)

64.86 H.I. Krebs, N. Hogan, M.L. Aisen, B.T. Volpe: Robot-aided neurorehabilitation, IEEE Trans. Rehabil. Eng. **6**, 75–87 (1998)

64.87 S.P. Buerger, H.I. Krebs, N. Hogan: Characterization and control of a screw-driven robot for neurorehabilitation, Proc. IEEE Int. Conf. Control Appl. (2001) pp. 388–394

64.88 D.J. Williams, H.I. Krebs, N. Hogan: A robot for wrist rehabilitation, IEEE 23rd Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (2001) pp. 1336–1339

64.89 L. Masia, H.I. Krebs, P. Cappa, N. Hogan: Whole-arm rehabilitation following stroke: Hand module, IEEE/RAS-EMBS 1st Int. Conf. Biomed. Robotics Biomechatron. (BioRob) (2006) pp. 1085–1089

64.90 J. Stein, H.I. Krebs, W.R. Frontera, S. Fasoli, R. Hughes, N. Hogan: Comparison of two techniques of robot-aided upper limb exercise training after stroke, Am. J. Phys. Med. Rehabil. **83**, 720–728 (2004)

64.91 H. Krebs, J. Palazzolo, L. Dipietro, M. Ferraro, J. Krol, K. Rannekleiv, B.T. Volpe, N. Hogan: Rehabilitation robotics: Performance-based progres-

sive robot-assisted therapy, Auton. Robots **15**, 7–20 (2003)
64.92 M.L. Aisen, H.I. Krebs, N. Hogan, F. McDowell, B. Volpe: The effect of robot-assisted therapy and rehabilitative training on motor recovery following stroke, Arch. Neurol. **54**, 443–446 (1997)
64.93 S. Fasoli, H. Krebs, J. Stein, W. Frontera, N. Hogan: Effects of robotic therapy on motor impairment and recovery in chronic stroke, Arch. Phys. Med. Rehabil. **84**, 477–482 (2003)
64.94 J.J. Daly, N. Hogan, E.M. Perepezko, H.I. Krebs, J.M. Rogers, K.S. Goyal, M.E. Dohring, E. Fredrickson, J. Nethery, R.L. Ruff: Response to upper-limb robotics and functional neuromuscular stimulation following stroke, J. Rehabil. Res. Dev. **42**, 723–736 (2005)
64.95 T.H. Wagner, A.C. Lo, P. Peduzzi, D.M. Bravata, G.D. Huang, H.I. Krebs, R.J. Ringer, D.G. Federman, L.G. Richards, J.K. Haselkorn, G.F. Wittenberg, B.T. Volpe, C.T. Bever, P.W. Duncan, A. Siroka, P.D. Guarino: An economic analysis of robot-assisted therapy for long-term upper-limb impairment after stroke, Stroke **41**, 2630–2632 (2011)
64.96 P.S. Lum, C.G. Burgar, P.C. Shor, M. Majmundar, H.F.M. Van der Loos: Robot-assisted movement training compared with conventional therapy techniques for the rehabilitation of upper limb motor function following stroke, Arch. Phys. Med. Rehabil. **83**, 952–959 (2002)
64.97 P.S. Lum, C.G. Burgar, H.F.M. Van der Loos, P.C. Shor, M. Majmundar, R. Yap: MIME robotic device for upper-limb neurorehabilitation in subacute stroke subjects: A follow-up study, J. Rehabil. Res. Dev. **43**, 631–642 (2006)
64.98 C.G. Burgar, P.S. Lum, A.M. Scremin, S.L. Garber, H.F. Van der Loos, D. Kenney, F. Shor: Robot-assisted upper-limb therapy in acute rehabilitation setting following stroke: Department of veterans affairs multisite clinical trial, J. Rehabil. Res. Dev. **48**, 445–458 (2011)
64.99 L.E. Kahn, M.L. Zygman, W.Z. Rymer, D.J. Reinkensmeyer: Robot-assisted reaching exercise promotes arm movement recovery in chronic hemiparetic stroke: A randomized controlled pilot study, J. Neuroeng. Neurorehabil. **3**, 12 (2006)
64.100 S. Hesse, C. Werner, M. Pohl, S. Rueckriem, J. Mehrholz, M.L. Lingnau: Computerized arm training improves the motor control of the severely affected arm after stroke: A single-blinded randomized trial in two centers, Stroke **36**, 1960–1966 (2005)
64.101 D. Reinkensmeyer, J. Emken, S. Cramer: Robotics, motor learning, and neurologic recovery, Annu. Rev. Biomed. Eng. **6**, 497–525 (2004)
64.102 F. Amirabdollahian, E. Gradwell, R. Loureiro, W. Harwin: Effects of the GENTLE/S robot mediated therapy on the outcome of upper limb rehabilitation post-stroke: Analysis of the Battle Hospital data, Proc. 8th Int. Conf. Rehabil. Robotics (2003) pp. 55–58
64.103 F. Amirabdollahian, R. Loureiro, E. Gradwell, C. Collin, W. Harwin, G. Johnson: Multivariate analysis of the Fugl-Meyer outcome measures assessing the effectiveness of GENTLE/S robot-mediated stroke therapy, J. Neuroeng. Rehabil. **19**, 4 (2007)
64.104 A.S. Merians, H. Poizner, R. Boian, G. Burdea, S. Adamovich: Sensorimotor training in a virtual reality environment: Does it improve functional recovery poststroke? The Rutgers arm, a rehabilitation system in virtual reality: A pilot study, Neurorehabil. Neural Repair. **20**, 252–267 (2006)
64.105 R. Colombo, F. Pisano, S. Micera, A. Mazzone, C. Delconte, M. Carrozza, P. Dario, G. Minuco: Robotic techniques for upper limb evaluation and rehabilitation of stroke patients, IEEE Trans. Neural Syst. Rehabil. Eng. **13**, 311–324 (2005)
64.106 R.J. Sanchez, J. Liu, S. Rao, P. Shah, R. Smith, S.C. Cramer, J.E. Bobrow, D.J. Reinkensmeyer: Automating arm movement training following severe stroke: Functional exercises with quantitative feedback in a gravity-reduced environment, IEEE Trans. Neural Syst. Rehabil. Eng. **14**, 378–389 (2006)
64.107 S.J. Housman, K.M. Scott, D.J. Reinkensmeyer: A randomized controlled trial of gravity-supported, computer-enhanced arm exercise for individuals with severe hemiparesis, Neurorehabil. Neural Repair **23**, 505–514 (2009)
64.108 D. Gijbels, I. Lamers, L. Kerkhofs, G. Alders, E. Knippenberg, P. Feys: The armeo spring as training tool to improve upper limb functionality in multiple sclerosis: A pilot study, J. Neuroeng. Rehabil. **8**, 5 (2011)
64.109 J. Zariffa, N. Kapadia, J.L. Kramer, P. Taylor, M. Alizadeh-Meghrazi, V. Zivanovic, R. Williams, A. Townson, A. Curt, M.R. Popovic, J.D. Steeves: Feasibility and efficacy of upper limb robotic rehabilitation in a subacute cervical spinal cord injury population, Spinal Cord **50**, 220–226 (2012)
64.110 L. Schwickert, J. Klenk, A. Stähler, C. Becker, U. Lindemann: Robotic-assisted rehabilitation of proximal humerus fractures in virtual environments: A pilot study, Z. Gerontol. Geriatr. **44**, 387–392 (2011)
64.111 S. Masiero, A. Celia, G. Rosati, M. Armani: Robotic-assisted rehabilitation of the the upper limb after acute stroke, Arch. Phys. Med. Rehabil. **88**, 142–149 (2007)
64.112 G. Fazekas, M. Horvath, A. Toth: A novel robot training system designed to supplement upper limb physiotherapy of patients with spastic hemiparesis, Int. J. Rehabil. Res. **29**, 251–254 (2006)
64.113 T. Nef, R. Riener: ARMin – Design of a novel arm rehabilitation robot, Proc. IEEE Int. Conf. Rehabil. Robotics (2005) pp. 57–60
64.114 E. Wolbrecht, J. Leavitt, D. Reinkensmeyer, J. Bobrow: Control of a pneumatic orthosis for upper extremity stroke rehabilitation, IEEE Eng. Med. Biol. Conf. (2006) pp. 2687–2693

64.115 J. Klein, S.J. Spencer, J. Allington, J.E. Bobrow, D.J. Reinkensmeyer: Optimization of a parallel shoulder mechanism to achieve a high force, low mass, robotic arm exoskeleton, IEEE Trans. Robotics **26**, 710–715 (2010)

64.116 J.L. Patton, G. Dawe, C. Scharver, F.A. Mussa-Ivaldi, R. Kenyon: Robotics and virtual reality: The development of a life-sized 3-D system for the rehabilitation of motor function, IEEE 26th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (2004) pp. 4840–4843

64.117 H. Huang, J. He: Utilization of biomechanical modeling in design of robotic arm for rehabilitation of stroke patients, IEEE 26th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc., Vol. 4 (2004) pp. 2718–2721

64.118 J.C. Perry, J. Rosen, S. Burns: Upper-limb powered exoskeleton design, IEEE/ASME Trans. Mechatron. **12**, 408–417 (2007)

64.119 R. Gopura, K. Kiguchi, Y. Li: SUEFUL-7: A 7DOF upper-limb exoskeleton robot with muscle-model-oriented EMG-based control, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 1126–1131

64.120 N. Jarrasse, M. Tagliabue, J.V.G. Robertson, A. Maiza, V. Crocher, A. Roby-Brami, G. Morel: A methodology to quantify alterations in human upper limb movement during co-manipulation with an exoskeleton, IEEE Trans. Neural Syst. Rehabil. Eng. **18**, 389–397 (2010)

64.121 B.-C. Tsai, W.-W. Wang, L.-C. Hsu, L.-C. Fu, J.-S. Lai: An articulated rehabilitation robot for upper limb physiotherapy and training, 2010 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 1470–1475

64.122 M.J. Johnson, H.F.M. Van der Loos, C.G. Burgar, P. Shor, L.J. Leifer: Experimental results using force-feedback cueing in robot-assisted stroke therapy, IEEE Trans. Neural Syst. Rehabil. Eng. **13**, 335–348 (2005)

64.123 D. Reinkensmeyer, C. Pang, J. Nessler, C. Painter: Web-based telerehabilitation for the upper-extremity after stroke, IEEE Trans. Neural Sci. Rehabil. Eng. **10**, 1–7 (2002)

64.124 X. Feng, J. Winters: UniTherapy: A computer-assisted motivating neurorehabilitation platform for teleassessment and remote therapy, IEEE Int. Conf. Rehabil. Robotics (2005) pp. 349–352

64.125 H. Sugarman, E. Dayan, A. Weisel-Eichler, J. Tiran: The jerusalem telerehabilitation system, a new low-cost, haptic rehabilitation approach, Cyberpsychol. Behav. **9**, 178–182 (2006)

64.126 L. Zollo, D. Accoto, F. Torchiani, D. Formica, E. Guglielmelli: Design of a planar robotic machine for neuro-rehabilitation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 2031–2036

64.127 L. Zollo, A. Salerno, M. Vespignani, D. Accoto, M. Passalacqua, E. Guglielmelli: Dynamic characterization and interaction control of the CBM-Motus robot for upper-limb rehabilitation, Int. J. Adv. Robotic Syst. **10**, 374 (2013)

64.128 O. Lambercy, L. Dovat, H. Yun, S.K. Wee, C. Kuah, K. Chua, R. Gassert, T. Milner, L.-T. Chee, E. Rurdet: Rehabilitation of grasping and forearm pronation/supination with the Haptic Knob, IEEE Int. Conf. Rehabil. Robotics (2009) pp. 22–27

64.129 L. Dovat, O. Lambercy, R. Gassert, T. Maeder, T. Milner, T.C. Leong, E. Burdet: HandCARE: A cable-actuated rehabilitation system to train hand function after stroke, IEEE Trans. Neural Syst. Rehabil. Eng. **16**, 582–591 (2008)

64.130 S. Hesse, H. Kuhlmann, J. Wilk, C. Tomelleri, S.G.B. Kirker: A new electromechanical trainer for sensorimotor rehabilitation of paralysed fingers: A case series in chronic and acute stroke patients, J. NeuroEng. Rehabil. **5**, 21 (2008)

64.131 C. Takahashi, L. Der-Yeghiaian, V. Le, S. Cramer: A robotic device for hand motor therapy after stroke, IEEE Int. Conf. Rehabil. Robotics (2005) pp. 17–20

64.132 C.N. Schabowsky, S.B. Godfrey, R.J. Holley, P.S. Lum: Development and pilot testing of HEXORR: Hand EXOskeleton rehabilitation robot, J. Neuroeng. Rehabil. **7**, 1–16 (2010)

64.133 S. Ito, H. Kawasaki, Y. Ishigure, M. Natsume, T. Mouri, Y. Nishimoto: A design of fine motion assist equipment for disabled hand in robotic rehabilitation system, J. Frankl. Inst. **348**, 79–89 (2011)

64.134 M. Mulas, M. Folgheraiter, G. Gini: An EMG-controlled exoskeleton for hand rehabilitation, Int. Conf. Rehabil. Robotics (2005) pp. 371–374

64.135 T. Kline, D. Kamper, B. Schmit: Control system for pneumatically controlled glove to assist in grasp activities, Int. Conf. Rehabil. Robotics (2005) pp. 78–81

64.136 B. Birch, E. Haslam, I. Heerah, N. Dechev, E. Park: Design of a continuous passive and active motion device for hand rehabilitation, IEEE 30th Annu. Int. Conf. Eng. Med. Biol. Soc. (2008) pp. 4306–4309

64.137 H. Yamaura, K. Matsushita, R. Kato, H. Yokoi: Development of hand rehabilitation system using wire-driven link mechanism for paralysis patients, IEEE Int. Conf. Robotics Biomim. (ROBIO) (2009) pp. 209–214

64.138 H. Taheri, J.B. Rowe, D. Gardner, V. Chan, D.J. Reinkensmeyer, E.T. Wolbrecht: Robot-assisted guitar hero for finger rehabilitation after stroke, IEEE Annu. Int. Conf. Eng. Med. Biol. Soc. (2012) pp. 3911–3917

64.139 S. Balasubramanian, J. Klein, E. Burdet: Robot-assisted rehabilitation of hand function, Curr. Opin. Neurol. **23**, 661–670 (2010)

64.140 B.R. Brewer, M. Fagan, R.L. Klatzky, Y. Matsuoka: Perceptual limits for a robotic rehabilitation environment using visual feedback distortion, IEEE Trans. Neural Syst. Rehabil. Eng. **13**, 1–11 (2005)

64.141 O. Lambercy, L. Dovat, H. Yun, S.K. Wee, C.W.K. Kuah, K.S.G. Chua, Z. Gassert, T.E. Milner, L.T. Chee, E. Burdet: Effects of a robot-assisted training of grasp and pronation/supination in chronic stroke: A pilot study, J. Neuroeng. Rehabil. **8**(1), 63 (2011)

64.142 S.P. Mazzoleni, M. Franceschini, S. Bigazzi, M.C. Carrozza, P. Dario, F. Posteraro: Effects of proximal and distal robot-assisted upper limb rehabilitation on chronic stroke recovery, NeuroRehabilitation **33**(1), 33–39 (2013)

64.143 T. Boonstra, H. Clairbois, A. Daffertshofer, J. Verbunt, B. van Dijk, P. Beek: MEG-compatible force sensor, J. Neurosci. Methods **144**, 193–196 (2005)

64.144 R. Gassert, R. Moser, E. Burdet, H. Bleuler: MRI/fMRI-compatible robotic system with force feedback for interaction with human motion, IEEE/ASME Trans. Mechatron. **11**, 216–224 (2006)

64.145 Z.J. Tang, S. Sugano, H. Iwata: Design of an MRI compatible robot for finger rehabilitation, Int. Conf. Mechatron. Autom. (ICMA) (2012) pp. 611–616

64.146 O. Unluhisarcikli, B. Weinberg, M. Sivak, A. Mirelman, P. Bonato, C. Mavroidis: A robotic hand rehabilitation system with interactive gaming using novel electro-rheological fluid based actuators, IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 1846–1851

64.147 A. Khanicheh, D. Mintzopoulos, B. Weinberg, A.A. Tzika, C. Mavroidis: MR_CHIROD v. 2: Magnetic resonance compatible smart hand rehabilitation device for brain imaging, IEEE Trans. Neural Syst. Rehabil. Eng. **16**, 91–98 (2008)

64.148 J. Izawa, T. Shimizu, T. Aodai, T. Kondo, H. Gomi, S. Toyama, K. Ito: MR compatible manipulandum with ultrasonic motor for fMRI studies, Proc. IEEE Int. Conf. Robotics Autom (ICRA) (2006) pp. 3850–3854

64.149 F. Sergi, H.I. Krebs, B. Groissier, A. Rykman, E. Guglielmelli, B.T. Volpe, J.D. Schaechter: Predicting efficacy of robot-aided rehabilitation in chronic stroke patients using an MRI-compatible robotic device, IEEE Annu. Int. Conf. Eng. Med. Biol. Soc. (2011) pp. 7470–7473

64.150 E. Buch, C. Weber, L.G. Cohen, C. Brawn, M.A. Dimyan, T. Ard, J. Mellinger, A. Caria, S. Soekadar, A. Fourkas, N. Birbaumer: Think to move: A neuromagnetic brain-computer interface (BCI) system for chronic stroke, Stroke **39**(3), 910–917 (2008)

64.151 E.T. Wolbrecht, V. Chan, D.J. Reinkensmeyer, J.E. Bobrow: Optimizing compliant, model-based robotic assistance to promote neurorehabilitation, IEEE Trans. Neural Syst. Rehabil. Eng. **16**, 286–297 (2008)

64.152 J.L. Patton, M. Kovic, F.A. Mussa-Ivaldi: Custom-designed haptic training for restoring reaching ability to individuals with poststroke hemiparesis, J. Rehabil. Res. Dev. **43**, 643–656 (2006)

64.153 R.G. Lovely, R.J. Gregor, R.R. Roy, V.R. Edgerton: Effects of training on the recovery of full weight-bearing stepping in the adult spinal cat, Exp. Neurol. **92**, 421–435 (1986)

64.154 H. Barbeau, S. Rossignol: Recovery of locomotion after chronic spinalization in the adult cat, Brain Res. **412**, 84–95 (1987)

64.155 M. Visintin, H. Barbeau, N. Korner-Bitensky, N. Mayo: A new approach to retrain gait in stroke patients through body weight support and treadmill stimulation, Stroke **29**, 1122–1128 (1998)

64.156 S. Hesse, C. Bertelt, M. Jahnke, A. Schaffrin, P. Baake, M. Malezic, K.H. Mauritz: Treadmill training with partial body weight support compared with physiotherapy in nonambulatory hemiparetic patients, Stroke **26**, 976–981 (1995)

64.157 A. Wernig, A. Nanassy, S. Muller: Maintenance of locomotor abilities following Laufband (treadmill) therapy in para- and tetraplegic persons: Follow-up studies, Spinal Cord **36**, 744–749 (1998)

64.158 A.L. Behrman, S.J. Harkema: Locomotor training after human spinal cord injury: A series of case studies, Phys. Ther. **80**, 688–700 (2000)

64.159 H. Barbeau: Locomotor training in neurorehabilitation: Emerging rehabilitation concepts, Neurorehabil. Neural Repair **17**, 3–11 (2003)

64.160 B. Dobkin, D. Apple, H. Barbeau, M. Basso, A. Behrman, D. Deforge, J. Ditunno, G. Dudley, R. Elashoff, L. Fugate, S. Harkema, M. Saulino, M. Scott: Weight-supported treadmill vs. overground training for walking after acute incomplete SCI, Neurology **66**, 484–493 (2006)

64.161 T.G. Hornby: Clinical and quantitative evaluation of robotic-assisted treadmill walking to retrain ambulation after spinal cord injury, Top. Spinal Cord Inj. Rehabil. **11**, 1–17 (2005)

64.162 L. Nilsson, K. Fugl-Meyer, L. Kristensen, B. Sjölund, K. Sunnerhagen: Walking training of patients with hemiparesis at an early stage after stroke: A comparison of walking training on a treadmill with body weight support and walking training on the ground, Clin. Rehabil. **15**, 515–527 (2001)

64.163 C. Werner, A. Bardeleben, K. Mauritz, S. Kirker, S. Hesse: Treadmill training with partial body weight support and physiotherapy in stroke patients: A preliminary comparison, Eur. J. Neurol. **9**, 639–644 (2002)

64.164 S. Hesse, C. Werner, H. Seibel, S. von Frankenberg, E. Kappel, S. Kirker, M. Käding: Treadmill training with partial body-weight support after total hip arthroplasty: A randomized controlled trial, Arch. Phys. Med. Rehabil. **84**, 1767–1773 (2003)

64.165 T. Brown, J. Mount, B. Rouland, K. Kautz, R. Barnes, J. Kim: Body weight-supported treadmill training versus conventional gait training for people with chronic traumatic brain injury, J. Head. Trauma. Rehabil. **20**, 402–415 (2005)

64.166 P.W. Duncan, K.J. Sullivan, A.L. Behrman, S.P. Azen, S.S. Wu, S.E. Nadeau, B.H. Dobkin, D.K. Rose, J.K. Tilson, S. Cen, S.H. Hayden: Body-weight-supported treadmill rehabilitation after stroke, N. Engl. J. Med. **364**, 2026–2036 (2011)

64.167 S. Freivogel, J. Mehrholz, T. Husak-Sotomayor, D. Schmalohr: Gait training with the newly developed *LokoHelp*-system is feasible for non-ambulatory patients after stroke, spinal cord and brain injury. A feasibility study, Brain Inj. **22**, 625–632 (2008)

64.168 Reha-Technology: http://www.rehatechnology.com/ (2012)

64.169 C. Werner, S. Von Frankenberg, T. Treig, M. Konrad, S. Hesse: Treadmill training with partial body weight support and an electromechanical gait trainer for restoration of gait in subacute stroke patients: A randomized crossover study, Stroke **33**, 2895–2901 (2002)
64.170 M. Pohl, C. Werner, M. Holzgraefe, G. Kroczek, J. Mehrholz, I. Wingendorf, G. Hoölig, R. Koch, S. Hesse: Repetitive locomotor training and physiotherapy improve walking and basic activities of daily living after stroke: A single-blind, randomized multicentre trial (DEutsche GAngtrainer-Studie, DEGAS), Clin. Rehabil. **21**, 17–27 (2007)
64.171 S. Freivogel, D. Schmalohr, J. Mehrholz: Improved walking ability and reduced therapeutic stress with an electromechanical gait device, J. Rehabil. Med. **41**, 734–739 (2009)
64.172 K.P. Westlake, C. Patten: Journal of neuroengineering and rehabilitation, J. Neuroeng. Rehabil. **6**, 18 (2009)
64.173 T.G. Hornby, D.D. Campbell, J.H. Kahn, T. Demott, J.L. Moore, H.R. Roth: Enhanced gait-related improvements after therapist- versus robotic-assisted locomotor training in subjects with chronic stroke: A randomized controlled study, Stroke; J. Cereb. Circ. **39**, 1786–1792 (2008)
64.174 J. Hidler, D. Nichols, M. Pelliccio, K. Brady, D.D. Campbell, J.H. Kahn, T.G. Hornby: Multicenter randomized clinical trial evaluating the effectiveness of the Lokomat in subacute stroke, Neurorehabil. Neural Repair **23**, 5–13 (2009)
64.175 M. Mihelj: Human arm kinematics for robot based rehabilitation, Robotica **24**, 377–384 (2006)
64.176 S. Hesse, C. Tomelleri, A. Bardeleben, C. Werner, A. Waldner: Robot-assisted practice of gait and stair climbing in nonambulatory stroke patients, J. Rehabil. Res. Dev. **49**, 613–622 (2012)
64.177 S. Hesse, C. Werner, A. Bardeleben: Electromechanical gait training with functional electrical stimulation: Case studies in spinal cord injury, Spinal Cord **42**, 346–352 (2004)
64.178 A. Picelli, C. Melotti, F. Origano, A. Waldner, A. Fiaschi, V. Santilli, N. Smania: Robot-assisted gait training in patients with parkinson disease a randomized controlled trial, Neurorehabil. Neural Repair **26**, 353–361 (2012)
64.179 H. Schmidt, S. Hesse, R. Bernhardt, J. Krüger: HapticWalker – A novel haptic foot device, ACM Trans. Appl. Percept. **2**, 166–180 (2005)
64.180 S. Jezernik, G. Colombo, M. Morari: Automatic gait-pattern adaptation algorithms for rehabilitation with a 4-DOF robotic orthosis, IEEE Trans. Robotics Autom. **20**, 574–582 (2004)
64.181 D. Reinkensmeyer, D. Aoyagi, J. Emken, J. Galvez, W. Ichinose, G. Kerdanyan, S. Maneekobkunwong, K. Minakata, J.A. Nessler, R. Weber, B.R. Roy, R. de Leon, J.E. Bobrow, S.J. Harkema, V.R. Edgerton: Tools for understanding and optimizing robotic gait training, J. Rehabil. Res. Dev. **43**, 657–670 (2006)
64.182 D. Surdilovic, R. Bernhardt: STRING-MAN: A new wire robot for gait rehabilitation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 2031–2036
64.183 J.F. Veneman, R. Ekkelenkamp, R. Kruidhof, F.C.T. van der Helm, H. van der Kooij: A series elastic- and bowden-cable-based actuation system for use as torque actuator in exoskeleton-type robots, Int. J. Robotics Res. **25**, 261–282 (2006)
64.184 J.A. Galvez, D.J. Reinkensmeyer: Robotics for gait training after spinal cord injury, Top. Spinal Cord Inj. Rehabil. **11**, 18–33 (2005)
64.185 D. Aoyagi, W.E. Ichinose, S.J. Harkema, D.J. Reinkensmeyer, J.E. Bobrow: An assistive robotic device that can synchronize to the pelvic motion during human gait training, IEEE Int. Conf. Rehabil. Robotics (2005) pp. 565–568
64.186 S. Jezernik, R. Scharer, G. Colombo, M. Morari: Adaptive robotic rehabilitation of locomotion: A clinical study in spinally injured individuals, Spinal Cord **41**, 657–666 (2003)
64.187 D. Aoyagi: Ph.D. Thesis Ser. (Department of Mechanical and Aerospace Engineering, University of California at Irvine 2006)
64.188 University of Twente, The Netherlands: http://www.utwente.nl/ctw/bw/research/projects/lopes/
64.189 J.F. Veneman, R. Kruidhof, E.E. Hekman, R. Ekkelenkamp, E.H. Van Asseldonk, H. van der Kooij: Design and evaluation of the LOPES exoskeleton robot for interactive gait rehabilitation, IEEE Trans. Neural Syst. Rehabil. Eng. **15**, 379–386 (2007)
64.190 G. Pratt, M. Williamson, P. Dillworth, J. Pratt, K. Ulland, A. Wright: Stiffness isn't everything, 4th Int. Symp. Exp. Robotics (1995)
64.191 D. Accoto, F. Sergi, N. Tagliamonte, G. Carpino, A. Sudano, E. Guglielmelli: Robomorphism: A nonanthropomorphic wearable robot, IEEE Robotics Autom. Mag. **21**(4), 45–55 (2014)
64.192 E.H. Van Asseldonk, J.F. Veneman, R. Ekkelenkamp, J.H. Buurke, F.C. Van der Helm, H. van der Kooij: The effects on kinematics and muscle activity of walking in a robotic gait trainer during zero-force control, IEEE Trans. Neural Syst. Rehabil. Eng. **16**, 360–370 (2008)
64.193 E. Van Asseldonk, R. Ekkelenkamp, J. Veneman, F. Van der Helm, H. Van der Kooij: Selective control of a subtask of walking in a robotic gait trainer (LOPES), IEEE Int. Conf. Rehabil. Robotics (2007) pp. 841–848
64.194 E.H. van Asseldonk, B. Koopman, J.H. Buurke, C.D. Simons, H. van der Kooij: Selective and adaptive robotic support of foot clearance for training stroke survivors with stiff knee gait, IEEE Int. Conf Rehabil. Robotics (2009) pp. 602–607
64.195 F. Sergi, D. Accoto, N.L. Tagliamonte, G. Carpino, E. Guglielmelli: A systematic graph-based method for the kinematic synthesis of non-anthropomorphic wearable robots for the lower limbs, Front. Mech. Eng. **6**(1), 61–70 (2011)
64.196 F. Sergi, D. Accoto, N.L. Tagliamonte, G. Carpino, S. Galzerano, E. Guglielmelli: Kinematic synthesis,

optimization and analysis of a non-anthropomorphic 2-DOFs wearable orthosis for gait assistance, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)
64.197 N.L. Tagliamonte, D. Accoto, F. Sergi, A. Sudano, D. Formica, E. Guglielmelli: Muscular activity when walking in a non-anthropomorphic wearable robot, IEEE Annu. Int. Conf. Eng. Med. Biol. Soc. (2014) pp. 3073–3076
64.198 D. Accoto, G. Carpino, F. Sergi, N.L. Tagliamonte, L. Zollo, E. Guglielmelli: Design and characterization of a novel high-power series elastic actuator for a lower limb robotic orthosis, Int. J. Adv. Robotics Syst. **10**, 359 (2013)
64.199 M. Peshkin, D.A. Brown, J.J. Santos-Munne, A. Makhlin, E. Lewis, J.E. Colgate, J. Patton, D. Schwandt: KineAssist: A robotic overground gait and balance training device, IEEE Int. Conf. Rehabil. Robotics (2005) pp. 241–246
64.200 HDT Global, USA: http://www.hdtglobal.com/services/robotics/medical/kineassist/
64.201 Univ. of Delaware, USA: http://www.udel.edu/udmessenger/vol17no1/stories/robotic_exoskeleton.html
64.202 D. Ferris: Powered lower limb orthoses for gait rehabilitation, Top. Spinal Cord Inj. Rehabil. **11**, 34–49 (2005)
64.203 J. Deutsch, J. Latonio, G. Burdea, R. Boian: Post-stroke rehabilitation with the Rutgers Ankle system – A case study, Presence **10**, 416–430 (2001)
64.204 S. Agrawal, A. Fattah: Theory and design of an orthotic device for full or partial gravity-balancing of a human leg during motion, IEEE Trans. Neural Syst. Rehabil. Eng. **12**, 157–165 (2004)
64.205 S.K. Banala, S.K. Agrawal: Gait rehabilitation with an active leg orthosis, Proc. ASME Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf. (2005)
64.206 J. Kawamura, T. Ide, S. Hayashi, H. Ono, T. Honda: Automatic suspension device for gait training, Prosthet. Orthot. Int. **17**, 120–125 (1993)
64.207 S. Banala, S.H. Kim, S. Agrawal, J. Scholz: Robot assisted gait training with active leg exoskeleton (ALEX), IEEE Trans. Neural Syst. Rehabil. Eng. **17**, 2–8 (2009)
64.208 I. Díaz, J.J. Gil, E. Sánchez: Lower-limb robotic rehabilitation: Literature review and challenges, J. Robotics **2011**, 759–764 (2011)
64.209 R.F. Erlandson, P. deBear, K. Kristy, M. Dijkers, S. Wu: A robotic system to provide movement therapy, Proc. 5th Int. Serv. Robot Conf. (1990) pp. 7–15
64.210 M.J. Rosen: Telerehabilitation, NeuroRehabilitation **12**, 11–26 (1999)
64.211 J. Borenstein, I. Ulrich: The guidecane – A computerized travel aid for the active guidance of blind pedestrians, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1997) pp. 1283–1288
64.212 I. Ulrich, J. Borenstein: The guidecane-applying mobile robot technologies to assist the visually impaired, IEEE Trans. Syst. Man Cybern. **31**(2), 131–136 (2001)
64.213 L. Zollo, K. Wada, H. Van der Loos: Special issue on assistive robotics, IEEE Robotics Autom. Mag. **20**, 16–19 (2013)
64.214 A. Tapus, M. Mataric, B. Scassellati: Socially assistive robotics – The grand challenges in helping humans through social interaction, IEEE Robotics Autom. Mag. **14**, 35–42 (2007)
64.215 National Robotics Initiative: The realization of co-robots acting in direct support of individuals and groups, available online at http://www.nsf.gov/pubs/2012/nsf12607/nsf12607.htm (2012)
64.216 C. Stanger, M. Cawley: Demographics of rehabilitation robotics users, Technol. Disabil. **5**, 125–138 (1996)
64.217 J. Hammel: The role of assessment and evaluation in rehabilitation robotics research and development: Moving from concept to clinic to context, IEEE Trans. Rehabil. Eng. **3**, 56–61 (1995)
64.218 J.J. Wagner, H.F.M. Van der Loos, L.J. Leifer: Dual-character based user interface design for an assistive robot, IEEE Int. Workshop Robot Hum. Commun. (RO-MAN) (1998) pp. 101–106
64.219 H.F.M. Van der Loos, J. Hammel, D.S. Lees, D. Chang, I. Perkash: Field evaluation of a robot workstation for quadriplegic office workers, Eur. Rev. Biomed. Tech. **5**, 317–319 (1990)
64.220 J.J. Wagner, H.F.M. Van der Loos, L.J. Leifer: Construction of social relationships between user and robot, Robotics Auton. Syst. **31**, 185–191 (2000)
64.221 M.J. Johnson, E. Guglielmelli, G.A. Di Lauro, C. Laschi, M.C. Carrozza, P. Dario: GIVING-A-HAND system: The development of a task-specific robot appliance. In: *Advances in Rehabilitation Robotics*, ed. by Z. Bien, D. Stefanov (Springer, Berlin, Heidelberg 2004) pp. 127–141
64.222 S. Groothuis, S. Stramigioli, R. Carloni: Lending a helping hand: Towards novel assistive robotic arms, IEEE Robotics Autom. Mag. **20**(1), 20–29 (2013)
64.223 Kinova: JACO Arm user guide, available online at http://www.robotshop.com (2010)
64.224 Assistive Innovations: iARM intelligent arm robot manipulator, available online at http://assistive-innovations.com (2010)
64.225 W. Yoon: Robotic arm for persons with upper-limb DisAbilities (RAPUDA), AIST Today **36**, 20 (2010)
64.226 T. Barrett Inc.: WAM Arm datasheet, available online at http://www.barrett.com (2012)
64.227 Elumotion: Elumotion RT2-Arm, available online at http://www.elumotion.com (2012)
64.228 G. Hirzinger, N. Sporer, A. Albu-Schaffer, M. Hahnle, R. Krenn, A. Pascucci, M. Schedl: DLR's torque-controlled light weight robot III–are we reaching the technological limits now?, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 1710–1716
64.229 H.F.M. Van der Loos, N. Ullrich, H. Kobayashi: Development of sensate and robotic bed technologies for vital signs monitoring and sleep quality improvement, Auton. Robots **15**, 67–79 (2003)
64.230 T. Sato, T. Harada, T. Mori: Environment-type robot system robotic room featured by behavior

media, behavior contents, and behavior adaptation, IEEE/ASME Trans. Mechatron. **9**, 529–534 (2004)
64.231 G. Romer, H.J.A. Stuyt, A. Peters: Cost-savings and economic benefits due to the assistive robotic manipulator (ARM), IEEE Int. Conf. Rehabil. Robotics (2005) pp. 201–204
64.232 M. Hillman, K. Hagan, S. Hagan, J. Jepson, R. Orpwood: The Weston wheelchair mounted assistive robot – the design story, Robotica **20**(2), 125–132 (2002)
64.233 B.V. FOCAL-Meditech: Personal Robot Bridgit, available online at http://www.focalmeditech.nl (2012)
64.234 J.F. Engelberger: Health-care robotics goes commercial: The HelpMate experience, Robotica **11**, 517–524 (1993)
64.235 P. Dario, C. Laschi, E. Guglielmelli: Design and experiments on a personal robotic assistant, Adv. Robotics **13**, 153–169 (1999)
64.236 European Union: Robot-Era: Implementation and integration of advanced robotic systems, available online at http://www.robot-era.eu/robotera (2012)
64.237 B. Graf, M. Hans, R.D. Schraft: Care-O-bot II – Development of a next generation robotic home assistant, Auton. Robots **16**, 193–205 (2004)
64.238 T.L. Chen, M. Ciocarlie, S. Cousins, P. Grice, K. Hawkins, K. Hsiao, C.C. Kemp, C.-H. King, D.A. Lazewatsky, H. Ngyen, A. Paepcke, C. Pantofaru, W.D. Smart, L. Takayama: Robots for humanity: A case study in assistive mobile manipulation, IEEE Robotics Autom. Mag. **20**(1), 30–39 (2013)
64.239 P. Dario, E. Guglielmelli, C. Laschi, G. Teti: MOVAID: A personal robot in everyday life of disabled and elderly people, Technol. Disabil. **10**, 77–93 (1999)
64.240 R.C. Simpson: Smart wheelchairs: A literature review, J. Rehabil. Res. Dev. **42**, 423–436 (2004)
64.241 S.P. Levine, D.A. Bell, L.A. Jaros, R.C. Simpson, Y. Koren, J. Borenstein: The navchair assistive wheelchair navigation system, IEEE Trans. Rehabil. Eng. E **7**, 443–451 (1999)
64.242 H. Yanco: Wheelesley: A robotic wheelchair system: Indoor navigation and user interface, Lect. Notes Comput. Sci. **1458**, 256–268 (1998)
64.243 P. Encarnação: Understanding and improving power mobility use among older adults: An overview of the canwheel program of research, Assist. Technol. Res. Pract. **33**, 210 (2013)
64.244 P. Viswanathan, A.K. Mackworth, J.J. Little, J. Hoey, A. Mihailidis: NOAH for wheelchair users with cognitive impairment: Navigation and obstacle avoidance help, AAAI Fall Symp. AI Eldercare New Solut. Old Prob. (2008) pp. 150–152
64.245 T.-V. How, R.H. Wang, A. Mihailidis: Evaluation of an intelligent wheelchair system for older adults with cognitive impairments, J. Neuroeng. Rehabil. **10**, 90 (2013)
64.246 R.A.M. Braga, M. Petry, L.P. Reis, A.P. Moreira: Intellwheels: Modular development platform for intelligent wheelchairs, J. Rehabil. Res. Dev. **48**(9), 1061–1076 (2011)
64.247 B.M. Faria, L.P. Reis, N. Lau: Adapted control methods for cerebral palsy users of an intelligent wheelchair, J. Intell. Robotics Syst. **77**(2), 1–14 (2014)
64.248 M. Bailey, A. Chanler, B. Maxwell, M. Micire, K. Tsui, H. Yanco: Development of vision-based navigation for a robotic wheelchair, IEEE Int. Conf. Rehabil. Robotics (2007) pp. 951–957
64.249 D. Ding, R.A. Cooper: Electric powered wheelchairs, IEEE Control Syst. **25**(2), 22–34 (2005)
64.250 K.M. Tsui, H.A. Yanco, D.J. Feil-Seifer, M.J. Matarić: Survey of domain-specific performance measures in assistive robotic technology, Proc. ACM 8th Workshop Perform. Metr. Intell. Syst. (2008) pp. 116–123
64.251 M.A. Peshkin, J.E. Colgate, W. Wannasuphoprasit, C.A. Moore, R.B. Gillerpie, P. Akella: Cobot architecture, IEEE Trans. Robotics Autom. **17**(4), 377–390 (2001)
64.252 J. Colgate, J. Edward, M.A. Peshkin, W. Wannasuphoprasit: *Cobots: Robots for Collaboration with Human Operators* (Northwestern University, Evanston 1996)
64.253 C.A. Moore, M.A. Peshkin, J.E. Colgate: Cobot implementation of virtual paths and 3D virtual surfaces, IEEE Trans. Robotics Autom. **19**(2), 347–351 (2003)
64.254 G. Lacey, S. MacNamara: User involvement in the design and evaluation of a smart mobility aid, J. Rehabil. Res. Dev. **37**(6), 709–723 (2000)
64.255 A.M. Dollar, H. Herr: Lower extremity exoskeletons and active orthoses: Challenges and state-of-the-art, IEEE Trans. Robotics **24**, 144–158 (2008)
64.256 H. Herr: Exoskeletons and orthoses: Classification, design challenges and future directions, J. Neuroeng. Rehabil. **6**, 21 (2009)
64.257 J.L. Pons: Rehabilitation exoskeletal robotics, IEEE Eng. Med. Biol. Mag. **29**, 57–63 (2010)
64.258 E. Strickland: Good-bye, wheelchair, hello, exoskeleton, http://spectrum.ieee.org/biomedical/bionics/goodbye-wheelchair-hello-exoskeleton
64.259 T.A. Swift, K.A. Strausser, A. Zoss, H. Kazerooni: Control and experimental results for post stroke gait rehabilitation with a prototype mobile medical exoskeleton, ASME Dyn. Syst. Control Conf. (2010)
64.260 K.A. Strausser, T.A. Swift, A. Zoss, H. Kazerooni: Prototype medical exoskeleton for paraplegic mobility: First experimental results, ASME Dyn. Syst. Control Conf. 2010, Cambridge (2010)
64.261 Argo-Medical: http://www.rewalk.com/ (2012)
64.262 T. Engineer: Rewalk is a new exoskeleton that lets paralysed people walk again, available online at http://wonderfulengineering.com/rewalk-is-a-new-exoskeleton-that-lets-paralysed-people-walk-again/
64.263 H. Kawamoto, Y. Sankai: Power assist system HAL3 for gait disorder person, Lect. Notes Comput. Sci. **2398**, 196–203 (2002)
64.264 K. Suzuki, Y. Kawamura, T. Hayashi, T. Sakurai, Y. Hasegawa, Y. Sankai: Intention-based walking

support for paraplegia patient, IEEE Int. Conf. Syst. Man Cybern., Vol. 3 (2006) pp. 2707–2713
64.265 R. Farris, H. Quintero, M. Goldfarb: Preliminary evaluation of a powered lower limb orthosis to aid walking in paraplegic individuals, IEEE Trans. Neural Syst. Rehabil. Eng. **19**(6), 652–659 (2011)
64.266 Rex-Bionics: http://www.rexbionics.com (2012)
64.267 R.F.F. Weir: Design of artificial arms and hands for prosthetic applications. In: *Standard Handbook of Biomedical Engineering and Design*, ed. by M. Kutz (McGraw-Hill, New York 2003) pp. 32.1–32.61
64.268 T.A. Kuiken, G.A. Dumanian, R.D. Lipschutz, L.A. Miller, K.A. Stubblefield: The use of targeted muscle reinnervation for improved myoelectric prosthesis control in a bilateral shoulder disarticulation amputee, Prosthet. Orthot. Int. **28**(3), 245–253 (2004)
64.269 T.A. Kuiken, L.A. Miller, R.D. Lipschutz, B.A. Lock, K. Stubblefield, P.D. Marasco, P. Zhou, G.A. Dumanian: Targeted reinnervation for enhanced prosthetic arm function in a woman with a proximal amputation: A case study, Lancet **369**, 371–380 (2007)
64.270 G.S. Dhillon, K.W. Horch: Direct neural sensory feedback and control of a prosthetic arm, IEEE Trans. Neural Syst. Rehabil. Eng. **13**, 468–472 (2005)
64.271 K.P. Hoffmann, K.P. Koch, T. Doerge, S. Micera: New technologies in manufacturing of different implantable microelectrodes as an interface to the peripheral nervous system, IEEE/RAS-EMBS Int. Conf. Biomed. Robotics Biomech. (BioRob) (2006) pp. 414–419
64.272 P.M. Rossini, S. Micera, A. Benvenuto, J. Carpaneto, G. Cavallo, L. Citi, C. Cipriani, L. Denaro, V. Denaro, G. Di Pino, F. Ferrari, E. Guglielmelli, K.-P. Hoffmann, S. Raspopovic, J. Rigosa, L. Rossini, M. Tombini, P. Dario: Double nerve intraneural interface implant on a human amputee for robotic hand control, Clin. Neurophys. **121**, 777–783 (2010)
64.273 S. Raspopovic, M. Capogrosso, F.M. Petrini, M. Bonizzato, J. Rigosa, G. Di Pino, I. Carpanedo, M. Controzzi, T. Boretius, E. Fernandez, G. Granata, C.M. Oddo, L. Citi, A.L. Ciancio, C. Cipriani, M.C. Carrozza, W. Jensen, E. Guglielmelli, T. Stieglitz, P.M. Rossini, S. Mieera: Restoring natural sensory feedback in real-time bidirectional hand prostheses, Sci. Transl. Med. **6**(222), 222ra19 (2014)
64.274 S.G. Meek, S.C. Jacobsen, P.P. Goulding: Extended physiologic taction: Design and evaluation of a proportional force feedback system. J. Rehabil. Res, Dev. **26**(3), 53–62 (1989)
64.275 H.M. Herr, J.A. Weber, S.K. Au, B.W. Deffenbaugh, L.H. Magnusson, A.G. Hofmann, B.B. Aisen: Powered ankle food prothesis, US Patent Ser 851 2415 B2 (2013)
64.276 Nature 442 (7099): http://www.nature.com/nature/journal/v442/n7099/index.html, 109–222, 2006
64.277 L.R. Hochberg, M.D. Serruya, G.M. Friehs, J.A. Mukand, M. Saleh, A.H. Caplan, A. Branner, D. Chen, R.D. Penn, J.P. Donoghve: Neuronal ensemble control of prosthetic devices by a human with tetraplegia, Nature **442**, 164–171 (2006)
64.278 C. Babiloni, V. Pizzella, C.D. Gratta, A. Ferretti, G.L. Romani: Fundamentals of electroencefalography, magnetoencefalography, and functional magnetic resonance imaging, Int. Rev. Neurobiol. **86**, 67–80 (2009)
64.279 G.E. Loeb, F.J. Richmond, L.L. Baker: The BION devices: Injectable interfaces with peripheral nerves and muscles, Neurosurg. Focus **20**, E2 (2006)
64.280 R.B. Stein, V. Mushahwar: Reanimating limbs after injury or disease, Trends Neurosci. **28**, 518–524 (2005)
64.281 P.F. Pasquina, P.R. Bryant, M.E. Huang, T.L. Roberts, V.S. Nelson, K.M. Flood: Advances in amputee care, Arch. Phys. Med. Rehabil. **87**, S34–S43 (2006)
64.282 H. Vallery, J. Veneman, E. van Asseldonk, R. Ekkelenkamp, M. Buss, H. van der Kooij: Compliant actuation of rehabilitation robots: Benefits and limitations of series elastic actuators, IEEE Robotics Autom. Mag. **15**, 60–69 (2008)
64.283 A. Basmajian, E.E. Blanco, H.H. Asada: The marionette bed: Automated rolling and repositioning of bedridden patients, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 1422–1427
64.284 F. Kasagami, H. Wang, I. Sakuma, M. Araya, T. Dohi: Development of a robot to assist patient transfer, IEEE Int. Conf. Syst. Man Cybern., Vol. 5 (2004) pp. 4383–4388
64.285 G.D. Abowd, M. Ebling, G. Hung, L. Hui, H.W. Gellersen: Context-aware computing, IEEE Pervasive Comput. **1**, 22 (2002)
64.286 M.J. Covington, W. Long, S. Srinivasan, A.K. Dev, M. Ahamad, G.D. Abowd: *Securing Context-Aware Applications Using Environment Roles* (Georgia Institute of Technology, Atlanta 2001)
64.287 R. Sanchez, D.E.R.I.C. Reinkensmeyer, P. Shah, J. Liu, S. Rao, R. Smith, J. Bobrow: Monitoring functional arm movement for home-based therapy after stroke, IEEE Annu. Int. Conf. Eng. Med. Biol. Soc. Chicago, Vol. 2 (2004) pp. 4787–4790
64.288 S.J. Housman, V. Le, T. Rahman, R.J. Sanchez, D.J. Reinkensmeyer: Arm-training with T-WREX after chronic stroke: Preliminary results of a randomized controlled trial, IEEE Int. Conf. Rehabil. Robotics (2007) pp. 562–568
64.289 A.J. Butler, C. Bay, D. Wu, K.M. Richards, S. Buchanan: Expanding tele-rehabilitation of stroke through in-home robot-assisted therapy, Int. J. Phys. Med. Rehabil. **2**(184), 2 (2014)
64.290 S.M. Linder, A.B. Rosenfeldt, R.C. Bay, K. Sahu, S.L. Wolf, J.L. Alberts: Improving quality of life and depression after stroke through telerehabilitation, Am. J. Occup. Ther. **69**(2), 6902290020 (2015)

64.291 F.H. Wilhelm, W.T. Roth: Ambulatory assessment of clinical anxiety. In: *Ambulatory Assessment: Computer-Assisted Psychological and Psychophysiological Methods in Monitoring and Field Studies*, ed. by J. Fahrenberg, M. Myrteck (Hogrefe Huber, Seattle 1996) pp. 317–345
64.292 E. Dishman: Inventing wellness systems for aging in place, Computer **37**, 34 (2004)
64.293 H. Zhou, H. Hu: Human motion tracking for rehabilitation—A survey, Biomed. Sig. Process. Control **3**, 1–18 (2008)
64.294 E. Papaleo, L. Zollo, S. Sterzi, E. Guglielmelli: An inverse kinematics algorithm for upper-limb joint reconstruction during robot-aided motor therapy, IEEE/RAS-EMBS Int. Conf. Biomed. Robotics Biomechatron. (BioRob) (2012) pp. 1983–1988
64.295 M. Zinn, B. Roth, O. Khatib, J. Salisbury: A new actuation approach for human friendly robot design, Int. J. Robotics Res. **23**, 379–398 (2004)
64.296 M. Nokata, K. Ikuta, H. Ishii: Safety evaluation method for rehabilitation robotics. In: *Advances in Rehabilitation Robotics*, ed. by Z. Bien, D. Stefanov (Springer, Berlin, Heidelberg 2004) pp. 187–198
64.297 N. Tejima: Risk reduction mechanisms for safe rehabilitation robots. In: *Advances in Rehabilitation Robotics*, ed. by Z. Bien, D. Stefanov (Springer, Berlin, Heidelberg 2004) pp. 187–198
64.298 H.F.M. Van der Loos: Design and engineering ethics considerations for neurotechnologies, Camb. Quar. Heal. Ethics **16**, 305–309 (2007)
64.299 G. Veruggio: The Roboethics Roadmap, available online at http://www.roboethics.org/site/modules/mydownloads/visit.php?cid=1&lid=37 (2007)
64.300 J.F. Engelberger: *Robotics in Service* (MIT Press, Cambridge 1989)
64.301 G. Colombo, M. Jorg, V. Dietz: Driven gait orthosis to do locomotor training of paraplegic patients, IEEE Annu. Int. Conf. Eng. Med. Biol. Soc., Vol. 4 (2000) pp. 3159–3163
64.302 I.T. Lott, E. Doran, D.M. Walsh, M. Hill: Telemedicine, dementia and Down syndrome: Implications for Alzheimer disease, Alzheimer's Dement. **2**, 179–184 (2006)
64.303 D. Reinkensmeyer, P. Lum, J. Winters: Emerging technologies for improving access to movement therapy following neurologic injury. In: *Emerging and Accessible Telecommunications, Information and Healthcare Technologies: Engineering Challenges in Enabling Universal Access*, ed. by J. Winters, C. Robinson, R. Simpson, G. Vanderheiden (Rehabiltitation Eng. Soc. North Am., Arlington 2002) pp. 136–150
64.304 M. Lotze, C. Braun, N. Birbaumer, S. Anders, L.G. Cohen: Motor learning elicited by voluntary drive, Brain **126**, 866–872 (2003)
64.305 J.E. Speich, J. Rosen: Medical robotics. In: *Encyclopedia of Biomaterials and Biomedical Engineering*, ed. by G.E. Wnek, G.L. Bowlin (Marcel Dekker, New York 2004)
64.306 D.J. Reinkensmeyer, J.L. Emken, S.C. Cramer: Robotics, motor learning, and neurologic recovery, Annu. Rev. Biomed. Eng. **6**, 497–525 (2004)
64.307 M. Oishi, I. Mitchell, H. Van der Loos: *Design and Use of Assistive Technology* (Springer, Berlin, Heidelberg 2010)

Multimedia Contents

65. Domestic Robotics

Erwin Prassler, Mario E. Munich, Paolo Pirjanian, Kazuhiro Kosuge

When the first edition of this book was published domestic robots were spoken of as a dream that was slowly becoming reality. At that time, in 2008, we looked back on more than twenty years of research and development in domestic robotics, especially in cleaning robotics. Although everybody expected cleaning to be the killer app for domestic robotics in the first half of these twenty years nothing big really happened. About ten years before the first edition of this book appeared, all of a sudden things started moving. Several small, but also some larger enterprises announced that they would soon launch domestic cleaning robots. The robotics community was anxiously awaiting these first cleaning robots and so were consumers. The big burst, however, was yet to come. The price tag of those cleaning robots was far beyond what people were willing to pay for a vacuum cleaner. It took another four years until, in 2002, a small and inexpensive device, which was not even called a cleaning robot, brought the first breakthrough: *Roomba*. Sales of the Roomba quickly passed the first million robots and increased rapidly. While for the first years after Roomba's release, the big players remained on the sidelines, possibly to revise their own designs and, in particular their business models and price tags, some other small players followed quickly and came out with their own products. We reported about theses devices and their creators in the first edition. Since then the momentum in the field of domestics robotics has steadily increased. Nowadays most big appliance manufacturers have domestic cleaning robots in their portfolio. We are not only seeing more and more domestic cleaning robots and lawn mowers on the market, but we are also seeing new types of domestic robots, window cleaners, plant watering robots, tele-presence robots, domestic surveillance robots, and robotic sports devices.

Some of these new types of domestic robots are still prototypes or concept studies. Others have already crossed the threshold to becoming commercial products.

For the second edition of this chapter, we have decided to not only enumerate the devices that have emerged and survived in the past five years, but also to take a look back at how it all began, contrasting this retrospection with the burst of progress in the past five years in domestic cleaning robotics. We will not describe and discuss in detail every single cleaning robot that has seen the light of the day, but select those that are representative for the evolution of the technology as well as the market. We will also reserve some space for new types of mobile domestic robots, which will be the success stories or failures for the next edition of this chapter. Further we will look into nonmobile domestic robots, also called smart appliances, and examine their fate. Last but not least, we will look at the recent developments in the area of intelligent homes that surround and, at times, also control the mobile domestic robots and smart appliances described in the preceding sections.

65.1 **Mobile Domestic Robotics** 1730
65.1.1 Domestic Floor Cleaning 1730
65.1.2 Domestic Window Cleaning 1737
65.1.3 Pool Cleaning 1739
65.1.4 Lawn Mowing 1740
65.1.5 Sports Robotics 1742
65.1.6 Tele-Presence 1745
65.2 **Enabling Technologies** 1747
65.2.1 Sensing and Obstacle Avoidance 1747
65.2.2 Localization and Mapping 1749
65.2.3 Navigation and Coverage 1753
65.3 **Smart Homes** 1754
65.3.1 Gator Tech Smart House 1755

65.3.2 Aware Home 1756
65.3.3 SELF – Sensorized Environment for LiFe 1756
Video-References 1757
References 1757

65.1 Mobile Domestic Robotics

The first mention of a domestic cleaning robot dates back to 1985. The device, nicknamed *Robby*, was developed by Hitachi starting in 1983 and officially carried the name HCR-00 (Fig. 65.1). Robby was equipped with a gyroscope to keep track of its position and a rotating sonar scanner for obstacle detection.

65.1.1 Domestic Floor Cleaning

Very similar to today's cleaning robots described further below, Robby already mapped its environment and used the map for path planning. HRC-00 remained a prototype like its successors HRC-01 to HRC-03.

First Generation of Domestic Cleaning Robotics (1985–1999)

Also in 1985, the Swedish appliance manufacturer Electrolux started the development of a concept vacuum cleaner *Stardust*. The device was equipped with eight fixed sonar sensors and one rotating sensors for obstacle detection. To maintain its orientation Stardust used an infrared sensor that tracked an infrared light bulb mounted to the ceiling. In 1988 Stardust was presented to the public at Domotechnica in Cologne, Germany, one of the largest fairs worldwide for domestic appliances [65.1].

Fig. 65.1 First generation of domestic floor cleaning robots (1985–1999)

About five years later, between 1989 and 1991, Panasonic undertook an effort to develop a domestic cleaning robot that led to two prototypes, one of them, *Brownie*, is shown in Fig. 65.1. Brownie was a battery-powered vacuum cleaner with a diameter of 40 cm, and at 18 kg, a super heavy weight compared to today's domestic cleaning robots. It was equipped with a gyroscope, a ring of sonar sensors, and a dust sensor. Likely, it was limitations in battery technology for driving such a heavy robot, as well as the total cost, which prevented Brownie from become anything more than a prototype.

Six more years went by before Minolta presented its *Minolta Cleaning Robot* and the first European player, Electrolux, its Robot Vacuum Cleaner, later called Trilobite, which was presented to the public in BBC's TV show *Tomorrows World* in 1996.

The devices had already become significantly smaller and lighter in these six years. The Minolta robot had dimensions of 321 mm × 320 mm × 170 mm and weighted about 8 kg. It was powered by nickel metal hybrid battery and used sonar and tactile sensors for obstacle detection, a cliff-sensor to discover staircases, and a gyroscope to track its position and orientation. Electrolux' robot also had a sophisticated sonar system, which let it follow contours and even return to a homing position. It already had the size and shape of later floor cleaning robots.

Five more domestic cleaning robots made their first appearance during this time, which we call the infancy of domestic cleaning robotics. Two of them only reached the proof of concept level: *AutoCleaner* from InMach Intelligent Machines – the German startup later developed low-cost navigation systems for the professional cleaning robot Robo40, which became commercially available in 2007 – and Koala form EPFL-Lami. AutoCleaner was the first domestic wet-cleaner proof of concept, which cleaned the floor using a rotating microfiber towel that was pulled through a water tank. Koala used a suction spout to reach corners. Two more robots reached at least the status of industrial prototypes, but were never commercialized: Dyson's *DC06* and Cye from *Probotics*. DC06 was a unique domestic

Fig. 65.2 Second generation of domestic floor cleaning robots (2000–2008)

cleaning robot. It differed not only in terms of the announced list price of approximately US$ 4000, but also in many other aspects: Dyson claimed that it had three onboard CPUs, more than 50 sensors for obstacle avoidance, cliff-detection, localization and more. On top of that came the unique Dual Cyclone cleaning technology. The DC06 never made it to the market. Neither did Cye. Cye was a small mobile robot that could pull and push a (semi-)regular vacuum cleaner. It was announced as the *first personal robot* which could not only vacuum the floor but also serve coffee or deliver mail. It is unclear if Cye ever delivered coffee or mail or vacuumed a real living room. It disappeared as quickly as it appeared. The last out of the nine first robots that we call first generation was the Kärcher RoboCleaner. Its successor RC3000 was amongst the first cleaning robots that had a docking station at which the robot could not only recharge its battery but also unload the dust, which it had collected. We will come back to the Kärcher RC3000 in the next section.

The first generation domestic floor cleaning robots shown in Fig. 65.1 were the ones presented to the public. However, they were by far not the only ones that were developed in the period between 1985 and 1999. By browsing over the patents in the field of cleaning robots during that period it becomes obvious that many of the big appliance and electronic device manufacturers worldwide performed research and development on domestic cleaning robots. Besides the ones mentioned above, one can find such big names as Nintendo, Matsushita, Sanyo, Samsung, Honda, Procter & Gamble, Electrolux, Philips, Henkel and more. So, although only a few players dared to go public, domestic cleaning was already on the radar of many international enterprises. What kept them away from taking the next steps is not hard to guess. It was the risk of getting into a rather conservative and tight market – the cleaning business is extremely conservative – with a semimature technology carrying a price tag that looked quite differently than the one on well-established technology, such as traditional vacuum cleaners.

The Second Generation of Domestic Cleaning Robotics (2000–2008)

Figure 65.2 shows the second generation of domestic cleaning robots. What made the difference between the first and the second generation? The amount of experimentation regarding the overall design decreased, with the majority of manufacturers settling on the disc shape. The number of sensors decreased, as well as their level of sophistication. Sophisticated and expensive sensors such as sonar largely disappeared. Most designs were limited to very few, simple, and inexpensive sensors such as bumpers, simple one-dimensional range sensors, typically based on infrared (IR), and cliff sensors. This was most likely a rather painful lesson learned from the designs in the first generation. What roboticists considered the state of the art in mobile robotics: environmental sensing, map building, range sensors with high quality angular and range res-

olution, and combined localization and mapping, were far too expensive to be built into a domestic cleaning robot. Those robots had to compete with regular vacuum cleaners in a price range of a few hundred dollars. This competition imposed very harsh cost limits for the robotic components that could be built into a domestic cleaning robot: 50 to 100 USD and often less. These cost constraints had a significant impact on a very fundamental expectation of what cleaning robots should achieve: systematic coverage of the area to be cleaned. Cleaning was understood to be synonymous with covering an area with a systematic and intuitive motion while applying some type of cleaning operation. Most of the patents filed for the first generation of domestic cleaning robots mentioned above described cleaning robots that were supposed to cover their workspace in a systematic fashion.

Systematic coverage, however, is impossible without absolute positioning and without decent knowledge of the environment. The second generation of domestic cleaning robots had to provide a solution for this dilemma of systematic coverage with cheap and simple sensors, which must not cost more than several tens of dollars.

The solution to this dilemma was already proposed in some early prototypes. The idea was to waive the requirement of systematic coverage, which would also involve intuitive motion patterns, and instead produce *semisystematic* coverage and *semi-intuitive* motion patterns. This semisystematic coverage was achieved by a combination of random motion, hard-coded motion patterns such as spiral- or meander-shaped motions, which – considered alone – reflects some systematic coverage, and some other systematic and intuitive motion patterns, for example, following contours of walls or other objects in the workspace. The theory of stochastic process states that the mean squared distance of a particle that performs a random walk with respect to the origin of its motion increases proportionally with time. That implies that the particle moving in confined space will cover that space in a finite amount of time.

First-generation domestic cleaning robots like the Electrolux Robotics Vacuum Cleaner (later Trilobite) or the Kärcher Robot Cleaner (later Kärcher RC 3000) already used random motion combined with hard-coded motion patterns to achieve a certain degree of coverage. But apparently the insight that the attribute *robotic* in front of *vacuum cleaner* was not enough of a sales argument to justify a price three to five times higher than that of a regular vacuum cleaner was not so easy to digest. Some of the second-generation cleaning robots were offered for a price on the order of 1500 USD or even more. In the end these robots shared the same fate: they become shelf warmers.

This is also due to the fact that in 2002 a robot was launched which was to herald a breakthrough in domestic cleaning robotics: Roomba. Its creator, iRobot Inc., had learned the lesson that others were still struggling with: if you want to sell a domestic appliance, better sell it for a price that is known for domestic appliances. For the sake of fairness one has to mention that the traditional vacuum cleaner manufacturers had to design the robots so they matched the quality and performance level that could be expected from the brand name. They could not take the risk of making their product too poor. Therefore it easily became over specified compared to what the market required at the time. New manufacturers such as iRobot in contrast had no brand name to defend [65.1].

If it is a totally new device, which may not only cause excitement but also concerns and reservations, sell it for less and not for more. When Roomba entered the market it was sold for 199 USD. That was a price, which did not cause customers to think about whether they really needed it, whether the quality was good enough, or whether the device would get into every corner of a room or underneath every couch or bed. The creators of Roomba were also smart enough not to call it a robot. This that prevented many customers from developing wrong expectation of what robots could or should do.

The robot technology that was built into the Roomba was everything but new or revolutionary. Roomba used a suspended front-shield for contact-sensing, a low-cost infrared range-sensor for contour-following, a cliff-sensor to prevent it from falling down stair-cases, and it could detect photoelectric barriers, so-called *virtual walls*, which kept it from leaving a room or a certain part of its workspace. Roomba further used the combination of random motion and hardcoded area-covering motion patterns to achieve a certain level of coverage. All this technology was known before Roomba.

Still Roomba can be seen as a milestone not only in domestic robotics but also in robotics at large. Why is that? It was the first time in robotics history that robotics was no longer synonymous with high-tech, high-price. Roomba showed that automation of everyday service tasks, such as domestic cleaning, could be achieved with a moderate effort in terms of hardware and at a decent price, given that one accepts some graceful degradation in the overall performance. There is no such thing as a free lunch, not even in robotics and this degradation in the performance is the price to be paid. What turned Roomba into a milestone was the cost-effective design, where the limitations of low-cost hardware – in Roomba's case the sensors – were balanced out by smart heuristics for problem-solving. This

Table 65.1 Technical specification of a selection of second-generation domestic cleaning robots

Manufacturer	iRobot	Sharper Image	Electrolux	Kärcher	Yujin
Model	Roomba	eVac	Trilobite 2.0	RC 3000	iclebo
Drive	Differential	Differential	Differential	Differential	Differential
Sensors	Suspended front shield as contact sensor, IR range sensor, four IR cliff sensors, dust sensor	Touch sensor, cliff sensor, IR for wall-following	180° sonar sensors (1 transmitter, 8 receivers), infrared cliff sensor, magnetic stripe detector, suspended front shield as contact sensor	Suspended front shield as contact sensor, four IR cliff sensor	Main brush, side brush anti-bacterial filter
Mapping and localization	n/a	n/a	n/a	n/a	n/a
Navigation/coverage	Random motion with bang and bounce, contour following, spiral motion	Random motion with bang and bounce, spot cleaning with star pattern	Wall-following, random motion with obstacle avoidance	Random motion with bang and bounce, spot cleaning with see-saw motion pattern	Heuristic pat-terns: random, circular, zig-zag, wall following
Cleaning technology	Side brush, two counter-rotating brushes, suction pump	Rotating brushes, vacuum pump	Rotating brush, suction pump	Rotating brush, suction pump	Main brush, side brush
Run time (min)	60–90	15–45	60	20–60	150
Performance	–	–	28 m^2/h	15 m^2/h	–
Docking/recharging station	Yes	No	Yes	Yes	no
Size (Ø/h) (cm)/(cm)	35/8,25	32/14	35/13	28/10	35/9
Weight (kg)	2,7	3	5	2	4
Year of launch	2002	2004	2004	2003	2005
Price range	250 USD	100 USD	1300 EUR	1350 EUR	530 USD

design of a commercially viable product, at a price point competitive with nonrobotized solutions, turned Roomba into the first successful domestic cleaning robot, and the most frequently sold robot in the past 50 years.

In Table 65.1 we give an overview of the technical specifications of some of these second generation robots. A more complete overview is included in 065-bib43

The idea to develop and commercialize a domestic cleaning robot at a price level of 199 USD, comparable to the price of traditional domestic appliances, was very appealing to consumers. However, it was not an overly profitable business model, even if Roomba was manufactured in a low-wage country at a cost of significantly less than 100 USD. Extreme cost pressure, however, often compromises the quality of a product. So a decision to adhere to this low price of below 200 USD may also have been a decision against the quality of the product. Today nearly a dozen models of Roomba are on the market, which differ mostly in terms of their cleaning technology and extra features. Their prices are in a range from 250 to 900 USD.

The Third Generation of Domestic Cleaning Robotics (2009–2012)

The second generation of domestic cleaning robots also provided their developers and distributors with a number of painful and partly contradictory insights:

- A domestic cleaning robot sold at a price that is significantly beyond the price of comparable non-robotic devices, runs a high risk of failing, because many customers may not be willing to pay extra money for a cleaning robot just because it is a robot.
- A domestic cleaning robot sold at the same price level as an inexpensive comparable appliance, runs a high risk of failing, because the low price may

compromise the quality and functionality of the product, and many customers may not be willing to pay even a rather low price for a product that is known to be of poor quality.
- For most customers – traditional, as well as technophiles – efficient cleaning requires systematic and efficient coverage of the workspace. A domestic cleaning robot that uses sophisticated sensors to achieve systematic and efficient coverage may easily end up with a price tag that is significantly above the price of a comparable nonrobotic device and hence runs a high risk of failure.
- A domestic cleaning robot, which refrains from using expensive sensors to achieve systematic coverage but instead uses low-cost sensors for collision avoidance, fall protection, and confinement as well as using random motion combined with some hard-coded motion patterns to achieve a certain level of coverage, runs a risk of not satisfying those customers who expect systematic cleaning and coverage.

From a developers point of view these lessons sound as if customers expect not less than to square the circle. The truth is that customers do not care about squares and circles. They expect value for money.

With the insights above one could classify potential customers of domestic cleaning robots into three categories:

- Customers who only care about the price and not so much about quality and functionality. The criterion for this class of customers was obvious: reduce the price as much as possible without ignoring that there may be a bottom line for what customers might expect in terms of quality and functionality.
- Customers who care about the price and quality, but are willing to adjust their expectations of functionality and efficiency. The criterion for such customers was to reduce the price but only to a certain extent, which does not compromise the quality of the product too much. Expectations of such customers in terms of efficiency (i. e., systematic coverage) can be compensated by auxiliary equipment such as automatic charging stations.
- Customers, who care about price, quality, and efficiency. Apparently these customers are the most demanding. None of the second-generation domestic cleaning robots has really managed to satisfy them. The criterion to serve them would be to develop a low-cost navigation system based on low-cost sensors that provides systematic coverage.

As satisfying the last group of customers, which is also likely the largest of the three groups, seemed nearly impossible, the third generation of domestic cleaning robots have diverged somewhat. Some manufacturers have focused on products that serve the first category of customers, some serve the second group and some even made an effort to square the circle.

May they be successful in the end or not, what cannot be overlooked is the explosion of the number of manufacturers and distributors of domestic cleaning robots in the past five years. At the end of the 2012, twenty-seven years after the first mention of a domestic cleaning robot to the public, Amazon alone listed more than 131 results under the key word *robot vacuum cleaner*, with 14 manufacturers and suppliers. The business to business (B2B) portal www.made-in-china.com lists more than 71 companies for *robot vacuum cleaner* and a total of 875 products. Some of these 875 robot vacuum cleaners look surprisingly similar to the products sold over B2C platforms under very-well known brand names.

In Fig. 65.3 we show some of the products, which primarily intend to serve customers of the first category and therefore stayed under a price of 200 EUR. Figure 65.3a shows a model series of six cleaning robots made by XRobot of the Chinese manufacturer Shenzhen Silver Star Intelligent Electronics Ltd. In B2B trading these robots cost between 64 and 102 USD per piece at a minimum order of 500 pieces. Most models of this series are also sold under the brand name of European and American enterprises. Figure 65.3b shows another series of still rather cheap robotic vacuum cleaners. The attentive reader may notice that the robots in the left and the right figure are not totally unique. Some just have different names and different prices. This is not entirely unintended. It just illustrates that in domestic cleaning robotics the value creation chain is no longer limited to the developers and manufacturers, just as in every other business.

Apart from a few minor details, all robots in Fig. 65.3 use very similar, though not to say the same, technology. They use very few and very cheap (contact, cliff, sometimes dirt) sensors and random motion combined with preprogrammed motion patterns. The vacuum technology consists of a rotating brush, sometimes combined with a small fan. Not surprisingly, robotic vacuum cleaners like the ones in Fig. 65.3 are often called Roomba-clones. The collection of robots in Fig. 65.3 is neither representative nor comprehensive.

Roomba has undoubtedly written robot history. It is the merit of Roomba and its developers that domestic cleaning robots are no longer considered as gadgets but as real appliances. Roomba has opened the door for all

Fig. 65.3 Third generation of domestic floor cleaning robots: low-cost, low-tech

Fig. 65.4 The seventh generation of Roombas

the cleaning robots mentioned above, be they Roomba-clones or not. And Roomba is the by far the best-selling robot ever.

In Fig. 65.4 we show some of the grandchildren and great-grandchildren of the very first Roomba, which started in 2002. The Roomba has gone through several facelifts in these ten years. The latest series, the Roomba 700 is the sixth generation of Roombas. It has matured quite a bit in many respects. It has matured in terms of handling, cleaning technology, obstacle avoidance and navigation technology. At the same time Roomba is delivered in more than a dozen versions, ranging from 250 to 900 USD. These versions differ in the design, in the sophistication of the user interface, in the navigation and coverage strategies, and in the details of the cleaning technology.

According to iRobot more than 6 million units have been sold in the ten years of its existence. Roomba has certainly satisfied quite a number of customers. It is built with good quality, and comes with a self-charging home base and other auxiliary equipment. What has not changed, however, is the basic strategy for covering the workspace. The sixth generation just like the first uses random motion with some precoded motion patterns and heuristics.

This raises one rather fundamental question. Will Roomba's success story continue? Will it maintain its market position? Or has Roomba possibly reached or even passed its summit? These questions will not be answered before the third edition of this book, as any statement in this direction would be pure speculation.

In any case, there are a few recent domestic cleaning robots, which demonstrate that covering a work space in a systematic fashion does not necessarily require expensive sensing, that would push the price for a domestic cleaning robot to a level which customers would not be willing to pay.

Four different key technologies enable the devices shown in Table 65.2 to localize themselves and navigate

Table 65.2 Systematic cleaning at low-cost. The information in this table was partly collected from the websites and the technical documentation of the manufactures and partly from public websites such as www.botroom.com, staubsaugerroboter-test.org, www.robotreviews.com, www.geek.com, gizmodo.com, www.engadget.com

Manufacturer	LG	Samsung	Neato Robotics	Iclebo	Evolution Robotics/iRobot
Distributor			Vorwerk	Philips	Dirt Devil EVO/iRobot
Model	Hom-Bot 3.0	Navibot SR 8895 Silencio	Neato XV-21	Philips FC9910/01 HomeRun/ iclebo smart	Mint 4200
CPU	32 bit	2 CPUs			ARM7
Drive	Differential	Differential	Differential	Differential	Differential
Sensors	Camera (ceiling), camera (floor), sonar & infrared for collision avoidance cliff sensors, gyroscope, acceleration	Range sensors camera, cliff sensors, collision sensors, gyro	1-D laser range finder, cliff sensor, gyroscope, acceleration,	Camera (ceiling), cliff sensor, infra red for collision detection, gyroscope,	Northstar Indoor GPS, cliff sensors, bumper, gyro
Mapping and localization	SLAM	SLAM based on ceiling pictures at 30 fps (Visionary Mapping System)	SLAM based on laser range finder (onboard Room Positioning System (RPS))	SLAM based on ceiling pictures	SLAM with Northstar, map-building
Navigation/coverage	Systematic	Systematic	Systematic	Systematic	Systematic
Cleaning technology	Brush roller, 2 side brushes	Brush roller, 2 side brushes	Bristled brush	Brush roller, 2 side brushes, micro-fiber mop	Dry or wet (or pre-moistened) microfiber cloths
Run time (min)	75	90	90	70	180
Performance	–	80 m^2/h	–	40 m^2/h	–
Docking/recharging station	Yes	Yes	Yes	Yes	No
Size (Ø/h) (cm)/(cm)	36/9	35/8	34× 34/10	35/10	31× 11/ 29
Weight (kg)	3,2	3,2	5	4	2
Year of launch	2011	2011	2012	2011	2010
Price range	US$ 799.00 (Amazon.com)	EUR 349.00 (Amazon.de)	US$ 350.00 (Amazon.com)	EUR 499.00 (Amazon.de)	US$ 174.00 (Amazon.com)

in an unknown domestic workspace and systematically cover that space:

- Visual odometry, with a camera pointing to the ceiling, combined with simultaneous localization and mapping (SLAM) used by LG Homebot, Samsung Navibot SR 8xxx, iclebo smart and Philips Homerun,
- Localization, using infra read patterns projected to the ceiling, and mapping based on contact information used by Mint,
- Proprioceptive motion estimation, using inexpensive inertial measurement units (including inexpensive gyroscopes and accelerometers)
- Simultaneous localization and mapping (SLAM), using, for example a one-dimensional laser rangefinder like Neato XV-xx.

These technologies by themselves are not entirely new. As a matter of fact methods such as simultaneous localization and mapping, visual odometry, and virtual landmark based navigation are not state of the technologies form a scientific point of view.

The true achievement of the developers of the above systems, which cannot be assessed high enough, is that they managed to reduce the cost of these technologies and at the same time make them robust enough for 24/7 operation. These key technologies will be described in

more detail in Sect. 65.1.2 Enabling Technologies. A review of domestic vacuum cleaning robots and some criteria to evaluate their performance are presented in VIDEO 727 and VIDEO 729.

65.1.2 Domestic Window Cleaning

Window cleaning does not seem to be significantly more pleasant housework as floor cleaning. Notwithstanding, robotic window cleaners have not experienced similar attention or progress as floor cleaners. As a matter of fact, today there are only three commercial robotic window cleaners on the market, shown in Fig. 65.5. www.made-in-china.com does not even list one entry for *window cleaning robot*, although two of the three commercial products are Chinese brands. Why is that so, given the fact that domestic floor cleaning has become a billion-dollar business? The answer to this question has two parts: an economic and a technical one.

The economic one is that windows in private homes are cleaned far less often than floors. Customers may hesitate to buy expensive equipment for a task that needs to be done once a month or even less. So the market for domestic window cleaners is presumably much smaller than that for floor cleaners.

The technical one refers to the technical hurdles that have to be overcome. The technical problems that robotic window cleaners have to face are:

- Adhesion to a vertical, fragile surface, which possibly needs to be moistened in order to be cleaned, and related, the power supply necessary to produce that adhesion.
- Requirements for cleaning performance; people may tolerate if the floor is not 100% clean, but nobody would buy a window cleaner that leaves streaks on the window.

While floor cleaners do not bother about gravity and falling down unless they are near staircases or ledges, gravity is an essential problem for window-cleaning robots, and the solutions are usually not very cheap. Special mechanisms have to be designed for secure motion. Typically special tether mechanisms prevent the robots from falling. Special locomotion mechanisms have to create enough adhesion force to hold a robot attached to a flat, vertical, damageable surface such as glass and at the same time move the body up and down and sideways. These mechanisms have to be small and light and create enough adhesion forces as well as low energy and resource consumption.

The two robotic window cleaners shown in Fig. 65.6, RACOON and QUIRL, are research prototypes, which were developed by Fraunhofer Institute for Manufacturing Engineering and Automation (IPA) in Germany. In these prototypes the adhesion problem was solved by means of suction cups. RACOON used caterpillar drives that were equipped with passive suction cups. Passive means that the system does not actively create a vacuum in the cup. Rather a small valve aerates or seals the suction cup depending on the position of the cup along the drive. Drives with passive suction cups have the advantage of moderate energy consumption. They have, however, one severe disadvantage. They tend to lose their adhesion after a while. The reason for this is the torque that acts on the center of gravity of the system. Due to this torque there is a traction force acting on the upper cups while at the same time pressure is exerted on the lower cups. Without any attractive force acting on the upper cups the adhesion there gets weaker and eventually the system falls. Therefore passive suction cups are rarely practical.

An apparent solution to this problem is the use of active suction pumps, which generate a vacuum under the upper suction cups. This solution prevents the system from falling. However, supplying the vacuum to the cups makes the system significantly more complex, heavier, and larger. Researchers at Fraunhofer IPA [65.2] have therefore invented a smart solution, which gets by with passive cups, but gets around the problem of decreasing adhesion. The solution uses a spacer at the rear of the vehicle. This spacer neutralizes the torque around the center of gravity which is typical for a systems with passive suctions cups. The spacer causes a traction force which acts on the lower suction cups. This traction force creates a torque around the spacer, which counteracts the torque around the center of gravity and also causes a pressure on the upper suction cubs. RACOON was presented at the Hannover Fair in 2002 and got quite some attention. But it never became a product. Neither became its successor QUIRL.

In QUIRL, the number of components, the weight, and the size of the system were significantly optimized. The main functions *cleaning*, *holding*, and *moving* were unified in one single component. QUIRL consisted of

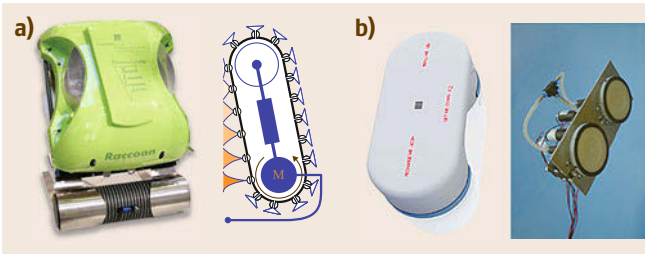


Fig. 65.6 **(a)** RACOON and **(b)** QUIRL, two early research prototypes of domestic window cleaning robots

Fig.65.5a–c Commercial robotics window cleaners (**a**) Windoro, (**b**) Winbot, (**c**) Hobot

two vacuum cups that were attached to a common frame which were driven by two separate motors, which rotated independently of each other. The overall motion of QUIRL could be controlled by selecting the velocities and rotational directions of the vacuum cups. If the motor of one vacuum cup was turned off and the cup did not rotate, QUIRL rotated around this fixed cup. If both motors and cups rotated in the same direction this led to an overall rotation of QUIRL about its vertical axis. If both drives rotated in the opposite direction at exactly the same velocity then QUIRL made a linear motion. If both drives rotated in the opposite direction but their velocities were not identical then the translational motion was superimposed by a rotational motion and QUIRL moved along a curved trajectory. In order to clean the surface some cleaning mechanism or tool needed to be fixed to the vacuum cups. By attaching, for example, specific cleaning towels in the cups the abrasion effect was increased and a very good cleaning performance could be achieved.

The field of robotic window cleaning has not made as much progress as domestic floor cleaning, but it has still made progress. In Fig. 65.5 we show three commercially available window cleaners – Windoro WCR I001, Winbot 7, and Hobot 168 – available today.

In 2010 the South-Korean enterprise Ilshim Global Co. Ltd. introduced its commercial window cleaner Windoro (WCR). Windoro consists of two modules, a navigator module and a cleaning module that are held together by two strong neodymium permanent magnets, whose distance can be adjusted. The navigator module and the cleaning module are placed on the inner and outer side of the window, respectively, like a sandwich with the glass pane in between. The two modules operate as a tandem. The navigator module has a differential drive system with two wide-based rubber wheels. Also the cleaning unit moves on two rubber wheels, which serve as spacers for the four spinning cleaning pads. When the navigator module moves, the cleaning module on the other side of the window moves with it. The linkage by permanent magnets obviously has one rather fundamental advantage. As long as the window does not exceed a certain thickness, namely 28 mm, the permanent magnets hold the robot safely attached to the window. Windoro cannot fall, even if its battery runs out of power. For that reason Windoro does not need any safety mechanism like a safety rope. Unfortunately there is a price to pay for this: it is very difficult and unhandy if not impossible to use Windoro if you cannot open the window that you want to clean.

Windoro first explores the width and height of the window before it then starts to clean the window from the top to the bottom in a zig-zag motion, with a velocity of 8 cm/s. While the robot moves over the window the cleaning module sprays a cleaning solution onto the window surface. With four spinning microfiber pads the cleaning module removes dirt and sprayed solution from the surface.

The second commercial window-cleaning robot is Winbot, a product of the Chinese company Ecovacs, which has established its own brand. Winbot was first introduced in 2011. The most recent version Winbot 7 was presented at the electronics fair CES (Consumer Electronics Show) 2013 in Las Vegas. Ecovacs also manufactures and distributes the robotic vacuum cleaner family Deebot.

Winbot uses two suction rings and a vacuum pump for the adhesion at the window. The outer ring also serves as a safety mechanism. If the air pressure in the ring increases, that means Winbot has reached the edge of a window plane, it backs up, turns around and moves in a different direction. A second suction cup serves as a safety anchor for Winbot while it moves. It is connected to Winbot by a safety rope and catches the robot if it falls. Winbot can move at a velocity of approximately 15 cm/s. Its drive system consists of two differentially driven anti-slip rubber tracks.

After it has been turned on, Winbot, like Windoro, first explores the height and width of the window, then calculates a zig-zag path that covers the window area and finally executes this path. Winbot has no *active* cleaning technology such as spinning pads. It uses two micro-fiber towels that are attached to two plates at the front and at the rear of the robot. The micro-fiber towel in the front has to be moistened before Winbot starts moving. It resolves and removes the dirt. A rubber blade behind the front towel removes the remaining moisture.

Fig. 65.7 Robotics pool cleaners

The micro-fiber towel in the rear of the robot finally dries the window.

The third, and most recent, system, Hobot 168, is a product of the Taiwanese company Hobot Technology Inc. It was first presented at IFA 2012 (Internationale Funk Ausstellung) in Berlin and launched in summer 2013. It looks surprisingly similar to Quirl in Fig. 65.5 and apparently uses a similar adhesion and locomotion mechanism, namely two rotating vacuum cubs, to move across the window.

The preceding paragraphs read as if the times are over where windows had to be cleaned by hand. Regrettably this is not the case. Although all three devices reasonably solved the adhesion problem their overall performance is modest. Several tests by housewives and magazines came to the same sobering conclusion: There is a lot of noise and very little cleaning. As a matter of fact, although the navigation problem on a vertical rectangular surface, which is free of obstacles apart from the window frame, seems to be a solvable one, Hobot 168 and Winbot showed a rather poor performance in terms of systematic coverage. After a semisuccessful effort to explore the width and height of the window, Winbot, in one of the tests, moved more or less erratically for several minutes, before it gave up somewhere along the road. Several of the currently available commercial window cleaning robots are reviewed in VIDEO 734, VIDEO 735, VIDEO 736, and VIDEO 737.

65.1.3 Pool Cleaning

While robotic floor cleaner and window cleaner were still struggling to get rid of the image of only being the crazy ideas of engineers and researchers at the beginning of the millennium, pool-cleaning robots were already well-established products. This may be due to the fact that the challenge of cleaning a rectangular pool is rather modest and so is the *robotic technology* used in robotic pool cleaners. It may also be due to the fact that pool owners belong to a class of customers who did not create the same price pressure as the ordinary housewife.

First patents on self-propelled pool cleaning devices date back to 1965, three years after the first industrial robot was installed. Ferdinand Chauvier, a South African engineer, could possibly be considered the father of automated pool cleaning. He developed several generations of devices for pool cleaning before he finally marketed *Kreepy Krauly*, the first automated pool cleaner in 1974. Kreepy Krauly was not only the first one of its kind but also the very first domestic service robot ever, 15 years before Joe Engelberg published his book *Robots in Service* at MIT Press in 1989 and coined the term *service robot*. Since then the technology of pool cleaning robots hasn't changed much.

As one can see in Fig. 65.7, most pool cleaners have track drives, which are operated differentially. Typically the motors that drive the tracks are also connected to the front and rear scrubbing brushes, which clean the pool surface, while the robot moves. While tracked drive system were rather common for the early generations of pool cleaners, newer pool cleaners also use wheel drives, for example, the Polaris 9400 from Zodiac Pool Systems. A wheel drive can be advantageous because the space between the bottom of the robot and the ground allows a better water flow and a higher throughput. Also Zodic claims that Polaris 9400 has higher maneuverability.

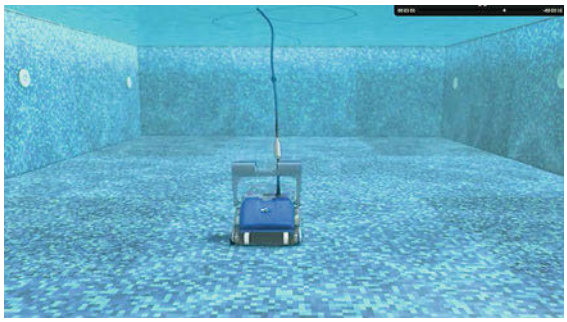


Fig. 65.8 Example of the navigation and cleaning strategy of a robotic pool cleaner

Since most sensors, which are used in mobile robots or in aerial robots do not work under water, the sensor modalities that are used by pool cleaners are manageable. Unfortunately the manufacturers do not disclose much about the sensor technology and technology in general used in their system. So we need to speculate a bit about how the behaviors shown in the commercials are internally implemented.

Most pool cleaners claim to be capable of avoiding obstacles. When they sense a collision, the back up for a certain distance, reverse the direction of motion and then continue, possibly on a track parallel to the one the led to the obstacle. They can *recognize* the walls of a pool, which form the borders of their workspace. They can even climb up the walls of a pool, float along the perimeter and then submerge again. They can *explore* the length and width of a pool. They can drive a certain distance before they change direction.

All these behaviors require a combination of several of the following sensors: *odometry* to measure traveled distance, an *inclinometer* to sense if the robot starts moving upward, e.g., when it keeps pushing against a wall and the front wheels start moving in a vertical direction, *contact sensors* to detect when the robot collides with an obstacle – be it a wall of the pool or a real obstacle on the ground of the pool – *sensor to measure the motor current*, which can be used in addition or instead of a collision sensor to detect if the robot pushes against an object, and possibly an *inertial measurement unit* to correct the heading when driving. Polaris and possibly other systems as well use an accelerometer to constantly determine its position in the pool. A sensor modality, which might be used underwater for obstacle detection and avoidance is laser, with wavelengths in the lower nano-meter range (e.g., 405 nm blue laser). But it is unknown if this principle has been considered for pool-cleaners.

In terms of *navigation and coverage* pool cleaners follow similar strategies to those floor cleaners, which we roughly classified as Roomba-clones. Earlier pool cleaners used random motion pattern. The newer ones shown in Fig. 65.7 use certain heuristics and strategies to perform some form of localization and exploration of the pool. The Maytronic's Dolphin, for example, first explores the length and the width of a pool. After being dropped into the pool and floating to the bottom of the floor it crosses the pool until it hits the first wall. With the support of a thruster it climbs up the wall until it reaches the surface. There it hovers to the side before it submerges again and moves back to the ground of the pool. Next it moves to the opposite side of the pool, climbs up the wall, hovers to the side and glides back to the ground. While moving on the ground from one wall to the other Dolphin measures the distance from wall to wall. Once it glides back from the second wall it moves halfway back toward the other wall. It then makes a 90° turn and repeats the exploration for the second set of opposite walls. After the exploration is completed, Dolphin knows the length and width of the pool and plans a pattern of parallel and orthogonal tracks as shown in Fig. 65.8 that in the end covers the entire pool.

Removing dirt and debris from the ground and the walls of a pool, requires loosening the dirt – if it is not loose already – and to soak it into a container, otherwise the dirt would only be circulated in the pool. The container is typically a jet pipe with a filter at one end that holds the dirt back while the water flows through the pipe back into the pool. Soaking in water and debris from the floor into a filter and pumping it back into the pool requires significant suction power. This is the reason why pool cleaners typically have an external power supply. This motor serves as a pump and as a thruster at the same time. Both effects together allow pool cleaners to easily climb up vertical walls. Using inflation at the bottom of the robot, and a water jet ejected at the top, enough traction power is created that both tracked and wheeled pool cleaners can drive up vertical surfaces.

As mentioned earlier, the cleaning technology, besides the water pump and the filter to keep back the debris, consists of a system of counter-rotating rubber brushes which brush the debris underneath the pool cleaner, where it ends up in the intake socket of the water pump. VIDEO 739 and VIDEO 740 compare a selection of home pool cleaning robots.

65.1.4 Lawn Mowing

Together with robotic pool cleaners and domestic floor cleaners, robotic lawnmowers today count as regular everyday products. People no longer consider them mystical pieces of technology, which have their own life and which at times become so autonomous that the user no longer knows what they are after.

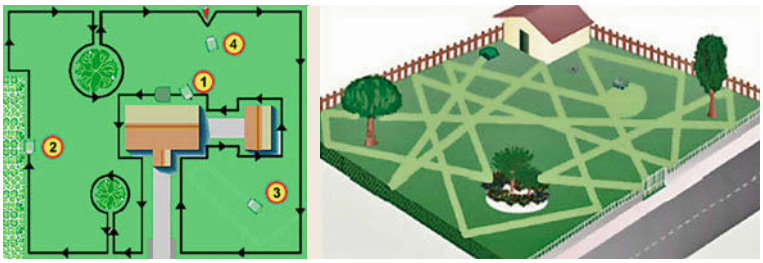


Fig. 65.9 Example for virtual fence and coverage strategy (random motion) of robotics lawnmowers

It is not surprising that robotic lawnmowers have a lot in common with domestic floor cleaners. They have to cover a workspace of a certain size with as little interaction with the owner as possible. They have to perform a certain operation to the surface such as cleaning the floor or mowing the lawn. They must not collide with any obstacles and if they do, they should at least not cause any damage. They must not get stuck anywhere in the environment and they should not leave the workspace without authorization.

The challenge for lawnmowers, much the same for domestic floor cleaners, is the systematic coverage of the workspace, which in turn requires precise positioning and mapping of the workspace. Given the fact that the price pressure is not as back-breaking as for robotic floor cleaners – there are not that many robotic lawnmowers which costs less than 1000 EUR – why not invest a little more in sensing and especially in position sensing and obtain a decent solution for the localization and coverage problem?

The answer to this question is: things are not that simple. Lawnmowers operate outdoors and none of the solution developed for the floor-cleaners in Fig. 65.2 will work. Regular GPS (global positioning system) has an accuracy of several meters and has a tendency to deliver erratic readings, which would lead to equally erratic motions of the lawnmowers. As a matter of fact Automower 220 from Husquana uses GPS, but only as an antitheft protection device. Differential GPS, which would provide accuracy below one meter, would be too expensive for an affordable robotics lawnmower. In a nutshell, absolute positioning or SLAM, which would be necessary to cover a large outdoor area like a garden in a systematic fashion, is not practical.

In order to cover their workspace robotics lawnmowers use similar strategies to the Roomba-like floor cleaners. They refer to heuristics, which do not provide an optimal performance but still show a decent result. Using sensors such as gyroscopes, digital compasses, or inertial measurement units, robotic lawnmowers follow a certain heading and cover the workspace as much as possible by parallel tracks. They move along a straight line until they hit an obstacle or reach the border of the workspace. There they back up to become clear from the obstacle, make a U-turn by 180° and drive back the way they came. Another heuristic that is frequently applied by robotic lawnmowers is the random motion shown also in Fig. 65.9. The mower moves along a straight line until it hits an obstacle or reaches the border of the workspace but then it does not just reverse but chooses a new direction randomly.

Since robotic lawnmowers move in open space, there is a danger that they leave their workspace and travel to areas where they are not supposed to be. For floor cleaners so-called virtual walls or fences solve this problem. Virtual walls are realized by infra read light beams emitted by some projectors, which can be placed in the workspace. The robots can sense these infrared light beams and consider them as obstacles, which evoke the typical obstacle avoidance behaviors. Lawnmowers use a similar technique, which is based on induction rather than light. To mark the border of the robot's workspace the owner has to place a wire around the area, which the robot must not leave (Fig. 65.9). This wire is connected to a low-voltage alternating current source. When the robot approaches the wire an inductivity sensor senses the current in the wire and causes the robot to reverse. Today most lawn mowers use a more sophisticated so-called *true in/out systems*, where the position is permanently tracked, not only when the robot approaches or passes the virtual fence [65.3].

The very first robotic lawnmowers had to be recharged manually. To avoid an all too frequent involvement of the human into the operation, Husqvarna – one of the pioneers in robotic lawn mowing – equipped its first mower with solar panels and called it *SolarMower*. SolarMower was released in 1995, and was one of the first robotics lawnmowers. Nowadays, all lawnmowers come with a base station where they can recharge their batteries without human intervention. Also solar panels are coming back as power source, however only as auxiliary power to increase the performance and runtime between two charges and not as main power supply.

Given the fact, that robotic lawnmowers do have poor positioning capabilities, it is somewhat tricky to guide them back to the docking station once the battery

Fig. 65.10 Robotic lawnmower

gets low and needs to be recharged. Some of the lawnmowers shown in Fig. 65.10 use special wires which radiate from the position of the docking station. This way a robot only needs to follow such a radiating wire to return to the docking station on the fastest way. Another strategy is to follow the wire at the workspace border. That will eventually lead the robot back to the docking station.

An exception is the Bosch Indego, which is equipped with mapping and localization capabilities. This allows Indego to plan a path that leads to a point close to the docking station. Then the robot can follow a wire for the docking maneuver [65.1].

To comply with the safety regulations and to avoid any injuries of humans or animals the cutting mechanism has to be very lightweight and designed such that it is guaranteed to stop if the device tilts or is lifted. The cutting mechanism of robotic lawnmowers typically consists of a rotating disc with three razor-like blades, which automatically retract into their mountings if the robot is stopped unscheduled, for example, if it hits an obstacle or is lifted.

Naturally the lightweight design limits the thickness of the grass that can be cut. Also it is important for the proper functioning of robotic lawnmowers that the lawn, which is to be cut, is not too high. This in turn requires a rather regular if not continuous operation of the lawnmowers. With regular use of the lawnmowers the grass cuttings are short enough to quickly decompose into nutritious compost, so there is no need to remove the cuttings after the lawn is mowed.

Figure 65.10 shows some of the better-known robotic lawnmowers on the market today. Besides the pioneers in robotic lawn mowing Husqvarna, Friendly Robotics and Zuchetti also a number of new players have entered the market, most noticeably the German automotive supplier Bosch and the Japanese car manufacturer Honda, which has set a milestone in non-industrial robotics with its humanoid robot Asimo.

We certainly do not claim that the collection in Fig. 65.10 is complete. Like for domestic floor cleaning, Chinese B2B portals such as www.made-in-china.com or www.alibaba.com list around a hundred products under the category of robotic lawnmowers and 35 suppliers; the market development for robotic lawnmowers in the past five years was not quite as overwhelming as for domestic floor cleaners but was still remarkable. In Table 65.3 we show the technical specifications of a selection of lawnmowers from Fig. 65.10.

65.1.5 Sports Robotics

A subdomain of domestic robotics that was not included in the first edition of this handbook, because it virtually did not exist, or it was not visible at that time, is *sports robotics*. What is a sports robot? Since an official definition of this term does not yet exist – at least we haven't found one – we take the liberty and provide such a definition here. We define a sports robot as a *robotic device*, which either *supports* the human user in their physical exercises as a *coach or a companion*, or acts as an opponent in a game. An important aspect of this definition is the physical exercise of the human, which is supported or challenged.

We would like to emphasize that the above definition does not include any form of entertainment robots, which play games such as soccer against each other but do not involve any human activity other than watching.

In Fig. 65.11 we show three examples of *robotic baseball players*. As the name says the *Headless Batsman* can act as a batter at least for exercising. It rather successfully hits baseballs thrown at it in many ways. The kinematic structure consists of *two arms, one leg and no head*, as its developer, an industrial designer from Robocross, likes to call it, and is made from auto parts, steel pipes, and pneumatic hoses. The pneumatic hoses are parts of its pneumatic actuation by an air

Headless Batsman | Pitching Machine | PhillieBot – a robotic pitcher

Fig. 65.11 Robotic baseball players

Table 65.3 Technical specification of a selection of robotic lawnmowers

Manufacturer	Husqvarna	Friendly Robotics	Zuchetti	Bosch	Honda
Model	Automower 220 AC	Robomow RL1000	Ambroggio	Indego	Miimo
CPU				ARM9 (32 bit) + PowerPC (32 bit)	
Drive	Differential	Differential	Differential 4 driving wheels	Differential	Differential
Sensors	Lift sensor collision/contact sensor	Rain sensor, lift sensor, collision/contact sensor	Grass sensors, safety sensors on handles, lift sensor, sensor for collision sensor and rollover	Lift sensor, collision/contact sensor, gyro, tilt sensor	3 × 360 degree bump sensors, 2 lift sensors
Navigation/coverage	Random	Random		Logicut navigation system: parallel tracks in parcels; localization, mapping, path planning	Random, parallel or mixed
Cutting technology	3 pivoting razor blades	3 razor blades	Rotating disc	3 pivoting blades	3 blades, which bend on hitting hard objects
Virtual fence and guides	Dual guide wires to return to the base station	Boundary wire	Boundary wire	Boundary wire	Boundary wire
Performance (m^2)	1800 (m^2)	2000 (m^2)	400 (m^2)	1000 (m^2)	3000 (m^2)
Ground speed			18 m/min	27 m/min	
Motor power (W)		3 × 150		2 × 20 + 80	2 × 25 + 56
Battery type and capacity	NiMH 30 W		Li-Ion, 6.9Ah	32 V Li-Ion, 3Ah	Li-Ion
Run time	45 min	2 h 30 min	3 h 30 min	50 min	
Docking/recharging station	Automatic docking and recharching		No	Automatic docking and recharching	Automatic docking and recharching
Theft protection	Pin code lock	Personal password		Alarm & Pin code	Pin code lock
Size (L×W×H)	71 × 55 × 30	87, 5 × 65 × 31			
Weight (kg)	10	25	7,9	11.1	
Year of launch				2012	
Price range	∼$ 2400,00	$ 1900,00		∼$ 1850,00	∼$ 2600,00

compressor. The *Headless Batsman* is fully controlled by a human operator via a remote control, which has three buttons: one to control the robot's hip, a second one that actuates the arms and a third one to lift

JoggoBot – a jogging companion RUFUS – a robotic running coach

Fig. 65.12 (a) JoggoBot and (b) RUFUS two robotic sports companions

and drop the inside shoulder, changing the trajectory of its swing. Similar mechanisms like *Headless Batsman* were developed at Hiroshima University and Tokyo University.

The counterpart of *Headless Batsman* is a pitching machine, which is operated with the same remote control. The barrel of the machine is sawn-off fire extinguisher whose other end is directly connected to a one-inch port poppet valve, which in turn is connected to an 8 bar air pressure tank. The air pressure tank is supplied from a screw compressor. The charging mechanism consists of a double rod actuator with a ring welded horizontally on to the end. Mounted above the ring is a magazine that holds up to 10 balls, which directly drop into the ring. For a pitch the actuator moves the ring with the ball over the edge of the barrel, where it falls on to a small guide. From there the balls rolls backward into the barrel. The actuated ring is covered by a steel strip, which holds back the other balls in the magazine until the actuated ring returns to its initial position. Since both mechanisms are remotely controlled, the human, who controls them, can actually sit in a chair next to the playground and let the robots play against each other. Such a use would clearly violate the above definition of a sports robot but does not seem to be the one that is primarily intended.

Another robot that can throw baseballs is Phillibot (VIDEO 748) developed in University of Pennsylvania's GRASP laboratory. Phillibot was developed in a couple of weeks using only spare parts in the GRASP lab. It uses a Segway as mobile base, a Barret arm, and pneumatically actuated wrist to create the necessary dynamics for the pitch. When the button is pressed the arm moves to the back of the robot and then accelerates its motion toward the target of its pitch. When the arm reaches the highest point of motion the pneumatic wrist cylinder delivers a burst of compressed carbon dioxide to snap the wrist forward and release the ball. What remains is the question of why use robotic equipment worth several tens of thousands of dollars to throw a baseball, given that pitching machines have existed for many years. According to the developers, the fact that Phillibot is mobile and its software can be tweaked to vary pitch velocity and trajectory was enough to justify the experimentation.

The two sports robots shown in Fig. 65.12 do not really seduce their users to sit in a chair and relax. They both serve as so-called *robotic running coaches* or *running companions*.

Researchers at the Royal Melbourne Institute of Technology in Australia have redesigned a commercially available Parrot AR Drone quadrocopter and turned it into an autonomous, flying running partner for joggers, called Joggobot [65.4]. Joggobot uses an integrated, front-mounted camera to detect and track a special patter printed on the T-shirt worn by the jogger. Joggobot takes off when the camera registers the pattern and rises to about the same height as the pattern on the t-shirt. An internal sensor determines Joggobot's altitude. Joggobot can be set into a *companion mode*, in which at flies at a steady pace at a relative distance of about three meters to the jogger or in a *coach mode* in which it flies at a slightly more challenging speed.

There are two features of Joggobot that make the device somewhat limited: first the capacity of the battery limits the flight time to 20 min, which in turns limits the time for exercise; for a short run, this is certainly ok, but for serious training this is insufficient. Second, Joggobot can only fly in a straight line, to let Joggobot follow an arbitrary path the jogger needs to remotely control Joggobot's flight path.

A slightly different concept of a jogging companion is pursued in RUFUS (VIDEO 747), which is developed by runfun (www.runfun.com), a German startup company. RUFUS is an electrically driven, automatically guided, ground vehicle that supports and guides a runner during his/her training. RUFUS plays the role of a *personal running coach*. It fulfills a similar function to a treadmill, which exposes its user to a varying strain by varying its velocity and inclination and thereby improves the fitness, endurance, and resilience of the cardio-vascular system of the user. Unlike a treadmill, RUFUS is not a stationary device, however. It drives ahead of the runner like a pace maker in a marathon or a fake rabbit in a dog race, and sets the runner's speed. RUFUS's velocity is set either manually or automatically via a training program.

If operated in *manual mode* the velocity is either set directly as a velocity set point or indirectly as a heartbeat set point. If the training guidance is based on the heartbeat, then RUFUS controls its velocity such that the runner exercises optimally and continuously within a certain heartbeat interval under a moderate stain of the cardiovascular system.

Fig. 65.13 Tele-presence robots and robotic avatars

This has a twofold use: on the one hand this prevents users from overstressing themselves through overambitious and intensive training modules, possibly from even injuring him or herself. Such a protection function is beneficial for unfit or untrained runners. On the other hand RUFUS facilitates optimal training effect and progress through a careful guidance of the training.

The training effect can be further improved if RUFUS is operated in the *program mode* instead of the manual mode. In this mode RUFUS executes complete training modules, for example pyramid speed interval workouts, which are customized to the user. Such training modules are typically elaborated on by physiotherapists or sports physicians. They can be downloaded to the RUFUS embedded PC like an app from an app store.

RUFUS has a major advantage over JoggoBot. It has a battery capacity that allows it driving for about six hours on a flat road without recharging.

65.1.6 Tele-Presence

In a world in which not only large, but even small and medium-sized enterprises operate globally, in which families are scattered over continents, in which ubiquitous presence seems to become an essential requirement for professional progress, and in which professional services are more and more delivered over the internet, tele-presence has become a fast growing market in the past years.

Robotics adds a very important aspect to plain *television* by turning it into *tele-presence*: embodiment and remotely controlled motion. As tele-presence is nothing but the combination of tele-vision and tele-operation using a robotic device, which are often called *tele-presence robots* or *robotic avatars*.

Tele-presence robots offer a whole spectrum of services and applications ranging from plain mobile video-conferencing systems to tele-surveillance, tele-diagnosing, and tele-care, to tele-teaching and tele-commuting. The term *tele-commuting* was coined by Scott Hassan, a Google developer of the early days, nowadays entrepreneur and investor, and founder of WillowGarage and Suitable Technologies, the manufacturer of Beam-RPD (see also Table 65.4).

A *tele-presence robot* typically consists of a mobile robotic platform that:

- Can be tele-operated through some user-interface,
- Carries a camera, which often can be actuated separately (via a pan-tilt unit) and allow the operator to actively explore the remote environment, and
- Carries a display, which allows those at the remote site to see the operator of the tele-presence system and communicate and interact with it.

Figure 65.13 shows a collection of such tele-presence robots. The devices range from a price of

Table 65.4 Technical specification of a selection of tele-presence robots

Manufacturer	Double Robotics	Vgo Communications	Gostai	Suitable Technologies
Model	Double	VGo	Jazz Connect	Beam RPD
Height	120–150 cm	120 cm	100.5 cm	5 feet, 2 inch
Weight	7 kg	9 kg	8 kg	45 kg
Screen size	9, 7″ (iPad)	6″	5″	17″
Camera/field of view			High resolution/ wide angle, up to 640 × 480 pixels at 25 fps	Two wide-angle HD cameras
Video conferencing	Open-tok	VGo video conf.	2-way audio and video for remote discussions	–
Network	WiFi	WiFi / 4G / LTE	WiFi	WiFi (two dual-band radios) / 4G
Remote Control	iPad App	VGo App	Intuitive control interface on internet browser	Beam software client, mouse, keyboard, or Xbox controller
Navigation	Tele-operated	Tele-operated	Tele-operated with obstacle detection	Tele-operated
Sensors	Gyroscope, accelerometer	Obstacle and cliff detection sensors	12 ultrasonic sensors, 4 IR receivers (for base docking), telemetric laser for autonomous navigation	–
Drive	Differential (10″ wheels)	Differential	Differential	Differential
Battery	Lithium ion	–	–	–
Run time (h)	8	12	5	8
Docking station	–	Yes auto-docking	Yes auto-docking	Yes

some US$ 1500 for TeleMe from MantaroBot to more than ten-fold that amount for Beam RPD from Suitable Technologies. The system RP-VITA (Remote Presence Virtual + Independent Telemedicine Assistant) emerging from a cooperation between InTouch Health and iRobot is available only for lease, at a monthly fee of $ 4700. Giraff is the result of a European research project lead by Giraff Technologies funded by the European Commission and is not commercially available yet.

Not all tele-presence systems shown in Fig. 65.13 can be classified as domestic robots. A system that clearly stands out and is by no means a domestic robot is *RP-VITA* by InTouch Health and iRobot. RP-VITA is a remote healthcare system. RP-VITA shall enable doctors to command any clinical, patient or care team management process remotely. RP-VITA has a full-fledged autonomous navigation system that allows the personnel to focus on the patient care task rather than on remote navigation. This feature has been awarded clearance by the US Food and Drug Association (FDA). RP-VITA further provides access to important clinical data to support physicians, nurses and other care personnel in their diagnosis and other medical workflow. For example RP-VITA connects with diagnostic devices such as ultrasound and comes equipped with the latest electronic stethoscope. So RP-VITA is in a class of its own, which may also justify the higher price.

Apart from RP-VITA, the tele-presence robots shown in Fig. 65.13, can all be classified as semiprofessional or domestic service robots. The functionalities and services they offer do not necessarily vary on the same scale as their prices. This can be seen by a comparison of two of the above systems: Double and Beam RPD (see also Table 65.4).

Double is not much more than a mobile iPad equipped with the video-conferencing system Opentok. The mobile base uses a Segway-like dual-wheeled drive system that can balance a pole, which holds the iPad. When Double stands still, two retractable kickstands are deployed and allow the system to put the control system in an idle mode and save energy. Double can

be remotely controlled and driven around a remote site through an app installed on a second iPad that enables communication with all known Doubles over the web. The height of the iPad holder can be remotely adjusted to enable communication at eye-level. Double Robotics list a number of potential services and applications for which Double could be used: Companies with sites at various locations can us Double to improve communication and collaboration between remote teams. Families can use Double to communicate with family members living abroad. Museums and art galleries can use Doubles to offer remote tours through their exhibitions.

Beam RPD uses two HD cameras with custom wide-angle lenses instead of the plain iPad camera. This gives Beam RPD peripheral vision that is comparable to a human's field of view. A digital zoom lets the operator further focus on details in the remote site. Beam uses an array of six microphones and audio processing algorithms, background noise reduction and echo cancellation. This equipment gives Beam an audio-quality, which obviously goes far beyond that of an iPad. Beam uses a 17-inch screen mounted at a height of 1.58 m that allows the display of a human face at its natural size and height. Another feature that goes beyond iPad standards is the WiFi connectivity. To provide reliable and seamless WiFi connectivity, Beam uses two dual-band radios and proprietary roaming algorithms. Altogether it is obvious that Beam RPD is far more than a movable iPad. It is left to the customers to decide whether this is worth a price which is an order of magnitude higher. VIDEO 741, VIDEO 742, VIDEO 744, and VIDEO 745 introduce several of the tele-presence robots available on the market today.

65.2 Enabling Technologies

The mass consumer market is very price-sensitive, so the price of the robot is key for the success of the product among consumers. Certain guidelines used in the consumer electronics market are relevant for the domestic robot market to provide a rough estimate of cost of the robot. Let's say that you want to develop a floor-care robot that would retail at $ 300, the *rule of thumb* is that your bill of materials (BOM) should be between $1/3-1/5$ of the retail price. In other words, your BOM should be within $ 60–$ 100! And the BOM must include all mechanical parts, electrical parts, battery, processor, memory, motors, assembly, packaging, user manuals, etc.

Given the extreme cost constraints outlined above, this chapter focuses on enabling technologies that are viable, from the cost point of view, to be included in a mobile domestic robot with a price lower than $ 1000 (or ideally below $ 500). These technologies are required to have a reliability level in line with the expected life time (and warranty) of the product; otherwise, no matter how good the technology is, if it stops working in an unreasonable period of time, the robot will be returned to the retailer. Special emphasis should be placed in the ease of manufacture of the technology. Difficult to manufacture components create delays in the production line, decreasing the yield of the product and eventually increasing the overall cost of production, leading to either eroding profit margins or a rise in the retail price.

Mobile robots need to sense and understand the environment in which they operate. The first key enabler is the capability of detecting obstacles and hazards to safely and accurate navigate around them. Section 65.2.1 describes the different available technologies for obstacle and hazard detection. The second key enabler is the ability to localize and create a map of the environment to intelligently plan actions and motions that allow the robot achieve its goal. Section 65.2.2 presents the technologies available for localization and mapping using a number of low-cost, yet powerful sensors. Section 65.2.3 discusses alternative approaches to coverage of the space implemented in commercial products.

65.2.1 Sensing and Obstacle Avoidance

Domestic robots aim to take care of tedious chores, interacting with a household that includes owners, children, babies, pets, and stationary objects such chairs, tables, walls, etc. Domestic robots must be safe in order to gain acceptance in our daily life: it is not tolerable to have a robot falling down the stairs or hurting a household member. Thus, robots must be equipped with drop/cliff sensors and proximity sensor that ensure proper operation while still satisfying the mentioned cost constraints.

Cliff Sensors

A number of solutions are present in current robots in the market. Off-the-shelf solutions are IR sensors from Sharp that consists of an emitter (light emitting diode LED) and a receiver (photodetector or position-sensitive-device PSD) that provides an output proportional to the distance to the object. The Roomba uses

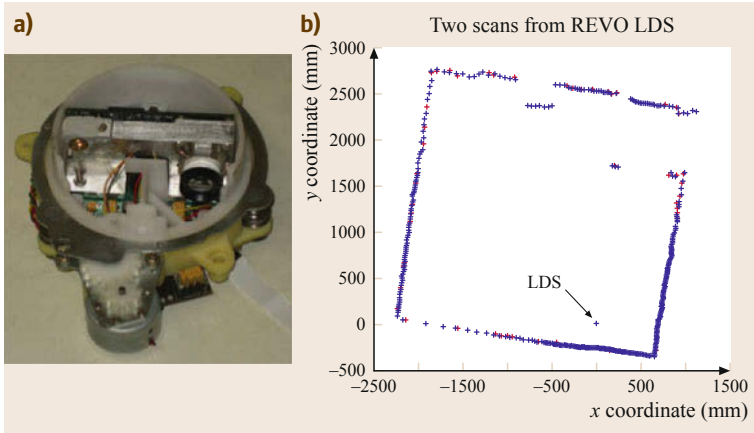


Fig.65.14a,b Laser Distance Sensor. **(a)** Prototype and **(b)** occupancy map generated with the sensor

a custom IR cliff sensor based on a similar principle that trades generic distance measurement with sensor cost. The Mint robot from Evolution Robotics employs a factory-calibrated mechanical hammer that triggers upon cliff detection. Solid-state sensors are usually more reliable that mechanical sensors, since they do not have moving parts, but have the drawback of a response dependent on the reflectivity of the surface in the IR spectrum and a dead-band in the response.

Contact and Proximity Sensors

Mechanical switches, called *bump sensors*, are commonly used for detecting when the robot gets in contact with obstacles. Bump sensors are cost-effective solutions providing the ability to stop the robot without damaging the obstacle. Touching obstacles is not desired unless performed very gently to ensure that the robot goes under curtains and bed skirts. IR and sonar sensors are frequently employed as touchless alternatives to the bump sensors by measuring the distance to obstacles. Both types of sensors are composed of an emitter and a receptor with an output proportional to the measured distance. IR sensors are usually more focused than sonars and less sensitive to multiple reflections on walls and other obstacles, but might lose thin obstacles such as chair legs. This type of sensors provides a point-wise measurement of distance to obstacles, so a robot needs a number of these sensors to obtain a dense representation of the obstacles in the environment. The information on obstacles and hazards (cliffs) is collected in occupancy grid maps and used for decision-making in systematic cleaning robots. A number of cost-effective dense distance measuring sensors have recently appeared in the market and will be discussed in the next sections. These dense distance measuring sensors have a cost on the order of tens of dollars, while the point distance measuring sensors have cost only a few dollars, so most of the domestic robots currently available on the market have yet to incorporate dense distance measuring sensors. The only exception is the Neato XV-21 that uses a low-cost laser distance measuring system.

Laser Distance Sensor

A low-cost Laser Distance Sensor was developed by Konolige and colleagues [65.5] using a laser point beam and a global shutter CMOS imaging sensor separated by a small baseline. The system operates by triangulation and achieves full 360 planar scan by rotating the optical assembly on a full circle. The sensor has a range of 0.2 to 6 m with an error < 3 cm at 6 m and an angular resolution of 1 degree, providing 4000 readings per second (up to 10 Hz) with a small size (approximate width of 10 cm shown in Fig. 65.14a) and low power (< 2 W). The sensor is eye safe and provides measurements that enable laser-based SLAM as shown by Fig. 65.14b.

Structured-Light Distance Sensors

Structured-light distance sensors consist of an emitter that projects a known pattern on the environment and a receptor that computes depth based on the deformation of the received pattern. The Kinect [65.6] interface to the Xbox game system uses a structured-light sensor from PrimeSense [65.7] thus showcasing the readiness of this sensor for consumer applications. The emitter consists of a laser with optics that projects a known pattern (Speckles [65.8, 9]) in near-IR light and a complementary metal-oxide-semiconductor (CMOS) IR camera that observes the pattern to estimate depth using triangulation. The emitter and the camera are calibrated during manufacturing assuming a rigid configuration. The speckles can be further shaped into ellipsis using optics with different focal lengths in x and y so that the orientation of the observed ellipsis is proportional to depth. Speckles of different sizes are used to obtain different depth accuracy depending on size. Figure 65.15a

shows the speckles pattern projected by the sensor and an image taken in the dark by photographer Audrey Penven showing the IR speckles. Figure 65.15b shows the components of the Kinect sensor and the corresponding RGB and depth images of a scene.

Time of Flight (TOF) Distance Sensors

Time of flight (TOF) sensors consist of a light source (usually a laser) that emits a continuous waveform and a special imaging sensor that measures the phase shift of the received signal in each pixel. The depth at each pixel is proportional to the phase shift. TOF sensors have been in the market for quite some time, but the need for allocating a large portion of the sensor to the decoding electronics has made it challenging to produce low-cost sensors at reasonable resolution. Some of the companies offering TOF sensor have been Mesa Imaging AG that produces the SwissRanger [65.10] sensors, Softkinetics [65.11], PMDVision.

Stereo Vision

Stereo vision is a well-known computer vision solution to the extraction of three-dimensional (3-D) depth maps in areas with sufficient texture to find image correspondences. As opposed to structured light or TOF sensors, stereo vision systems are totally passive, but require a calibrated stereo rig, and their performance depends on the level of external illumination and on the amount of texture present in images.

The selection of the optimal dense mapping sensor depends on the application. The laser distance sensor provides reasonable information for laser-based SLAM and obstacle detection and avoidance; however, it only provides range information on a plane as opposed to the dense 3-D range offered by the structured light, TOF, or stereo systems. The structured light sensor uses a simple imaging sensor but requires additional computation to estimate depth in each pixel while the TOF sensor computes depth in each pixel at the expense of a sensor with lower fill factor. The stereo system does not require additional lighting, but requires an additional camera and a computation module to extract depth. Other parameters to consider are the maximum and minimum range that the sensor provides to ensure that it fits the requirements of the application in terms of mapping and obstacle detection.

65.2.2 Localization and Mapping

A robot that knows its location and understands its surroundings is able to plan intelligent maneuvers to achieve its goals. Localization and mapping are basic primitives that enable smart and efficient behavior. Early successful robots like the Roomba chose to sacrifice localization and mapping in order to achieve an appealing retail price since the localization and mapping technology was either too expensive or not existent at the time (many of the dense sensors presented in the previous section were developed after the Roomba). In recent years, a number of low-cost, yet powerful simultaneous localization and mapping (SLAM) technologies have been developed and integrated in floor-care products. These technologies are described in the following sections.

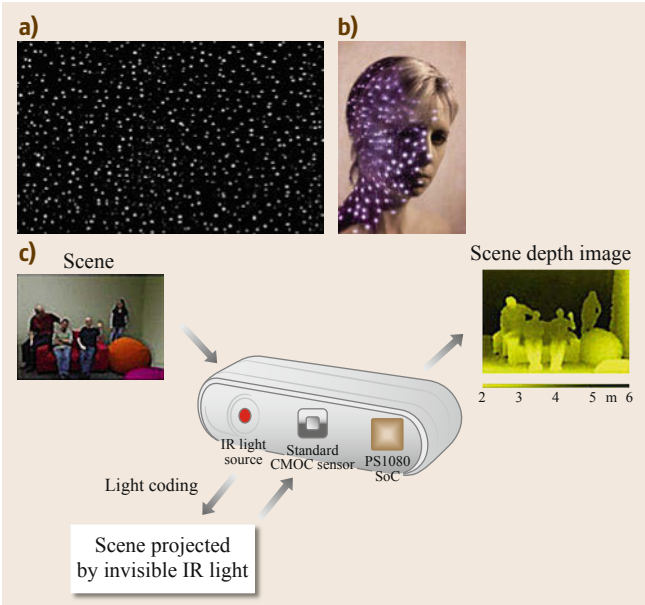


Fig.65.15a–c Primesense sensor: (a,b) Speckles pattern projected by the laser emitter. (c) Block diagram of the Kinect sensor

Vector Field SLAM

The Mint robot by Evolution Robotics uses active beacons for localization. The Northstar beacon projects two IR spots onto the ceiling that are modulated to simplify the detection of the spots with the sensor. The spots are invisible to the human eye and thus do not produce visual clutter. While placing the beacon can still be regarded as a modification to the environment, surveys among our customers suggest that the vast majority largely accepts setting up the beacon prior to running the robot [65.12]. The Northstar sensor uses 3 photodiodes to compute the direction to the spots from the measured current through the photodiodes. The photodiodes can be sampled at high frequency to detect the modulated spot frequencies for data association. The sensor is quite inexpensive and suitable for cost-sensitive applications. However, the sensor suffers from multipath since light not only reaches the sensor directly but also through reflections from walls

and other furniture, making the position computed by the sensor inadequate to be directly used for localization and mapping. The sensor is augmented with a localization method that learns the light distribution in the room through a simultaneous localization and mapping (SLAM) approach [65.13]. Figure 65.16 shows the Mint robot cleaning the environment depicting the operation of the Northstar localization system and the problem of multipath. The figure also shows the Mint robot and the Northstar cube and the Northstar sensor.

In vector field SLAM the spatial variation of continuous signals is learned and simultaneously used for localizing the robot by fusing information from dead-reckoning (odometry and gyro) and Northstar. In the following, this method is introduced and tailored toward measurements obtained from Northstar. The signal field is represented as a regular grid of fixed node positions $b_i = (b_{i,x}, b_{i,y})^{\mathrm{T}}$, $i = 1 \ldots N$, where each node $m_i \in R^4$ holds the expected Northstar positions of both spots when placing the robot at b_i and pointing it in a fixed orientation $\theta_0 = 0$. Vector field and robot pose are then estimated through the application of SLAM.

Let the robot path be a time series of poses $x_0 \ldots x_{\mathrm{T}}$, $x_t \in \mathrm{SE}(2)$, i. e., the set of rigid transformations in the horizontal plane, and let $x_0 = (0, 0, 0)^{\mathrm{T}}$. At each time step $t = 1 \ldots T$ the robot receives a motion input u_{t} with covariance R_{t} and a measurement $z_t = (z_{x1}, z_{y1}, z_{x2}, z_{y2})^{\mathrm{T}}$ of the two Northstar spot positions with covariance Q_t. The spot positions are also each affected by the rotational variability denoted $c = (c_x, c_y)^{\mathrm{T}}$. The rotational variability models errors in measuring the direction to the spots caused by not having the sensor perfectly level.

The SLAM problem is solved with the ESEIF-SLAM that is constant time and requires memory linear in the size of the area explored by the robot. The method has been implemented in the processor of Mint, an ARM 7 processor with 64 kByte RAM [65.13]. Vector field SLAM has been extended to address covering larger areas by using more Northstar beacons [65.15] and to provide a solution for the re-localization of the robot after it has been kidnapped or paused and resumed [65.16].

Figure 65.18 presents the map obtained by the robot after a cleaning run in a home environment of 125 m^2 covered by three Northstar beacons marked with the pink and green disks (units are in meters). The robot navigated in the home by following a cleaning strategy based on systematically covering sectors of the environment. As long as at least one beacon is visible to the robot, the strategy moves the robot onto a neighboring region until no space is left to clean. At the end the robot follows along the perimeter of detected obsta-

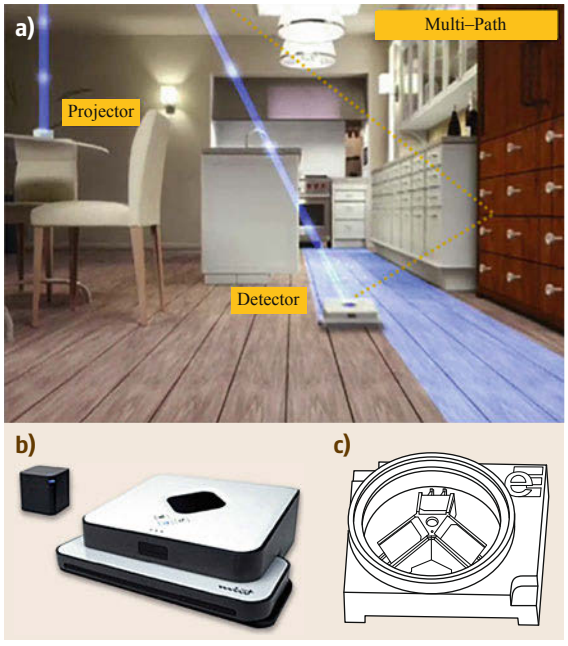


Fig.65.16a–c Mint robot. **(a)** Normal operation of the Mint robot using Northstar (the *yellow path* indicates multipath). **(b)** Mint robot and Northstar cube. **(c)** Northstar sensor

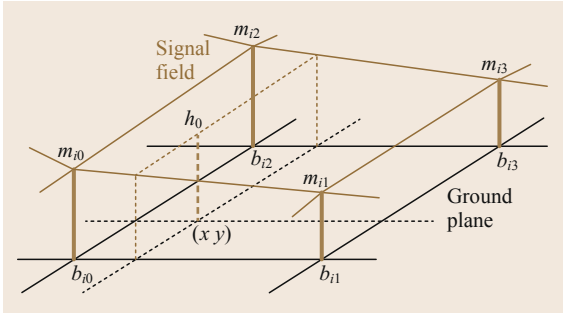


Fig. 65.17 Vector field SLAM with Northstar

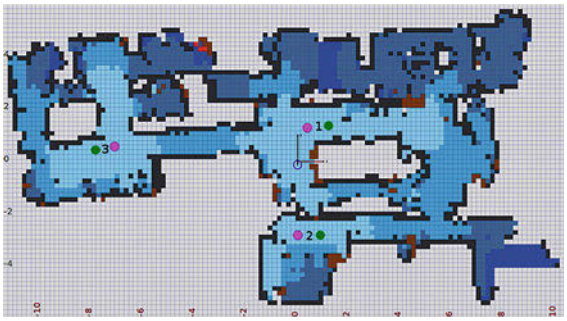


Fig. 65.18 Robot map obtained with vector field SLAM in a 125 m^2 home environment

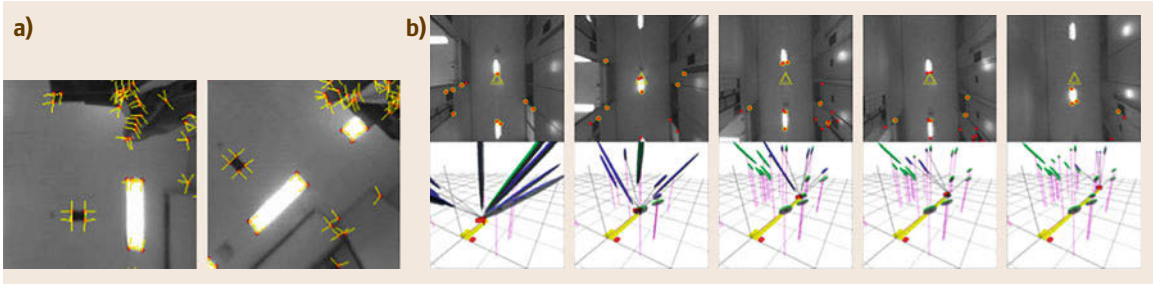


Fig. 65.19 **(a)** cv-SLAM (after [65.14]). **(b)** Features extracted by the vision front-end. Map obtained with the EKF

cles for a thorough cleaning around walls and furniture. As the robot moves through the environment it creates an occupancy grid map using the position information from localization. Each visited cell is classified into one of the following categories: obstacles (drawn in black), floor changes (drawn in brown), hazards (drawn in red), and free space (drawn in shades of blue that indicate visibility of the spots, the lighter the blue, the better the visibility).

Visual SLAM

Visual localization and mapping is attractive for a variety of applications due to the rich input and the low cost and footprint of visual sensors. The difficulties lie in robustly extracting a critical subset of information from the high-rate visual data stream and processing it efficiently to yield useful output. Despite steady increases in the computational power of most platforms, such challenges are nonetheless exacerbated by the limited processing and storage provided by low-cost, embedded systems appropriate for low-power applications or consumer products. Many state-of-the-art approaches to visual SLAM depend on a per-frame processing rate sufficiently high, relative to the speed of camera motion, to permit strong temporal assumptions on the image sequence. This section reviews two approaches to visual SLAM focused on domestic robots. Ceiling vision similar to the work of *Jeong* and *Lee* [65.14] can be found in the Navibot from Samsung, the Roboking from LG, and the Iclebo/Homerun from Yujin Robotics/Philips. Furthermore, visual SLAM systems have been developed for low-cost and embedded systems [65.17, 18].

cv-SLAM

The ceiling vision SLAM [65.14] system is composed of a camera facing upward, looking toward the ceiling. The system extracts corner features using a Harris detector and matches features using correlation. Feature matching achieves invariance to view point changes by training a set of multiview descriptors of the features as matching proceeds during the run. The main orientation(s) of the feature are further used as descriptors of the feature to ensure a two-dimensional (2-D) rotation invariant feature matching that can be used for relocalization. The localization and mapping backend of the system is based on an extended Kalman filter (EKF) that fuses visual and dead-reckoning information (odometry and gyro) and that tracks the pose of the robot in a two-and-a-half-dimensional (2.5-D) (x, y, θ) and the 3-D position of the features (landmarks of the map). The orientation(s) of the features are also represented in 3-D and tracked with the EKF to predict changes in the feature patch due to motion of the robot (as the robot moves, features on the ceiling will undergo a rotation while features on the walls will experience a shearing transformation in addition to rotation).

Figure 65.19a shows corner points and their estimated orientations in views with rotation only. Figure 65.19b presents snapshots from a sequential map-building experiment on a corridor.

vSLAM

The vSLAM [65.17, 18] system from Evolution Robotics is designed for a low-cost robotic platform equipped with simple odometry and a single camera. Figure 65.20 shows the block diagram of the system. Operating primarily as a view recognition engine, the visual measurement front-end requires only occasional, weak assumptions on processing rate, and intrinsically provides robust loop closing when previously mapped areas are revisited. The visual measurements and odometry are fused in the back-end in a graph representation and optimized incrementally. The SLAM graph complexity is bounded during operation using variable elimination and constraint pruning, with heuristic schedules in order to keep optimization and storage costs commensurate with explored area, rather than with time of exploration, all while causing minimal loss in mapping and localization accuracy. The system has been evaluated on real datasets with planar ground-truth reference, showing that the system operates successfully even at frame rates below 2 Hz, running on an ARM9 processor with 64 MB of RAM [65.18].

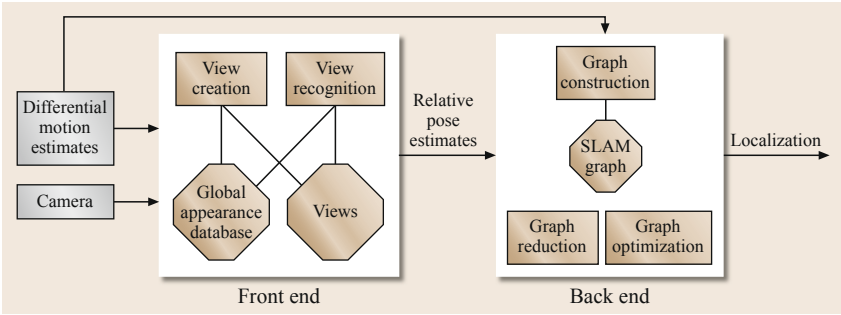


Fig. 65.20 vSLAM block diagram

View recognition engines [65.17–22] have proven attractive components for SLAM systems because they permit robust and flexible loop closing. Instead of making correspondences between individual features or measurements as in the cv-SLAM system, view recognition engines typically match constellations of features or entire images, without requiring tracking. vSLAM [65.17] creates views by first matching SIFT [65.23] features on pairs of incoming images, and second computing structure and motions estimates using bundle adjustment. The SIFT features are stored in a local database for each view and on a global database for the complete map. View recognition is performed by a feature lookup in the global database that provides a set of candidate view matches. Feature lookup in the local view database followed by robust outlier rejection and a local bundle adjustment completes the view recognition and visual pose estimation process.

Commonly used graph SLAM methods for agglomerating sensor information often incur computation and storage costs that grow with time, rather than with space explored. For a robot operating for extended periods within a limited spatial area – typical of practical applications – this is an undesirable trade-off. vSLAM [65.18] instead applies probabilistically sound graph reduction methods that limit the complexity of the graph to a linear factor of the complexity of the explored space. Past poses of the robot that are not used for view recognition can be marginalized out of the estimation, and their incident constraints are collapsed back into the graph.

Figure 65.21 presents the results of running the vSLAM system in two sequences, one collected in a regular household (*right*) (same house as the one shown in Fig. 65.18 for vector field SLAM) and the other collected in a large warehouse environment (*left*). The first set of plots show the ground truth path of

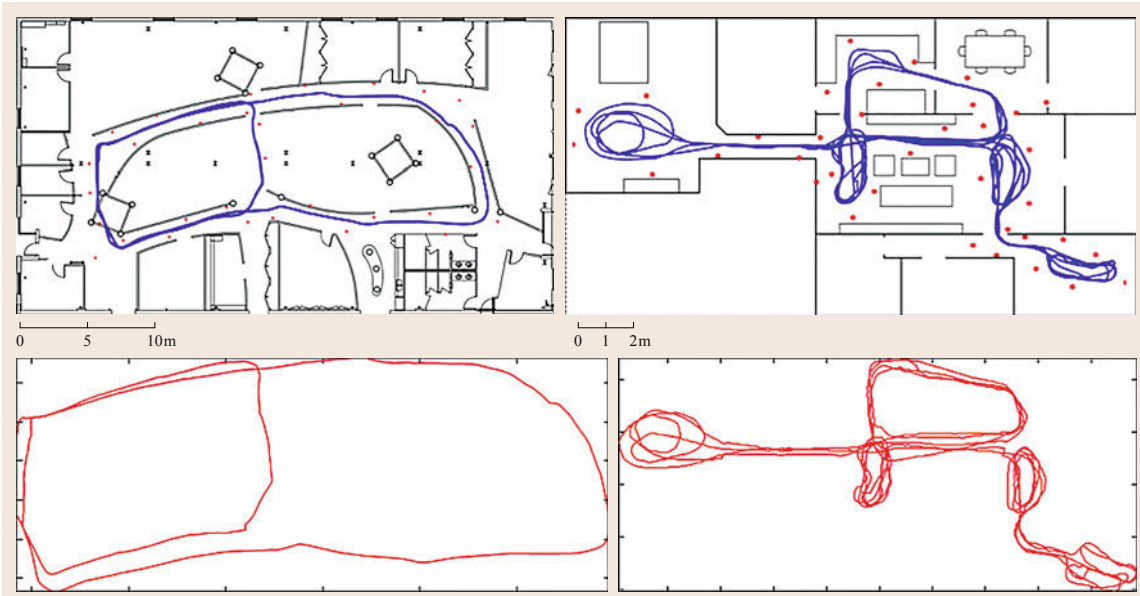


Fig. 65.21 vSLAM results on a warehouse environment and in a household environment

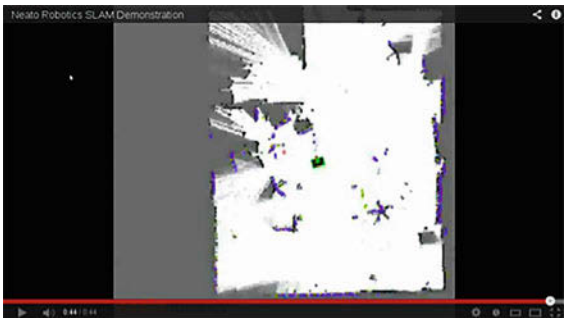


Fig. 65.22 Map created with the Neato XV-11 robot (www.youtube.com/watch?v=zodC8EFvh7g)

the robot and the floor plan of the environments. The second set of plots shows the trajectory of the robot estimated with vSLAM. The warehouse environment extends over an area of 24×12 m and the house over an area of 20×9 m. The estimated trajectory has a root mean square (RMS) error of 44 cm in the warehouse case and of 28 cm in the household case.

Laser-Based SLAM

The previous sections presented a number of low-cost solutions to the SLAM problem in which the data association was (almost) perfect. In the case of Vector-Field SLAM, the modulation of the spots of the Northstar cube ensures a unique identification of the spot. In the case of visual SLAM, the visual front-end incorporates a number of checks to ensure that misrecognitions are very rare. Laser-based SLAM has the characteristic that the measurements acquired with the laser are not unique since it is possible to obtain similar measurements in a variety of places in a household (just aiming the laser to a wall will provide measurements that are indistinguishable from a measurement to a different wall). In addition to the data association problem, laser distance measurement sensors, such as the one presented in Sect. 65.2.1, provide both range and bearing to the landmarks. Both Northstar and cameras give only bearing measurements to the spots or features.

The laser-based SLAM literature is quite extensive [65.25]. The estimation back-end could be an extended Kalman filter (EKF), a Particle Filter, or a GraphSLAM system. Several data association algorithms are available, some of them are proactive (or even greedy) [65.26, 27] and some are lazy [65.28] in assigning correspondences between measurements and landmarks. The selection of the algorithm to implement in the domestic robot would be guided by the trade-off between computational resources and the performance requirements. Figure 65.22 presents a map obtained with the Neato XV-11 robot.

65.2.3 Navigation and Coverage

The Trilobite robot vacuum cleaner by Electrolux was one of the pioneer robots to be commercialized. The trilobite was equipped with a sophisticated custom sonar sensor system that allowed the robot to navigate without touching obstacles (or touching them very gently). The coverage strategy consisted of two stages: first, explore the perimeter of the room to estimate the area to clean and second, cover the room with wall-to-wall diagonal passes. The perimeter exploration stage assumes that the robot would eventually traverse through the complete boundary of the area to clean and return to the charging station (Fig. 65.23a).

As mentioned previously, the cost constraints forced early robots like the Roomba to eschew advanced features like localization and mapping in order to offer a product at a price point accepted by consumers. Nevertheless, the navigation strategy implemented in the Roomba was quite effective in covering the space, especially when running in single rooms or in presence of large amount of clutter. The Roomba uses a spiral pattern in open area to optimally cover the space without localization until it hits an obstacle (Fig. 65.23b). Then it selects random orientation and continues moving in straight line until reaching the next obstacle where it selects another random orientation to continue with the same behavior. The spiral can be triggered when the robot travels for a certain distance without having found any obstacles. Many of the other random robots present in the market have a strategy that follows similar principles to the Roomba one.

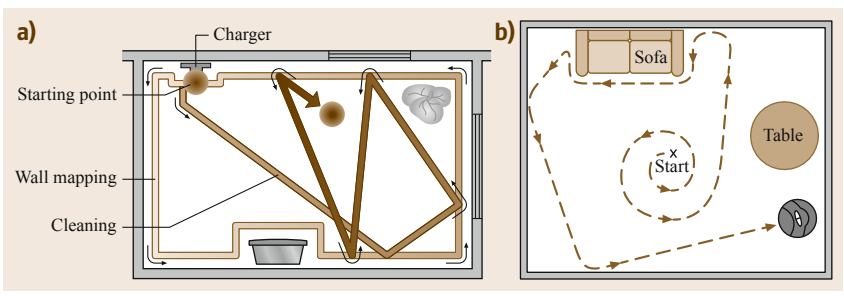


Fig.65.23a,b Coverage strategies. Trilobite strategy. Roomba strategy [65.24]

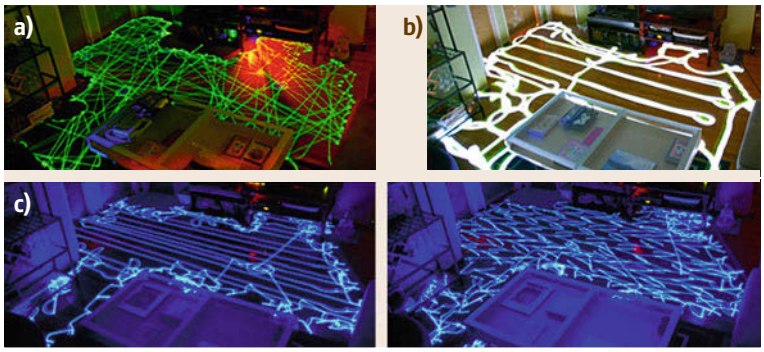


Fig.65.24a–c Long exposure images [65.29] showing the coverage strategies of the **(a)** Roomba, **(b)** Neato, and **(c)** Mint

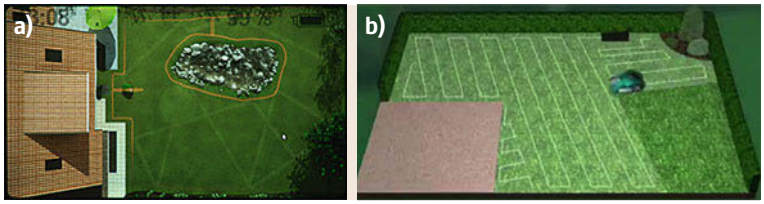


Fig.65.25a,b Coverage strategies for lawnmowers. **(a)** Tango [65.30] by John Deere and **(b)** Indego [65.31] by Bosch

The systematic robots on the other hand take advantage of the localization system to plan efficient cleaning paths to maximize coverage in the least amount of time, pause for charging on the docking station and resume cleaning from the last cleaned spot, and intelligently navigate from room to room. Mint takes the strategy of focusing first on open area cleaning, covering the space in parallel straight passes to traverse as much open space as fast as possible, and then performing a final cleaning on the perimeter and around the obstacles. This perimeter cleaning step enables Mint to uncover portals to new areas that were not encountered during open area cleaning. Other robots like the Navibot from Samsung, the Roboking from LG and the iClebo from Yujin/Philips use a similar strategy without the final perimeter cleaning. The Neato XV-11, on the other hand, performs first an exploration and cleaning of the perimeter of the environment in order to create a good localization map and then completes the coverage of the open area. Figure 65.24 shows long exposure pictures of different robots: Roomba, Neato, Mint sweeping and Mint mopping. These long exposure pictures show the trajectory of the robots during normal cleaning operation.

The lawnmower robots use similar coverage strategies to the ones of vacuum cleaner robots. One element common to all lawnmower robots is the usage of an embedded wire to define the perimeter of the lawn. Within the boundaries defined by the wire, some robots like the Automower from Husqvarna or the Robomow from Friendly Robotics use a complete random coverage strategy. The Tango from John Deere combines straight motions in random directions with spiral motions (a-la-Roomba). The Indego from Bosch is the only robot that attempts to systematically cover the space by first exploring the perimeter to estimate the size of the lawn and then traversing the interior with parallel straight passes. Systematic coverage is enabled by fusing sensory information from wheel odometry and an inertial measurement unit (IMU) by means of probabilistic reasoning with prior knowledge. Figure 65.25 depicts the coverage strategies of the Tango and the Indego.

65.3 Smart Homes

Several attempts have been made in the literature to define the term smart home, for example, in [65.32] the term is defined as the *the latest expression of the various ways in which technology in the home has developed*. In [65.33], the notion of a smart home is defined more explicitly as:

a residence equipped with computing and information technology which anticipates and responds to the needs of the occupants, working to promote their comfort, convenience, security and entertainment through management of technology within the home and connections to the world beyond.

Smart homes typically comprise elements such as network of sensors and actuators, and also entire robotic systems.

In 1999, a company called e2 Home was established by Ericsson and Electrolux to explore the possibilities of smart homes [65.34]. E2 Home built several houses with smart home devices using IT technologies. Through the exploration, several issues relating to smart homes from business point of view have been unveiled, which include the difficulties relating to the complexity of the system, starting up the new business model, handling of intellectual property rights (IPR) for the third parties contents, and the consumer's needs. The company was liquidated in 2004.

Current smart home technology includes video monitoring, motion detectors, fall detectors, pressure mats, environment control, health monitoring such as blood pressure, pulse rate, body temperature, weight, and human computer interaction (HCI) technology, for example, to recognize gestures. Smart homes also often refer to houses connected to a smart grid, which is defined as [65.35]:

> *a developing network of new technologies, equipment, and controls working together to respond immediately to our 21st century demand for electricity.*

In this case, the smart home is defined as [65.35]:

> *a residence with the capability of efficiently controlling generated solar energy and power consumption making it ideal for vehicle power supply and management.*

The development of smart homes requires a number of technical questions and challenges to be addressed [65.32]: how to convert current home structures and architectures into smart homes, how to standardize smart home components, for example, sensor networks, how to keep the equipment cost at a reasonable level, and how to deal with security and privacy issues.

In the following sections we will describe a number of prominent smart home developments, the Aware Home at Georgia Institute of Technology, the Gator Tech Smart House at the University of Florida, and the sensorized environment for life (SELF) at AIST (the Japan National Institute of Advanced Industrial Science and Technology) (Figs. 65.26, 65.27). These developments show that current smart home technology goes significantly beyond existing home automation. Accounting for the significant increase in the elderly population in the near future, many of today's smart home developments pay special attention to improving the quality of life of elderly people. The descriptions below will also address the question on how to integrate robotics systems into smart home concepts.

65.3.1 Gator Tech Smart House

The Gator Tech Smart House (Fig. 65.26) was built at the University of Florida [65.36]. It addresses the needs of elderly people to live independently and maintain dignity and quality of life at older ages. The house is equipped with many smart devices such as smart floors, tracking the motion of the occupants of the house, smart blinds, automatically adjusting ambient light, smart display, smart cameras, smart phones that can act as a remote control to other appliances, location tracking, smart leak detectors, and smart beds. The exterior of the house has a smart mailbox which alerts residence if mail is delivered and a smart front door which can sense home owners using an radio frequency identification (RFID) tag, allowing keyless entry to home owners.

The kitchen of the Gator Tech Smart House includes a smart microwave, which uses RFID on food packages. This allows the microwave to adjust the settings for cooking the meal. It also informs the resident about the readiness of the meal. The kitchen further comprises a smart refrigerator that monitors food availability and consumption, and detects expired food items. The smart refrigerator can create shopping lists automatically and has an integrated meal preparation advisor based on items in the refrigerator and pantry.

The implementation of such a complex system has raised a number of technical issues and questions, including the development of the smart devices, data handling of networks of sensors, and interconnecting smart devices to other devices in the environment. These questions led to some new research tracks on smart houses, primarily grouped into pervasive computing and mobile computing network research.

Fig. 65.26 Georgia Institute of Technology: Aware Home

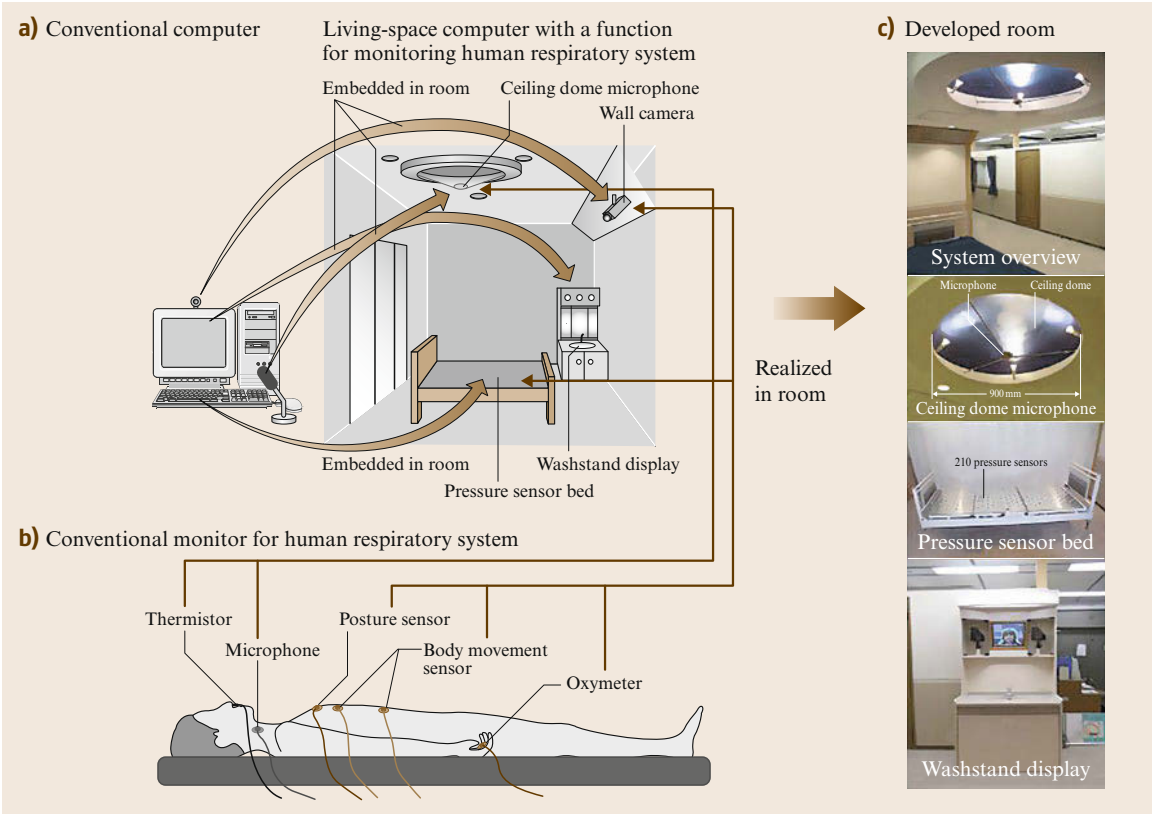


Fig. 65.27 AIST, the sensorized environment for life

65.3.2 Aware Home

Aware Home is a living laboratory for research in ubiquitous computing for everyday activities. This project is conducted at Georgia Institute of Technology [65.37]. The major objective of the Aware Home project is to build an environment that is capable of being aware and keeping track of the states and activities of its inhabitants. Aware Home creates a partnership between the resident and the surrounding sensing and computing technologies. This opens several fields of research, not only from the technology point of view, but also in terms of the social aspects of the inhabitants. The main research agenda of Aware Home spans human-centered and technology-centered research, software engineering, and social implications.

Technology and application-centered research focuses on sensor networks, distributed computing, context awareness and ubiquitous sensing, individual interaction with the home, smart floors, and finding lost objects. Research on context awareness is inspired by the fact that humans communicate with each other very success-fully by referring to what is called shared context. For communication between humans and computer systems, this shared context must be made explicitly. Sensor systems, which facilitate the extraction of context, need to be developed.

This human-centered research focuses on support for the elderly and other social issues. A key concept in sup-porting the elderly is *aging in place*. Aware Home is designed to support the elderly and allow them to be independent instead of moving to elderly care facilities. Supporting the elderly leads to the study on cognitive support such as reminding them when to take medication, guiding them when they lose their way, and locating lost items.

65.3.3 SELF – Sensorized Environment for LiFe

SELF stands for Sensorized Environment for LiFe [65.1]. The objectives of SELF are to develop a network of sensors which are embedded in the environment, information gathering using the networked sensors, storing and analyzing the information, and the reporting of useful information to assist and maintain good health. The basic advantages

of SELF, due to the embedded nature of the sensors are:

1. No limit on size, weight, or power source,
2. It does not disturb the human,
3. It does not impose physical restrictions, and
4. Sensors are rarely broken since they are fixed to the environment.

SELF can be viewed as a system that monitors a person's behavior or activity and represents the data objectively in an approach known as self-externalization. SELF is motivated by the fact that humans sometimes cannot notice a change in their condition which affects their health without a medical doctor. Therefore, the use of network sensors to monitor human behavior and report useful information that greatly affects the health status will further improve quality of life.

The SELF study considers behavior as a means of communication, and sensors embedded into the environment as one way to observer a person's behavior. The SELF implementation consists of a bed with a sensor, a ceiling with microphones, a washstand with a display, etc. The bed with sensors can determine the time the subject sleeps and wakes up, their posture during sleeping, and their breathing pattern. The microphones attached to the ceiling can detect snoring or normal breathing sounds. Based on the monitored data, the washstand display is used as an output device to provide the subject's health status and thus create feedback to the subject.

Video-References

VIDEO 727 Robotic Vacuum Cleaners Reviewed by Click – Spring 2014
available from http://handbookofrobotics.org/view-chapter/65/videodetails/727

VIDEO 729 How would you choose the best Robotic Vacuum Cleaner?
available from http://handbookofrobotics.org/view-chapter/65/videodetails/729

VIDEO 731 Husqvarna Automower versus competitors
available from http://handbookofrobotics.org/view-chapter/65/videodetails/731

VIDEO 734 Windoro window cleaning robot review
available from http://handbookofrobotics.org/view-chapter/65/videodetails/734

VIDEO 735 WINBOT W710 versus HOBOT 168
available from http://handbookofrobotics.org/view-chapter/65/videodetails/735

VIDEO 736 Winbot window cleaning robot
available from http://handbookofrobotics.org/view-chapter/65/videodetails/736

VIDEO 737 Serbot Robot Clean Ant Profi
available from http://handbookofrobotics.org/view-chapter/65/videodetails/737

VIDEO 739 Home Pool Cleaner Review – 5 types of robotic cleaners
available from http://handbookofrobotics.org/view-chapter/65/videodetails/739

VIDEO 740 Automatic pool cleaner reviews
available from http://handbookofrobotics.org/view-chapter/65/videodetails/740

VIDEO 741 Telepresence robot in action
available from http://handbookofrobotics.org/view-chapter/65/videodetails/741

VIDEO 742 Double robotics – overview
available from http://handbookofrobotics.org/view-chapter/65/videodetails/742

VIDEO 744 Test-Driving Beam, the telepresence robot
available from http://handbookofrobotics.org/view-chapter/65/videodetails/744

VIDEO 745 Beam's new Palo Alto store lets telepresence robots sell themselves. Literally
available from http://handbookofrobotics.org/view-chapter/65/videodetails/745

VIDEO 746 This robot is your running coach – Joggobot
available from http://handbookofrobotics.org/view-chapter/65/videodetails/746

VIDEO 747 RUFUS – your personal running coach
available from http://handbookofrobotics.org/view-chapter/65/videodetails/747

VIDEO 748 PhillieBot robot gives first pitch at a Phillies game
available from http://handbookofrobotics.org/view-chapter/65/videodetails/748

References

65.1 P. Ljunggren: Intelligent machines, Personal Communication

65.2 R.D. Schraft, M. Hägele, K. Wegener: *Service-Roboter-Visionen* (Hanser, Munich 2004)

65.3 A. Albert: Robert-Bosch GmbH, Personal Communication

65.4 Howstuffworks: http://electronics.howstuffworks.com/gadgets/home/robotic-vacuum2.htm

65.5 K. Konolige, J. Augenbraun, N. Donaldson, C. Fiebig, P. Shah: A low-cost laser distance sensor, Proc. Int. Conf. Robotics Autom. (ICRA), Pasadena (2008)

65.6 Gearfuse: http://www.gearfuse.com/robotic-vacuum-paths-mapped-and-compared-with-long-exposure-pictures/

65.7 P. Ridden: Joggobot turns a quadrocopter into a running companion, Gizmag, June 11 http://www.gizmag.com/joggobot-autonomous-quadrocopter-running-partner/22899/ (2012)

65.8 B. Freedman, A. Shpunt, J. Arieli: Distance-varying illumination and imaging techniques for depth mapping, US Patent 29 0698 (2010)

65.9 A. Shpunt, A. Zlesky: Depth-varying light fields for three dimensional sensing, US Patent 2008/10 6746 (2008)

65.10 Microsoft (Xbox): http://www.xbox.com/en-US/KINECT

65.11 Apple (Primesense): http://www.primesense.com/

65.12 J.-S. Gutmann, K. Culp, M. Munich, P. Pirjanian: The social impact of a systematic floor cleaner, ARSO Munich (2012)

65.13 J.-S. Gutmann, E. Eade, P. Fong, M. Munich: Vector field SLAM – localization by learning the spatial variation of continuous signals, Trans. Robotics **28**(3), 650–667 (2012)

65.14 W. Jeong, K. Lee: CV-SLAM: A new ceiling vision-based SLAM technique, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005)

65.15 J.-S. Gutmann, D. Goel, M. Munich: Scaling vector field SLAM to large environments, Proc. IAS, Jeju (2012)

65.16 J.-S. Gutmann, P. Fong, M. Munich: Localization in a vector field map, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012)

65.17 N. Karlsson, E.D. Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian, M.E. Munich: The vSLAM algorithm for robust localization and mapping, Proc. Int. Conf. Robotics Autom. (ICRA) (2005)

65.18 E. Eade, P. Fong, M. Munich: Monocular graph SLAM with complexity reduction, IROS (2010)

65.19 M. Cummins, P. Newman: Accelerated appearance-only SLAM, Proc. Int. Conf. Robotics Autom. (ICRA), Pasadena (2008)

65.20 E. Eade, T. Drummond: Monocular slam as a graph of coalesced observations, Proc. 11th IEEE Int. Conf. Comput. Vis. (ICCV'07), Rio de Janeiro (2007)

65.21 E. Eade, T. Drummond: Unified loop closing and recovery for real time monocular slam, Proc. Br. Mach. Vis. Conf. (BMVC'08) Leeds (2008) pp. 53–62

65.22 K. Konolige, J. Bowman, J.D. Chen, P. Mihelich, M. Calonder, V. Lepetit, P. Fua: View-based maps, Proc. Robotics Sci. Syst. Seattle (2009)

65.23 D.G. Lowe: Distinctive image features from scale-invariant keypoints, Proceedings IJCV (2004)

65.24 Pmdtechnologies GmbH: http://www.pmdtec.com/

65.25 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)

65.26 J. Neira, J. D. Tard os: Data association in stochastic mapping using the joint compatibility test, IEEE Trans. Robotics Autom. **17**(6), 890–897 (2001)

65.27 S. Thrun, D. Fox, W. Burgard: A probabilistic approach to concurrent mapping and localization for mobile robots, Mach. Learn. **31**, 29–53 (1998)

65.28 D. Hähnel, S. Thrun, B. Wegbreit, W. Burgard: Towards lazy data association in SLAM, Proc. 11th Int. Symp. Robotics Res. (ISRR), Sienna (2003) pp. 421–431

65.29 John Deere: http://www.deere.com/wps/dcom/en_INT/products/equipment/autonomous_mower/autonomous_mower.page

65.30 Robert Bosch GmbH: http://www.bosch-indego.com/

65.31 E. Prassler, K. Kosuge: Domestic robotics. In: *Springer Handbook of Robotics*, ed. by B. Siciliano, O. Khatib (Springer, Berlin, Heidelberg 2008) pp. 1253–1281

65.32 D. Gann, J. Barlow, T. Venables: *Digital Futures: Making Homes Smarter. Report published by Chartered Institute of Housing* (Chartered Institute of Housing, Coventry 1999)

65.33 R. Harper: *Inside the Smart Home* (Springer, London 2003)

65.34 Japan Electronics and Information Technology Industries Association: http://www.eclipse-jp.com/jeita/

65.35 Gator Tech Smart House: http://www.icta.ufl.edu/gt.htm

65.36 Aware Home Research Initiative: http://www.awarehome.gatech.edu/

65.37 T. Hori, Y. Nishida, T. Suehiro, S. Hirai: SELF-Network: Design and implementation of network for distributed embedded sensors, IEEE/RSJ Int. Conf. Intell. Robots Syst. IROS (2000)

Multimedia Contents

66. Robotics Competitions and Challenges

Daniele Nardi, Jonathan Roberts, Manuela Veloso, Luke Fletcher

This chapter explores the use of competitions to accelerate robotics research and promote science, technology, engineering, and mathematics (STEM) education. We argue that the field of robotics is particularly well suited to innovation through competitions. Two broad categories of robot competition are used to frame the discussion: human-inspired competitions and task-based challenges. Human-inspired robot competitions, of which the majority are sports contests, quickly move through platform development to focus on problem solving and test through game play. Task-based challenges attempt to attract participants by presenting a high aim for a robotic system. The contest can then be tuned, as required, to maintain motivation and ensure that the progress is made. Three case studies of robot competitions are presented, namely robot soccer, the UAV challenge, and the DARPA (Defense Advanced Research Projects Agency) grand challenges. The case studies serve to explore from the point of view of organizers and participants, the benefits and limitations of competitions, and what makes a good robot competition.

This chapter ends with some concluding remarks on the natural convergence of human-inspired competitions and task-based challenges in the promotion of STEM education, research, and vocations.

66.1 **Introduction** 1760
66.2 **Overview** 1760
66.2.1 Aims and Targeted Participants ... 1761
66.2.2 Competition Challenges 1761
66.2.3 Types of Robots 1761
66.2.4 Competition Scenarios 1761
66.2.5 Safety 1762
66.2.6 Qualification 1762
66.2.7 Evaluation 1762
66.2.8 Organization 1762
66.2.9 Sponsors 1762
66.3 **Competitions Inspired by Human Competitions** 1762
66.3.1 Case Study: Robot Soccer 1764
66.4 **Task-Oriented Competitions** 1769
66.4.1 Case Study: The UAV Challenge 1769
66.4.2 Case Study: The DARPA Challenges 1774
66.5 **Conclusion and Further Reading** 1780
66.5.1 Further Reading 1780
Video-References 1781
References 1781

The idea of promoting science and technology through competitions became pervasive at the end of the 20th century [66.1]. Since then, competitions have been promoted in several fields, taking a broad range of formats, goals, and target audiences. The main aims underlying this trend are to create a motivating framework for science and technology, and to support the development of skills, ideas, and technical solutions.

66.1 Introduction

These objectives fit particularly well with the field of robotics, since the design and implementation of robots is a great challenge for a competition. Robots are physical artifacts, whose design and implementation require creativity, technical skills, and scientific knowledge. Arguably, Robotics is nowadays the field where competitions have the strongest impact.

The competitions that aim at the design and implementation of robots cover the whole spectrum of the educational chain, from young pupils of the elementary schools to senior researchers, such as the authors of this chapter. Moreover, there is a large world of hobby robot competitions, that we will not specifically address in this chapter. The focus of this chapter are those initiatives that are specifically targeting autonomous robotics research and education.

Another ingredient that makes robotic competitions very successful is that robotics requires a great deal of integration. Consequently, building a winning team requires a substantial cooperation among team members with very different skills and backgrounds. This creates an ideal playground to address teamwork within educational programs, and an engineering challenge for higher level students and researchers. The level of performance that robots achieve in the competitions is often compelling, as compared with the prototypes that are developed within conventional research projects. In this respect, competitions provide an outstanding testbed for scientific and technical solutions.

Robotic competitions can have great appeal for a general audience; thus, robotic competitions are also a very effective tool to bridge the gap between citizens and the progress of science and technology.

There are several features that can be used for the classification and presentation of competitions. We have chosen to distinguish between competitions where the contest mirrors a traditional human contest and competitions where robots are challenged to perform some task. The task whose achievement, once mature, would extend the possible uses of robots in our society.

Next we present an overview including past robot competitions and considerations that apply across all robot contests. We then focus on human-inspired contests with a case study on robot soccer. Later we review task-orientated contests with case studies on the UAV challenge and the DARPA grand challenges. The chapter ends with concluding remarks on lessons learned thus far in robot competitions.

66.2 Overview

The history of robot competition begins with the Micromouse contests. In 1977, the IEEE Spectrum magazine announced their intention to hold an *Amazing Micromouse Competition*. The contest took the form of a time trial where a small wheeled robot *mouse* raced to complete a maze in the shortest possible time. The first event to be held was in New York in 1979 and saw over 6000 entries. However, the field rapidly shrank to 15 competing *mice*. This first contest of these early autonomous robots was won by a simple wall-following robot. While a valid victory, the rules for future events were swiftly changed to move the goal point from the edge of the maze to the center, ensuring that the path search aspect of the problem was emphasized [66.2]. The micromouse contest then became very popular across Europe, then later Japan and then back to the USA by the mid-1980s. Micromouse events continue to held to this day, including contests from India and South Korea. The contest which began as a struggle to develop a device that could reliably traverse corridors without jamming, is now a highly optimized contest where precision machines complete the maze in under 5 s [66.3].

Seeing the success of the micromouse competition and the power to inspire science and technology education, Dean Kamen conceived the For Inspiration and Recognition of Science and Technology or *FIRST* robot competition. In the FIRST robot competitions high school teams would compete to build and program a robot to complete a challenge. The original challenge in 1992, titled *Maize Craze*, involved herding balls into a home goal. Since then the challenge, from stacking containers to throwing flying discs, has been changed year to year while keeping the basic structure of the contest the same. The contest, like the Micromouse contests, have expanded internationally and continue to this day. In 2014, the competition is projected to involve 68 000 students [66.4].

Meanwhile, in the early 1990s, the advances and limitations of game playing computers, and the need to evaluate competing algorithms in mobile robotics converged toward the concept of a new competition for academic robotics: robot soccer [66.5–7]. We will continue this discussion in Sect. 66.3.

In 1995, restlessness at the pace of innovative change since the space race of the 1960s sparked aerospace entrepreneur Peter Diamandis to propose the X-Prize. Ansari X-Prize was a $10 million prize to demonstrate a functional privately developed space pas-

senger plane [66.8]. Rekindling the idea of racing to achieve an ambitious task was captivating and caused ripples through the scientific community [66.1]. The inspiration spread to other fields and funding bodies resulting in many initiatives such as the UAV challenge and the DARPA grand challenge which we discuss in Sect. 66.4.

Robotics is well suited to innovation through competition. As this handbook testifies the field of robotics is broad and covers research in mechanical design, electronics, control theory, signal processing, software engineering, and machine learning to name a few. In addition to the breadth of the field the nature of the research also requires one or more agents interacting with the physical environment, thereby limiting the use of techniques common to other technical fields to share and validate their research.

Standard techniques such as: standardized tests (which are commonplace in science and engineering), shared test data sets (such as those used in image processing and machine learning communities), and reference implementations (used in computer science algorithm evaluation) are invaluable in robotics but cannot cover the full scope of a robot system operating in the world. Indeed, many components used in robotic systems can be tested in isolation with these same techniques; however, as the components of a robot are integrated into a system of systems interacting with the physical world the limits of the above methods becomes evident. In terms of test methodology: unit and integration level testing are well served; however, system, dynamic and, in particular, acceptance testing of innovation across the field was missing. An answer came in a resurgence back to the roots of robotics and in favor of robot competitions.

The range of robot scientific competitions is very broad, and in this section we look at them from different perspectives, aiming at providing an overview of the settings that have been developed so far. To this end, in the following we address a number of aspects that are relevant to the design and implementation of a robot competition.

66.2.1 Aims and Targeted Participants

As suggested in the introduction, the competitions that are addressed in this chapter are those focussed on research and education. These two aspects are often deeply inter-related. On one hand, there is always an educational component in any research involving a team with different backgrounds and different levels of expertise. Moreover, if one looks at competitions as an educational tool, they represent one of the most advanced forms of education; consequently, they specifically attract teachers and mentors that are involved in research on education. Although research and education are related to one another, the balance between research and education is naturally significantly different across the numerous robot competitions.

66.2.2 Competition Challenges

Each competition relies on a challenge and there are several ways of categorizing them. While competitions are nowadays mostly among robots and not robot-vs-human, a large group of competitions take inspiration from human competitions, ranging from soccer, to the Olympic Games. These competitions have an explicit or implicit underlying assumption of comparing robot performance with human performance. The other kind of challenge aims at a proposing a specific problem to be addressed using machines the hope is that by racing to achieve the aim, new capabilities can be matured faster into future robots.

66.2.3 Types of Robots

Another key element for classification of competitions is the robot type. Any kind of robot is conceivable for use in competition. Robot choice is driven by the competition format and targeted participants. The use of commercially available robots and robot kits enables less mechanically focused participants to move forward quickly. The use of homogeneous sets of robots enables competitions to be fought on algorithmic innovations over robot manufacturing skill. An unrestricted category of robot type can be useful for some task-based challenges, where a novel physical design may be the key in simplifying the problem and achieving the objective (such as in rescue robotics).

66.2.4 Competition Scenarios

The scenario is largely determined by the challenge addressed, be it a competition among soccer robots or among service robots operating in a home environment; consequently, there is a broad variety. In some cases, the scenario is fixed and known before the competition, such as in robot soccer; in other cases, only a general description is provided beforehand and the actual competition scenario is not known in advance as, for example, in search and rescue competitions. The competition scenario has deep implications not only on the setup of competition venues, but also on the participating teams. Practice environments for teams in lead up to competition must be representative, while not prohibitive to construct. From Lego robot soccer to the DARPA Robot challenge simple readily available hard-

ware store materials have been the material of choice for contest environments.

66.2.5 Safety

In some cases, the competition raises safety issues for the participants. This is the case for example of UAV competitions, and proper measures need to be taken in order to ensure a satisfactory level of safety. Moreover, the organizers must carefully address regulations for deployment of unmanned vehicles and/or use restricted access areas for the competition.

66.2.6 Qualification

Competitions with a large number of participants, such as education programs, can require regional and qualifying rounds similar to traditional sports leagues. When contests require a substantial effort and expense to participate funding agencies may make grants available to teams which succeed at a qualifying round to contribute to the cost to compete. Diverse selection criteria for satisfactory progress may well be considered in this case including justification of sound financial backing to complete the competition.

66.2.7 Evaluation

Evaluation is one of the key features of competitions both in terms of education (students must deliver a working solution) and in terms of research, where the integration of components is not a straightforward engineering practice. However, competitions also provide a great opportunity for developing benchmarks and new test methods: performance evaluation is an open issue in robotics and competitions are providing significant insights into the development of reliable and repeatable measures and testbeds.

66.2.8 Organization

Competitions are typically originated by a perceived need by a related scientific interest group, such as a conference (e.g., AAAI, ICRA), a research funding agency (e.g., DARPA, ESA) or an entrepreneurial private company. Once established an ongoing competition founded by the first two groups will often be ran through a dedicated not-for-profit foundation. However, increasingly, education is a large business and competitions are becoming a significant activity therein; moreover, companies that help running competitions as projects are now on the market and production companies are now looking at competitions as viable tools for beyond state-of-the-art product development (not restricted to robotics) and for promoting their products.

66.2.9 Sponsors

The organization of a competition requires a substantial effort and budget. There are, however, several sources for acquiring the necessary resources. A first category of sponsors are schools that undertake competitions as part of their educational activity. The other natural sponsor are the funding agencies that institutionally provide support to research in robotics. The involvement of funding agencies can be many fold, ranging from the whole creation of a competition to the support of specific event, in particular at the local level. Another kind of sponsorship comes from public institutions that may regard competitions as a mean to drive citizens' attention on scientific and technological development. Robot manufacturers are also interested in sponsoring competitions that promote the use of their products. Besides promoting specific competitions, companies are also generally interested in supporting competitions as a form of advertisement. This is obviously related to the kind of visibility competitions can reach on the media. Moreover, companies may sponsor teams of young students as a form of recruitment and supporting education.

As the earlier discussion shows, a successful robotic competition not only requires to put forward a grand challenge or target specific societal needs, but it also requires a careful design and implementation.

66.3 Competitions Inspired by Human Competitions

In this section, we focus on competitions among robots that take inspiration from human competitions. The challenges that have been driving a lot of research in Artificial Intelligence, such as chess, go, checkers, are inspired by human competitions/games, but they do not address the interaction with physical world. On the other hand, robotic competitions that are inspired by human competitions are typically connected with sports. A variety of sports have been addressed, from Olympic games [66.9, 10], to robot soccer [66.11, 12], to other competitions, such as sailing [66.13] or racing [66.14].

The distinguishing feature of this category of competitions is the grand challenge not only of emulating human performance, but to do this in the context of a human competition. Consequently, the robot is in principle humanoid, although several other platforms have been used, creating games for robots on wheels, given that, at this stage, it is a much simpler and accessible locomotion. It is worth noticing that a big push to this kind of competitions was given by companies such as Sony or Honda, which developed humanoid platforms also as showcases of their research and development activities.

One additional challenge that characterizes the human-inspired competitions is that several of them involve multiple robots. This feature has several significant consequences, not only in terms of the requirements on the teamwork among the robots, but also on the teamwork of the development team.

The scenario of these competitions is mostly outdoor; however, due to both technological and practical reasons often they take place indoor. On the one hand, platforms that are waterproof, work on grass or uneven terrain are unavailable or much more difficult to realize; on the other hand, practical considerations suggest to keep participation affordable and accessible.

One important feature of human-inspired competitions is that they aim also at providing some entertainment for the general public. Therefore, a suitable balance between technical difficulties and level of performance is needed. It is worth noticing how the general public is more keen to accepting the limitations of robots in playing soccer than in accomplishing a mission in Fukushima plant. Consequently, even games with slow and somewhat disappointing performance can generate participation, interest, and even strong emotions in the audience.

Human-inspired competitions span over the whole spectrum of participants, including both research oriented and more educational targets. In all cases, the type of competition creates an interest both in young generations that are naturally attracted by their ludic character, and in more mature researchers that aim toward the scientific challenges that they embody. A consequence of the intuitive appeal exerted by human-inspired competitions is that they easily attract large numbers of participants. Indeed, a frequent pattern of participation is to start with games and then move to other kinds of competitions that target socially or business relevant goals. In some cases, this endeavor is encouraged by the presence in the same event of both types of competitions.

The evaluation of the performance, in the case of human-inspired competitions, is very clear and amounts to winning the game. There are, however, examples of specific capabilities associated with the game that are individually evaluated.

Given the reference provided by human competitions, human-inspired robot competitions typically have a permanent nature and a structure with local qualifying events. Consequently, there is an organization body managing the competition across the years and the main ones are nonprofit organizations (NPO)s. There are, however, commercial opportunities and companies that are interested in the deployment and sale of their robots. At present, these companies are mostly sponsoring the events, and often provide special offers for their products to the teams that enter the competition.

As mentioned earlier, human-inspired competitions have a strong appeal to the general public. Consequently, public organization often endorse and support them within their initiatives aiming at public awareness and promotion of science and technology. According to a common pattern, this may also attract large companies that invest some of their marketing budget in supporting socially and scientifically targeted events.

The ultimate consequence of the above sketched scenario is that the human-inspired competitions have a large impact on the general public. We have seen reports on human-inspired competitions in the media not only in the sections that typically cover science and technology, but also in the general news, and sometimes, even in connection with sport reports.

Besides this somewhat indirect effect, there are two major direct impacts of human-inspired competitions: on education and research. The renovation and advancement of the school system is embracing competitions for two major reasons: they motivate pupils and they teach key skills that are typically not addressed by conventional curricular activities. Human-inspired robotic competition perfectly fit both purposes. As for research, the scientific challenges, that have been put forward, whether or not they will turn out to be too ambitious, have gathered the attention of significant number of researchers, have driven a large body of research and have given rise to a number of side-effect developments: they are therefore fulfilling a role that is nowadays essential in the overall effort for advancing science and technology.

Robot soccer has a significant role within this category of competitions. The idea of robot soccer was discussed by several researchers in the early 1990s [66.5, 6]. Shortly afterward, Robot soccer competitions have been designed, in several venues, including Federation of International Robot-soccer Association (FIRA) [66.11, 15] and RoboCup [66.7, 12, 16].

In the next section, we address in more detail the RoboCup competitions. RoboCup remarkably started with the statement of a challenge, by *Hiroaki Kitano* and the initial RoboCup cofounders [66.7], namely

Table 66.1 Five subleagues of the RoboCup soccer games

League	Soccer players	Perception	Reasoning	Actuation	Communication
Simulation (Fig. 66.1)	(11) Software agents, 2-D (two-dimensional) and 3-D (three-dimensional) biped	Server engine	Client per team	Physical motion and perception	Explicit
Small-size (Fig. 66.2)	(5) Custom-built wheeled robots	Centralized overhead camera	Centralized offboard	Fast ball/robot speed, kicking, dribbling, passes, goal shots	Dedicated Radio
Middle-size (Fig. 66.2)	(5) Custom-built wheeled robots	Onboard, omnidirectional	Large-space control	Kicking, goal defense	Dedicated WiFi
Standard platform (SPL) (Fig. 66.3)	(5) Purchased legged robots	Onboard, directional range	Distributed	Legged, kicking, goal defense	Dedicated WiFi
Humanoid (Fig. 66.4)	(3,4,5) Custom-built, or purchased, biped multisized robots	Onboard, directional range	Distributed	Legged, kicking, goal defense	Dedicated WiFi

that *By the year 2050, there would be a robot soccer team that would be able to beat the human World Cup team.* With such long-term goal, RoboCup is an ongoing competition, that aims at research, educational, and engineering challenges through a set of testbeds that implement soccer games among robots. The competition keeps evolving every year, by yearly evaluating the performances of the robots and generating roadmaps toward its goal.

66.3.1 Case Study: Robot Soccer

The first organized RoboCup competitions took place in 1997 [66.16], and significant progress is noted every year. Interestingly, starting in 2007, and yearly thereafter, a team of five human players (volunteers from the RoboCup Board of Trustees) plays a demonstration game with the best robot team of the middle-size league, which we introduce below. Furthermore, each RoboCup competition event includes a symposium, which highlights research and development contributions relevant to the RoboCup competitions, and also to general robotics problems. As of 2014, the robots can play the basics of the game, with great speed and accuracy, but are clearly largely noncompetitive with human players. There is still a long way to go for RoboCup to reach its goal, but RoboCup continues to achieve annually increasing success, scope, and wide participation.

RoboCup soccer games are structured in five subleagues, where robots have different sizes and different capabilities, as overviewed (Table 66.1).

Many scientific challenges are addressed through different league settings, including teamwork and strategic decision making, in particular with systems of multiple robots in adversarial environments. To this end, the setup of each competition has evolved along the years from the initial proposal [66.7]. The simulation, small-size, and middle-size leagues were the initial leagues in 1997. The standard platform league, started as a demonstration in 1998 and as a competition in 1999, was played with the Sony AIBO quadruped robots until 2007, and since 2008 with the Aldebaran NAO robots. Finally, the Humanoid Robot League started in 2000 with team-built biped robots that could barely stand on one leg to kick, probably to the most compelling derivative in improvement, leading into 2012 to biped robots of a variety of sizes that play complete games.

We now briefly describe the individual goals and set up of the soccer leagues, aiming at discussing the technical challenges and solutions devised. A summary of the complete technical contributions is found in the RoboCup symposia proceedings [66.16–32] (VIDEO 385).

Simulation

The RoboCup simulation league was developed from the very beginning of RoboCup, as an 11-player two-team simulation server environment, the SoccerServer, conceived and implemented by *Noda* [66.33]. The simulation league consists of a client-server architecture in which each of the two client teams is given, as an input by the SoccerServer, the state of the game (players state and ball position), and generates actions for the simulated robots, which are passed to the server for simulation of the new state of the world. Given the controlled realistic simulated environment, the goal of the league is to enable the researchers to abstract from the complications of real hardware and focus on teamwork architectures and coordination, strategic decision making, and learning in complete teams of 11 players. Initially, the league was a 2-D simulation, and since 2009, and following its roadmap to the RoboCup goal, it was extended to include a 3-D competition, leading to the current two subleagues: the 2-D and 3-D Simulation Leagues (Fig. 66.1).

Fig.66.1a,b Simulation leagues: **(a)** 2-D and **(b)** 3-D

Fig.66.2a,b RoboCup soccer leagues: **(a)** small-size robots and **(b)** middle-size robots

In the 2-D simulation subleague, the SoccerServer simplifies the environment by representing robots as basic circular elements, though with orientation and different capabilities in terms of perception and actuation. The client teams focus on effective team composition and on the strategic aspects of the game. In the 3-D Simulation subleague, the players are simulated biped humanoids. The teams need to address low-level control of such humanoid robots, including the creation of their basic behaviors, such as biped walking, running, kicking, turning, and standing up. In addition, the teams design and implement multi-agent higher level behaviors for realistic humanoid robot teams. The 3-D simulation league is currently tightly connected to the standard platform league, as the simulated humanoid robots are based on the characteristics of the Aldebaran NAO humanoid robots, as used in the RoboCup standard platform league (see in the following). The goal is to directly transfer techniques and approaches from simulation into real robots. Both the 2-D and 3-D servers are open software projects.

Small Size

The RoboCup small-size league (SSL) has been in place since the initial RoboCup in 1997. The league is set up as a field of predefined dimensions for teams of custom soccer robots of predefined small dimensions. Initially the teams were of three robots, and later have been increased to five robots. The initial playing field was a green ping-pong table, which was selected pragmatically by the founders in 1996, in order to make the specifications of the newly created RoboCup competition accessible to any research team in the world. A few years later, the field became green carpeted with white line markings, and its size has incrementally grown to the recently increased size of 6 m × 9 m. Since its beginning, the game is played with an orange golf ball and an overhead vision system. Robots are custom built according to the specifications of the league, currently to fit within a 180 mm diameter circle, no higher than 15 cm (Fig. 66.2).

The teams are allowed to use at least one camera overhead of the field. Robots have colored marks on their tops to enable visual identification and tracking. Off-board computation by each team uses the processed images to plan and then wirelessly communicate the team actions as the motion of the small robots on the field. The RoboCup SSL league offers challenges and significant contributions in: hardware design, colored-based real-time image processing (of the colored ball and colored marker identifiers of the robots), and in team strategy planning.

The RoboCup SSL robot hardware has evolved from the early simple and brittle robots in 1997 to the sophisticated, fast, robust, kicking, and dribbling, omnidirectional robot platforms of the current days. The hardware of the RoboCup SSL clearly pioneers small robot design, in contrast with all other much larger research robots in other applications. The league also significantly evolved in the vision processing and the playing strategy. Although the global view of the complete playing field is a clear advantage in capturing a complete view of the position of all the players and ball, it proved to be quite challenging to process in real time at 60 Hz, 10 fast moving robots, and a small orange

golf ball. As the league evolved, and some teams succeeded in devising vision processing algorithms, since 2010 such solutions became a shared vision system as part of the league setup that can be used by any team [66.34]. This SSL vision is available to all researchers and can be set to be used in other applications that may need global overhead perception. In terms of strategy, the teams aimed at passing and positioning well on the field since the beginning of the league. Both the capabilities of individual robots and teamwork became increasingly sophisticated and effective over the years, with the ability to dribble, to flat or chip kick, to perfectly navigate collision free at high speeds in a space crowded with opponents as dynamic obstacles, and to coordinate the team with effective defense, accurate passes, and precise aim at narrow openings of the goals. VIDEO 387 shows an example of the multi robot teamwork by the CMU Dragons.

The ongoing research directions aim at learning the opponent strategy and responding to it. Through the common vision systems, all games are logged and now available for researchers to challenge learning algorithms with the complex positional, fast game data. Furthermore, the league aims at playing games of eleven players on a much larger field, also in ad-hoc teams, where the players are elements of different and previously unknown research teams.

Middle Size

The RoboCup middle-size league (MSL) started in the first RoboCup event in 1997. The league is set on a much larger field than the SSL one, initially of the size of 3 by three ping-pong tables. Currently, two MSL teams of six wheeled robots play on a 18 m × 12 m green carpeted field, with an official FIFA winter ball (orange).

MSL clearly differs from the SSL in its lack of centralized overhead perception and offboard computation. Each MSL robot is onboard fully autonomous, equipped with onboard sensors, computation, and actuation. Through wireless communication the robots can establish interteam cooperation and receive all referee commands. These two SSL and MSL leagues, were designed to offer frameworks for different robotics research interests in perception, planning, and mechanical robot design.

The MSL robots have evolved from a variety of initial experimental designs to the very effective, fast omnidirectional robots with creative kicking and defending actuation devices, in particular also in the goalie robots. At some point, robots had omnidirectional cameras that allowed each robot to compute the position of all the players on the field, for avoiding obstacles in its motion planning. Wireless communication allows for sharing state information and supports the implementation of cooperation strategies. Currently, teamwork accomplishments are based on the effective selection of robots to roles and attacking passes. In particular, the recent rules of the game force the robot to pass the ball in the opponents middle field before shooting to the goal. In this way, cooperation and passes have become a key feature for a successful team. The MSL goalie robots have been of special note, as teams build special hardware for them to allow for kicking and to effectively defend the long and well-aimed shots from the attackers. The MSL is compelling for the largest field size in RoboCup and robots of dimensions and speed that also allow humans to test play with the robots. Since RoboCup 2007, a team of volunteer members of the RoboCup Board of Trustees always plays a game against and with the MSL winning team. Such demonstrations are well appreciated by the competition audience and participants and provide evidence to the progress of the performances of the robots.

Standard Platform

The standard platform league started with the goal of providing a common hardware platform, so that the teams can focus their development on the algorithms and compare the performance of their solutions on the same hardware. Thus, in the standard platform league, all teams use identical robots, which happened to be legged robots. In the standard platform league all the challenges of the middle-size league are present (i. e., the robots operate onboard fully autonomously), but on a platform that requires a sophisticated motion control. The motion of the robot also affects perception and causes a greater uncertainty in the estimation of the environment and, thus, in basic capabilities such as ball tracking and localization of the robot in the field. The field evolved over the years: it is currently 6 m × 9 m and the game is played by two teams of five players with an orange ball.

The league started with the Sony AIBO (Fig. 66.3), in its early stages of development in 1998. After the first year demonstration, the league attracted several researchers, and underwent a substantial development both in terms of the platform, that evolved in three major revisions after the initial prototype, and in terms of the rules and requirements for the field. The AIBO robots had a very high number of actuators (16–18) to support 4-legged locomotion and body pose. At each revision, robustness of mechanical design, sensors, and computational power, steadily improved, thus, allowing to increase the size of the field (from 3 m × 4 m, to 4 m × 6 m) and to remove artificial landmarks, side barriers and strict requirements on the lighting.

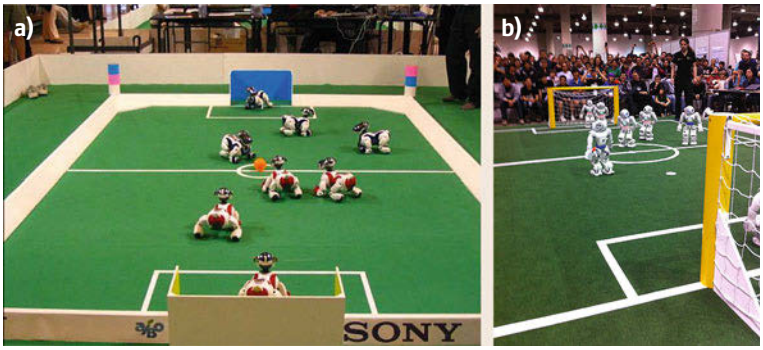


Fig.66.3a,b Standard platform league: **(a)** AIBOs and **(b)** NAOs

Both the extremely advanced features of the platform and the availability of software developed by the RoboCup teams made the AIBO a very popular platform in research laboratories to test the performance of new legged locomotion techniques, perception localization and teamwork. As the performance of the robots improved over the years, not only the rules have evolved to meet new technical and scientific challenges, but also the games became very attractive for the general audience.

When Sony announced the discontinuation of the production of the AIBOs, RoboCup issued a Call for Platform that ended up with the choice of the current standard platform: the humanoid NAO by Aldebaran Robotics (Fig. 66.3), which has been used since 2008. The set up of the field did not change substantially; however, the biped platform required not only new locomotion, but also redesign of all the basic components. Again the platform evolved over the years, with corresponding changes of the field and rules. The current version of the NAO robot is 58 cm tall humanoid robot, with 25 degrees of freedom. The processing is provided by an Intel ATOM 1.6 GHz CPU (located in the head) that runs a Linux kernel and supports Aldebaran's proprietary middleware (NAOqi) and a second CPU (located in the torso). The platform is equipped with various communication devices, including voice synthesizer, LED lights, and two speakers, two cameras (one pointing toward the feet to control the kicking), tactile sensors and sonar rangefinders, Ethernet and Wi-Fi. The capabilities of the current platform allowed to enlarge the field to the current size (to 6 m × 9 m) and to remove from it all artificial features. Each game is played by five robots, plus an additional coaching robot outside the field. A section of the competition has recently started, where teams are formed by robots that are designed by different participants.

The adoption of a common platform has several advantages. First of all, the hardware is chosen so that the teams have a powerful, yet affordable robot, and speed up the development by sharing the same infrastructure. Moreover, a standard hardware platform system makes a systematic comparison of different technical solutions more accurate. Finally, a common platform supports significantly community development that allows to reach the critical mass needed to develop complex robotic systems.

Humanoid

In the humanoid league, autonomous robots with a human-like body plan and human-like body senses play soccer against each other. The robots are designed and built by the competing teams, and include some of the best autonomous humanoid robots in the world.

The humanoid league was established in RoboCup 2002, Fukuoka. The later start, as compared with the other leagues is connected with the development of humanoid robots. The first demonstration of the early humanoid robots took place in RoboCup 2000 and Honda ASIMO first showed the ability to kick the ball, as a demonstration, in RoboCup 2002. Since then, the league started, initially addressing basic skills such as walking and kicking: the soccer game was limited to penalty kicks. In the first years of the league, very different robots were competing, but soon a separation in subleagues was gradually introduced, based on the size of the robots and on the constraints on the structure of the robot aiming to enforce dynamic walking. In 2005 the first two against two games were started in the smallest robot subleague. Since 2008, the game had three players in the kid-size and the league made a significant progress in terms of the number of participating teams and the capability of the hardware platforms. In the current setting, the robots of the humanoid league are divided into three size classes (Fig. 66.4): KidSize (30–60 cm height), TeenSize (90–120 cm), and AdultSize (130 cm and taller). In the KidSize soccer competition, teams have three players, in the TeenSize two players (keeper and forward) and in AdultSize soccer, a striker robot plays against a goal keeper robot in a dribble and kick competition. The size of the field is 4 m × 6 m and the ball is of different size for

Fig.66.4a–c Humanoid Robots: **(a)** KidSize, **(b)** TeenSize, **(c)** AdultSize

the different subleagues. Unlike humanoid robots outside the Humanoid League, the task of perception and world modeling is not simplified by using nonhuman-like range sensors.

The challenges addressed by the Humanoid League embrace those arising in the realization of fully autonomous humanoid robots in the standard platform league; however, the overall design, including the hardware, makes it possible to aim at a richer range of solutions. More specifically, key issues addressed in the Humanoid League include dynamic walking, running, kicking the ball while maintaining balance, visual perception of the ball, other players, and the field, self-localization, and team play.

Conclusion

RoboCup is recognized as a major forum for research and education through competitions. The main international yearly event gathers about 200 teams, 1500 participants (not including Junior), and tenths of thousands visitors. Regional events are also held worldwide, some of them of a size comparable to that of the main one. RoboCup is not only soccer, since it aims at fostering the development of new solutions also in applications that have a direct impact in our society: RoboCup Rescue, where robots compete to search and rescue in disaster scenarios, RoboCup@Home, where robots perform tasks that help people in daily life, Logistic League and the RoboCup@Work, where robots operate in industrial settings. Moreover, RoboCup provides a link to young generations through RoboCup Junior, a set of competitions specifically designed for the introduction of children to robotics and organized within the major event.

RoboCup has a profound impact on research. Researchers enter RoboCup competitions to validate their newly designed approaches to the broad spectrum of problems that arise from the grand goal of building teams of autonomous robots. Algorithms and robots are put to test in adversarial environments not previously seen, to show effective performance and advance the scientific and engineering state of the art in Robotics. There is a countless number of publications that present, in the most qualified Artificial Intelligence and Robotics venues, scientific contributions arising from the technical challenges tackled in RoboCup and validated through RoboCup games and platforms. The approaches and solutions developed in RoboCup have impacted many research areas. Particularly significant examples are the fields of humanoid robotics, multi-agents and multirobots, and performance metrics for Robotics.

RoboCup has brought significant contributions in education. The students, who form the teams, are the driving force of the effort to design and implement the components of rather complex integrated systems. Learning about cutting-edge solutions and experimenting them is a very effective way to develop creativity and design skills. Moreover, engineering within a team, in the framework of a competition, is a unique experience that goes far beyond the standard practice of academic curricula. In addition, RoboCup organized several Summer Schools, with a substantial hands-on component that has also been subsequently adopted in many other events. While some of the above features are not unique to RoboCup, the blending of ingredients and the environment provided by RoboCup have proven extremely rich, attractive and successful.

A few examples of particularly significant impacts that RoboCup had within our society are hinted below. Aldebaran Robotics NAO robot won RoboCup 2007 Open Call for a standard platform and afterward it has become popular through the standard platform league. Today NAO is deployed worldwide by research laboratories in robotics, becoming a commercial success. Kiva Systems are revolutionizing automation by using hundreds of mobile robots for storing, moving, and sorting inventory. In March 2012, Kiva Systems was acquired by Amazon. RoboCup and robot soccer were credited by Kiva's CEO for the success of the company. The search and rescue robot Quince, during the rescue oper-

ations in the Fukushima–Daiichi Nuclear Power Plant, could go up to the second to fifth floors in the nuclear reactor buildings. The robot was designed and evaluated through RoboCup Rescue competitions, showing excellent mobility in comparison with the other competing robots.

66.4 Task-Oriented Competitions

In this section, we consider competitions that take their inspiration from common, but important tasks, currently undertaken by people. Competitions in this class tend to be focussed of dirty, dangerous and dull activities. Robotics of this kind is known as *Field Robotics* with robots typically having to operate in unstructured or semistructured environments, which often change over time.

The robots in these competitions are normally vehicles, such as ground vehicles (often cars), submersibles, or unmanned aerial vehicles. There is even a current high-profile competition to send a robot to the Moon, the Google Lunar X-Prize.

Unlike the competitions inspired by humans, task-oriented competitions are not driven by spectator experience. Spectators are welcomed at some of the competition events but not at all. Some are closed to competitors, organizers, and media only.

Task oriented competitions are held to drive innovation, raise the profile of a particular area of robotics, or to overcome nontechnical hurdles such as regulation and insurance which would otherwise stifle individual research projects. A competition is a more efficient method of addressing these issues than a collection of disparate research projects across the world. This type of competition also attracts professionals in robotics or soon to-be professionals such as university students. They are hence a useful way to increase interest in industry participation and you will see major corporations with interests in the relevant technology and applications areas supporting or sponsoring a competition.

Unlike robot competitions inspired by sporting contests, task oriented competitions often happen as one-off events.

Task-based competitions in which the exact manner by which the task is to be accomplished is unknown, or significantly beyond the state of field, are often referred to as a *Challenge*. Challenges often have large cash prizes for the team that accomplishes the task reflecting the substantial investment of effort envisaged to claim the prize by the prize authority.

Ultimately even substantial cash prizes are almost always surpassed by the accumulated cost of the winning team's labour, equipment, donations by sponsors and overheads. Instead most participants are driven by prestige of conquering the challenge, *doing the impossible* and claiming the prize, or at least attempting to.

In recent years, the success of the robot competition format has lead to a number of contests including the MAGIC Competition [66.35], RoboCup Rescue [66.36], ELROB [66.37], and the AUVSI competitions [66.38].

A push to robot competitions to task-oriented robot challenges has also been given within the framework of scientific conferences such as AAAI and ICRA. Within this context, service robotics has also been a source of inspiration, addressing tasks such as serving drinks or preparing sushi. RoboCup@Home combines multiple challenges facing service robotics into a structured competition framework [66.39].

In the following sections, we examine at two representative task oriented competitions, the UAV challenge and the DARPA Challenge. While the robots differ significantly the drivers and approach to competition has many parallels.

66.4.1 Case Study: The UAV Challenge

Flying Robot Competitions

Competitions that encourage students to develop their skills with flying robots in exciting and challenging ways have been developed by universities and other research and education organizations since the early 1990s. One of the first, longest running and most successful flying robotics competitions is the International Aerial Robotics Competition (IARC) [66.40]. It was first held in 1991 at Georgia Institute of Technology. The aim of the competition is to push forward the state of the art in autonomy in flying robots. Teams are given challenges based on real-world scenarios, typically mock ups of disaster response or covert military style operations. A new challenge is announced once the current scenario is completed. Many scenarios take a number of years to be successfully demonstrated. Since 1991, IARC has seen six different competition scenarios. Nearly all of the scenarios involve interaction with ground elements meaning that the aerial robot must avoid obstacles and pick something up. This means that the successful platforms of IARC are nearly always rotary wing vehicles such as helicopters, ducted fan VTOL vehicles or in more recent year, multiro-

tor platforms. IARC is open to university-based teams from around the world. The past three IARC missions have involved the aerial robots having to navigate inside buildings as well as free-flying outdoors.

Another long running event is the International Micro Air Vehicles (IMAV) competition, which has been running since 1997 [66.41]. This competition focusses on very small unmanned aircraft operating either indoors, or in outdoor areas around an airfield or built environment. The IMAV events consist of a number of subevents, each aiming to push technology to solve specific research challenges, such as autonomy, perception, and flight dynamics. As reported in [66.41], the level of autonomy demanded to complete events has been increasing in recent years.

The Singapore Amazing Flying Machine Competition (SAFMC) is organized by Singapore's DSO National Laboratories and Science Centre Singapore and was first run in 2009 [66.42, 43]. It is typical of the type of competition event that is comprised of a number of different categories, from a basic radio-controlled flying competition to a category encouraging university level teams to design and build automated aircraft to navigate complex indoor obstacle course.

The UAV Challenge Outback Rescue is an international flying robot challenge that was first run in 2007 [66.44]. It is different to other competitions in that the aircraft have to be flown a considerable distance from the take-off and landing location. The competition revolves around a simulated scenario of a lost walker in the Australian outback with the start of the search area being 4 km from the take-off location and the furthest point of the search area being 6.5 km away. These distances require autonomous operation of the aircraft and reliable communications back to the ground stations – all things that stretch the capability of university and amateur teams [66.45, 46].

In 2013, the Wildlife Conservation UAV Challenge was launched [66.47]. This competition has a similar format to that of the UAV Challenge Outback Rescue, but is focussed on a scenario to find poachers of endangered animals in Africa. The aim of this challenge is to drive the cost down, and the reliability up, of small robot aircraft. The scoring system hence rewards teams heavily for low-cost designs, long duration capability and autonomy.

There are numerous other flying robot competitions run each year, including the USA-European micro UAV flight competition [66.48], the AUVSI Seafarer Annual Student UAS Competition [66.49], the International Universities MINI UAV Competition [66.50], the Chinese Robotics Competition which has a flying robot category [66.51], and the Annual Search and Rescue Cyber Physical Systems Challenge, where teams build ground vehicles and the organizers provide a video feed from a UAV hovering above the indoor search area [66.52, 53]. A final category of flying robot competitions are the type held solely for participation within single universities and which are typically part of core course work in robotics or extension course work. A representative example of this type of internal university competition is given by *Schochmann* [66.54].

Motivation for Flying Robot Competitions

The stated aim of the longest running flying robot contest, International Aerial Robotics Competition, is to drive innovation [66.40], particularly in the area of autonomy. Many of the flying robot competitions and challenges of the past decades have focussed on this technology development or innovated uses of existing technology. However, not all flying robotics competitions and challenges are created for this reason alone, with some being linked to the education of high-school or university students who are studying robotics, while others are conceived to overcome nontechnologically based problems. There are a number of key issues that are potential inhibitors to the adoption of flying robots in the civilian context. In the mid-2000s, the general public considered flying robots as a military-only tool, being expensive and dangerous. This was largely due to a number of popular movies at the time depicting such things and high media interest in the use of unmanned aircraft by the USA in the Middle East conflicts of the time. In the 2010s, the public was concerned with privacy issues surrounding the use of drones (as they have become known in the media) and the fear that small unmanned aircraft could be misused by criminals or terrorists [66.55].

A second significant issue is that of airspace regulation. Flying robots will share the sky with manned aircraft. The airspace inhabited by manned aviation is highly regulation across the world. Before flying robots can be routinely used for civilian missions, both the airspace regulator of a particular country, and the general public in that country must be comfortable that flying robots will be as safe as current conventionally piloted aircraft. This issue is the single most significant hurdle to the adoption of unmanned aircraft in civilian airspace. A third issue, strongly related to the regulation and safety issue, revolves around the appropriate training, certification, and insurance of unmanned aircraft. Insurance companies work on precedence and past accident rates in order to determine risk and hence set insurance premiums. In a new industry, such as that of unmanned aircraft, with little data on which to basis risk probabilities it is difficult for insurers to cover operators. Finally, as the unmanned aircraft industry grows, it will need people with the correct skills. These new

Fig. 66.5 **(a)** The aim of the Search and Rescue Challenge is to locate and drop a water bottle to a walker, Outback Joe. The mannequin wears bright clothes to help the teams find him using computer vision. **(b)** The course showing the Mission Boundary (*yellow*) and the Search Area (*green*). The airfield is at the top of the picture

Engineers, operators, technicians, and researchers will need to be trained in unmanned aircraft-specific areas of expertise.

The UAV Challenge Outback Rescue competition was created in Australia in 2007 [66.44] to attempt to simultaneously address all these major inhibitors to civilian use of flying robots. These broad aims meant that the targeted participants were from a wide cross-section of the community and included, the general public, high-school students, university students, radio-controlled aircraft hobbyists, the Australian aviation regulator (CASA), government, research institutes and universities, aerospace industry, and media.

Competition Formats and Challenges

The majority of flying robot competitions and challenges consist of multiple subevents or competitions, usually aimed at either different age groups of competitors or different skill levels. The level of complexity and the difficulty of the specific tasks set are set so that younger, less experienced students will almost always succeed to some degree and are hence competing against others like a sporting event. On the other hand, the events aimed at older or more experienced entrants are often deliberately conceived to be impossible to complete the first year that a team attempts it. These events are often referred to as Challenges while the easier events, where a team is expected to finish, are known as Competitions. A case in point is the UAV Challenge Outback Rescue that comprises a difficult Challenge, that is open to teams from around the world, and a high-school competition, that is far simpler, but still challenging, for high-school aged students. The two events share a common theme to encourage the high-school students to consider progressing to the Challenge in later years. The scenario for the UAV Challenge is that of locating a lost walker and dropping him a water bottle. Such a scenario is appealing to the general public, the media and the government.

The UAV Challenge Outback Rescue's major event is known as the Search and Rescue Challenge and was unique in the world at the time it began in 2007 in that the competition took place over a large area of farmland and public roads. The lost walker was represented by a mannequin and was placed in a farmer's field approximately 4–6.5 km from a public airport in the regional town of Kingaroy in Queensland, Australia. The aim of the Search and Rescue Challenge was for a team to use their flying robot to find the lost walker (Fig. 66.5) and then drop him a water bottle. If a team managed those tasks, returned their aircraft safely to the airport and the water bottle landed intact less than 100 m from the bushwalker, they would win $ 50 000. The Search and Rescue scenario was created to be as realistic as possible with competitors given a briefing document at the start of the day outlining where the lost walker was last seen and what he was wearing. Teams were under time pressure and given time limits on setup, mission time and pack up time. All of these restrictions were designed to make the scenario as close to a real rescue as it could be. Teams were evaluated on their mission performance and on their documentation meaning that even if no team completed the mission, their could still be a points winner, who would be awarded the title of First Place and receive a smaller cash prize. This is typical of Challenge-type events where a number of years may go by without a team completing the main challenge. Another typical feature of this event it is that it is not set up as a spectator event because the nature of the Challenge makes it almost impossible to follow if you are not directly involved in the action.

The high-school competition of the UAV Challenge Outback Rescue is known as the Airborne Delivery Challenge as takes place on the airport site and within the airport boundary. This competition is therefore set up to be a spectator friendly event. This scenario required high-school students to pilot their unmanned aircraft over the same walker, who is located in sight of the general public close to the air-side fence. Teams are required to drop a package to the walker, but the pilot cannot select when to make the drop. Instead, a second team member located in a tent and with no line of sight

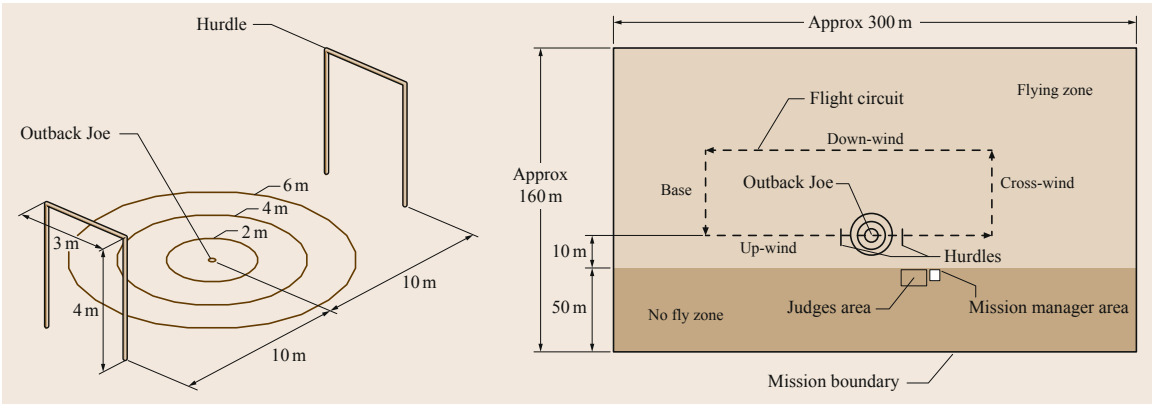


Fig. 66.6 The Airborne Delivery Challenge course showing the hurdles and the flying zone

to the bushwalker has to make the drop (Fig. 66.6). The aim of this scenario is to force teams to use some technology to help assess when to drop the package. Many teams are successful and a points system determines the winner.

Qualification and Precompetition Testing

Some flying robot competitions and challenges have a qualification element to them which means that teams must demonstrate a certain level of competence prior to the actual event. This requirement implies that many of these competitions can be considered to be much longer in duration than the relatively short events that take place on the competition days at the end. The UAV Challenge Outback Rescue for example, takes place over a long time period leading up to the event. Since 2007, the Search and Rescue Challenge has been held both over a single year and as a 2-year long campaign. In both forms of the event, the high-school and the international open event, teams must provide documentation to the organizers to demonstrate their competence, their progress, and their treatment of safety. Safety is the priority issue for the organizers and the teams at flying robot competitions and challenges. In the case of true UAV Challenge, the unmanned aircraft are operated over farmland, over public roads and close to the general public who come to spectate (Fig. 66.5).

Teams first register for the UAV Challenge and are then asked to provide documentation, known as Deliverables. The first deliverable must show a basic system design and an analysis of risk. The second deliverable must show the final design and adherence to the mandatory safety features of the UAV system. For the 2-year long version of the Search and Rescue Challenge, a third deliverable was added where teams were asked to show documented evidence of at least 5 h of autonomous flight of their unmanned aircraft. This final set of documentation was introduced to ensure that only teams that were capable of actually flying for a sustained period of time in the search range qualified for the event. By its nature, the qualification process significantly reduces the size of the field as a competition or challenge progresses. For example, in 2011–2012, 68 teams registered to take part in the Search and Rescue Challenge, with 61 submitting a the first deliverable document. Of those, 53 teams from ten countries, qualified to continue. At the second deliverable stage, 23 teams qualified to continue to the final deliverable phase where nine teams qualified and were invited to the actual flying event. Between that announcement and the event, four of the nine teams pulled out due to crashes during testing and other issues leaving just five teams to compete during the event (Fig. 66.7,66.8).

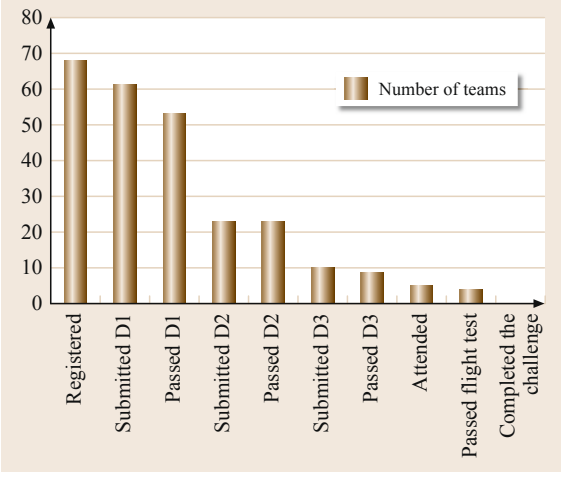


Fig. 66.7 The number of teams declines as the UAV Challenge progresses. In 2011–2012, 68 teams registered and throughout the 18 months of the competition teams dropped away leaving only 5 at the event

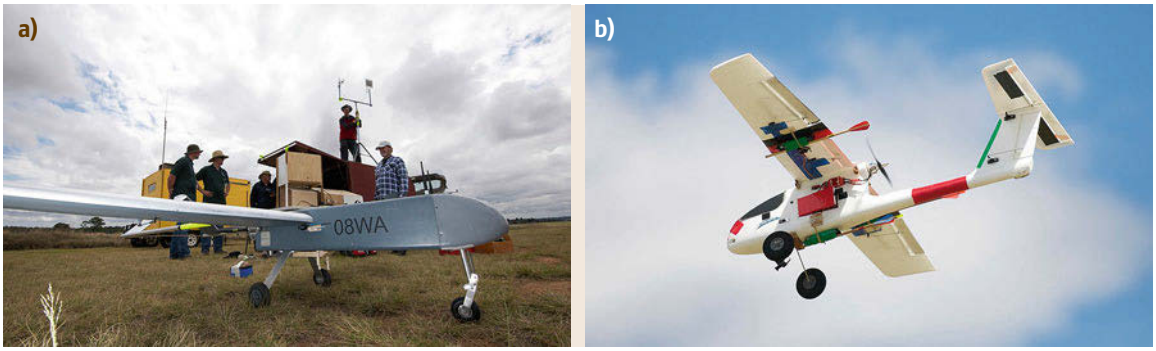


Fig.66.8a,b **(a)** First place team from the 2012 Search and Rescue Challenge, Canberra UAV. **(b)** Winners of the 2012 Airborne Delivery Challenge, MUROC Hawks

It is common at flying robot events to test teams prior to the actual competition flights. In the UAV Challenge Search and Rescue event, the final phase of qualification involves the teams being tested or scrutineered. The scrutineering process aims to test both the safety systems of the unmanned aircraft, the supporting systems and the team themselves, including any safety pilots. The scrutineering is undertaken by industry professionals and consists of a static scrutineering test where the team is interviewed, the aircraft and systems inspected and the flight termination tested. Teams that pass scrutineering are then invited to fly their aircraft within the boundary of the airport where the ability of the team to fly autonomous is tested. Only teams that pass this final hurdle are invited to launch a search and rescue mission and compete for the grand prize. In 2011–2012 Search and Rescue Challenge, four teams passed the final scrutineering tests meaning that just over 5% of teams that originally registered qualified for the final mission. In comparison in the 2013–2014 event, this qualification rate increased to 17%.

Regulations and Safety

Many flying robot competitions are carried out indoors. As well as guaranteeing that bad weather will not cause problems for the event, a major reason to hold a competition indoors is that in most countries, this means that the official airspace regulator does not need to be involved in any way. Unmanned aircraft flown indoors pose no threat to manned aviation flying above. However, a number of high-profile flying robot competitions do take place outdoors, including IARC and the UAV Challenge Outback Rescue. The latter event must be undertaken within the specific regulations for the operation of unmanned aircraft in Australia. These regulations were launched in 2002 by Australia's civilian air regulator, the Civil Aviation Safety Authority (CASA) and are known as CASR Part 101. The regulations outline the operating rules for three defined classes of unmanned aircraft, micro UAS under 100 g, small UAS 100 g to 150 kg and large UAS 150 kg and above. The rules of the UAV Challenge dictate that the competing team's cannot enter an aircraft in the large UAS class. Normal commercial operation of small UAS would require the operating business to obtain an Operators Certificate from CASA. However, the competition is classified Sport and Recreation by CASA and hence an Operator Certificate is not required. CASA treats the UAV Challenge organizers as the body accountable for the safe conduct of the event and adherence to the relevant regulations. The UAV Challenge organizers have developed the event rules to specifically meet the requirements of the regulations and to hence ensure that if a team is following the rules, they are in turn not in breach of the regulations. Further to this, the organizers have also developed a detailed Policy and Procedures manual that outlines how the UAV Challenge will be conducted to ensure safety and adherence to the regulations by the organizers themselves. Developing the manual and the supporting rules has been a major activity of the organizing committees over the years that the event has been run and is a significant output from the whole UAV Challenge activity.

What makes the UAV Challenge different to many other unmanned aircraft competitions is that because the event takes place largely beyond visual range of the team's ground controllers, and an aircraft could be up to 7 km from the ground station, there is a higher potential for a so-called fly-away than there is during the more common style of events where the flying takes place in close proximity to the ground station. A fly-away is where the aircraft flies out of the designated competition area and cannot be commanded to return. Such an incident could be seen as a major setback to the developing unmanned aircraft industry. In the first 2 years of the UAV Challenge, the mitigation for fly-away was to mandate the use of a flight

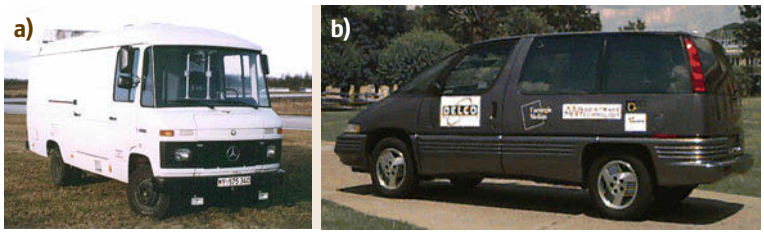


Fig.66.9a,b DARPA grand challenge vehicle ancestry. Pioneering autonomous vehicles of 1980s, 1990s. **(a)** 1985: UBM's Vamors. **(b)** 1995: CMU's Navlab 5 used for the *No Hands Across America* trial

termination mode whereby any aircraft that crossed the defined competition boundary or lost contact with the ground station had to automatically perform a fast dive into the ground. All aircrafts were tested during the scrutineering sessions to ensure that this behavior was built into the systems. Since, 2010, the requirements have kept step with the more common industry approach to lost communications and teams are now allowed to deploy aircraft that will automatically return to a predetermined location in the competition area should communications be lost between the ground station and the aircraft. The requirement to automatically terminate a flight on competition boundary crossing remains and is still the last line of defence against a fly away. Many teams have lost aircraft during the testing and development of their flight termination systems and this requirement is still one of the most challenging of the UAV Challenge.

Organization and Sponsorship

The nature of flying robot competitions and challenges often require large teams of organizers with complementary skills. For example, the UAV Challenge Outback Rescue is organized and run by two committees whose members typically come from the organizations that provide support for the event. The first committee is the Steering Committee and runs all nontechnical aspects of the UAV Challenge event such as equipment hire, securing sponsorship, liaising with local government and carrying out event communications and media activities. This committee is also responsible for the event finances. The second committee is the Technical Committee. This committee is responsible for developing and releasing the rules for the UAV Challenge. The committee also liaises with air regulator and works with the UAV Challenge Steering Committee on the event organization. At the event, the Technical Committee is responsible for the airside part of the UAV Challenge which includes selection of the technical team that runs all technical issues relating to the event, scrutineering of the teams, the supervision and scheduling of the actual competition flights and the coordination of aerodrome activities during the event times. The Technical Committee meets regularly throughout the year with initial meetings held to formulate the rules and incorporate lessons learned from the previous year. The Technical Committee reports to the Steering Committee.

Each time the UAV Challenge is run, sponsors are sought to cover the costs of the prize money and the hire of facilities. The organizations that come together to run the UAV Challenge provide the time of their staff as an in-kind contribution. The total budget of a UAV Challenge is approximately $125 000 plus the staff time of the organizers. It is hence a considerable undertaking and requires a significant number of sponsors to break even. The UAV Challenge has been well supported over the years, attracting the sponsorship of major aviation companies such as Boeing and Lockheed Martin as well as unmanned aircraft companies such as Insitu. The safety regulator, CASA, has also sponsored the event.

66.4.2 Case Study: The DARPA Challenges

In 2001, the United States Congress challenged the Department of Defence with the following mandate [66.56]:

> *It shall be a goal of the Armed Forces to achieve the fielding of unmanned, remotely controlled technology such that by 2015, one-third of the operational ground combat vehicles of the Armed Forces are unmanned.*

In response, a little over a year later, the Defense Advanced Research Projects Agency (DARPA), the primary research and development organization for the United States Department of Defense, announced that there would be a race.

The race, to be titled the *DARPA grand challenge* was to drive a robot car 300 miles from Los Angeles to Las Vegas in the fastest time to claim the $1-million dollar prize [66.57].

The aim of the DARPA grand challenge was to reinvigorate and extend autonomous vehicle technologies from the 1980s and 1990s to future defense vehicles. Initiatives such as the Universität der Bundeswehr Munich (UBM)'s Vamp/Vamors vehicles, University of Parma and Carnegie Mellon University's NavLab's research program (shown in Fig. 66.9) had demonstrated

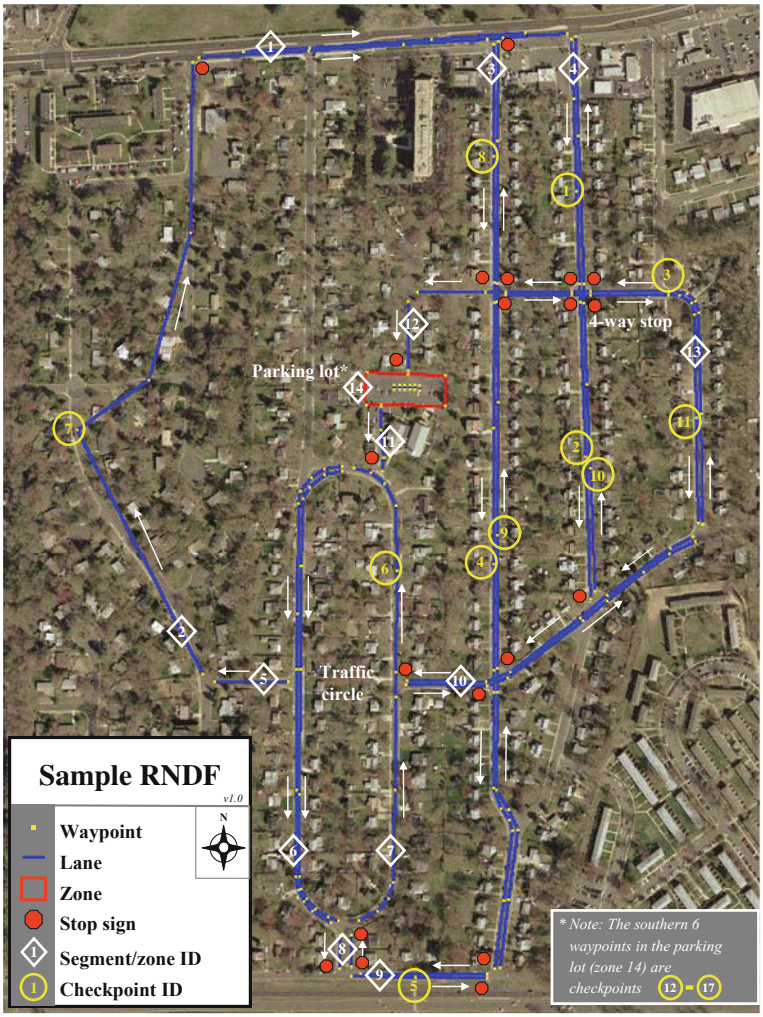


Fig. 66.10 A sample route network definition file (RNDF) provided by DARPA for the 2007 Challenge (overlaid on aerial imagery)

impressive feats of road vehicle autonomy. Trials across thousands of highway kilometres had demonstrated lane keeping, obstacle avoidance and even automatic overtaking (passing) [66.58–60].

The problem was, close to a decade later, the capability of current technology full-sized, fully autonomous vehicles was unknown. The impressive trials of the late 1980s and 1990s concentrated on highway driving where the primary task was computer vision-based lane keeping. There were indoor and outdoor robots navigating through challenging *obstacle field* style environments [66.61, 62]. There were also teleoperated vehicles already in use by the defense forces. The question was, given computation power tracking with Moore's law over the intervening years and many other advances (such as the use of LIDAR), what would robot car be capable of?

DARPA aimed to [66.63]:

- *Demonstrate an autonomous vehicle able to travel over rugged terrain at militarily relevant speeds and distances.*
- *Accelerate autonomous ground vehicle technology development in the areas of sensors, navigation, control algorithms, hardware systems, and systems integration.*
- *Attract and energize a wide community of participants not previously associated with Department of Defense programs or projects to bring fresh insights to the autonomous vehicle problem.*

Potential participants were drawn to the challenge. Many participants could also see the potential for related technologies in the automotive industry and for road safety [66.64].

Teams would develop a vehicle that could follow a simple route across a course representing typical ter-

rain encountered by defense vehicles. The route map would be a set of paths specified by GPS way-points. The vehicle would be then given a list of destinations in the map and instructed to drive to each location. A typical route map is shown in Fig. 66.10 overlaid on the satellite image.

For this first competition, as for those that followed, the rules stated that the robot must be a fully autonomous and unmanned ground vehicle. Each vehicle was fitted with a wireless remote E-stop to bring the vehicle to a halt. On the base vehicle was added sensing, processing, and control. Some vehicles were built from the ground up. Most teams fit actuators to the manual controls of an existing vehicle, some built upon disability aid kits fitted to the vehicle. A couple teams were able to tie into existing *by-wire* capabilities of the vehicle, a notable example being Victor Tango's 2007 challenge entry *Odin* [66.65]. In addition to GPS, vehicles needed to have sensing for obstacle avoidance, in which LIDAR sensors dominated. As the vehicles were to be completely autonomous, all sensor processing, and control had to be done on-board so a certain amount of computer power was incorporated. Then the practicalities would necessitate other devices such as power generators and additional air conditioning, if required. The general anatomy of a DARPA grand challenge vehicle is illustrated at Fig. 66.11. This vehicle, from the 2007 DARPA Urban Challenge, was at the upper end of the spectrum for additional equipment with LIDAR, video cameras and automotive radar, as well as a generator and supplementary air conditioning required for a blade server computer housed in the rear compartment [66.66].

2004 DARPA Grand Challenge

DARPA hired *Score international*, an off-road race organizer, to develop the course [66.67]. A 142-mile course was laid out through the Mojave Desert. The competition was structured as a time trial to complete the course in the shortest time under 10 h. One hundred and six teams applied.

In March 2004, 25 teams turned up for the competition. In the days leading up to the race a qualifier, inspection and demonstration (QID) round was conducted. This event was primarily to verify that the vehicles were fit to compete and capable of operating safely. By race day, the 15th of March 2004, a field of 15 vehicles had qualified and were ready to race.

As stated earlier, a primary goal of the competition was to gauge the state the art in the field. Steep elevation, blind turns, and sheer drops adjacent to the course were included to challenge the vehicles. While the degree of difficulty was low by human off-road driving standards, the course was designed to be interesting and make for a good competition [66.67]. Expecting something akin to human driven competition, DARPA even advised the media to set up at the finishing line where the action was expected to take place [66.68].

However, the course and the autonomous driving problem proved too difficult for the robot vehicles. No vehicle made it beyond the first seven miles of the course. After a promising start the vehicles soon became stuck or had to be disabled by a safety official after deviating from the course. One vehicle's wheel caught on fire, another flipped around a bend another began circling around in the starting chute [66.69].

Inexperience was the chief cause of failure on the 2004 contest. The new nature of the challenge and the stretch of mobile robotics to a full-sized off-road vehicle introduced too many complications for the robots systems.

The challenge stood unbeaten, the prize money remained unclaimed. The real world complications and the size of the challenge were now apparent. Also apparent, however, was the curiosity sparked within the research community and the general public.

Three months after the first competition, DARPA announced the 2005 DARPA Grand Challenge – a rerun of the event, this time for a $2-million dollar prize.

2005 DARPA Grand Challenge

With a slightly longer lead time, DARPA held a participants' conference and conducted site visits to each team to gauge progress. At the participants' conference team representatives and DARPA officials discussed the draft race rules, clarifying rule interpretations and technical specifications. Out of 195 entrants, DARPA officials flew out to attend 118 site visits.

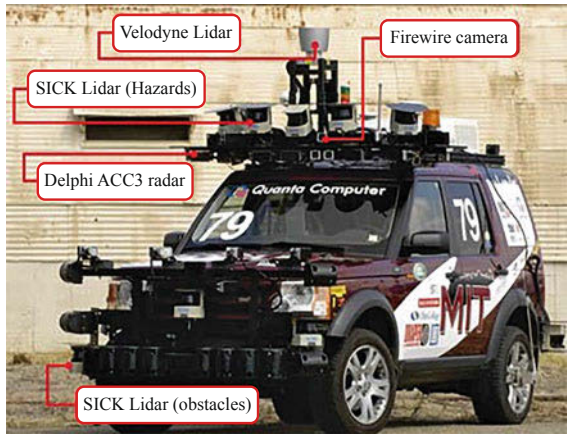


Fig. 66.11 Anatomy of a DARPA grand challenge vehicle. 2007 DARPA Urban Challenge entrant MIT's *Talos* was on the upper end of the additional equipment spectrum

From the site visits 43 teams were selected to participate in the national qualifying event (NQE). During the qualifying event, which was twice as long as 2004, the teams had to complete a miniature version of the course which was created at the California Speedway (Fontana, California, Fig. 66.12).

By race day, 23 robots were selected as starters in the race. The vehicles queued up to start at 5-min intervals. DARPA officials paused slow-moving vehicles to permit faster robots to pass.

The difference between the 2004 and 2005 competitions was stark. Eighteen months on from the first competition, five robots drove 132 miles across the Mojave Desert to complete the course [66.63].

The race winner *Stanley*, from the Stanford Racing team, claimed the prize – completing the course in 6 h, 53 min and 58 s (Fig. 66.13) [66.70]. Second and third places were taken by robots from Carnegie Mellon University (CMU) teams: *Red Team* and *Red Team Too* [66.63]. *Kat-5* from Team Gray came forth, a remarkable achievement from a team founded by Louisianan insurance executives. And last, but not least, the 16-ton *TerraMax* truck by the Ochkoch Truck Corporation which completed the course the following day after spending the night *paused* on the course.

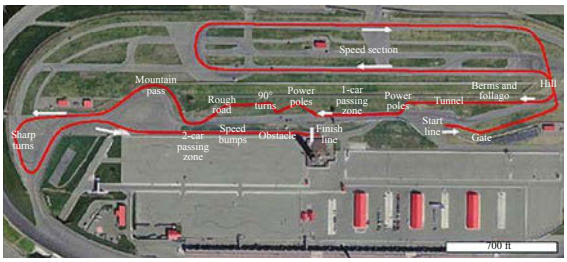


Fig. 66.12 Miniature version of the course which was created at the California Speedway

Fig. 66.13 Winner of the 2005 grand challenge Stanford Racing's robot *Stanley*

It was a trail blazing event which captivated the public with news images of unoccupied race vehicles kicking up dust as they drove across the Mojave desert sand.

During the qualifiers and the race some common weaknesses were also identified. The vehicles required GPS for navigation as all the course information were given GPS coordinates, however the vehicles also appeared to be overly reliant on a strong, correct, GPS signal. During the race vehicles drove under high voltage power lines which caused GPS loss. Vehicles deviated from the course or even crashed due to loss or multipath signals. Several robots were also able to make significant progress *driving blind* after sensor failures meant that local obstacle avoidance was no longer functioning correctly.

In 2006, DARPA announced the next grand challenge, the DARPA *Urban Challenge*. The contest this time required fully autonomous vehicles to drive on an urban street network.

2007 DARPA Urban Challenge

In the DARPA Urban Challenge, robot vehicles were to follow the Californian driver's handbook. Unlike the previous challenge where vehicles were not intentionally presented with moving obstacles (other robots were paused when they became too close to each other) the aim of the contest was to test the vehicles' ability to drive in traffic. This time vehicles were expected to keep to lanes, queue in traffic, park, merge, pass, and even observe intersection precedence (i. e., yield to cars already at an intersection) [66.71].

In a new element in the 2007 Challenge, DARPA introduced seed funding of up to $1-million dollars for per team for 11 teams based on a competitive application process to bootstrap the cost of development of the increasingly complex robot platform. The seed funding attracted new entrants to the competition in line with DARPA's aim to attract a wide community of participants [66.72].

On November 3, 2007, the urban challenge event (xUCE) was held in Victorville, California. This event set loose, simultaneously, for the first time, 11 full-size autonomous vehicles on a closed course (Fig. 66.14).

The UCE was held on a closed course within the decommissioned George air-force base. The course was predominantly the street network of the residential zone of the former base with several graded dirt roads added for the competition. The contest was cast as a race against time to complete three missions. The missions were different from each team but were designed to require each team to drive 60 miles to finish the race. Penalties for erroneous or dangerous behavior were also converted into time penalties.

Most roads were paved with a single lane in each direction as typical for a suburban street. A couple of roads had two lanes of traffic in each direction to simulate an arterial road or highway. One road, in the southeastern corner of the network, was a raised dirt road constructed especially for the event.

This time all 11 qualifying robots were allowed to interact in the UCE course simultaneously, with additional traffic vehicles driven by professional drivers. For the third time, like anxious parents, the researchers manned the fences helpless as the robots, whose development had consumed the past 12 months of their lives, one by one, drove away. There were some tense moments and even a fender bender [66.66, 73]. However, powering through the mission objectives with a driving style described as *a soccer mom with some place to be* [66.74], CMU's Tartan racing's robot *Boss* (Fig. 66.15), crossed the finish line [66.73] to claim the prize. 2005 winning team Stanford Racing came second with *Junior* while team Victor Tango's robot *Odin* secured third place. Six robots completed the race [66.72].

Review

DARPA endorses the use of competitions to cause paradigm shifting advances in science and technology. As stated DARPA's 2007 Fiscal year report [66.72]:

Fig. 66.14 DARPA starters

Fig. 66.15 Tartan racing's robot Boss crossing the finishing line

The program achieved its program goals and stimulated interest in the programs and projects of interest to the DoD Science and Technology (S&T) community. It was successful in attracting considerable joint investment by the participants and their sponsors, effectively leveraging Government investment in the program. The technical challenge was carefully defined and staged to bring coherence to the community and increase the chance for cross-fertilization among competing groups. The solicitation and qualification process was successful in attracting a large pool of strong teams with participation from the defense industry, automotive industry, academia, as well as a number of smaller organizations. This investment in expanding the community will continue to pay dividends as DoD benefits from a strengthened commercial sector autonomous vehicle technical community. The program has been successful in attracting many young people to work on S&T problems in areas affecting national security, and benefits are expected to accrue for many years as this group enters the work force.

Although DARPA's approach may raise the question of whether research initiated by the competitions should be supported beyond competition, DARPA's view is that its role is to kick start ground breaking research not bring it to completion [66.1]. They encourage research teams and commercial organizations to team up to bring the technology to market for defense vehicles [66.72].

Velodyne LIDAR, a dominant sensing technology in the 2007 grand challenge, was developed with road safety potential in mind by Team DAD from the 2005 DARPA grand challenge [66.75]. The invention is a good example of the technological pull DARPA was wanting to achieve with the competition.

Another direct outcome of the competitions is the Google self-driving car project, which is based on the research and researchers from grand challenge teams [66.76]. Many vehicle manufacturers, such as Ford, General Motors, Volkswagon, and Toyota among others, have either entered the competition, partnered with teams or employ competition alumni to continue to develop and apply related technologies to road safety [66.64, 70, 72].

The DARPA Robotics Challenge

Buoyed by the success of the grand challenges, DARPA now hopes to accelerate the development of humanoid rescue robotics. In April, 2012 DARPA proposed the DARPA robotics challenge. The Challenge is to develop an adaptable ground robot capable of completing tasks in a dangerous and degraded environments, using hand

Fig. 66.16 DARPA robotics challenge conceptual illustration

tools to driving vehicles for disaster relief operations (as illustrated in Fig. 66.16 [66.77].

Similar to the first grand challenge, an ambitious goal has been set to inspire and stretch the state of the art.

Though it is important to note that the new robotics challenge is a separate research program from the previous grand challenges, some best practices and lessons learned from the previous Challenges have been adopted. Entrants are divided into four tracks. Tracks A and D compete to develop hardware and software of a complete robotic system. Tracks B and C compete to develop software first to run on a common simulated robot. Tracks B and C will then compete in a virtual competition. In 2014, successful Track B and C teams will be able to develop and deploy the software to a common robot platform provided by DARPA. The entrants can also apply for seed funding (similar to the), which differentiates teams between Tracks A and B, who have DARPA seed funding, and C and D who are self-funded [66.77].

The idea of using a common platform across teams and a simulator-based challenge addresses the observation that successful teams in the 2007 challenge had arrived upon similar sensors and sensor configurations. So a future challenge could remove the significant hardware development burden by allowing teams to compete on a common platform. The motivation is to once again broaden field of participants [66.77].

The new robotics challenge also introduces access to a remote human supervisor. Previous challenges were strictly fully autonomous. A criticism had been that due to the extreme lack of contextual information available to most robotic systems, simple ambiguities that arise can impede a robot unnecessarily if a supervisor can correct the problem. The lack of supervisory control meant that in previous challenges vehicles would appear *overly cautious*.

For example, many vehicles in the 2004 Challenge were incapacitated by trivial problems such as the robot misjudging which side of a road boundary they were actually on. Full autonomy should be the ultimate aim; however, allowing limited supervisory contact would have permitted the 2004 contest to continue allowing more trial time for teams.

Conclusion

DARPA never aimed to create the ultimate autonomous vehicle. The true aim of the competition was to advance the science and technology research and education underpinning the original Congressional mandate on autonomous vehicles.

Of course, the exercise has come with considerable cost. From the 2007 Fiscal report [66.72] the true dollar cost to DARPA of the competition came in at around 25 million dollars. Three and a half million dollars were paid directly to the winning teams as prize money. A further $ 11 million was paid in *Track A* grants to teams, with the remaining $ 10.5 million was used to host the participants' conference, site visits, qualifying round, and stage the final event.

Notwithstanding these costs and the necessarily limited nature of these competitions – which create an artificial environment to focus a broad group of experimenters on a common goal, for a given period of time – they have brought clear benefits for DARPA, and the field of robotics.

Regardless of the percentage of autonomous vehicles fielded by the Department of Defense in 2015, the experience shared by participants from: research institutions, car manufacturers, defense contractors, technology companies, schools, a loud speaker company [66.75] and even insurance executives [66.78] to attempt the challenge of creating a full size, fully autonomous, self-driving car has gathered the field for an even greater race. The race between companies such as Google, Mercedes Benz, Toyota and their technology suppliers to bring these vehicles to market not only for defense vehicles but for all road vehicles.

An outcome beyond the dreams of the forward thinkers in the 2001 Congress.

66.5 Conclusion and Further Reading

In this chapter, we have discussed several types of robot competitions highlighting their impact on research and education. Here we summarize the manifold impacts of competitions as well as the major issues that need to be addressed to design and run them.

Robotics competitions with ambitious aims can provide significant breakthroughs in the development of science and technology.

To this end, competition models based on one time challenges have the advantage of being able to headline a fresh new goal. The drawback for the organizer and participants is the significant effort to build systems and infrastructure that may be too specialized to be used after the competition for subsequent challenges. On the other hand, challenges that are developed across the years can benefit from a continuous development, as long as they suitably build on yearly achievements, without loss of focus on the long-term roadmaps. The two approaches as outlined in the case studies demonstrate the merit of both as well as the unquestionable impact on science and technology.

Robots are complex systems composed of many mechanical, electrical, software, and algorithmic components and as such are hard to test and compare. In isolation a test can be derived to evaluate some components or subcomponents. For example, one can test the degrees of freedom of a mechanical design, or the bandwidth and latency of a data bus. Image processing and mapping algorithm development have made great strides using shared datasets to compare results. The performance of a robot system as a whole would often fall back to a qualitative assessment presented as a video accompanied by the encouraging words of the researcher. Competitions provide the motivation and the forum for robotics researchers to undertake the significant effort (in time, compatibility, and transportation) to develop and validate innovative solutions on complex systems under controlled conditions.

Robot competitions have a positive impact on robotics research groups in that the team work requirement within and between groups means that a better standard of core software is achieved across the group. Good tools in terms of software libraries and best practice by the more experienced software engineers in a group are often adopted across the team enabling algorithms to make it onto robot hardware sooner. To this end, it is critical that competitions support the sharing of solutions and tools, by developing community infrastructure and resources. This kind of gently enforced openness keeps competitions competitive, fosters interactivity between teams and encourages diversity of participants by minimising the barriers to entry.

Robot competitions can play a tremendous role in education. They are attractive to students and teachers, while providing an interdisciplinary framework to support the development of several types of technical skills, including teamwork. Maintaining a good balance between competing and learning becomes a major concern when the focus of the competition is education. In international competitions with younger students, differences in culture and language need to be carefully taken into consideration to maintain fairness and inclusiveness.

Last, but not least, robot competitions have a significant impact on public opinion and the public understanding of the science and technology. Simple distinctions such as autonomous versus teleoperated robotics are often mistaken or overlooked by the news media and the public. Human-inspired competitions have attracted many spectators and the attention of the media, and they certainly contribute to make scientific progress more accessible, through the language of sports. Grand challenges, such as the DARPA Challenges have been extensively covered by the media, thus, contributing, although from a different perspective to disseminate the awareness on the progress of robotics in the society.

The aforementioned outcomes do not come without a significant effort. A successful competition must clearly define a target participation and suitably support it, by making the entry level accessible to a large number of participants. The technical challenge needs to be appropriately compelling and attractive; moreover, the competition setting must ensure that winning is achieved through innovation and scientific progress. This is achieved by designing the requirements on the robots and on the competition scenario as well as through a well-designed performance assessment.

The organization of competitions also comes with formidable challenges: logistics for transporting robots and teams, venue and infrastructure for the competition event, safety of participants and spectators. The financing and administration, often underestimated, also requires substantial preplanning, know-how, and resources.

This notwithstanding, we expect the use of competitions to motivate, educate, and foster research innovation only to increase in coming years as competition has proven to be a particularly useful tool for robotics research.

66.5.1 Further Reading

Research results of robot competitions are often accumulated into a journal special issue. The reader is

encouraged to explore the articles as they are often approachable and can include interesting details of events during the contests. For the DARPA grand challenge the reader may refer to the *Journal of Field Robotics* Volume 23 (2006) Issues 8 and 9 or the *Springer Tracts in Advanced Robotics* Volume 36. For the DARPA the reader may refer to the *Journal of Field Robotics* Volume 25 (2008) Issues 8 and 9 or the *Springer Tracts in Advanced Robotics* Volume 56. In future, similar special issues are likely to be available on the DARPA robotics challenge.

US Innovation Programs [66.79] is interesting review on the broader discussion of competitions in funding science and technology research.

Video-References

VIDEO 385 Brief history of RoboCup robot soccer
available from http://handbookofrobotics.org/view-chapter/66/videodetails/385

VIDEO 387 Multirobot teamwork in the CMDragons RoboCup SSL team
available from http://handbookofrobotics.org/view-chapter/66/videodetails/387

References

66.1 D.D. Stine: U.S. innovation programs. In: *Federally Funded Innovation Inducement Prizes*, ed. by D.D. Stine (Nova Press, Hauppauge 2011) pp. 1–38

66.2 R. Allan: The amazing Micromice: See how they won. Probing the innards of the smartest and fastest entries in the Amazing Micro-Mouse Contest, IEEE Spectrum **16**(9), 62–65 (1979)

66.3 E. Ackerman: Meet the new world's fastest micromouse robot (IEEE Spectrum Magazine, November 2011)

66.4 FIRST Organization, available from http://www.usfirst.org/

66.5 A.K. Mackworth: On seeing robots. In: *Computer Vision: Systems, Theory and Applications*, ed. by A. Basu, X. Li (World Scientific, Singapore 1993) pp. 1–13

66.6 M.K. Sahota: Reactive deliberation: An architecture for real-time dynamic control in dynamic environments, Proc. 12th Natl. Conf. Artif. Intell. (AAAI) (1994) pp. 1304–1308

66.7 H. Kitano (Ed.): *Proc. IROS-96 Workshop on RoboCup* (1996)

66.8 Ansari X Prize foundation, available from http://space.xprize.org/ansari-x-prize

66.9 World Robot Olympiads, available from http://www.wroboto.org/

66.10 Robogames (formerly ROBOlympics), available from http://robogames.net/

66.11 Federation of International Robot Soccer Federation, available from http://www.fira.net/

66.12 RoboCup, available from http://www.robocup.org/

66.13 World Robotic Sailing Championship, available from http://www.roboticsailing.org/

66.14 International Autonomous Robot Racing Competitions, available from http://robotracing2012.com/

66.15 J.-H. Kim (Ed.): *A Booklet on MIROSOT'96 (Micro Robot World Cup Soccer Tournament)*, (MIROSOT Organizing Committee, 1996)

66.16 H. Kitano (Ed.): *Robot Soccer World Cup I* (Springer, Berlin, Heidelberg 1998)

66.17 M. Asada, H. Kitano (Eds.): *Robot Soccer World Cup II* (Springer, Berlin, Heidelberg 1999)

66.18 M. Veloso, E. Pagello, H. Kitano (Eds.): *RoboCup 1999: Robot Soccer World Cup III* (Springer, Berlin, Heidelberg 2000)

66.19 P. Stone, T. Balch, G. Kraetzschmar (Eds.): *Robot Soccer World Cup IV* (Springer, Berlin, Heidelberg 2001)

66.20 A. Birk, S. Coradeschi, S. Tadokoro (Eds.): *Robot Soccer World Cup V* (Springer, Berlin, Heidelberg 2002)

66.21 G. Kaminka, P. Lima, R. Rojas (Eds.): *Robot Soccer World Cup VI* (Springer, Berlin, Heidelberg 2003)

66.22 D. Polani, B. Browning, A. Bonarini, K. Yoshida (Eds.): *Robot Soccer World Cup VII* (Springer, Berlin, Heidelberg 2004)

66.23 D. Nardi, M. Riedmiller, C. Sammut, J. Santos-Victor (Eds.): *Robot Soccer World Cup VIII* (Springer, Berlin, Heidelberg 2005)

66.24 A. Bredenfeld, A. Jacoff, I. Noda, Y. Takahashi (Eds.): *Robot Soccer World Cup IX* (Springer, Berlin, Heidelberg 2006)

66.25 G. Lakemeyer, E. Sklar, D.G. Sorrenti, T. Takahashi (Eds.): *Robot Soccer World Cup X* (Springer, Berlin, Heidelberg 2007)

66.26 U. Visser, F. Ribeiro, T. Ohashi, F. Dellaert (Eds.): *Robot Soccer World Cup XI* (Springer, Berlin, Heidelberg 2008)

66.27 L. Iocchi, H. Matsubara, A. Weitzenfeld, C. Zhou (Eds.): *Robot Soccer World Cup XII* (Springer, Berlin, Heidelberg 2009)

66.28 J. Baltes, M. Lagoudakis, T. Naruse, S. Shiry Ghidary (Eds.): *Robot Soccer World Cup XIII* (Springer, Berlin, Heidelberg 2010)

66.29 J. Ruiz del Solar, E. Chown, P.-G. Ploeger (Eds.): *RoboCup 2010: Robot Soccer World Cup XIV* (Springer, Berlin, Heidelberg 2011)

66.30 T. Roefer, N.M. Mayer, J. Savage, U. Saranli (Eds.): *Robot Soccer World Cup XV* (Springer, Berlin, Heidelberg 2012)

66.31 X. Chen, P. Stone, L.E. Sucar, T. Van der Zant (Eds.): *RoboCup 2012: Robot Soccer World Cup XVI, Mexico City* (Springer, Berlin, Heidelberg 2013)

66.32 S. Behnke, M. Veloso, A. Visser, R. Xiong (Eds.): *Robot Soccer World Cup XVII* (Springer, Berlin, Heidelberg 2014)

66.33 H. Kitano, M. Tambe, P. Stone, M. Veloso, S. Coradeschi, E. Osawa, H. Matsubara, I. Noda, M. Asada: The RoboCup synthetic agent challenge, Proc. IJCAI-97 (1997)

66.34 S. Zickler, T. Laue, O. Birbach, M. Wongphati, M. Veloso: SSLvision: The shared vision system for the RoboCup Small Size League, Lect. Notes Artif. Intell. **5949**, 425–436 (2010)

66.35 E. Olson, J. Strom, R. Morton, A. Richardson, P. Ranganathan, R. Goeddel, M. Bulic, J. Crossman, B. Marinier: Progress toward multirobot reconnaissance and the MAGIC 2010 competition, J. Field Robotics **29**(5), 762–792 (2012)

66.36 A. Jacoff, R. Sheh, A.-M. Virts, T. Kimura, J. Pellenz, S. Schwertfeger, J. Suthakorn: Using competitions to advance the development of standard test methods for response robots, Proc. 2012 Perform. Metrics Intell. Syst. Workshop (PerMIS'12) (2012)

66.37 ELROB competition: http://www.elrob.org/

66.38 AUVSI competition: http://www.auvsifoundation.org/Competitions/CompetitionCentral

66.39 T. Wisspeintner, T. van der Zant, L. Iocchi, S. Schiffer: RoboCup@Home: Scientific competition and benchmarking for domestic service robots, Interact. Stud. **10**(3), 393–428 (2009)

66.40 R.C. Michelson: Autonomous Aerial Robots, Unmanned Syst. **29**(10), 38–42 (2011)

66.41 A. Visser: The international micro air vehicle flight competition as autonomy benchmark, Robotics Compet.: Benchmarking, Technol., Transfer Educ. Workshop – Eur. Robotics Forum (2013)

66.42 DSO National Laboratories, The Science Centre Singapore: Singapore amazing flying machine, Competition Challenge Booklet (2014)

66.43 M.J. Er, S. Yuan, N. Wang: Development control and navigation of Octocopter, 10th IEEE Int. Conf. Control Autom. (ICCA) (2013) pp. 1639–1643

66.44 R. Roberts, J. Walker: Flying robots to the rescue, IEEE Robotics Autom. Mag. **17**(1), 8–10 (2010)

66.45 D. Erdos, A. Erdos, S.E. Watkins: An experimental UAV system for search and rescue challenge, Aerosp. Electronic Syst. Mag. IEEE **28**(5), 32–37 (2013)

66.46 D.C. Macke, S.E. Watkins, T. Rehmeier: Creative Interdisciplinary UAV Design, IEEE Potentials Mag. **33**(1), 12–15 (2014)

66.47 D. Werner: Making way for unmanned aircraft, Aerospace Am., January, 28–37 (2014)

66.48 W. Oomkens, M. Mulder, M.M. Van Paassen, M.H.J. Amelink: UAVs as aviators: Environment skills capability for UAVs, IEEE Int. Conf. Syst. Man Cybern. (2008) pp. 2426–2431

66.49 AUVSI Seafarer Chapter: 2014 Rules for AUVSI Seafarer Chapter 12th Annual Student UAS (SUAS) Competition (2013)

66.50 B. Vidolov, J. De Miras, S. Bonnet: AURYON – A mechatronic UAV project focus on control experimentations, Int. Conf. Comput. Intell. Model., Control Autom., 2005 Int. Conf. Intell. Agents, Web Technol. Internet Commer., Vol. 1 (2005) pp. 1072–1078

66.51 Z. Fucen, S. Haiqing, W. Hong: The object recognition and adaptive threshold selection in the vision system for landing an unmanned aerial vehicle, Int. Conf. Inf. Autom. (ICIA) (2009) pp. 117–122

66.52 Cyber Alaska: Rules of The First Annual CYBERAlaska CyberPhysical Systems (2013)

66.53 O. Lawlor, M. Moss, S. Kibler, C. Carson, S. Bond, S. Bogosyan: Search and rescue robots for integrated research and education in cyber-physical systems, 7th IEEE Int. Conf. e-Learn. Ind. Electron. (ICELIE) (2013) pp. 92–97

66.54 R. Schochmann, M. Suchy, J. Pilka, I. Pestun, M. Kusenda: The impact of student's projects: A self reflection, 14th Int. Conf. Interact. Collab. Learn. (ICL) (2011) pp. 553–555

66.55 W. Grespin: Drones in our world, Part VI: Barriers to adoption, Diplomatic Courier (2014)

66.56 United States Congress: National Defense Authorization Act for Fiscal Year 2001 (United States Congress, Washington 2001), Sect. 217, p. 2549

66.57 Defense Advanced Research Projects Agency (DARPA): DARPA Grand Challenge Announcement 2004, http://archive.darpa.mil/grandchallenge04/media/announcement.pdf (2003)

66.58 E.D. Dickmanns, V. Graefe: Applications of dynamic monocular machine vision, Mach. Vis. Appl. **1**(4), 241–261 (1988)

66.59 D.J.T. Pomerleau: Rapidly adapting machine vision for automated vehicle steering, IEEE Expert Mag. **11**(2), 19–27 (1996)

66.60 R. Gregor, M. Lutzeler, M. Pellkofer, K.-H. Siedersberger, E.D. Dickmanns: EMS-Vision: A perceptual system for autonomous vehicles, IEEE Trans. Intell. Transp. Syst. **3**(1), 48–59 (2002)

66.61 S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte, D. Schultz: Probabilistic algorithms and the interactive museum tour-guide robot minerva, Int. J. Robotics Res. **19**, 972–999 (2000)

66.62 M.W.M.G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, M. Csorba: A solution to the simultaneous localization and map building (SLAM) problem, IEEE Trans. Robotics Autom. **17**(3), 229–241 (2001)

66.63 Defense Advanced Research Projects Agency (DARPA): Grand Challenge 2005 Report to Congress, http://archive.darpa.mil/grandchallenge/docs/Grand_Challenge_2005_Report_to_Congress.pdf (2005)

66.64 J.R. McBride, J.C. Ivan, D.S. Rhode, J.D. Rupp, M.Y. Rupp, J.D. Higgins, D.D. Turner, R.M. Eustice: A perspective on emerging automotive safety applications, derived from lessons learned through participation in the DARPA Grand Challenges, J. Field Robotics **25**(10), 808–840 (2008)

66.65 A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, C. Reinholtz, D. Hong, A. Wicks, T. Alberi, D. Anderson, S. Cacciola, P. Currier, A. Dalton, J. Farmer, J. Hurdus, S. Kimmel, P. King, A. Taylor, D.V. Covern, M. Webster: Odin: Team VictorTango's entry in the DARPA Urban Challenge, J. Field Robotics **25**(8), 467–492 (2008)

66.66 L. Fletcher, S. Teller, E. Olson, D. Moore, Y. Kuwata, J. How, J. Leonard, I. Miller, M. Campbell, D. Huttenlocher, A. Nathan, F.R. Kline: The MIT–Cornell collision and why it happened, J. Field Robotics **25**(10), 775–807 (2008)

66.67 Defense Advanced Research Projects Agency (DARPA): DARPA Grand Challenge FAQ 2004, http://archive.darpa.mil/grandchallenge04/faq.asp (2004)

66.68 Defense Advanced Research Projects Agency (DARPA): DARPA Grand Challenge FAQ 2007, http://archive.darpa.mil/grandchallenge04/media_briefing.pdf (2007)

66.69 Defense Advanced Research Projects Agency (DARPA): DARPA Grand Challenge Final Data 2004, http://archive.darpa.mil/grandchallenge04/media/final_data.pdf (2004)

66.70 S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, P. Mahoney: Stanley: The robot that won the DARPA Grand Challenge, J. Field Robotics **23**(9), 661–692 (2006)

66.71 Defense Advanced Research Projects Agency (DARPA): DARPA Grand Challenge FAQ 2007, http://archive.darpa.mil/grandchallenge/faq.asp (2007)

66.72 Defense Advanced Research Projects Agency (DARPA): DARPA Fiscal Year 2007 Annual Report, http://archive.darpa.mil/GRANDCHALLENGE/docs/DDRE_Prize_Report_FY07.pdf (2008)

66.73 C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M.N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T.M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, D. Ferguson: Autonomous driving in urban environments: Boss and the urban challenge, J. Field Robotics **25**(8), 425–466 (2008)

66.74 P. Henderson: SUV with mind of its own wins robot car race, Reuters (Nov. 4 2007)

66.75 Velodyne Inc.: Velodyne Lidar Origins, avaliable online at http://velodynelidar.com/lidar/hdlabout/origins.aspx (2005)

66.76 J. Markoff: Google cars drive themselves, in traffic, available online at http://www.nytimes.com/2010/10/10/science/10google.html (October 2010)

66.77 Defense Advanced Research Projects Agency (DARPA): 2012 DARPA Robotics Broad Agency Announcement (BAA), https://www.fbo.gov/index?s=opportunity&mode=form&id=5cf3cc1f46103cc0aba5d0f9412cab25&tab=core&_cview=1 **12**(39) (2012)

66.78 L. Grossman: Building the best driverless robot car, Time (online): http://www.time.com/time/magazine/article/0,9171,1684543,00.html, (November 2007)

66.79 A.E. Berger, M.C. Koepel (Eds.): *U.S. Innovation Programs* (Nova Press, Hauppauge 2011)

Multimedia Contents

Part G Robots

Part G Robots and Humans

Ed. by Daniela Rus

67 Humanoids
Paul Fitzpatrick, Genoa, Italy
Kensuke Harada, Tsukuba, Japan
Charles C. Kemp, Atlanta, USA
Yoshio Matsumoto, Tsukuba, Japan
Kazuhito Yokoi, Tsukuba, Ibaraki, Japan
Eiichi Yoshida, Tsukuba, Ibaraki, Japan

68 Human Motion Reconstruction
Katsu Yamane, Pittsburgh, USA
Wataru Takano, Tokyo, Japan

69 Physical Human–Robot Interaction
Sami Haddadin, Hannover, Germany
Elizabeth Croft, Vancouver, Canada

70 Human–Robot Augmentation
Massimo Bergamasco, Pisa, Italy
Hugh Herr, Cambridge, USA

71 Cognitive Human–Robot Interaction
Bilge Mutlu, Madison, USA
Nicholas Roy, Cambridge, USA
Selma Šabanović, Bloomington, USA

72 Social Robotics
Cynthia Breazeal, Cambridge, USA
Kerstin Dautenhahn, Hatfield, UK
Takayuki Kanda, Kyoto, Japan

73 Socially Assistive Robotics
Maja J. Matarić, Los Angeles, USA
Brian Scassellati, New Haven, USA

74 Learning from Humans
Aude G. Billard, Lausanne, Switzerland
Sylvain Calinon, Martigny, Switzerland
Rüdiger Dillmann, Karlsruhe, Germany

75 Biologically Inspired Robotics
Fumiya Iida, Cambridge, UK
Auke Jan Ijspeert, Lausanne, Switzerland

76 Evolutionary Robotics
Stefano Nolfi, Rome, Italy
Josh Bongard, Burlington, USA
Phil Husbands, Brighton, UK
Dario Floreano, Lausanne, Switzerland

77 Neurorobotics: From Vision to Action
Patrick van der Smagt, Munich, Germany
Michael A. Arbib, Los Angeles, USA
Giorgio Metta, Genoa, Italy

78 Perceptual Robotics
Heinrich Bülthoff, Tübingen, Germany
Christian Wallraven, Seoul, Korea
Martin A. Giese, Tübingen, Germany

79 Robotics for Education
David P. Miller, Norman, USA
Illah Nourbakhsh, Pittsburgh, USA

80 Roboethics: Social and Ethical Implications
Gianmarco Veruggio, Genoa, Italy
Fiorella Operto, Genoa, Italy
George Bekey, Arroyo Grande, USA

Robots and Humans covers some of the most recent advances concerned with human–robot interaction, ranging from designing biologically inspired robots, to programming and safety issues for human–robot interaction, and ethical issue brought forth by robotics. Our field's future vision for technology is the leap from personal computers to personal robots, in a world where robots exist pervasively and work side by side with humans. Over the past 50 years we have made great strides in robotics, as the previous parts of this Handbook show. However, there are still new capabilities that need to be developed and existing capabilities that need to be improved to create a world in which robots and humans work together. Robot bodies should be easily integrated into our living environments. Robots should be safe to be around. Robots should take commands from human users easily. Robots should be functionally capable. Robots should engage humans to help mitigate error states and task uncertainties. Meeting these challenges will bring robots closer to our vision of pervasive robotics.

The topics in Part G are essential for creating robots that operate in human-centered environments. The chapters cover organically human-centered and life-like robots and include hardware design, control (of locomotion, manipulation, and interaction), perception, user interfaces, and social and ethical implications for robotics. As such, Part G builds on all of the Handbook's previous parts. The connection with the chapters in Robot Foundations (Part A) and Robot Structures (Part B), particularly Mechanisms and Actuation (Chap. 4), Sensing and Estimation (Chap. 5), Motion Planning (Chap. 7), Robotics Systems Architectures and Programming (Chap. 12), AI Reasoning Methods for Robotics (Chap. 14), Robot Hands (Chap. 19), Limbed Systems (Chap. 17), Modular Robots (Chap. 22) and Wheeled Robots (Chap. 24) are essential. There is also an important dependency on Sensing and Perception (Part C), Manipulation and Interfaces (Part D), and Moving in the Environment (Part E) as these chapters cover some of the basic algorithms and technologies required by robots operating in human-centered environments.

Today's approach to computation has progressed naturally from desktop computing, to mobile computing, to pervasive computing, ultimately leading to computation for interaction with the physical world. In other words, Part G of the Handbook on Robotics presents a snapshot of the field's advances for creating machines in our own image that are smart and obedient. With this overview of Part G, we now provide a brief synopsis of each chapter.

Chapter 67, Humanoids, provides a concise overview of the state of the art in creating human-like machines, starting with a historical and philosophical discussion on the human example and understanding intelligence. These machines are capable of bipedal locomotion and manipulation. In addition, humanoid robots can take advantage of their entire body (e.g., lifting objects while moving the body to compensate for the load). Control issues specific to humanoids include static and dynamic stability during a mobile manipulation action. Humanoid robots interact with humans via perception systems and should be capable of interpreting behavior and mood as well as via speech systems for interpreting human language.

Chapter 68, Human Motion Reconstruction, describes the most recent algorithms that rely on data from motion capture systems to synthesize human-like motion models for machines. The chapter introduces two models for information capture: optical motion capture and mechanical motion capture. The skeleton model is computed on the captured data using kinematics and dynamics models. Musculoskeletal models can also be computed for motion analysis involving muscles. The chapter discussed model identification techniques, segmentation of human motion, classification of human motion, as well as temporal sequencing and linguistic classification of human motion. The chapter concludes with a survey of techniques for motion reconstruction for robots.

Chapter 69, Physical Human-Robot Interaction, discusses what happens when humans and robots share the same workspace and come into contact with one another. Safety can be jeopardized by mechanical failures, operator errors, or software problems. Designing and controlling robots with safety guarantees is a very important precondition for physical human–robot interaction (pHRI). The chapter discusses safety standards for pHRI and surveys several approaches to safety.

Chapter 70, Human Robot Augmentation, introduces wearable robotics systems that enhance the capabilities of the humans wearing them, while allowing the humans to control them to enhance their performance in a task-specific way. There are three types of devices that deliver human augmentation:

i) Prostheses, for the functional replacement of human limbs
ii) Powered orthoses, which function is to actively operate in parallel with unhealthy human joints
iii) Robotic exoskeletons that operate in parallel with healthy human limbs.

The chapter describes prostheses, powered orthoses and exoskeletons for upper limb, lower limbs and whole body structures. State-of-the-art devices together with their functionalities and main robotics components are presented for each class of wearable system. Critical design issues and open research aspects are reported.

Chapter 71, Cognitive Human Robot Interaction, considers human(s), robot(s), and their joint actions as a cognitive system. This chapter surveys the state of the art in modeling, algorithms, and design guidelines to enable the design of systems where humans and robots can interact. The chapter describes the representations and actions that allow robots to participate in joint activities with people and the models of joint activity for human-robot interaction. The chapter also describes recent research on developing a deeper understanding of human expectations and cognitive responses to robot actions. This chapter draws on research questions and advances from a wide range of fields including computer science, cognitive science, linguistics, and robotics.

Chapter 72, Social Robotics, discusses the creation of robots designed to interact with people in a social and emotional way for applications in education, entertainment, health care, etc. These robots' embodiment is life-like and they are capable of multi-modal communication. The chapter uses a case study of the Kismet robot to illustrate the idea of a machine capable of projecting emotion. The chapter surveys some basic approaches to giving robots sociocognitive skills such as shared attention, emotional empathy, and mental perspective tasking.

Chapter 73, Socially Assistive Robotics, covers the design of human-robot systems that use verbal and non-verbal expression and communication to interact and assist users through social rather than physical interaction. Socially assistive robotics (SAR) focuses on the challenges of providing motivation, coaching, training, and rehabilitation through non-physical interaction; such systems have been validated in hands-off stroke rehabilitation, social skill training of children with autism, and eldercare, among others. This chapter reviews the critical societal issues that have motivated research into socially assistive robotics, describes the reason why physical robots rather than virtual agents are essential to this, highlights the major research issues within this area, describes the primary application domains and populations where SAR research has shown an impact, and closes with some of the ethical and safety issues.

Chapter 74, Learning from Humans, covers important topics about how we approach teaching robots what to do by providing either good or bad examples. This is an intuitive approach to robot control and learning of tasks with the potential of opening robot applications to non-expert users. The chapter starts by discussing what it means for a robot to learn a new skill and surveys several learning approaches for this. Biologically oriented learning approaches such as conceptual models of imitation learning and neural models of imitation learning are also introduced. The chapter concludes with several open issues in robot programming by demonstration.

Chapter 75, Biologically Inspired Robots, starts with a discussion of the difference between bio-inspired and biomimetic robots. Bio-inspired robots seek to reproduce a natural phenomenon but not necessarily the underlying means. Biomimetic robots reproduce both the natural phenomenon and the means. The chapter starts with a survey of bio-inspired morphologies. A survey of bio-inspired sensors from vision to audio, touch, smell, and taste is then presented. Bio-inspired actuators include locomotion topics such as crawling, legged locomotion, climbing swimming, flying, and manipulation. The chapter also examines bio-inspiration in the context of control and planning. This includes behavior-based architectures, learning robots, evolving robots, and developing robots. Finally, the chapter examines how to make machines self-sustaining from an energy point of view.

Chapter 76, Evolutionary Robotics, discusses a method for creating robots that is inspired by the Darwinian principle of selective reproduction of the fittest captured by evolutionary algorithms. The chapter introduces the evolutionary computation method and presents several case studies. Because the evolutionary method requires intensive computation, it is often run in simulation and the final result is transferred to the robot platform. However, in the recent past there were several successes with running evolutionary algorithms in real time on physical platforms to achieve complex behaviors such as learning to walk.

Chapter 77, Neurobotics: From Vision to Action, looks at how neuroethology, the study of brain mechanisms for animal behavior, inspires the creation of a central nervous system for a machine. The chapter presents several case studies that sample this bio-inspired design of robot controllers structurally and functionally: the optic flow in bees and robots, visually guided behaviors in frogs and robots, and navigation in rats and robots. The chapter then examines the role of the cerebellum in motion control and the role of mirror systems. The chapter concludes by observing the brain has evolved to serve as action-oriented percep-

tion, which is a useful guiding principle for building robot controllers.

Chapter 78, Perceptual Robotics, discusses principles derived from high-level cognitive processing in vision in the human brain that have led to results in robotics and computer vision. The chapter is focused on the technical realization of perceptual functions in robots by examining relationships between biological perception and robotics systems. Object recognition is used as an exemplary task and issues such as representation, structural description models, neural representations, as well as recognition and learning algorithms are discussed. Next, example-based movement representations are examined to establish the connection between the functionality of the visual cortex and a class of computer vision algorithms.

Chapter 79, Robotics for Education, provides an overview of the key ingredients that make successful education robots possible. It begins with a survey of robot tournaments, which are the most prevalent mechanism for educating young students in robotics. Robot tournaments have already impacted tens of thousands of students across diverse geographic and age group boundaries. Next, the chapter technology standards required for effective use of robots in education. Educational robot devices consist of both hardware (pre-assembled or as kits or components) and software (both as source code and programming environments). The chapter surveys the physical robot platforms that have achieved notable success, and the software that enabled the interaction with young students (that is, the low-level controllers that interface those platforms to high-level computation and the top-level programming environments.) Finally, the chapter describes how conventional analytical tools may be used to evaluate unconventional educational programs that use robots.

Chapter 80, Roboethics: Social and Ethical Implications, discusses the social and ethical implications in a society where robots coexist with humans. Starting with a philosophical introspection into the dangers of unlimited use of technology, the chapter argues for the need for a new ethics of robots. Many other factors, including the cultural differences in societies ready to accept robots, define this need. The chapter surveys the code of ethics, privacy, accuracy, intellectual property, and access, which could be adopted by robotics. A taxonomy of robot types, along with socio-ethical issues for each robot type is then presented.

Multimedia Contents

67. Humanoids

Paul Fitzpatrick, Kensuke Harada, Charles C. Kemp, Yoshio Matsumoto, Kazuhito Yokoi, Eiichi Yoshida

Humanoid robots selectively immitate aspects of human form and behavior. Humanoids come in a variety of shapes and sizes, from complete human-size legged robots to isolated robotic heads with human-like sensing and expression. This chapter highlights significant humanoid platforms and achievements, and discusses some of the underlying goals behind this area of robotics. Humanoids tend to require the integration of many of the methods covered in detail within other chapters of this handbook, so this chapter focuses on distinctive aspects of humanoid robotics with liberal cross-referencing.

This chapter examines what motivates researchers to pursue humanoid robotics, and provides a taste of the evolution of this field over time. It summarizes work on legged humanoid locomotion, whole-body activities, and approaches to human–robot communication. It concludes with a brief discussion of factors that may influence the future of humanoid robots.

67.1 **Why Humanoids?** 1789
67.1.1 The Human Example 1790
67.1.2 The Pleasing Mirror 1790
67.1.3 Understanding Intelligence 1790
67.1.4 Human Environments 1790
67.1.5 Human Interaction 1792
67.1.6 Entertainment, Culture, and Surrogates 1792
67.2 **History** 1792
67.3 **What to Immitate?** 1794
67.3.1 Body Parts 1794
67.3.2 Mechanics 1794
67.3.3 Sensors 1794
67.3.4 Other Characteristics 1795
67.4 **Locomotion** 1795
67.4.1 Bipedal Locomotion 1795
67.4.2 Other Various Locomotion Styles 1797
67.4.3 Localization and Navigation Among Obstacles 1798
67.4.4 Generating Motions when in Contact with an Object 1799
67.5 **Whole-Body Activities** 1801
67.5.1 Coarse Whole-Body Motion 1801
67.5.2 Generating Dynamically Stable Motions 1803
67.5.3 Generating Whole-Body Motions with Various Tasks 1805
67.5.4 Generating Motions Including Multiple Contacts 1807
67.6 **Morphological Communication** 1809
67.6.1 Expressive Morphology and Behavior 1809
67.6.2 Interpreting Human Expression 1811
67.6.3 Physical Interaction and Developmental Robotics 1812
67.7 **Conclusions and Further Reading** 1813
Video-References 1813
References 1813

67.1 Why Humanoids?

Throughout history, the human body and mind have inspired artists, engineers, and scientists. The field of humanoid robotics focuses on the creation of robots that are directly inspired by human capabilities (Chap. 75). These robots usually share similar kinematics to humans, as well as similar sensing and behavior. The motivations that have driven the development of humanoid robots vary widely. For example, people have developed humanoid robots to serve as general-purpose mechanical workers, entertainers, and test-beds for theories

from neuroscience and experimental psychology [67.1–3].

Notably, while this chapter focuses on robots that have been explicitly designated as humanoid robots by their creators, the lines between these robots and others can be blurry. Many robots share characteristics with humans, or have been inspired by humans.

67.1.1 The Human Example

On a daily basis, humans perform important tasks that are well beyond the capabilities of current robots. Moreover, humans are generalists with the ability to perform a wide variety of distinct tasks. Roboticists would like to create robots with comparable versatility and skill. Considering the physical and computational mechanisms that enable a person to perform a task is a common approach to automating it. Exactly what to borrow from the human example is controversial. The literal-minded approach of creating humanoid robots may not be the best way to achieve some human-like capabilities (Chap. 65). For example, dishwashing machines bear little similarity to the manual dishwashing they replace.

67.1.2 The Pleasing Mirror

Humans are humanity's favorite subject. A quick look at popular magazines, videos, and books should be enough to convince any alien observer that humanity is obsessed with itself. The nature of this obsession is not fully understood, but aspects of it have influenced the field of humanoid robotics.

Humans are social animals that generally like to observe and interact with one another [67.4]. Moreover, people are highly attuned to human characteristics, such as the sound of human voices and the appearance of human faces and body motion [67.5–7]. Infants show preferences for these types of stimuli at a young age, and adults appear to use specialized mental resources when interpreting these stimuli. By mimicking human characteristics, humanoid robots can engage these same preferences and mental resources.

Humanity's interest in itself has been reflected in media as diverse as cave paintings, sculpture, mechanical toys, photographs, and computer animation. Artists have consistently attempted to portray people with the latest tools at their disposal. Robotics serves as a powerful new medium that enables the creation of artifacts that operate within the real world and exhibit both human form and behavior [67.8].

Popular works of fiction have frequently included influential portrayals of humanoid robots and human-made humanoid creatures. For example, *Karel Čapek*'s science fiction play Rossum's Universal Robots (R.U.R.) from 1920 centers around the story of artificial people created in a factory [67.9]. Many other works have included explicit representations of humanoid robots, such as the robot Maria in *Fritz Lang*'s 1927 film *Metropolis* [67.10], and the thoughtful portrayal of humanoid robotics by *Isaac Asimov* in works such as *The Caves of Steel* from 1954 [67.11]. The long history of humanoid robots in science fiction has influenced generations of researchers, as well as the general public, and serves as further evidence that people are drawn to the idea of humanoid robots.

67.1.3 Understanding Intelligence

Many researchers in the humanoid robotics community see humanoid robots as a tool with which to better understand humans [67.3, 12]. Humanoid robots offer an avenue to test understanding through construction (*synthesis*), and thereby complement the *analysis* provided by researchers in disciplines such as cognitive science.

Researchers have sought to better immitate human intelligence using humanoid robotics [67.13]. Developmental psychologists, linguists, and others have found strong links between the human body and human cognition [67.14]. By being embodied in a manner similar to humans, and situated within human environments, humanoid robots may be able to exploit similar mechanisms for artificial intelligence (AI). Researchers are also attempting to find methods that will enable robots to develop autonomously in a manner akin to human infants [67.15]. Some of these researchers use humanoid robots that can physically explore the world in a manner similar to humans [67.16].

67.1.4 Human Environments

People inhabit environments that accommodate human form and human behavior [67.17, 18]. Many important everyday objects fit in a person's hand and are light enough to be transported conveniently by a person. Human tools match human dexterity. Doors tend to be a convenient size for people to walk through. Tables and desks are at a height that is well matched to the human body and senses. Humanoid robots can potentially take advantage of these same accommodations, thereby simplifying tasks and avoiding the need to alter the environment for the robot [67.19]. For example, humanoid robots and people could potentially collaborate with one another in the same space using the same tools [67.20]. Humanoid robots can also interface with machinery that does not include drive-by-wire controls, as shown by the teleoperated robot in the cockpit of a backhoe in Fig. 67.1 [67.21].

Robots with legs and human-like behavior could potentially traverse the same environments that humans traverse, such as rugged outdoor environments and the industrial plant shown in Fig. 67.2, which has stairs and handrails designed for human use [67.22]. In addition to mobility advantages, legs have the potential to help in other ways. For example, legs could enable a humanoid robot to change its posture in order to lean into something, pull with the weight of its body, or crawl under an obstacle [67.23, 24].

The Fukushima Daiichi nuclear disaster, which occurred in March 2011 in Japan, is a compelling example scenario for robots. The disaster resulted in human environments that were unsafe for humans. Robots capable of performing diverse tasks in these environments via remote control would have been valuable. Future humanoid robots may be able to access similar environments using narrow passageways, ladders, and other environmental features designed for people (Fig. 67.3). Likewise, they may be able to remotely perform tasks involving control panels, valves, and tools designed for people. This type of scenario

Fig. 67.1 The humanoid robot HRP-1S (HRP: humanoid robot project) driving a backhoe (courtesy Kawasaki Heavy Industries, Tokyu Construction and National Institute of Advanced Industrial Science and Technology (Japan) (AIST)). The robot can be teleoperated by a human operator to control the backhoe remotely. The same robot could potentially interface with many different unmodified machines

Fig. 67.3 An image of supposed disaster-response scenario DARPA Robotics Challenge

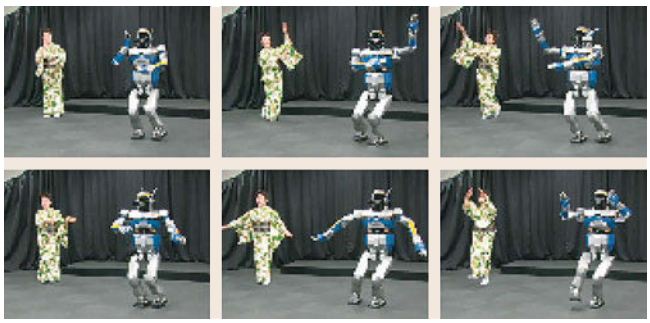


Fig. 67.4 The humanoid robot HRP-2 dancing with a human (after [67.25]). The human is a master of a traditional Japanese dance whose dancing was recorded by a motion-capture system, and transformed for use by the robot

Fig. 67.2 HRP-1 operating in a mockup of an industrial plant (courtesy Mitsubishi Heavy Industries)

Fig. 67.5 Actroid (courtesy Kokoro), an android designed for entertainment, telepresence, and media roles

Fig. 67.6 Atlas robot provided as a platform for the DARPA Robotics Challenge (courtesy Boston Dynamics)

has inspired the DARPA (Defense Advanced Research Projects Agency) Robotics Challenge in which robots will compete by performing related tasks. Notably, DARPA plans for some teams to compete using Atlas humanoid robots from Boston Dynamics Fig. 67.6.

67.1.5 Human Interaction

People are accustomed to working with other people. Many types of communication rely on human form and behavior. Some types of natural gestures and expression involve subtle movements in the hands and face (Chap. 72). People can interpret eye gaze and facial expressions without training. Humanoid robots can potentially simplify and enhance human–robot interaction by taking advantage of the communication channels that already exist between people.

Similarly, people already have the ability to perform many desirable tasks. This task knowledge may be more readily transferred to humanoid robots than to a robot with a drastically different body. This is especially true of cultural actions centered around the human form (Fig. 67.4).

Fig. 67.7 Petman is a humanoid robot that Boston Dynamics developed to test chemical protection clothing for the US military (after [67.26])

67.1.6 Entertainment, Culture, and Surrogates

Humanoid robots are inherently appropriate for some applications. For example, robots that resemble humans could play roles in entertainment, such as theater, theme parks, and companionship for adults (Fig. 67.5). Realism in form and function could make humanoid robots preferable to wax figures and animatronics.

A humanoid robot could serve as an avatar for telepresence, model clothing, test ergonomics, or serve other surrogate roles that fundamentally depend on the robot's similarity to a human. For example, Boston Dynamics developed the humanoid robot Petman to test clothing that is intended to protect military personnel from chemical agents (Fig. 67.7). Robotic prostheses and cosmeses also have a close relationship to humanoid robotics, since they seek to directly replace parts of the human body in function and form (Chap. 64).

67.2 History

There is a long history of mechanical systems with human form that perform human-like movements. For example, Al-Jazari designed a humanoid automaton in the 13th century [67.27], Leonardo da Vinci designed a humanoid automaton in the late 15th century [67.28], and in Japan there is a tradition of creating mechanical dolls called Karakuri ningyo that dates back to at least the 18th century [67.29]. In the 20th century, animatronics became an attraction at theme parks. For example, in 1967 Disneyland opened its Pirate's of the Caribbean ride [67.30], which featured animatronic pirates that play back human-like movements synchronized with audio. Although programmable, these humanoid animatronic systems moved in a fixed open-loop fashion without sensing their environments.

In the second half of the 20th century, advances in digital computing enabled researchers to incorporate

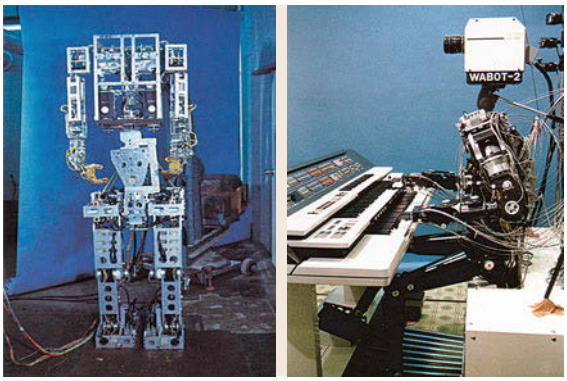


Fig. 67.8 **(a)** WABOT-1 (1973) and **(b)** WABOT-2 (1984; courtesy Humanoid Robotics Institute, Waseda University)

significant computation into their robots for sensing, intelligence, control, and actuation. Many roboticists developed isolated systems for sensing, locomotion, and manipulation that were inspired by human capabilities. However, the first humanoid robot to integrate all of these functions and capture widespread attention was Waseda robot (WABOT-1), developed by Ichiro Kato et al. at Waseda University in Japan in 1973 (Fig. 67.8).

The WABOT robots integrated functions that have been under constant elaboration since: visual object recognition, speech generation, speech recognition, bimanual object manipulation, and bipedal walking. For instance, WABOT-2's ability to play a piano, publicized at the Tsukuba Science Expo in 1985, stimulated significant public interest.

In 1986, Honda began a confidential project to create a humanoid biped. Honda grew interested in humanoids, perhaps seeing in them devices of complexity comparable to cars with the potential to become high-volume consumer products one day. In 1996, Honda unveiled the Honda Humanoid P2, the result of this confidential project. P2 was the first full-scale humanoid capable of stable bipedal walking with onboard power and processing. Successive designs reduced its weight and improved its performance (Fig. 67.9). Compared to humanoids built by academic laboratories and small manufacturers, the Honda humanoids were a leap forward in sturdiness, using specially cast lightweight high-rigidity mechanical links, and harmonic drives with high torque capacity.

In parallel with these developments, the decade-long *Cog project* began in 1993 at the MIT Artificial Intelligence laboratory in the USA with the intention of

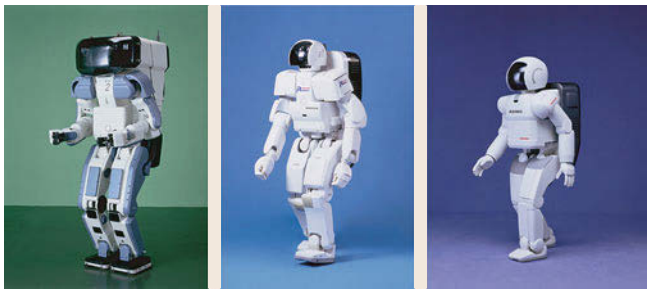


Fig. 67.9 **(a)** Honda P2 (180 cm tall, 210 kg), **(b)** P3 (160 cm, 130 kg), and **(c)** advanced step in innovative mobility (glossnoidx-ASIMOadvanced step in innovative mobility) (120 cm, 43 kg) (after [67.31]; courtesy Honda)

Fig. 67.10 The humanoid robot Cog used neural oscillators in conjunction with compliant torque-controlled arms to perform a variety of everyday tasks with human tools, such as crank turning, hammering, sawing, and playing a snare drum (after [67.32]; courtesy Sam Ogden)

creating a humanoid robot that would, *learn to think by building on its bodily experiences to accomplish progressively more abstract tasks* [67.13]. This project gave rise to an upper-body humanoid robot whose design was heavily inspired by the biological and cognitive sciences (Fig. 67.10). Since the inception of the Cog project, researchers across the world have initiated many humanoid robotics projects and formed communities devoted to developmental robotics, autonomous mental development (AMD [67.33]), and epigenetic robotics [67.34].

As of the early 21st century, a large number of companies and academic researchers have become involved with humanoid robots and created new humanoid robots with distinctive characteristics.

67.3 What to Immitate?

Humanoid robots come in a variety of shapes and sizes that immitate different aspects of human form and behavior (Fig. 67.11). As discussed, the motivations that have driven the development of humanoid robots vary widely. These diverse motivations have led to a variety of humanoid robots that selectively emphasize some human characteristics, while deviating from others.

67.3.1 Body Parts

One of the most noticeable axes of variation in humanoid robots is the presence or absence of body parts. Some humanoid robots have focused solely on the head and face, others have a head with two arms mounted to a stationary torso, or a torso with wheels (for example, Fig. 67.12). Some humanoid robots even combine a highly-articulate face with arms, legs, and a torso.

67.3.2 Mechanics

Humanoid robots immitate various mechanical aspects of the human body, such as its kinematics, dynamics, geometry, material properties, and actuation. As such, humanoid robotics is closely related to the field of human biomechanics.

Humanoid robots often consist of rigid links with kinematics that approximate the kinematics of the human musculoskeletal system. Even a rigid-link model of human kinematics can have a very high number of degrees of freedom (DOF). A humanoid robot typically imitates degrees of freedom that are pertinent to its intended use. For example, humanoid robots rarely attempt to immitate the human shoulder's ability to translate or the flexibility of the human spine [67.35, 36]

The human hand serves to further illustrate these issues. Modern humanoid robots frequently have two arms, each with seven degrees of freedom, but their hands vary considerably (Chap. 19). The human hand is highly complex with over 20 DOFs (i. e., approximately four DOFs per finger and a five-DOF thumb) in a very compact space with a compliant exterior, dense tactile sensing, and low distal mass. Researchers have approximated the human hand with varying levels of accuracy, including the anatomically correct testbed (ACT) hand, the 20-DOF Shadow Hand, the 12-DOF DLR-Hand-II (DLR: Deutsches Zentrum für Luft- und Raumfahrt), the 11-DOF Robonaut hand, and the 2-DOF Cog hand [67.37–41]. The ACT hand represents the high-fidelity end of the spectrum, since it approximates the bone structure, inertial properties, kinematics, and actuation of the human hand.

Actuation is another property of humanoid robots that varies considerably. Human actuation consists of a complex, highly-redundant system of variable-stiffness muscles. In contrast, many humanoid robots use a stiff, position-controlled actuator at each joint. There are early exceptions, such as the use of series elastic actuators in Cog's arms [67.32, 42], and various forms of compliant actuation have now become common in the arms of humanoid robots.

67.3.3 Sensors

Humanoid robots have made use of a variety of sensors including cameras, three-dimensional (3-D) cameras, laser rangefinders, microphone arrays, lavalier microphones, and pressure sensors. Some researchers choose to immitate human sensing by selecting sensors with

Fig. 67.11 Kismet is an example of a humanoid head for social interaction

Fig. 67.12 The NASA (National Aeronautics and Space Administration) Robonaut consists of an upper body placed on a wheeled mobile base

clear human analogs and mounting these sensors on the humanoid robot in a manner that mimics the placement of human sensory organs. As discussed in Sect. 67.6, this is perhaps most evident in the use of cameras. Two to four cameras have often been mounted within the heads of humanoid robots with configurations similar to human eyes.

The justifications for this bias towards human-like sensing include the impact of sensing on natural human–robot interaction, the proven ability of the human senses to support human behavior, and aesthetics. For example, with respect to human–robot interaction, nonexperts can sometimes interpret the functioning and implications of a human-like sensor, such as a camera, more easily. Similarly, if a robot senses infrared or ultraviolet radiation, the robot can see a different world than the human. With respect to behavior, placement of sensors on the head of the robot allows the robot to sense the world from a vantage point that is similar to that of a human, which can be valuable for finding objects that are sitting on a desk or table.

Prominent humanoid robots have added additional sensors without human analogs. For example, Kismet used a camera mounted in its forehead to augment the two cameras in its servoed eyes, which simplified common tasks such as tracking faces. Similarly, versions of ASIMO have used a camera mounted on its lower torso that looks down at the floor in order to simplify obstacle detection and navigation during locomotion.

67.3.4 Other Characteristics

Other common forms of variation include the size of the robot, the extent to which the robot attempts to appear like a human, and the activities the robot performs. The remainder of this chapter provides examples from three active areas of humanoid robotics research: locomotion, whole-body activities, and morphological communication.

67.4 Locomotion

Bipedal walking is a key research topic in humanoid robotics (see also Chap. 48, *Legged Robots*, for a review of this topic in the context of locomotion in general). Legged locomotion is a challenging area of robotics research, and bipedal humanoid locomotion is especially challenging. Some small humanoid robots are able to achieve statically stable gaits by having large feet and a low center of mass, but large humanoids with a human-like weight distribution and body dimensions typically need to balance dynamically when walking bipedally.

67.4.1 Bipedal Locomotion

Currently the dominant methods for bipedal legged locomotion with humanoids make use of the zero-moment point (ZMP) criterion to ensure that the robot does not fall over [67.43]. As discussed in detail in Chap. 48, control of the robot's body such that the ZMP sits within the support polygon of the robot's foot ensures that the foot remains planted on the ground, assuming that friction is high enough to avoid slipping. The ZMP can be used to plan walking patterns that make the robot dynamically stable while walking. Conventionally, biped locomotion had been offline generated by solving an ordinally differential equation with respect to the motion of the COG (center of gravity) given a desired trajectory of the ZMP. Recently, several extensions have been done for the ZMP based biped gait generation as shown in the following:

Realtime Walking Pattern Generation

By solving the ordinally differential equation in realtime, biped gait is generated in realtim [67.44, 45]. Since the realtime walking pattern generator enables us to change the landing positions of foot in realtime, it is used in various situations; in [67.46], the landing position of the foot changes in accordance with the hand reaction force as will be described more concretely in the subsection of *manipulation*. As shown in Fig. 67.13 [67.47], the walking pattern is generated in realtime in accordance with the amount of external disturbance applied to the robot. In this case, after the torso of a robot is pushed by a human, biped gait for a few steps is generated in realtime to recover the balance. In [67.48], the humanoid robot ASIMO walks in the environment with moving obstacles. By using the estimation of the object motion, the walking pattern of the robot is generated in realtime. Fig. 67.14 [67.49] shows the biped locomotion on uneven terrain. In this experiment, the shape of the environment is measured by a laser range sensor. According to the shape information of the environment, the landing position on uneven terrain is calculated in realtime.

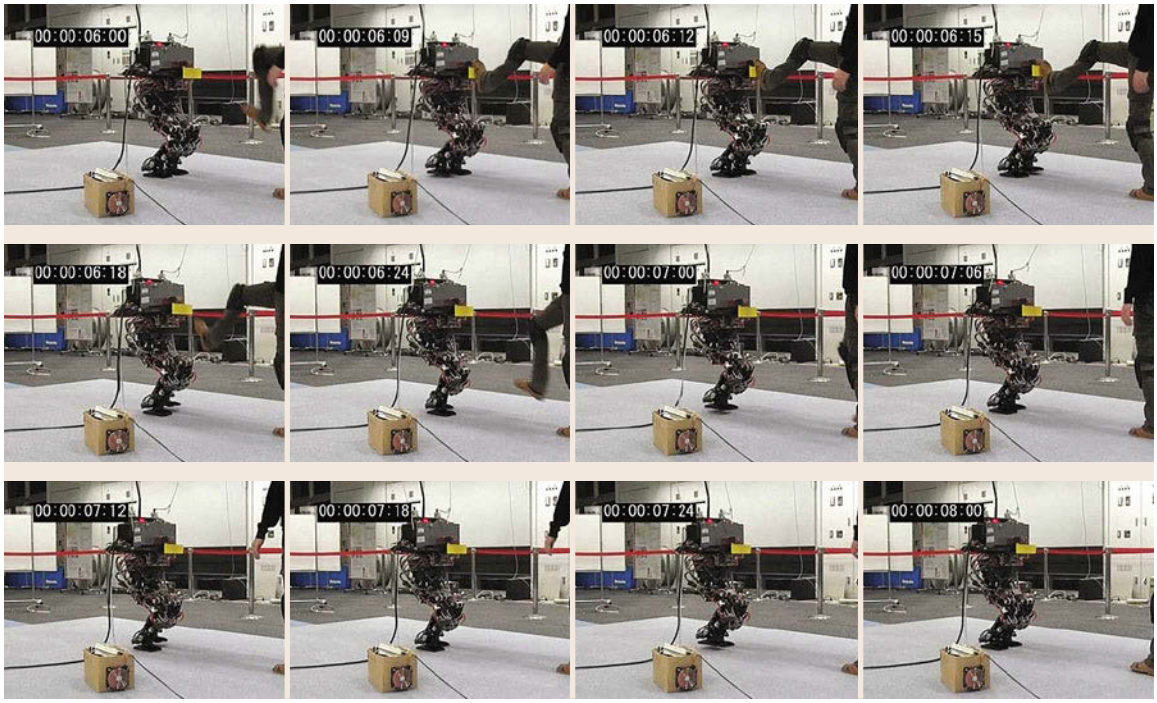


Fig. 67.13 Experiment of push recovery

Running

By additionally considering the flight phase to the ZMP based walking pattern generator, running motion of biped robot is generate [67.50–52]. Fig. 67.15 [67.52] shows an example of running motion by a biped humanoid robot. Running motion of a human-sized humanoid robot with 7 km/h is realized.

Fig. 67.14 Experiment on biped gait on uneven terrain

Extension of ZMP Based Method

Althogh the ZMP is a two dimentional information defined for the interaction between a robot and the ground surface, a robot applies 6 dimensional force/moment onto the ground. Hence, just by regulating the position of the ZMP, it is impossible to control all dimension of the interaction force/moment. More concretely speaking, the robot may slip on the ground surface or may loose contact with the ground. Research on biped locomotion considering full 6 dimensional force/moment has been don [67.53, 54].

Human-Like Walking Motion

The biped locomotion generated just by using the ZMP may not be a human-like one. Challenge has been done to generate a human-like biped gait of a humanoid robo [67.55]. Fig. 67.16 [67.55] showws a biped locomotion where single toe support, knee stretching and human like swing leg trajectory are applied. It is compared with the human walking motion where the model belongs to *Walking Studio Rei*.

Force/Moment Controller

Bipedal walking needs to be robust to unexpected disturbances encountered during the execution of planned

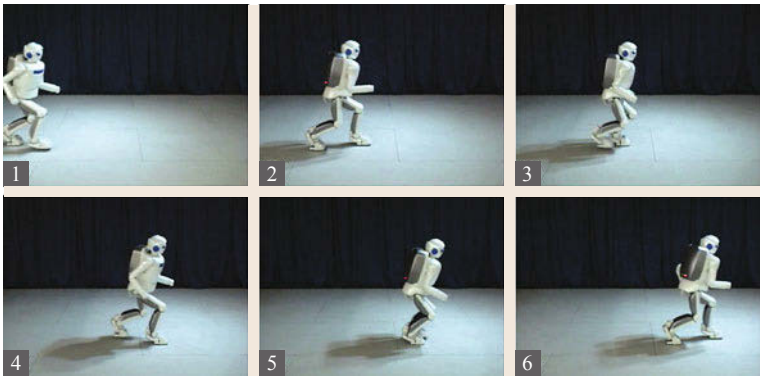


Fig. 67.15 Example of running motion

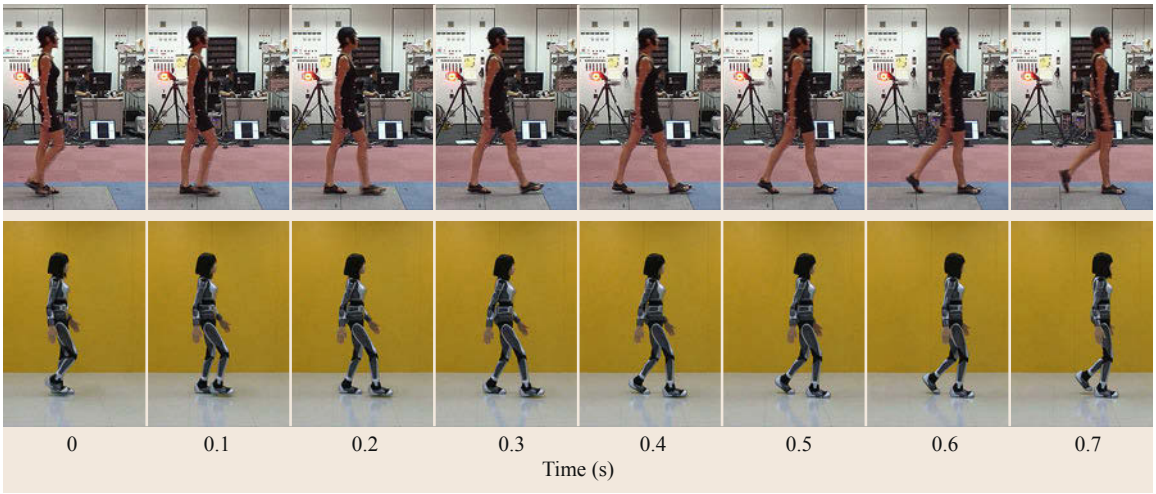


Fig. 67.16 Human like walking motion compared with human motion

walking patterns. In these situations, walking can sometimes be stabilized with feedback control and appropriate sensin [67.56]. Many humanoid robots, such as Honda's ASIMO, make use of accelerometers, gyroscopes, and six-axis force/torque sensors to provide feedback to the robot during locomotion.

Force/torque sensors have long been applied to manipulators for the implementation of force control, but force/torque sensors with sufficient robustness to handle foot impact for a full-size humanoid robot are relatively new. When the foot of the robot touches down, the foot receives an impact which can disturb its walking. This impact can be rather large, especially when the robot is walking quickly. Some feet now incorporate a spring and damper mechanism as shown in Fig. 67.17 in order to mitigate these problems.

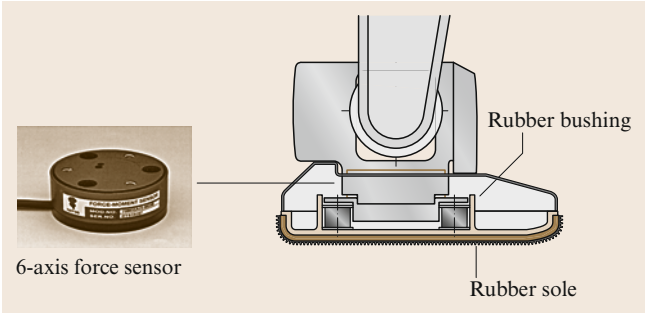


Fig. 67.17 Example of a humanoid foot structure for legged locomotion that uses compliance and force/torque sensing

Passive-Gait-Based Approach

Alternative to the ZMP-based approach, researchers have begun to use the principles of bipedal passive-dynamic walkers to develop powered bipedal walkers that walk with high efficiency in a human-like way by exploiting natural dynamics (Fig. 67.18 [67.57]).

67.4.2 Other Various Locomotion Styles

Most humanoid robots have two legs and two arms. Here, in addition to the legs, the arms can be used to

Fig. 67.18 These robots from Delft, MIT and Cornell (*left* to *right*) are designed to exploit their natural dynamics when walking (after [67.57]; courtesy Steven H. Collins)

enhance the mobility of a humanoid robot. For example, when a robot walks while grasping a handrail, the contact could potentially increase the stability of the robot. Attempt [67.58] have been done to generate the motion of a humanoid robot by using the arms in addition to the legs to enhance its mobility of it. As shown in Fig. 67.19 [67.58], a humanoid robot walking on uneven terrain sometimes uses to increase the robot's stability.

A human-scale robot should expect to fall from time to time in realistic conditions. A humanoid robot may fall down due to a large disturbance even if the motion is planned carefully and a sophisticated feedback controller is applied to the robot. In this event, the robot could be damaged significantly during a fall, and could also damage the environment or injure people who are nearby. An important area of research is how to control the robot's fall in order to gracefully recover or minimize damage. The Sony QRIO (Quest for cuRIOsity) can control its falling motions in order to reduce the impact of touch down [67.50], although it is of a relatively small size (which simplifies the problem). *Fujiwara* et al. developed a falling motion controller for a human-size humanoid robot that is falling backwards [67.59]. Figure 67.20 shows an example of a controlled falling motion. The general problem is still very much an active area of research. Similarly, there is also the issue of getting back up again [67.60] (Fig. 67.21).

67.4.3 Localization and Navigation Among Obstacles

In order for a humanoid robot to walk in unmodeled environments, localization and obstacle detection are essential. Wheeled robots encounter similar issues while navigating, but full bipedal humanoids have more-specialized requirements. For example, bipedal humanoids have the ability to control contact with the world through their highly articulate legs.

Artificial landmarks can simplify localization. As shown in Fig. 67.22, Honda's ASIMO uses a camera mounted on its lower torso that looks down at the floor to find artificial markers for position correction [67.61]. Accurate positioning is important for long-distance navigation and stair climbing, since slippage usually occurs while walking and accumulated positional and directional errors can lead to severe failures.

Obstacle avoidance is also an important function for locomotion. Disparity images generated by stereo vi-

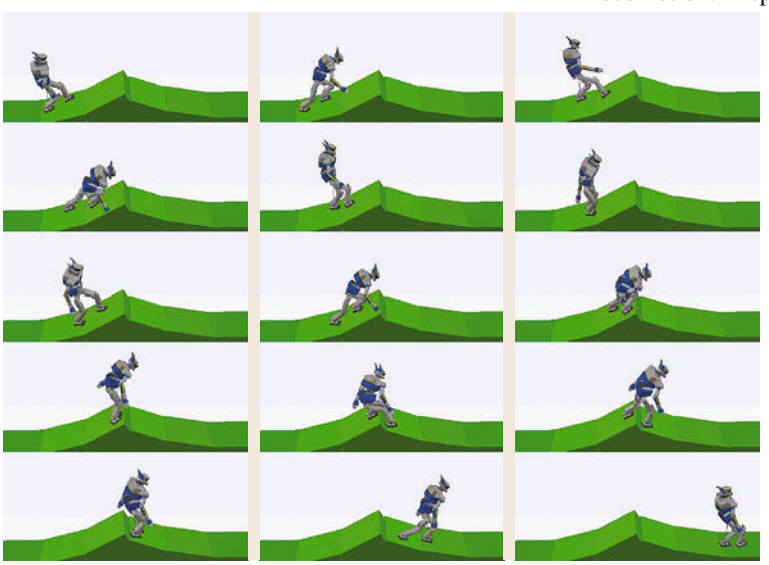


Fig. 67.19 Humanoid robot walking on uneven terrain by utilizing hand contact

Fig. 67.20 Example of controlled falling-down motion

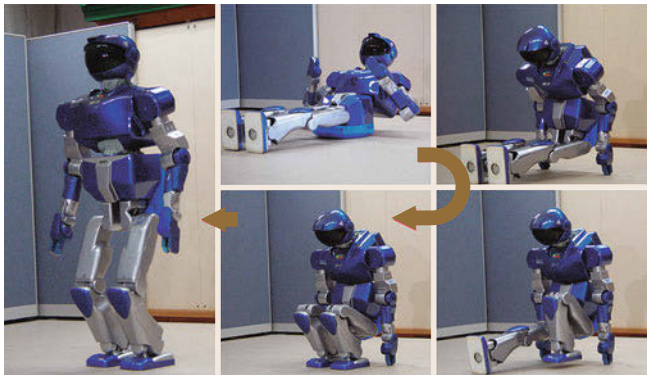


Fig. 67.21 The humanoid robot HRP-2P getting up from a lying-down position ►

sion have been utilized for this purpose. For example, the plane segment finder [67.62] developed by *Okada* et al. helps detect traversable areas. Figure 67.23 shows the result of detecting clear areas of the floor plane appropriate for gait generation.

Humanoids require a great deal of computation due to the need for sophisticated sensing and control. Customized computational hardware may help mitigate this problem. For example, Sony's humanoid robot QRIO is equipped with a field-programmable gate array (FPGA) to generate disparity maps in real time from the stereo cameras. This real-time vision system has been used to detect floor areas, stair steps, and obstacles for navigation [67.63, 64].

67.4.4 Generating Motions when in Contact with an Object

Many approaches to whole-body motion generation assume that the robot is only in contact with the ground. When a humanoid robot's hands make contact with the environment, it can no longer maintain balance using the conventional ZMP property defined by the center of pressure of the supporting feet [67.65]. This leads to significant challenges for whole-body activities, especially since the properties of the environment with which the robot is making contact may not be known in advance.

Harada et al. have introduced generalized ZMP (GZMP) as a method of handling some of these issues, such as the hand reaction forces generated from contact with the environment [67.66]. Researchers have developed methods that directly make use of the six-dimensional force/torque acting on the robot at the hands, which can be sensed with conventional force/torque sensors placed at the wrists [67.67]. Researchers have also developed specialized methods for generating stable robot motion while an object is being manipulated [67.46, 65, 68–70].

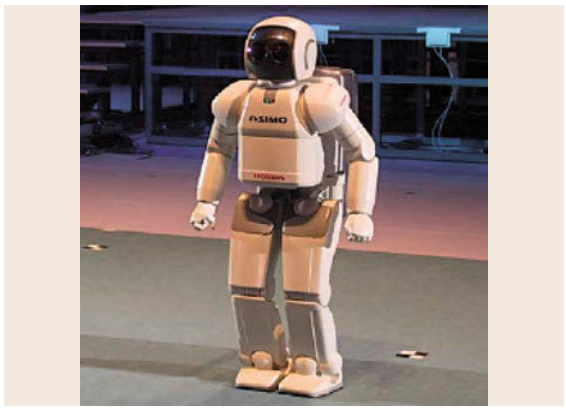


Fig. 67.22 ASIMO and artificial landmarks on the floor

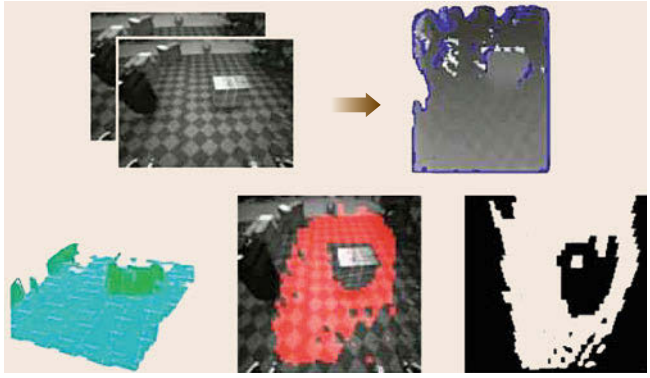


Fig. 67.23 Plane segment finder for detecting traversable floor area

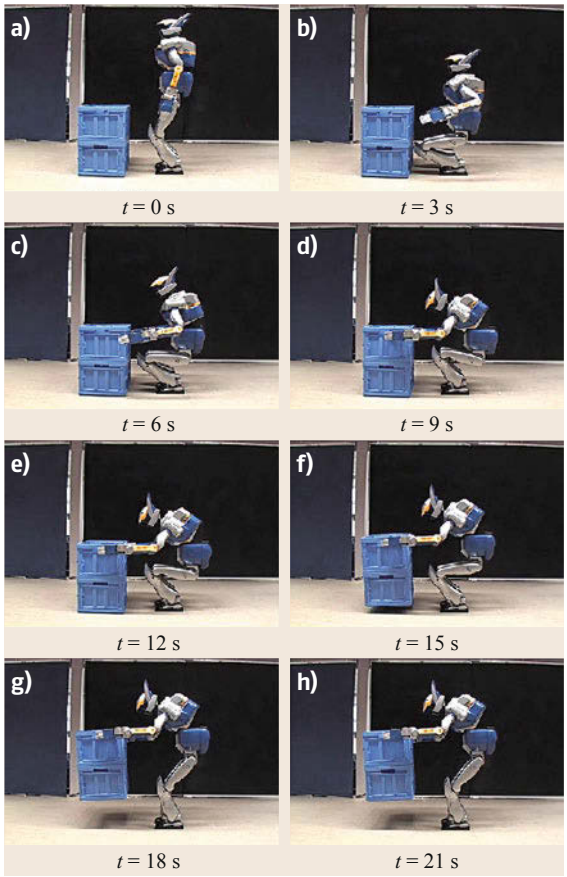


Fig. 67.24 (a–h) Lifting an object while moving the waist to compensate for the load (after [67.71])

Carrying an Object

In a manner analogous to the previously described methods, coarse motions that do not consider the hand reaction forces can be modified [67.65, 68, 71, 72]. Figure 67.24 shows an experimental result of carrying an object that weighs 8 kg [67.71]. Based on measurements of the hand reaction force, the position of the waist is modified to compensate for the load and maintain stability.

Pushing an Object

As another example of using force sensing to adapt behavior, consider the problem of pushing a large object placed on the floor. For such a task, if the gait pattern is determined before the robot actually moves, the robot may not stay balanced if the weight of the object or the friction coefficient between the object and the floor is different from the predicted values. To address this problem, the gait pattern can be adaptively changed depending on the output of a force sensor at the end of the arms in order to handle changes in the object's weight and the friction coefficient [67.46].

Figure 67.25 shows an experimental result for this approach [67.46]. In the experiment, the table weighs about 10 kg. Even though the motion of the table is disturbed externally during the experiment, the robot stays balanced by adaptively changing its gait pattern based on the measured forces.

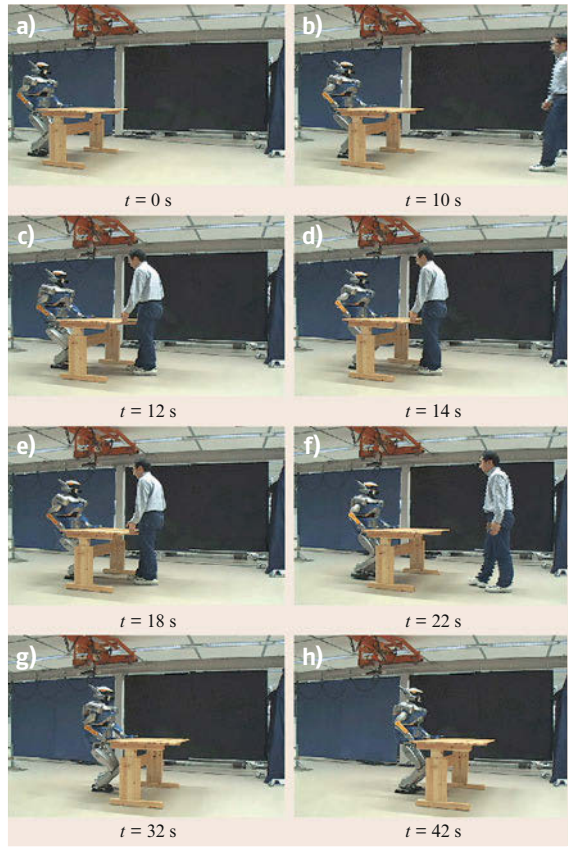


Fig. 67.25 (a–h) Example of pushing manipulation and cooperation (after [67.46])

67.5 Whole-Body Activities

The two previous sections have focused on humanoid locomotion and manipulation separately. This section outlines whole-body motions that require coordinated control of arms and legs to perform various tasks, such as carrying a bulky objects, climbing a ladder, or going through narrow spaces with contact supports on the environment. Humanoid motion is characterized by their redundancy and underactuation (Chaps. 10 and 17). Unlike fixed industrial robots, it has a floating base (usually it is set at the pelvis) that can only be controlled through the leg locomotion or multiple contact motion involving arms and legs. It is therefore essential how to define the desired task and to generate the motion that achieves it. Since humanoids have a redundant structure, a general approach is first to generate coarse motion, and then to transform it into a whole-body coordinated joint trajectory that is executed by a controller maintaining the stability through sensor feedback, as illustrated in Fig. 67.26. The first half of this section addresses the first two components of this general approach.

The latter half of this section deals with issues related to various complex whole-body motion that the basic conversion methods addressed in Sect. 67.5.2 could not always resolve. Resolution of various concurrent tasks, including those expressed as inequality and dynamic constraints is presented in Sect. 67.5.3. This framework can be applied to a reaching task while keeping the visibility of the object, as well as footstep planning. Finally, motion generation including multiple contacts is introduced as an advanced topic in Sect. 67.5.4. A wide variety of its application is expected to extend the activity fields of humanoids in cluttered environments where the humanoid should maintain its balance by supporting its body on non-coplanar contact points.

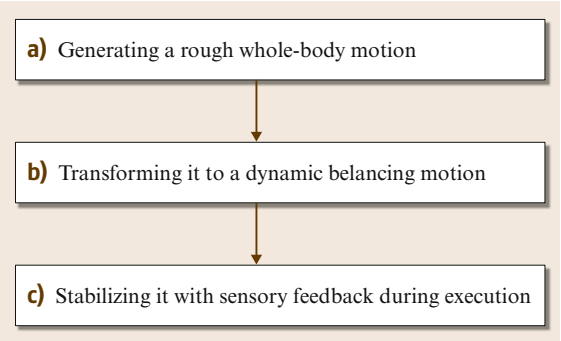


Fig. 67.26 Overview of motion-generation stages for a balancing robot

67.5.1 Coarse Whole-Body Motion

There are several ways to generate coarse humanoid motion:

1. Using motion capture system
2. Using graphical user interface (GUI)
3. Using automated motion planning
4. Using abstract task specification.

Using Motion Capture System

As humanoid robot has a human-like structure, a natural and common way of motion generation is using measured human motion. Motion retargeting from recorded human motion to digital characters is a well-studied area in computer animation. Typically a human subject performs actions while wearing easily detected markers on his or her body. The motion of these markers is recorded by cameras placed in the room, and software then infers the 3-D positions of these markers over time. A number of studies have been reported to convert the capture motions to humanoid whole-body motions through learning [67.73] and optimization [67.74, 75]. Figure 67.27 shows an example: the captured motions of a woman performing a Japanese traditional dance [67.76] that the performance in Fig. 67.4 is based on. Kinematic similarity allows using the captured motion as a reference for a humanoid's whole-body motion, by computing the corresponding joint angles from forming virtual links with several markers. However, due to dynamic differences, such as mass distributions and torque generation, captured motions are generally not stable nor feasible when applied to a humanoid. It is therefore necessary to adapt them to humanoid body as explained in Sect. 67.5.2.

Using GUI

Tools such as those used in character animation for computer graphics can also be used to design movements for humanoid robots. If the designer were forced to control each of the many degrees of freedom independently or to takes care of the balance, the process would be tedious and inefficient. One solution that enables non robotics expert to design robot motion is key-pose based approach on GUI, which allows the designer to define the *key-poses* of the desired motion with the help of inverse kinematics of the end effectors. The interface take care of the interpolation and dynamic balance compensation so that the input motion is feasible for the humanoid as explained later in Sect. 67.5.2. Figure 67.28 illustrates the overview of a GUI interface developed as *Choreonoid* for this purpose [67.77, 78].

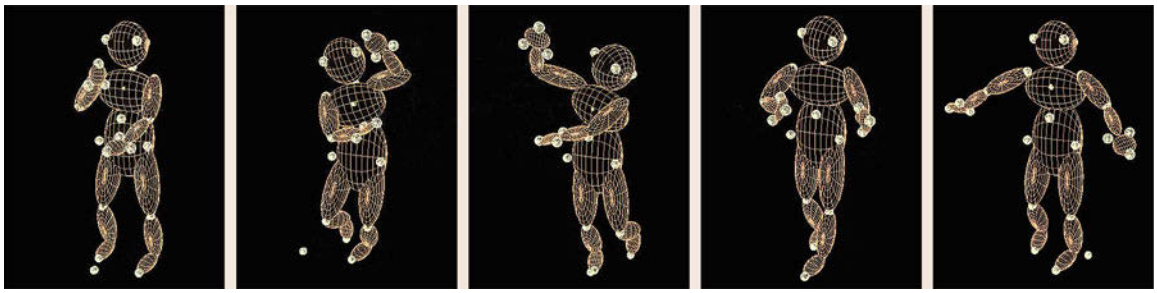


Fig. 67.27 A sequence of captured motion of dancing (after [67.76])

Using Automated Motion Planning

The previous two methods are based on mainly joint angles from a motion capture system or a GUI. If the purpose of the robot motion is going from one configuration to another without collisions and how the humanoid moves does not really matter, automated motion planning can provide efficient solutions (Chaps. 36 and 47).

Fast path-planning techniques such as rapidly-exploring random trees (RRT) can compute basic collision-free postures with static balance [67.79, 80] or walking paths [67.81] automatically with a simplified model. Figure 67.29 illustrates a humanoid carrying an object whose lower body is modeled by a bounding box. Given geometric models of the humanoid and the environment, initial and goal configurations, the planning system automatically searches for a path free of collisions at both upper and lower bodies. This coarse path can be converted into dynamically stable whole-body motion a walking pattern generator including upper-body motion compensation as described in Sect. 67.5.2. For further reading on humanoid motion planning, the readers are referred to a book dedicated to this subject [67.82].

Using Abstract Task Specifications

Tasks for a humanoid to execute are not always specified in the joint space, but often in the workspace. For instance, if the humanoid wants to grasp an object on the table or floor, this task is expressed as the hand position and orientation in Cartesian space [67.83–87]. Another example is teleoperation: it is easier for the operator to guide an operational point, such as the end-effector or the head of the humanoid, rather than to give a whole-body joint configuration [67.88]. These tasks are represented in an abstract way by a smaller number of DOF than the redundant structure of humanoid. Figure 67.30 shows a motion of bimanual manipulation based on abstract representation of motion as a sequence of attractor points acting in the task space [67.89].

On the other hand, other constraints such as balancing or joint limits should also be taken into account to generate a whole-body motion to achieve the task.

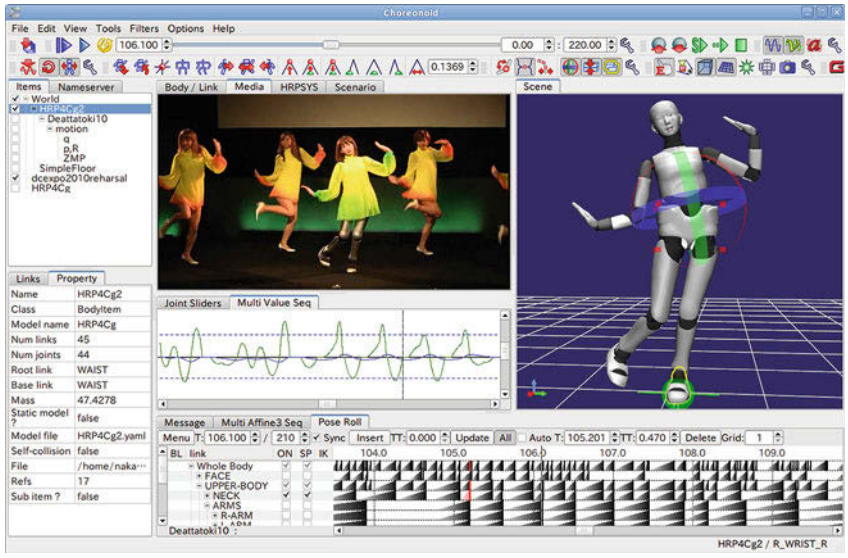


Fig. 67.28 A motion choreography tool working within the Choreonoid framework. In this tool, whole body motions of biped humanoid robots can be created with key-frame editing similar to computer graphics (CG) character animations (after [67.78])

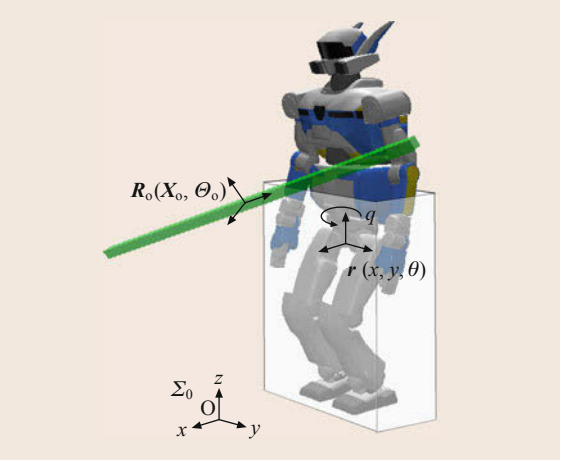


Fig. 67.29 Humanoid modeled by rectangle box with a bar. In the first stage the geometric and kinematic path planner generates collision-free path for the 9-DOF system including robot waist on the plane (3-DOF) and object (6-DOF)

A mechanism is therefore necessary that derives a humanoid positioning as well as a whole-body target posture from the abstract goal specification. Whereas the previously introduced GUI including automatic balancing is useful for cases with large free space, cluttered environments require integration of search techniques to compute a valid whole-body posture from the task [67.90, 91]. Motion generation for multiple tasks and constraints is later discussed in Sect. 67.5.3.

67.5.2 Generating Dynamically Stable Motions

The methods presented in Sect. 67.5.1 can be useful when generating coarse motions for a humanoid robot, such as dance performance, manipulating or walking. However, for some of these methods the motions generated will not take into account the dynamic stability of the robot, and may result in the robot falling over. This subsection presents some approaches for converting coarse motions to dynamically-stable motions, dynamic balancing algorithms and task-balance functional decomposition. In the former approach, all the joints including upper body are involved for whole-body balancing based on the reference motion, like kicking motion that needs upper body motion compensation. The latter uses mainly lower body for balancing or walking while upper body takes care of the desired tasks such as manipulation.

Dynamic Balancing

A framework called autobalancer is one of the pioneering studies for the dynamic whole-body balancing for humanoids. It all joint angles at every sample in time by solving a quadratic programming (QP) optimization problem in order to convert a given motion to a balanced one [67.92]. This method can be effective for a motion in which static balancing is dominant, such as when the humanoid is standing. The autobalancer calculates a whole-body motion first by fixing the center of gravity (COG) on the vertical axis which passes through a point in the support polygon of the humanoid. Then it keeps inertia moments around the COG at acceptable values in order to satisfy the balancing conditions. This technique was combined with a fast sampling-based motion planner to derive a dynamically motion by exploring configurations with balance constraints [67.79].

Resolved momentum control (RMC) [67.93] is a framework for whole-body control based on the linear and angular momentum of the entire robot. The robot is regarded as a single rigid body whose linear and angular momentum is to be controlled. At each point in time, this framework uses least squares to find joint velocities that will achieve the desired linear and angular momentum of the robot. Elements of the momentum can also be left unspecified as free variables, which is often done in practice with elements of the angular momentum. In addition to elements of the momentum, resolved momentum control requires that desired velocities for the feet be specified. This method has been applied to teleoperation [67.88] or stable reaching or kicking motions [67.93]. Other methods like dynamics

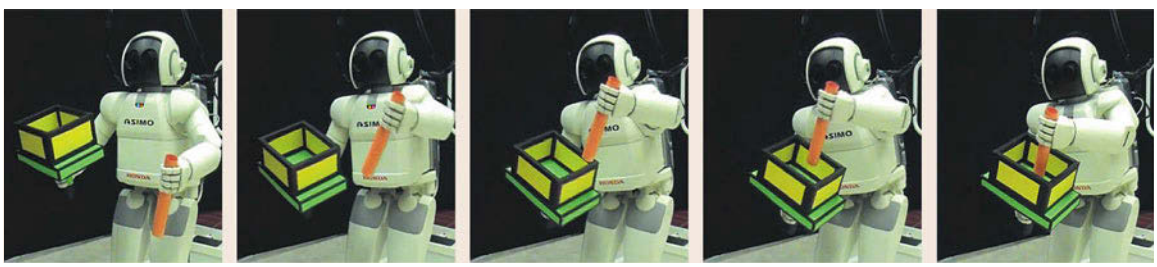


Fig. 67.30 Bimanual manipulation by humanoid robot ASIMO based on a motion representation using attractor dynamics in task space (after [67.89])

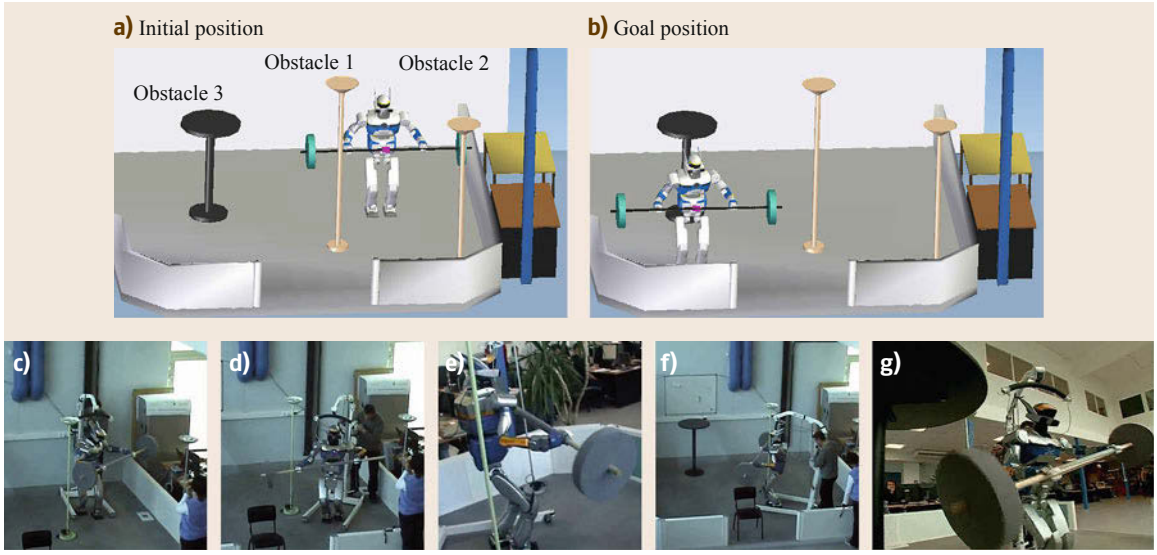


Fig.67.31a–g A 3-D collision-free motion for bar-carrying task by humanoid robot HRP-2 from starting **(a)** an intial position to final configuration (goal position) **(b)** using whole-body motion (after [67.81]). The robot rotates the bar horizontally to make the bar go through a gap between poles whose distance is shorter than the of the bar **(c–e)**. By making use of the concave part of the carried object **(f)** for 3-D collision avoidance, it arrives at the goal configuration with another avoidance motion **(g)** (VIDEO 594, VIDEO 598)

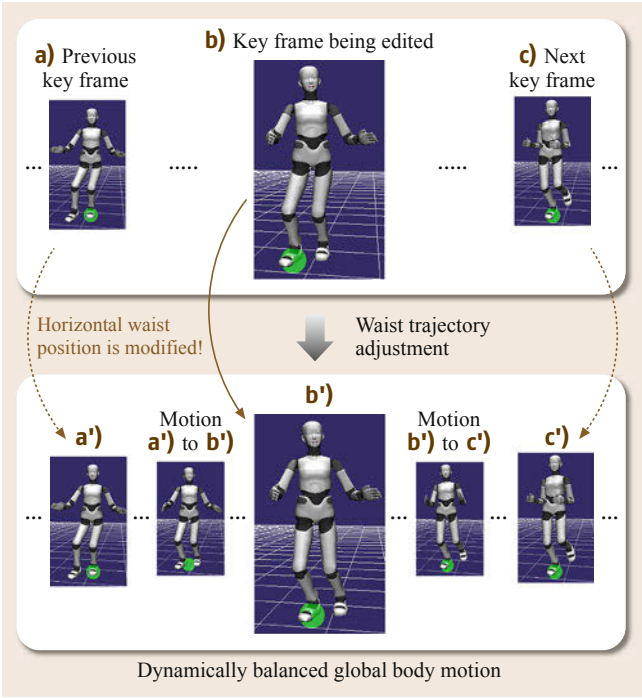


Fig. 67.32 Waist trajectory adjustment, which is automatically processed immediately after every time key poses are modified. As a result, horizontal waist positions of key poses are slightly modified and a dynamically balanced motion is obtained (after [67.77])

filter that makes reference motions dynamically feasible by a humanoid [67.94], upper-body motion compensation for Waseda bipedal humanoid (WABIAN) [67.95] have also been proposed as this type of approach.

Task–Balance Functional Decomposition

In this approach, dynamic stability is maintained during whole-body motion based on pattern generation or a balance compensation by the legs to maintain the ZMP inside the foot support area, while upper body is in charge of specified tasks such motions as manipulation or designed movement.

An iterative two-stage motion planning method has been proposed for a humanoid to perform manipulation and locomotion at the same time [67.81]. At the first stage, the motion planner generates the upper-body motion with a walking path of the bounding box of the lower body as in Fig. 67.29. The second stage overlays the desired upper-body motion on the dynamically stable walking motions generated by a dynamic walking pattern generator based on preview control of ZMP for a linear inverted pendulum model [67.96] (Chap. 48 and Sect. 67.4). This upper-body motion during walking induces errors in resulting ZMP from the reference, which may make the humanoid instable. By applying the preview control again to this ZMP error the necessary compensation motion can be computed as a horizontal offset on the waist position. If the resulting

whole-body motion is not collision-free, the planning process goes back to the first stage to reshape and this procedure is repeated until a valid motion is obtained. Figure 67.31, VIDEO 594 and VIDEO 598 show the resulting collision-free manipulation motion of a bar-shaped object in an environment populated with obstacles.

Lower-body motion compensation for dynamic balance is also used for GUI-based motion designing [67.77]. The GUI accepts a sequence of key-poses for the desired motion as the input and interpolates between them to compute the whole-body motion of the humanoid. Since the generated motion is not dynamically stable in general, both the key poses and the interpolated motion are adjusted to be dynamically balanced by applying the waist trajectory adjustment in such a way that the trajectory of the ZMP from body motion is always inside the foot support area. The adjustment only modifies the horizontal waist position of the key poses and the interpolated ones, for the adjusted motion to be as close as the original one (Fig. 67.32). The adjustment is automatically and immediately done every time a user has finished an edit operation so that the user can see the resulting motion.

67.5.3 Generating Whole-Body Motions with Various Tasks

The main purpose of the methods in previous section was to make the given coarse motion dynamically stable. One can think of a case the task is only given in an abstract manner, for instance reaching the end-effector in specified position and orientation in workspace, or aligning a camera axis in a direction. This section takes a step forward in order to generate automatically the motion to achieve the specified tasks by taking into account such constraints as balance, foot positions or joint limits at the same time. Generalized inverse kinematics technique with task priority and its extension is utilized as a key tool for local whole-body motion generation (Chap. 10).

The main particularities of humanoid robot from the viewpoint of inverse kinematics are the following: necessity of dynamic balancing, changing fixed root joint and floating base frame. Those issues should be dealt with appropriately depending on the task of the robot. Some extensions for more complex tasks including inequality constraints, footstep planning and dynamics are also mentioned at the last part of this section.

Dynamic Balancing and Walking

Some examples are shown the whole-body motion generation based on task priority generalized inverse kinematics. As shown in Chap. 10, this framework accomplishes first the task with the highest priority and then tries to achieve those with lower priority at the best in the null space of the higher-priority tasks.

Tasks are specified locally as a velocity in workspace, such as hand velocity to reach the target. The balance constraint can therefore be expressed as the velocity of the center of mass (COM). The ZMP-based pattern generator has the advantage that it outputs the velocity of the COM of dynamically stable walking motion from the reference ZMP trajectory, which can be easily integrated into this inverse kinematics framework by using COM Jacobian [67.97]. Figure 67.33 shows the whole-body reaching motion including a step to take a ball localized by a vision system [67.85]. The high priority is assigned to COM and foot motion to avoid falling in this example. As can be seen, the legs are used not only for stepping but also bending to reach a lower position in a manner coordinated with the upper body. The left arm moves backwards as the result of balancing task. Whole-body motions for manipulation of daily-life tools [67.84] or object pushing/lifting [67.98–100], and also for self-collision avoidance [67.101] have been implemented also based on a similar framework.

Another example is given in Fig. 67.34, VIDEO 595 and VIDEO 599 where the humanoid

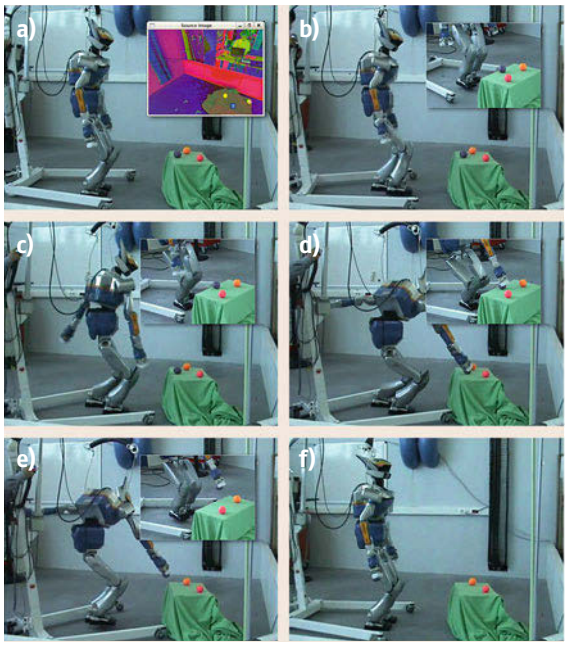


Fig. 67.33 (a–f) A whole-body grasping motion generated through task-priority generalized inverse kinematics (after [67.85]). Upper and lower bodies coordinate to achieve the desired grasping task while making a step and maintaining the balance

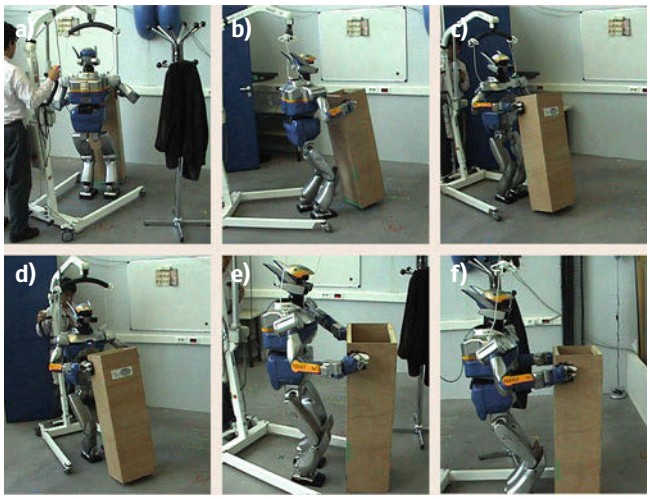


Fig.67.34a–f Experiments of whole-body *pivoting* manipulation. Starting from the initial position **(a)** with obstacle at right-hand side, the humanoid robot manipulates the object backwards away from the wall **(b)**. After switching motion direction to forward **(c)**, the robot continues to manipulate the object to the goal position by avoiding the obstacle (the hanger) **(d–f)** (VIDEO 595, VIDEO 599)

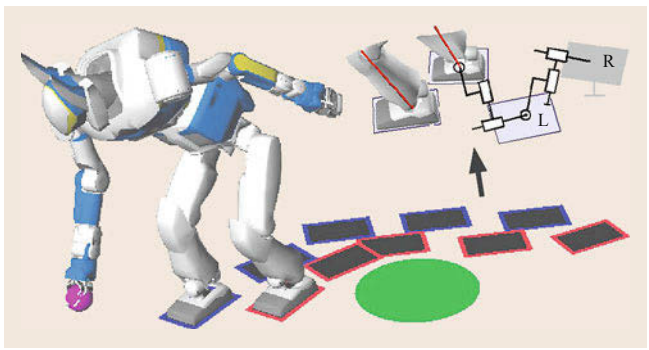


Fig. 67.35 Footstep planning modeled as a whole-body inverse kinematic problem (after [67.102]) (VIDEO 596, VIDEO 600)

execute *pivoting* manipulation to carry a bulky object without lifting [67.86]. In this case, a coarse path of the object towards its goal position is fist planned to compute the trajectory of the hands that perform the manipulation. Then foot positions are determined along the object path, from which the COM trajectory is derived using the dynamic walking pattern generator. Those tasks are provided to the inverse kinematics to generate the coordinated arm and leg motion for this complex manipulation. In addition to the dynamic balance, the change of the root joint is also considered to compute the whole-body motion when the support leg changes during walking. The same framework has been applied to a motion for catching a moving object during walking where the task of visual tracking by the end-effector is integrated with dynamic walking motion [67.103].

Floating base frame of a humanoid sometimes brings difficulties in determining the whole-body configuration from a specific abstract task specified in workspace. In the methods mentioned above, the goal configuration is derived as a result of repeated computation of local generalized inverse kinematics. However, there are often cases where goal whole-body configurations is first needed to be used popular motion planning techniques searching in configuration space such as sampling-based planning. Some methods that derive the goal configuration based on inverse kinematics can be useful for this purpose [67.90, 91, 104] or a precomputed reachability map that characterizes the capacity of reaching in discretized workspace around the robot [67.105, 106].

Extensions for Complex Tasks

The whole-body motion generation with tasks can be extended to cope with more complex tasks such as stepping and those expressed as inequality or dynamic constraints.

One extension particular to humanoid is incorporating stepping in the framework of whole-body generalized inverse kinematics. *Kanoun* et al. [67.102] introduced an augmented robot structure by introducing *virtual* planar links attached to a foot that represents footsteps as illustrated in Fig. 67.35, VIDEO 596 and VIDEO 600. This modeling makes it possible to solve the footstep planning as a problem of inverse kinematics, and also to determine the final whole-body configuration. After planning the footsteps, the dynamically stable whole-body motion including walking can be computed by using the method presented earlier.

Task-priority generalized inverse kinematics for redundant robots in Sect. 10.3 usually models tasks as equalities so that the operational points can achieve the desired velocity. However, tasks are sometimes given as inequalities: keeping the hand out of some area to avoid collisions or robot view obstruction, respecting joint limits, or maintaining the COM inside the foot support area. Inequality tasks have usually been transformed into more restrictive equality constraints through potential fields. A method for extending the task-priority inverse kinematics for those inequality tasks is proposed to remove this limitation based on a sequence of QP optimization [67.87]. This method searches for the optimal sets for the sequence of QPs by minimizing the error to the desired equality tasks in such a way that inequality ones can also be satisfied at a desired priority. This method allows the humanoid to perform such

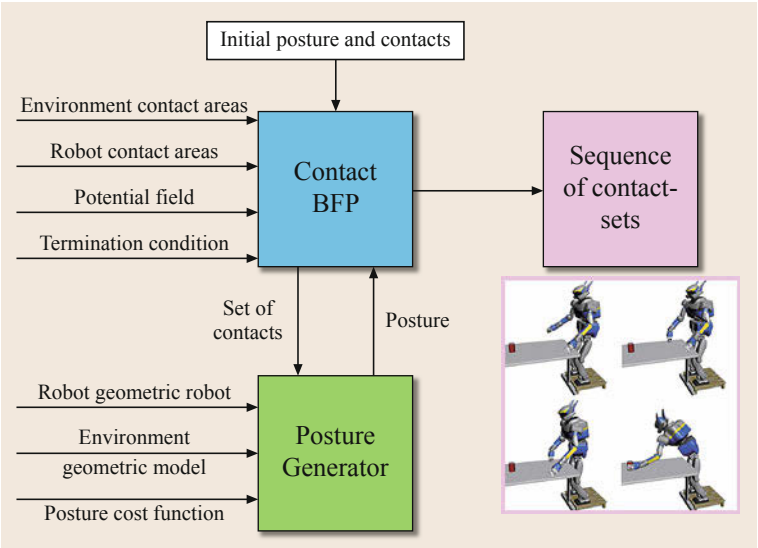


Fig. 67.36 Framework of contact motion planner composed of contact best-first-planner (BFP) and posture generator (after [67.107])

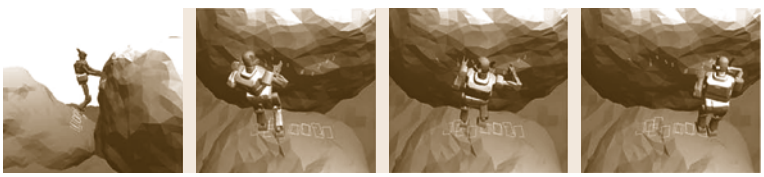


Fig. 67.37 A sequence of planned whole-body motion including multiple contacts on a irregular terrain (after [67.108])

a task of reaching its arm towards an object on the floor without obstructing its view.

This approach based on cascaded QP can be generalized to generate whole-body motions including dynamic equality and inequality tasks [67.109]. In addition to inverse kinematics considered so far, inverse dynamics is also integrated to the task-priority whole-body motion generation framework. By using this method, the dynamic balance can be addressed directly without converting the dynamic ZMP constraint into COM velocity via a pattern generator. This approach assumes a torque-controlled humanoid as opposed to position controlled ones that are often the case for platform currently used. However, dynamic whole-body motion generation with various tasks is being actively studied owing to not only recent progress of robot hardware [67.110] but also increasing interests on more complex tasks including multiple contacts presented in the next section.

67.5.4 Generating Motions Including Multiple Contacts

The whole-body motions presented so far basically supposes only the contacts between the humanoid's feet and the floor. Looking at our daily life however, contacts other than with feet occurs often, for example pass through narrow spaces or to support the body when reaching a distant place on the desk. Since a humanoid is high affinity to environments designed for humans, its application fields could be expanded by exploiting the contacts as much as possible rather than by avoiding them as is often the case in motion planning. This section addresses planning and control of whole-body motions with multiple contacts that have been intensively studied in recent years. The role of planner is to derive a global sequence of configuration with multiple contacts to reach the goal, whereas the controller generates dynamically stable motions to transit from one contact state to another.

Motion Planning for Multiple Contacts

Multicontact nongaited have been proposed that are applicable to legged robots including humanoids [67.107, 108, 112]. By defining a stance as a finite set of contacts between the robot and the environment, the planner generates a sequence of stances that can reach the goal. During the planning, possible transitions from a stance are explored by sampling another stance with a feasible and stable robot configuration, as shown in Fig. 67.36 [67.107]. Figure 67.37 shows a resultant locomotion planned using this planning method [67.108]. A more generalized framework is proposed to deal with multirobot and multiobject systems [67.113]. This

Fig. 67.38 Dynamic multicontact motion generated through global trajectory optimization (after [67.111], VIDEO 597)

generalization allows for a common description and treatment of locomotion and manipulation problems, either for a single robot or for multiple collaborating robots.

The output of the planner is a sequence of statically stable contact stances that can be executed through quasi-static motions. In order to generate fast but dynamically stable motions, a global optimization approach has been proposed [67.111]. This method parameterizes joint trajectory by B-Spline function to convert the infinite trajectory problem to semi-infinite one so that optimization technique can be applied. The whole-body motion is generated through nonlinear optimization to minimize square torque and execution time, by taking into account such constraints as joint torque limits and dynamic multiple contact stability. As a result, the generated motion is much faster than quasi-static ones. An example of resultant motion is shown in Fig. 67.38 and VIDEO 597. Trajectory optimization approach can also be applied for lifting of a very heavy object by generating a weight-lifting motion by always respecting physical constraints [67.114].

Even though the feasibility of multicontact motions like in Fig. 67.38 have been validated with experiments, the optimization process is time-consuming and cannot cope with errors or disturbances during execution. It is therefore necessary to build a controller that ensures execution of multicontact whole-body motions.

Stability Measure for Motion with Multiple Contacts

Before discussing controller, it is worth mentioning first dynamic stability measure for non coplanar contacts and its usage. Although the ZMP is well-known dynamic stability criteria, it can only be applied to contacts on a flat plane. Stability margin for mobile robots [67.118] is only applicable for static gait for legged robots. A generalized version of ZMP (GZMP) has therefore been proposed [67.119] by extending the ZMP by considering interaction forces other than floor-feet contacts. The stable region for multiple contacts can be obtained by considering the infinitesimal displacement and the moment about the edges of the convex hull of the supporting points. *Hirukawa* et al. proposed another criterion called contact wrench sum (CWS), which is the sum of the gravity and the inertia wrench applied to the COG of the robot. The humanoid is stable if it is inside the polyhedral convex cone of the contact wrench between the feet of a robot and its environment [67.115, 120]. Based on this criterion, walking

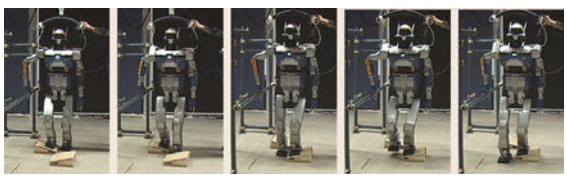


Fig. 67.39 Walking on a rough terrain using a support of arm on a handrail (after [67.115])

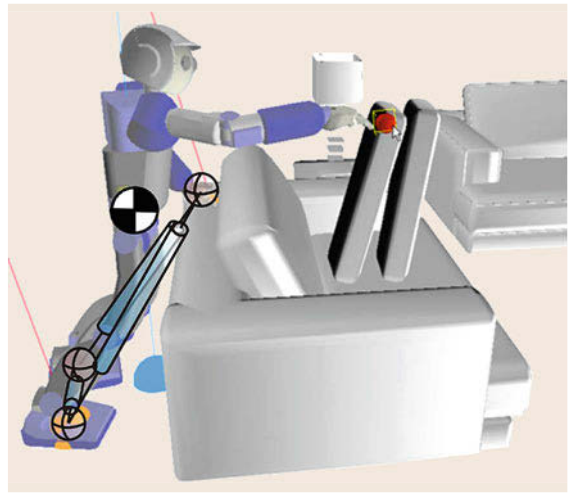


Fig. 67.40 Real-time simulation of a multicontact behavior with user-enabled interactive control of the robot's right hand. A virtual linkage model is overlaid capturing the internal force behaviors acting between supporting bodies (after [67.116])

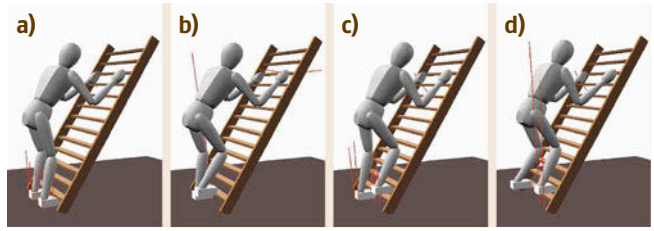


Fig.67.41a–d A digital figure climing a ladder by using whole-body controller with task priority with multiple contacts (after [67.117])

Fig. 67.42 Snapshots from multicontact planning and control of HRP-2 ingress in a car (after [67.121]). From the initial posture, supported on the steering handle and the seat by an arm, the robot finally succeeds in entering the car

on a rough terrain by supporting a handrail has been performed (Fig. 67.39).

Controlling Whole-Body with Multiple Contacts

The execution of the multicontact motions presented earlier by a humanoid requires the control based on sensor feedback to absorb unexpected disturbance or modeling errors.

Khatib et al. extended their framework of operational space approach that enables a humanoid robot to perform motions that simultaneously meet prioritized objectives in existence of multiple contacts [67.116, 117]. A torque-based approach for the control of internal forces is suggested and integrated into the framework for whole-body prioritized multitasking, thus enabling the unified control of COM maneuvers, operational tasks, and internal-force behavior (Fig. 67.40). Figure 67.41 shows an example of multicontact behavior on a real-time simulator. *Hyon* et al. proposed another approach of passivity-based controller that can adapt to unknown external forces applied to arbitrary contact points without sensing the contact forces by using a torque-controlled robot [67.122].

Another real-time controller based on linear QP optimization has been proposed for whole-body motion with multiple contacts [67.121, 123]. The motion is constrained by the free-floating whole-body dynamics of the humanoid robot without using a reduced model such as inverted pendulum since the motions we aim at are more general than bipedal walking. The optimization process also incorporates such constraints as nonsliding condition, actuation torque limits, contact forces within friction cones, and avoidance of undesirable self-collisions and collisions with the environment. Taking a target contact stance and also necessary sensory information as its input, the controller can compute feedback control commands to execute the whole-body motion with multiple contacts in real time, on less than 100 Hz control loop. Figure 67.42 shows a simulation result of complex motion of a humanoid entering into a car based on the control method.

67.6 Morphological Communication

Humans evaluate each others' state through body posture and movement. It is quite natural to extend this form of communication to include robots that share our morphology.

67.6.1 Expressive Morphology and Behavior

Humanoids can communicate with people through expressive morphology and behavior. As with people, humanoid robots integrate communicative and noncommunicative functionality. For example, the arms and hands of a robot can reach and grasp, but also point and gesture. Heads for humanoid robots are an especially important example of these overlapping roles, and have had an important impact on humanoid robotics and robotics in general [67.124].

The head of a humanoid robot has two main functions:

- To orient directional sensors as needed for the purposes of perception, while leaving the main body free to meet other constraints such as maintaining balance and gait. Cameras and sometimes microphones are usefully oriented in this way.
- To strike expressive poses, along with the rest of the body. Even if a robot head is not intended to be expressive, it will be interpreted as being so by humans – particularly as a cue to the robot's presumed locus of visual attention. It is also possible to deliberately engineer an approximate *face* that can be an important line of communication with humans (Chap. 72).

Locus of Attention

Eyes can be one of the most expressive components of a humanoid robot. For humans, eye movements are both expressive and important for sensing. Humanoid robots

have the option to factor these two roles by moving eyes that are only for display, and using sensors placed elsewhere. Most humanoid robots, however, use head-mounted servoed cameras that play both expressive and sensory roles. These mechanisms exhibit different degrees of biological realism, for example, the Kismet head captured many of the expressive components of human eye movements, while having a nonhuman-like camera arrangement that simplified some forms of perception (Fig. 67.43).

Many humanoid robots use biologically inspired, foveated vision systems, which provide a wide field of view with low detail, combined with a narrow field of view with high detail (Fig. 67.44). With appropriate control strategies to fixate the narrow field of view on task-salient regions detected in the wide field of view, these robots achieve a practical compromise between resolution and field of view. Additionally, the configuration of the eyes communicates the robot's locus of attention in an intuitive way. Many systems use four cameras, with a narrow- and wide-angle camera for each of the robot's eyes, but some researchers have also used special-purpose space-variant cameras modeled after the space-variant receptor densities in the human eye [67.125].

The eye movements of some humanoids are modeled explicitly after human eye movements. An example of a model of this kind is shown in Fig. 67.45. These

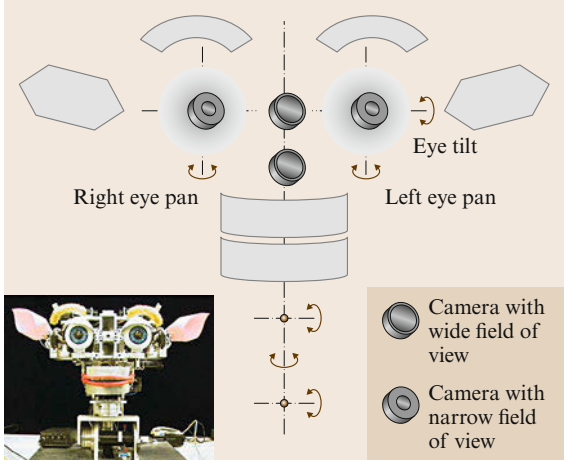


Fig. 67.43 On Kismet, foveal vision was implemented using cameras in the eyes, and peripheral vision used unobtrusive cameras on the head (after [67.124]). This achieved good expression of locus of attention, while simplifying the process of differentiating egomotion from motion of objects (since the head moved less frequently and more slowly than the eyes). This is an example of a partial decoupling of expressive and functional concerns, showing that many different levels of humanoid *fidelity* are possible

bio-inspired approaches to active vision typically have four types of visual behavior:

Saccades. These are high-velocity movements to fixate a new target or *catch up* with a fast-moving target. From a control point of view, these movements are *ballistic* (at least in humans) – once initiated, they continue without responding to changing stimuli.

Smooth Pursuit. These are movements to continuously track a moving target. They apply at low velocities. These movements respond constantly to visual feedback about the target's location. A fast-moving target may also trigger small saccades.

VOR and OKR. The vestibulo-ocular reflex and optokinetic response work to stabilize the direction of gaze in the presence of movement of the head and body, using inertial and visual information respectively.

Vergence. This movement drives the relative angle of the two eyes so that the same target is centered in both. This only applies to two-eyed systems that have this freedom of motion. For conventional stereo algorithms,

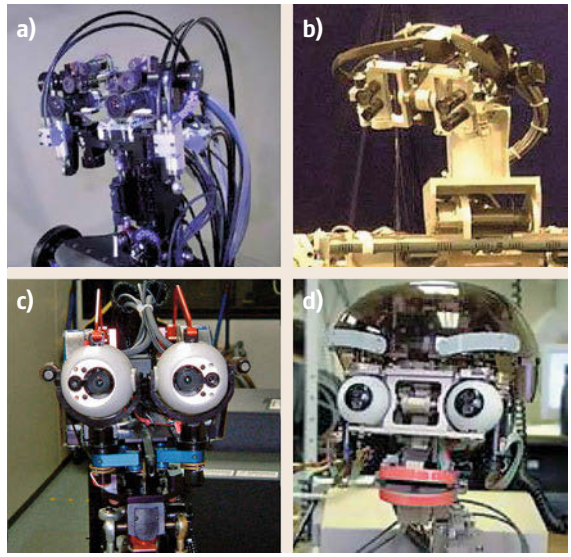


Fig.67.44a–d The heads of humanoid robots come in many forms. A popular arrangement is to have two cameras per *eye*, as a crude approximation of foveal and peripheral vision in humans. **(a)** Biomimetic oculomotor control investigated on DB (after [67.126]). **(b)** Cog's head (after [67.127]). **(c)** The double-camera arrangement can be arranged in a less-double-barreled appearance (after [67.128]; see *Ude* et al. [67.128] for more examples and an analysis) ATR; Humanoid head developed by ATR and SARCOS. **(d)** The Infanoid robot (after [67.129])

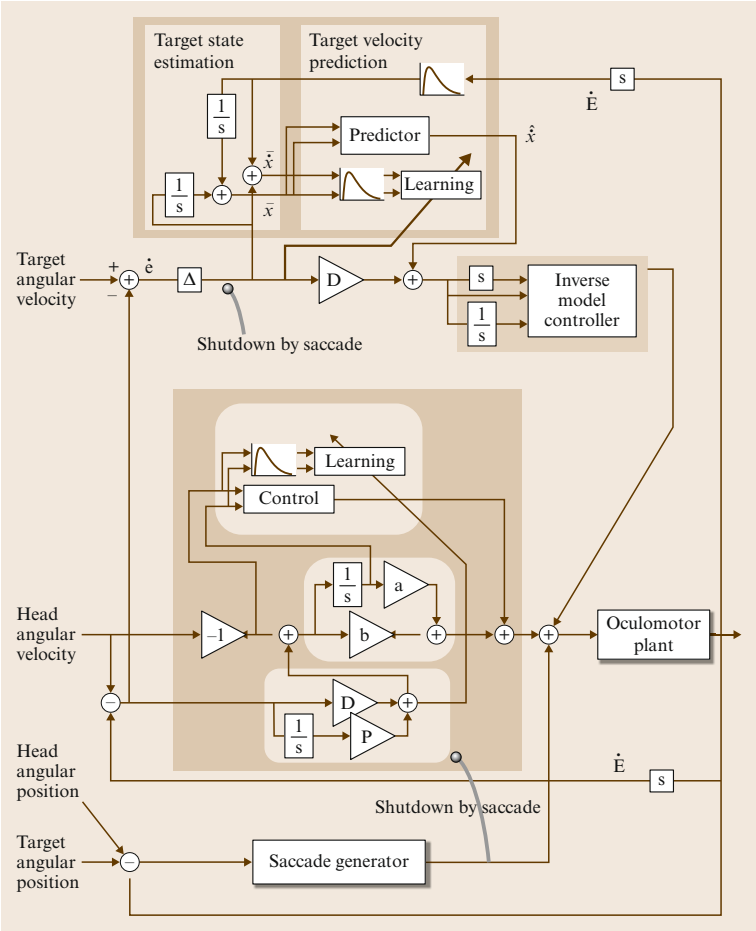


Fig. 67.45 A biomimetic control model (after [67.130]), that integrates saccading, smooth pursuit, the vestibular-ocular reflex (VOR), and the optokinetic response (OKR). Smooth pursuit and VOR/OKR commands are summed, with periodic corrections to errors in position made by saccades

vergence is a disadvantage, since the algorithms are simplest when the cameras remain parallel. Other algorithms are possible, but it is currently quite common not to use vergence.

67.6.2 Interpreting Human Expression

The interpretation of human expression is essential for many forms of natural human communication that could be valuable for humanoid robots.

Posture and Expression

The recognition and interpretation of the location and pose of humans is important, since humanoids are often expected to work in human environments. Algorithms for the following functions have been incorporated in various humanoids:

- Person finding
- Person identification
- Gesture recognition
- Face pose estimation.

ASIMO has used these functions to perform a prototypical reception task as shown in Fig. 67.46. The robot can find and identify a person, then recognize gestures such as *bye-bye*, *come here*, and *stop*, which are utilized for performing reception tasks. In general, such func-

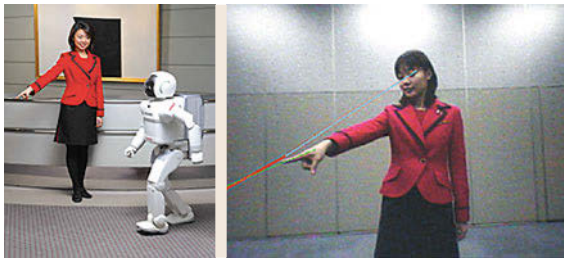


Fig. 67.46 ASIMO recognizing a pointing gesture during a reception task

tions on a humanoid are not yet robust, and are active areas of research.

Speech Recognition

Speech is a natural, hands-free mode of communication between humans, and potentially between robots and humans. Speech recognition is a popular interface utilized for commanding a humanoid, and many off-the-shelf packages are now available. However the use of microphones embedded in the robot is problematic, because general-purpose speech recognition software is usually optimized for utterances captured by a microphone that is close to the speaker. In order to achieve sufficient recognition performance in natural interaction situations between a humanoid and a human new methods for speech recognition are being investigated. These methods compensate for sources of noise, such as the robot's motors and air flow in the environment, by using multiple microphones and multimodal cues [67.131]. However, at the time of writing researchers often circumvent these issues by using a headset, lavalier, or handheld microphones.

Auditory Scene Analysis

In order to attain more-sophisticated human–robot interaction, researchers have been developing methods for computational auditory scene analysis on a humanoid robot. The objective of this research is to understand an arbitrary sound mixture including nonspeech sounds and voiced speech, obtained by microphones embedded in the robot. Beyond speech recognition, this also involves sound-source separation and localization.

As for sound recognition, sound categories such as *coughing*, *laughing*, *beating by hand*, *adult's voice*, and *child's voice* have been shown to be recognizable using maximum-likelihood estimation with Gaussian mixture models. This function has been utilized during interactions between the HRP-2 and a human [67.132].

Multimodal Perception

Sound-source separation can be achieved by beam forming. In order to perform beamforming effectively, sound-source localization is essential. Vision can be utilized for finding the talker within the field of view. *Hara* et al. used a camera and an eight-channel microphone array embedded in the head of HRP-2, and succeeded in speech recognition in the presence of multiple sound sources by using sound source separation [67.133]. Figure 67.47 shows a scenario in which speech recognition is taking place with television (TV) sound playing in the background.

When integrated with speech recognition, vision can also help resolve the ambiguities of speech. For instance, the ambiguity of demonstrative pronouns such as *this* or *that* can sometimes be resolved by recognizing pointing gestures. Similarly, the face and gaze direction can be used to realize communication via eye contact, so that the humanoid only replies when a human is looking at it and talking to it [67.132]. Multimodal interaction with these functions has also been demonstrated by HRP-2, as shown in Fig. 67.48.

67.6.3 Physical Interaction and Developmental Robotics

Humanoid robots typically use methods for perception and interaction that are established in fields such as computer vision and dialogue systems. There is also an emerging research field called developmental robotics or epigenetic robotics in which human-like perception and interaction abilities are obtained through physical interaction with the real environments including humans [67.134–136]. In developmental robotics, researchers aim at studying the developmental mechanisms, architectures and constraints that allow life-long and open-ended learning of new skills and new knowledge in embodied machines. Much of the research in this field utilizes humanoid robots, such as the iCub shown in Fig. 67.36. As in human children, learning is expected to be cumulative and of progressively in-

Fig. 67.47 HRP-2 recognizing speech with background noises (TV sound)

Fig. 67.48 HRP-2 recognizing face and gaze direction for communication via eye contact

creasing complexity, and to result from self-exploration of the world in combination with social interaction. The typical methodological approach consists in starting from theories of human development elaborated in fields such as developmental psychology, cognitive science, neuroscience, developmental and evolutionary biology, and linguistics, then to formalize and implement them in robots. The experimentation of those models in robots allows researchers to confront them with reality, and as a consequence developmental robotics also provides feedback and novel hypothesis on theories of human development.

67.7 Conclusions and Further Reading

Because of the integrative nature of humanoid robotics, this chapter has avoided details and formalisms and liberally cross-referenced other chapters within the handbook that can provide the reader with deeper coverage of many of the areas of robotics on which humanoid robots depend. Additionally, this chapter references work within the humanoid robotics community and related communities.

Humanoid robotics is an enormous endeavor. The emulation of human-level abilities in a human-like robot serves as a grand challenge for robotics, with significant cultural ramifications. The motivations for humanoid robotics are as deep as they are diverse. From the earliest cave drawings, humanity has sought to represent itself. Robotics is one of the most recent mediums for this ongoing fascination. Besides this deep societal motivation, humanoid robots offer unique opportunities for human–robot interaction, and integration into human-centric settings.

Over the last decade, the number of humanoid robots developed for research has grown dramatically, as has the research community. Humanoid robots have already gained a foothold in the marketplace as robots for entertainment and research (e.g., the Robo-One competition and the NAO from Aldebaran). Given the special properties of humanoid robots, they seem likely to further increase in number as their capabilities improve and their costs go down. Robots with human characteristics, and technologies related to humanoid robotics, also appear destined to proliferate. Will human-scale, legged robots with human form become commonplace, as so often imagined by science fiction? Only time will tell.

Video-References

VIDEO 594 3-D collision-free motion combining locomotion and manipulation by humanoid robot HRP-2
available from http://handbookofrobotics.org/view-chapter/67/videodetails/594

VIDEO 595 Whole-body pivoting manipulation
available from http://handbookofrobotics.org/view-chapter/67/videodetails/595

VIDEO 596 Footstep planning modeled as a whole-body inverse kinematic problem
available from http://handbookofrobotics.org/view-chapter/67/videodetails/596

VIDEO 597 Dynamic multicontact motion
available from http://handbookofrobotics.org/view-chapter/67/videodetails/597

VIDEO 598 3-D collision-free motion combining locomotion and manipulation by humanoid robot HRP-2 (experiment)
available from http://handbookofrobotics.org/view-chapter/67/videodetails/598

VIDEO 599 Regrasp planning for pivoting manipulation by a humanoid robot
available from http://handbookofrobotics.org/view-chapter/67/videodetails/599

VIDEO 600 Footstep planning modeled as a whole-body inverse kinematic problem (experiment)
available from http://handbookofrobotics.org/view-chapter/67/videodetails/600

References

67.1 H. Inoue, S. Tachi, Y. Nakamura, K. Hirai, N. Ohyu, S. Hirai, K. Tanie, K. Yokoi, H. Hirukawa: Overview of humanoid robotics project of METI, Proc. 32nd Int. Symp. Robotics (ISR) (2001)

67.2 Y. Kuroki, M. Fujita, T. Ishida, K. Nagasaka, J. Yamaguchi: A small biped entertainment robot exploring attractive applications, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 471–476

67.3 C.G. Atkeson, J.G. Hale, F.E. Pollick, M. Riley, S. Kotosaka, S. Schaal, T. Shibata, G. Tevatia, A. Ude, S. Vijayakumar, M. Kawato: Using humanoid robots to study human behavior, IEEE Intell. Syst. **15**(4), 46–56 (2000)

67.4 K.F. MacDorman, H. Ishiguro: The uncanny advantage of using androids in social and cognitive science research, Interact. Stud. **7**(3), 297–337 (2006)

67.5 V. Bruce, A. Young: *In the Eye of the Beholder: The Science of Face Perception* (Oxford Univ. Press, Oxford 1998)

67.6 R. Blake, M. Shiffrar: Perception of human motion, Annu. Rev. Psychol. **58**, 47–73 (2007)

67.7 J.F. Werker, R.C. Tees: Influences on infant speech processing: Toward a new synthesis, Annu. Rev. Psychol. **50**, 509–535 (1999)

67.8 D. Hanson, A. Olney, I.A. Pereira, M. Zielke: Upending the uncanny valley, Nat. Conf. Artif. Intell. (AAAI), Pittsburgh (2005)

67.9 K. Čapek: *R.U.R. (Rossum's Universal Robots), A Play in Introductory Scene and Three Acts* (eBooks, Adelaide 2006), translated into English by D. Wyllie

67.10 Metropolis, directed by Fritz Lang (DVD) (Kino Video, 1927)

67.11 I. Asimov: *The Caves of Steel* (Bantam, New York 1954)

67.12 B. Adams, C. Breazeal, R.A. Brooks, B. Scassellati: Humanoid robots: A new kind of tool, IEEE Intell. Syst. **15**(4), 25–31 (2000)

67.13 R.A. Brooks, L.A. Stein: Building brains for bodies, Auton. Robotics **1**(1), 7–25 (1994)

67.14 R.W. Gibbs Jr.: *Embodiment and Cognitive Science* (Cambridge Univ. Press, Cambridge 2006)

67.15 M. Lungarella, G. Metta: Beyond gazing, pointing, and reaching: A survey of developmental robotics, Proc. Third Int. Workshop Epigenet. Robotics (2003) pp. 81–89

67.16 G. Metta, G. Sandini, D. Vernon, D. Caldwell, N. Tsagarakis, R. Beira, J. Santos-Victor, A. Ijspeert, L. Righetti, G. Cappiello, G. Stellin, F. Becchi: The RobotCub project – An open framework for research in embodied cognition, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2005)

67.17 E. Grandjean, K. Kroemer: *Fitting the Task to the Human*, 5th edn. (Routledge, London 1997)

67.18 W. Karwowski: *International Encyclopedia of Ergonomics and Human Factors*, 2nd edn. (CRC, Boca Raton 2006)

67.19 R. Brooks, L. Aryananda, A. Edsinger, P. Fitzpatrick, C. Kemp, U.-M. O'Reilly, E. Torres-Jara, P. Varshavskaya, J. Weber: Sensing and manipulating built-for-human environments, Int. J. Humanoid Robotics **1**(1), 1–28 (2004)

67.20 W. Bluethmann, R. Ambrose, M. Diftler, S. Askew, E. Huber, M. Goza, F. Rehnmark, C. Lovchik, D. Magruder: Robonaut: A robot designed to work with humans in space, Auton. Robotics **14**(2/3), 179–197 (2003)

67.21 K. Yokoi, K. Nakashima, M. Kobayashi, H. Mihune, H. Hasunuma, Y. Yanagihara, T. Ueno, T. Gokyuu, K. Endou: A tele-operated humanoid operator, Int. J. Robotics Res. **22**(5/6), 593–602 (2006)

67.22 K. Yokoi, K. Kawauchi, N. Sawasaki, T. Nakajima, S. Nakamura, K. Sawada, T. Takeuchi, K. Nakashima, Y. Yanagihara, K. Yokohama, T. Isozumi, Y. Fukase, K. Kaneko, H. Inoue: Humanoid robot applications in HRP, Int. J. Humanoid Robotics **1**(3), 409–428 (2004)

67.23 B. Thibodeau, P. Deegan, R. Grupen: Static analysis of contact forces with a mobile manipulator, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 4007–4012

67.24 J. Gutman, M. Fukuchi, M. Fujita: Modular architecture for humanoid robot navigation, Proc. 5th IEEE/RAS Int. Conf. Humanoid Robotics (2005) pp. 26–31

67.25 S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, K. Ikeuchi: Generating whole body motions for a biped humanoid robot from captured human dances, IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3905–3910

67.26 Boston Dynamics: http://www.bostondynamics.com (2013)

67.27 M.E. Rosheim: *Robot Evolution: The Development of Anthrobotics* (Wiley, New York 1994)

67.28 M.E. Rosheim: *Leonardo's Lost Robots* (Springer, Berlin, Heidelberg 2006)

67.29 T.N. Hornyak: *Loving the Machine: The Art and Science of Japan's Robots* (MIT Press, Cambridge 2006)

67.30 J. Surrell: *Pirates of the Caribbean: From the Magic Kingdom to the Movies* (Disney, New York 2005)

67.31 K. Hirai, M. Hirose, Y. Haikawa, T. Takenaka: The development of Honda humanod robot, IEEE Int. Conf. Robotics Autom. (ICRA) (1998) pp. 1321–1326

67.32 M. Williamson: Robot Arm Control Exploiting Natural Dynamics, Ph.D. Thesis (MIT, Cambridge 1999)

67.33 J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, E. Thelen: Autonomous mental development by robots and animals, Science **291**(5504), 599–600 (2001)

67.34 J. Zlatev, C. Balkenius: Why "epigenetic robotics"?, Proc. 1st Int. Workshop Epigenet. Robotics Model. Cogn. Dev. Robotics Syst., ed. by C. Balkenius, J. Zlatev, H. Kozima, K. Dautenhahn, C. Breazeal (Lund Univ. Press, Lund 2001) pp. 1–4

67.35 Y. Sodeyama, I. Mizuuchi, T. Yoshikai, Y. Nakanishi, M. Inaba: A shoulder structure of muscle-driven humanoid with shoulder blades, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2005) pp. 4028–4033

67.36 I. Mizuuchi, M. Inaba, H. Inoue: A flexible spine human-form robot-development and control of the posture of the spine, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS), Vol. 4 (2001) pp. 2099–2104

67.37 M. Vande Weghe, M. Rogers, M. Weissert, Y. Matsuoka: The ACT hand: Design of the skeletal structure, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

67.38 L.Y. Chang, Y. Matsuoka: A kinematic thumb model for the ACT hand, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1000–1005

67.39 C. Lovchik, M.A. Diftler: The robonaut hand: A dexterous robot hand for space, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 907–912

67.40 M.J. Marjanović: Teaching an Old Robot New Tricks: Learning Novel Tasks via Interaction with People and Things, Ph.D. Thesis (MIT, Cambridge 2003)

67.41 C. Ott, O. Eiberger, W. Friedl, B. Bäuml, U. Hillenbrand, C. Borst, A. Albu-Schäffer, B. Brunner, H. Hirschmüller, S. Kielhöfer, R. Konietschke, M. Suppa, T. Wimböck, F. Zacharias, G. Hirzinger: A humanoid two-arm system for dexterous manipulation, Proc. IEEE-RAS Int. Conf. Humanoid Robot. (2006) pp. 276–283

67.42 G. Pratt, M. Williamson: Series elastic actuators, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS), Pittsburg, Vol. 1 (1995) pp. 399–406

67.43 M. Vukobratović, J. Stepanenko: On the stability of anthropomorphic systems, Math. Biosci. **15**, 1–37 (1972)

67.44 M. Morisawa, K. Harada, S. Kajita, S. Nakaoka, K. Fujiwara, F. Kanehiro, K. Kaneko, H. Hirukawa: Experimentation of humanoid walking allowing immediate modification of foot place based on analytical solution, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 3989–3994

67.45 K. Nishiwaki, S. Kagami: High frequency walking pattern generation based on preview control of ZMP, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 2667–2672

67.46 K. Harada, S. Kajita, F. Kanehiro, K. Fujiwara, K. Kaneko, K. Yokoi, H. Hirukawa: Real-time planning of humanoid robot's gait for force controlled manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 616–622

67.47 J. Urata, K. Nshiwaki, Y. Nakanishi, K. Okada, S. Kagami, M. Inaba: Online walking pattern generation for push recovery and minimum delay to commanded change of direction and speed, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 3411–3416

67.48 J. Chestnutt, M. Phillipe, J.J. Kuffner, T. Kanade: Locomotion among dynamic obstacles for the honda ASIMO, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007) pp. 2572–2573

67.49 K. Nishiwaki, J. Chestnutt, S. Kagami: Autonomous navigation of a humanoid robot over unknown rough terrain using a laser range sensor, Int. J. Robotics Res. **31**(11), 1251–1262 (2012)

67.50 K. Nagasaka, K. Kuroki, S. Suzuki, Y. Itoh, J. Yamaguchi: Integrated motion control for walking, jumping and running on a small bipedal entertainment robot, IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 3189–3194

67.51 B.-K. Cho, S.-S. Park, J.-H. Oh: Stabilization of a hopping humanoid robot for a push, Proc. Int. Conf. Humanoid Robots (Humanoids) (2010) pp. 60–65

67.52 R. Tajima, D. Honda, K. Suga: Fast running experiments involving a humanoid robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 1571–1576

67.53 H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, M. Morisawa: A universal stability criterion of the foot contact of legged robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1976–1983

67.54 S.-H. Hyon, G. Cheng: Disturbance rejection for biped humanoids, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2668–2675

67.55 K. Miura, M. Morisawa, F. Kanehiro, S. Kajita, K. Kaneko, K. Yokoi: Human-like walking with toe supporting for humanoids, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 4428–4435

67.56 S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, K. Yokoi: Biped Walking stabilization based on linear inverted pendulum tracking, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 4489–4496

67.57 S.H. Collins, A.L. Ruina, R. Tedrake, M. Wisse: Efficient bipedal robots based on passive-dynamic walkers, Science **307**, 1082–1085 (2005)

67.58 K. Bouyarmane, A. Kheddar: Multi-contact stances planning for multiple agents, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5353–5546

67.59 K. Fujiwara, F. Kanehiro, S. Kajita, K. Kaneko, K. Yokoi, H. Hirukawa: UKEMI: Falling motion control to minimize damage to biped humanoid robot, IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 2521–2526

67.60 H. Hirukawa, S. Kajita, F. Kanehiro, K. Kaneko, T. Isozumi: The human-size humanoid robot that can walk, lie down and get up, Int. J. Robotics Res. **24**(9), 755–769 (2005)

67.61 Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, K. Fujimura: The intelligent ASIMO: System overview and integration, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2002) pp. 2478–2483

67.62 K. Okada, S. Kagami, M. Inaba, H. Inoue: Plane segment finder: Algorithm implementation and applications, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 2120–2125

67.63 K. Sabe, M. Fukuchi, J.-S. Gutmann, T. Ohashi, K. Kawamoto, T. Yoshigahara: Obstacle avoidance and path planning for humanoid robots using stereo vision, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004)

67.64 J. Gutman, M. Fukuchi, M. Fujita: Real-time path planning for humanoid robot navigation, Proc. Int. Jt. Conf. Artif. Intell. (2005) pp. 1232–1238

67.65 K. Harada, S. Kajita, K. Kaneko, H. Hirukawa: Pushing manipulation by humanoid considering two-kinds of ZMPs, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 1627–1632

67.66 K. Harada, S. Kajita, K. Kaneko, H. Hirukawa: Dynamics and balance of a humanoid robot during manipulation tasks, IEEE Trans. Robotics **22-3**, 568–575 (2006)

67.67 H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, M. Morisawa: A universal stability criterion of the foot contact of legged robots – Adios ZMP, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1976–1983

67.68 Y. Hwang, A. Konno, M. Uchiyama: Whole body cooperative tasks and static stability evaluations for a humanoid robot, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2003) pp. 1901–1906

67.69 T. Takenaka: Posture control for a legged mobile robot, Jap. Pat. 1023 0485 (1998)

67.70 K. Inoue, H. Yoshida, T. Arai, Y. Mae: Mobile manipulation of humanoids – Real-time control based on manipulability and stability, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 2217–2222

67.71 K. Harada, S. Kajita, H. Saito, M. Morisawa, F. Kanehiro, K. Fujiwara, K. Kaneko, H. Hirukawa: A humanoid robot carrying a heavy object, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1724–1729

67.72 T. Takubo, K. Inoue, K. Sakata, Y. Mae, T. Arai: Mobile manipulation of humanoid robots – control method for CoM position with external force, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2004) pp. 1180–1185

67.73 D. Kulic, D. Lee, C. Ott, Y. Nakamura: Incremental learning of full body motion primitives for humanoid robots, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2008) pp. 326–332

67.74 K. Yamane, S.O. Anderson, J.K. Hodgins: Controlling humanoid robots with human motion data: Experimental validation, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2010) pp. 504–510

67.75 J.-H. Oh, J.-W. Heo: Upper body motion interpolation for humanoid robots, Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatro. (AIM) (2011) pp. 1064–1069

67.76 S. Nakaoka, A. Nakazawa, F. Kanehiro, K. Kaneko, M. Morisawa, K. Ikeuchi: Task model of lower body motion for a biped humanoid robot to imitate human dances, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2005) pp. 2769–2774

67.77 S. Nakaoka, S. Kajita, K. Yokoi: Intuitive and flexible user interface for creating whole body motions of biped humanoid robots, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2010) pp. 1675–1682

67.78 S. Nakaoka: Choreonoid: Extensible virtual robot environment built on an integrated GUI framework, Proc. IEEE/SICE Int. Symp. System Integration (SII) (2012) pp. 79–85

67.79 J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, H. Inoue: Dynamically-stable motion planning for humanoid robots, Auton. Robots **12**(1), 105–118 (2002)

67.80 S. Dalibard, A. Nakhaei, F. Lamiraux, J.-P. Laumond: Whole-body task planning for a humanoid robot: a way to integrate collision avoidance, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2009) pp. 355–360

67.81 E. Yoshida, C. Esteves, I. Belousov, J.-P. Laumond, T. Sakaguchi, K. Yokoi: Planning 3D collision-free dynamic robotic motion through iterative reshaping, IEEE Trans. Robotics **24**(5), 1186–1198 (2008)

67.82 K. Harada, Yoshida Ei, K. Yokoi (Eds.): *Motion Planning for Humanoid Robots* (Springer, Berlin, Heidelberg 2010)

67.83 H. Janßen, M. Gienger, C. Goerick: Task-oriented whole body motion for humanoid robots, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2005) pp. 238–244

67.84 K. Okada, M. Kojima, Y. Sagawa, T. Ichino, K. Sato, M. Inaba: Vision based behavior verification system of humanoid robot for daily environment tasks, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2006) pp. 7–12

67.85 E. Yoshida, O. Kanoun, C. Esteves, J.-P. Laumond, K. Yokoi: Task-driven support polygon reshaping for humanoids, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2006) pp. 827–832

67.86 E. Yoshida, M. Poirier, J.-P. Laumond, O. Kanoun, F. Lamiraux, R. Alami, K. Yokoi: Pivoting based manipulation by a humanoid robot, Auton. Robots **28**(1), 77–88 (2010)

67.87 O. Kanoun, F. Lamiraux, P.-B. Wieber: Kinematic control of redundant manipulators: Generalizing the task priority framework, IEEE Trans.Robotics **27**(4), 785–792 (2011)

67.88 E. Neo, K. Yokoi, S. Kajita, K. Tanie: A framework for remote execution of whole body motions for humanoid robots, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2004) pp. 608–626

67.89 M. Toussaint, M. Gienger, C. Goerick: Optimization of sequential attractor-based movement for compact behaviour generation, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2007) pp. 122–129

67.90 N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, R. Dillmann: Humanoid motion planning for dual-arm manipulation and re-grasping tasks, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2009) pp. 2464–2470

67.91 F. Kanehiro, E. Yoshida, K. Yokoi: Efficient reaching motion planning and execution for exploration by humanoid robots, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 1911–1916

67.92 S. Kagami, F. Kanehiro: AutoBalancer: An online dynamic balance compensation scheme for humanoid robots, Workshop Algorithmic Found. Robotics (2000) pp. 79–89

67.93 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Resolved momentum control: Humanoid motion planning based on the linear and angular momentum, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2003) pp. 1644–1650

67.94 K. Yamane, Y. Nakamura: Dynamics filter – concept and implementa tion of online motion generator for human figures, IEEE Trans. Robotics Autom. **19**(3), 421–432 (2003)

67.95 Jin'ichi Yamaguchi, E. Soga, S. Inoue, A. Takanishi: Development of a bipedal humanoid robot – Control method of whole body cooperative dy-

namic biped walking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1999) pp. 368–374
67.96 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Biped walking pattern generation by using preview control of zero-moment point, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 1620–1626
67.97 T. Sugihara, Y. Nakamura: Whole-body cooperative balancing of humanoid robot using COG jacobian, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2002) pp. 2575–2580
67.98 H. Harada, S. Kajita, F. Kanehiro, K. Fujiwara, K. Kaneko, K. Yokoi, H. Hirukawa: Real-time planning of humanoid robot's gait for force controlled manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 616–622
67.99 H. Harada, S. Kajita, H. Saito, M. Morisawa, F. Kanehiro, K. Fujiwara, K. Kaneko, H. Hirukawa: A Humanoid robot carrying a heavy object, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 1712–1717
67.100 M. Stilman, K. Ishiwaki, S. Kagami: Learning object models for whole body manipulation, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2007) pp. 174–179
67.101 H. Sugiura, M. Gienger, H. Janssen, C. Goerick: Real-time collision avoidance with whole body motion control for humanoid robots, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2007) pp. 2053–2058
67.102 O. Kanoun, J.-P. Laumond, E. Yoshida: Planning foot placements for a humanoid robot: A Problem of Inverse Kinematics, Int. J. Robotics Res. **30**(4), 476–485 (2011)
67.103 O. Stasse, B. Verrelst, A. Davison, N. Mansard, F. Saidi, B. Vanderborght, C. Esteves, K. Yokoi: Integrating walking and vision to increase humanoid autonomy, Int. J. Humanoid Robotics **5**(2), 287–310 (2008)
67.104 K. Okada, T. Ogura, A. Haneda, J. Fujimoto, F. Gravot, M. Inaba: Humanoid motion generation system on HRP2-JSK for daily life environment, Proc. IEEE Int. Conf. Mechatron. Autom. (2005) pp. 1772–1777
67.105 F. Zacharias, C. Borst, G. Hirzinger: Capturing robot workspace structure: Representing robot capabilities, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2007) pp. 3229–3236
67.106 N. Vahrenkamp, E. Kuhn, T. Asfour, R. Dillmann: Planning multi-robot grasping motions, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2010) pp. 593–600
67.107 A. Escande, A. Kheddar: Contact planning for acyclic motion with tasks constraints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 435–440
67.108 K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, B. Wilcox: Motion planning for legged robots on varied terrain, Int. J. Robotics Res. **27**(11-12), 1325–1349 (2008)
67.109 L. Saab, N. Mansard, F. Keith, J.-Y. Fourquet, P. Soueres: Generation of dynamic motion for anthropomorphic systems under prioritized equality and inequality constraints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 1091–1096
67.110 C. Ott, C. Baumgärtner, J. Mayr, M. Fuchs, R. Burger, D. Lee, O. Eiberger, Alin Albu-Schäffer, M. Grebenstein, G. Hirzinger: Development of a biped robot with torque controlled joints, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2010) pp. 167–173
67.111 S. Lengagne, J. Vaillant, E. Yoshida, A. Kheddar: Generation of whole-body optimal dynamic multi-contact motions, Int. J. Robotics Res. **32**(9/10), 1104–1119 (2013)
67.112 A. Escande, A. Kheddar, S. Miossec: Planning support contact-points for humanoid robots and experiments on HRP-2, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2006) pp. 2974–2979
67.113 K. Bouyarmane, A. Kheddar: Humanoid robot locomotion and manipulation step planning, Adv. Robotics **26**(10), 1099–1126 (2012)
67.114 H. Arisumi, S. Miossec, J.-R. Chardonnet, K. Yokoi: Dynamic lifting by whole body motion of humanoid robots, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2008) pp. 668–675
67.115 K. Koyanagi, H. Hirukawa, S. Hattori, M. Morisawa, S. Nakaoka, K. Harada, S. Kajita: A pattern generator of humanoid robots walking on a rough terrain using a handrail, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2008) pp. 2617–2622
67.116 L. Sentis, J. Park, O. Khatib: Compliant control of multicontact and center-of-mass behaviors in humanoid robots, IEEE Trans. Robotics **26**(3), 483–501 (2010)
67.117 J. Park, O. Khatib: Contact consistent control framework for humanoid robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1963–1969
67.118 S. Hirose, H. Tsukagoshi, K. Yoneda: Normalized energy stability margin and its contour of walking vehicles on rough terrain, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2001) pp. 181–186
67.119 K. Harada, S. Kajita, K. Kaneko, H. Hirukawa: Dynamics and balance of a humanoid robot during manipulation task, IEEE Trans. Robotics **22**(3), 568–575 (2006)
67.120 H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, M. Morisawa: A universal stability criterion of the foot contact of legged robots – Adios ZMP, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1976–1983
67.121 K. Bouyarmane, J. Vaillant, F. Keith, A. Kheddar: Exploring humanoid robots locomotion capabilities in virtual disaster response scenarios, Proc. IEEE/RAS Int. Conf. Humanoid Robotics (2012) pp. 337–342
67.122 S.-H. Hyon, J.G. Hale, G. Cheng: Full-body compliant human-humanoid interaction: Balancing in the presence of unknown external forces, IEEE Trans. Robotics **23**(5), 884–898 (2007)
67.123 K. Bouyarmane, A. Kheddar: Using a multi-objective controller to synthesize simulated hu-

manoid robot motion with changing contact configurations, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2011) pp. 4414–4419

67.124 C. Breazeal, A. Edsinger, P. Fitzpatrick, B. Scassellati, P. Varchavskaia: Social constraints on animate vision, IEEE Intell. Syst. **15**, 32–37 (2000)

67.125 F. Berton, G. Sandini, G. Metta: Anthropomorphic visual sensors. In: *The Encyclopedia of Sensors*, Vol. X, ed. by M.V. Pishko, C.A. Grimes, E.C. Dickey (American Scientific, Stevenson Ranch 2006) pp. 1–16

67.126 T. Shibata, S. Vijayakumar, J. Conradt, S. Schaal: Biomimetic oculomotor control, Adapt. Behav. **9**, 189–208 (2001)

67.127 B. Scassellati: *A Binocular, Foveated Active Vision System, Vol. AIM-1628* (MIT, Cambridge 1998)

67.128 A. Ude, C. Gaskett, G. Cheng: Foveated vision systems with two cameras per eye, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Orlando (2006)

67.129 H. Kozima: Infanoid: A babybot that explores the social environment. In: *Socially Intelligent Agents: Creating Relationships with Computers and Robots*, ed. by K. Dautenhahn, A.H. Bond, L. Canamero, B. Edmonds (Kluwer, Amsterdam 2002) pp. 157–164

67.130 T. Shibata, S. Schaal: Biomimetic gaze stabilization based on feedback-error-learning with nonparametric regression networks, Neural Netw. **14**(2), 201–216 (2001)

67.131 P. Heracleous, S. Nakamura, K. Shikano: Simultaneous recognition of distant-talking speech to multiple talkers based on the 3-D N-best search method, J. VLSI Signal Process. Syst. Arch. **36**(2–3), 105–116 (2004)

67.132 J. Ido, Y. Matsumoto, T. Ogasawara, R. Nisimura: Humanoid with interaction ability using vision and speech information, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2006)

67.133 I. Hara, F. Asano, H. Asoh, J. Ogata, N. Ichimura, Y. Kawai, F. Kanehiro, H. Hirukawa, K. Yamamoto: Robust speech interface based on audio and video information fusion for humanoid HRP-2, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2004) pp. 2402–2410

67.134 M. Lungarella, G. Metta, R. Pfeifer, G. Sandini: Developmental robotics: A survey, Connect. Sci. **15**(4), 151–190 (2003)

67.135 B. Scassellati, C. Crick, K. Gold, E. Kim, F. Shic, G. Sun: Social development, Comput. Intell. Mag. (IEEE) **1**(3), 41–47 (2006)

67.136 M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, C. Yoshida: Cognitive developmental robotics: A survey, IEEE Trans. Auton. Mental Dev. **1**(1), 12–34 (2009)

Multimedia Contents

68. Human Motion Reconstruction

Katsu Yamane, Wataru Takano

This chapter presents a set of techniques for reconstructing and understanding human motions measured using current motion capture technologies. We first review modeling and computation techniques for obtaining motion and force information from human motion data (Sect. 68.2). Here we show that kinematics and dynamics algorithms for articulated rigid bodies can be applied to human motion data processing, with help from models based on knowledge in anatomy and physiology. We then describe methods for analyzing human motions so that robots can segment and categorize different behaviors and use them as the basis for human motion understanding and communication (Sect. 68.3). These methods are based on statistical techniques widely used in linguistics. The two fields share the common goal of converting continuous and noisy signal to discrete symbols, and therefore it is natural to apply similar techniques. Finally, we introduce some application examples of human motion and models ranging from simulated human control to humanoid robot motion synthesis.

68.1 **Overview** 1819
68.2 **Models and Computations** 1820
68.2.1 Motion Capture Technology 1820
68.2.2 Skeleton Model and Computation 1821
68.2.3 Musculoskeletal Model and Computation 1823
68.2.4 Model Identification 1824
68.3 **Reconstruction for Understanding** 1825
68.3.1 Segmentation of Human Motion 1825
68.3.2 Categorization of Human Motion 1825
68.3.3 Temporal Sequence of Human Motion 1826
68.3.4 Linguistic Interpretation of Human Motion 1827
68.3.5 Theory of Communication 1828
68.4 **Reconstruction for Robots** 1829
68.4.1 Reconstruction for Virtual Humans 1829
68.4.2 Controller Optimization from Physiological Model 1829
68.4.3 Reconstruction for Physical Robots 1829
Video-References 1830
References 1831

68.1 Overview

Human motions are valuable resources for robots as they implicitly represent human strategies for control and planning. They are particularly useful when the control objective is difficult to formulate as a cost function, such as conveying expressions or moving in a particular style. Many researchers have used human motions for teaching new tasks to robots [68.1] and generating robot motions with human-like styles [68.2].

In order to use human motions for robotics applications, we first have to reconstruct the motion from raw measurement from a motion capture system. For this purpose, we can apply many techniques and algorithms developed in robotics such as kinematics, dynamics, parameter identification, and statistical models. In some applications such as robot learning, reconstructed motion data represented as sequences of joint angles are often not useful. In this case, the motion must be interpreted as higher level representation such as a sequence of behaviors.

This chapter reviews some of the algorithms for reconstructing and interpreting human motions for robotics applications. The next section describes the

kinematic and dynamic computations for human motion reconstruction. As these algorithms were originally developed for robot manipulators, we need substantial adaptation to human body due to its complexity. In Sect. 68.3, we present an overview of statistical approaches for understanding human motions as discrete behaviors, and use them for communication between humans and robots. The technique presented here has a close relationship to natural language processing that deals with the similar problem of converting continuous signal (sound) to discrete symbols (words). Finally, we review a few examples of using human motions to generate human-like motions on virtual humans as well as humanoid robots.

68.2 Models and Computations

Motion capture refers to measuring motions in the Cartesian space. The subjects are usually humans but animals or robots may also be used. Many motion capture technologies are commercially available with varying costs and capabilities. We will give a brief overview of the most popular motion capture systems: optical and mechanical.

68.2.1 Motion Capture Technology

Optical Motion Capture

Optical motion capture systems are popular thanks to the minimum disruption to motions, although high-end systems can be expensive because of high-resolution, high-speed cameras and elaborate software. Furthermore, offline data cleanup is often necessary to handle occasional occlusion and mislabeling. These systems are widely used in many fields including biomechanics, entertainment, and robotics.

Figure 68.1 shows the principle of optical motion capture. Multiple markers (white circles) are attached to the subjects' body (gray) as *landmarks* to be detected by the cameras surrounding the capture area. The three-dimensional (3-D) marker positions are obtained by computing the intersection of lines connecting the cameras' optical center and the markers on the image plane.

Human whole-body motion capture typically requires more than 40 markers. The markers are usually spherical and retro-reflective, which means that they reflect the incoming light directly back to the light source. The subject experiences very little disruption during motions because the markers are light and small, although they can be a problem in motions that involve physical contacts with other subjects or the environment.

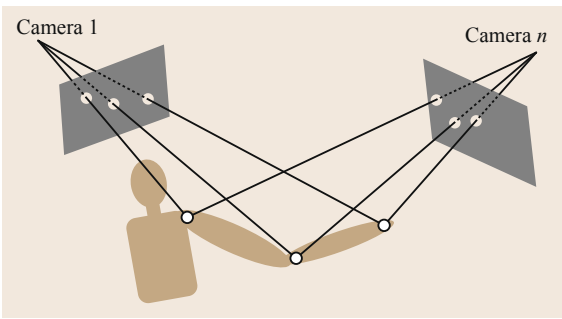


Fig. 68.1 Principle of optical motion capture

The capture area is surrounded by three or more cameras with near-ultrared light source. Because the markers are retro-reflective, they can be easily detected in the camera images as bright disks by simple image processing with little effect from the lighting condition. The internal and external camera parameters are identified through a system-specific calibration process before each motion capture session.

Human motions are reconstructed in three steps. First, the markers are detected from the camera images. Then the Cartesian positions of the markers can be computed by using the marker positions from multiple cameras. Finally, joint angle data are computed by performing inverse kinematics computation (VIDEO 762). This step will be described in Sect. 68.2.2.

The complexity of the software is due to the three-dimensional (3-D) reconstruction process where it is difficult to identify the correspondence between the markers in multiple images because all markers look identical. Furthermore, some of the markers may be lost due to occlusion and fast motions. In practice, careful marker and camera placement is required to obtain consistent measurements.

Mechanical Motion Capture

Another family of motion capture technology is mechanical, where a mechanical device is attached to the subject's body to directly measure the joint angles. Earlier systems used mechanical linkages consisting of rigid links and joint angle sensors, which were bulky and therefore significantly restricted the subjects' mobility. However, more recent systems use flexible suites with embedded bend sensors that do not cause as much restriction. The suites are equipped with additional inertial sensors to measure the global position.

An advantage of this technology is that it can directly measure the joint angles, and therefore is more robust than optical systems. It also allows unbounded capture area, possibly in outdoor environments. On the other hand, the inertial sensors used for measuring the global motion of the body may suffer from drifts that result in inaccurate global position measurements. Drifts are usually corrected assuming that at least one of the foot segments is fixed to the environment, but this is not the case in motions involving flights.

Future of Motion Capture

Development of markerless, realtime, and low-cost motion capture systems is an active research area in both academia and industry. In fact, several commercial products with some of those features are becoming available.

For example, the OpenStage system [68.3] from Organic Motion realizes realtime markerless motion capture. Kinect [68.4] has been developed for gaming interface but it can potentially be used as a low-cost, realtime motion capture system. Some researchers are working on even more challenging motion capture techniques, such as markerless motion capture using monocular vision [68.5].

68.2.2 Skeleton Model and Computation

Once motion capture data are obtained, we can analyze the data by computing the joint angles and torques, provided that kinematics and dynamics models of the subject are available. While a number of algorithms have been developed for motion analysis and simulation of robots, applying these algorithms to the human body is not straightforward.

The human body is much more complex than most mechanical systems. It consists of a number of elements such as bones, muscles, organs, and skin, most of which are deformable and difficult to model themselves. Many joints are also difficult to model and simulate because of the complex constraints. For example, the center of rotation of the shoulder joint moves along with the joint movement. The knee and elbow joints are constrained by rolling contacts and ligaments between bones.

In practice, therefore, it is unrealistic to model all the details due to computational complexity, and the human body is typically approximated by articulated rigid body models familiar to the robotics community. We can then apply the kinematics and dynamics algorithms for mechanical systems as described in this subsection. Figure 68.2 shows an example of detailed human skeleton model [68.6], which consists of 155 degrees of freedom (DOF) excluding the fingers.

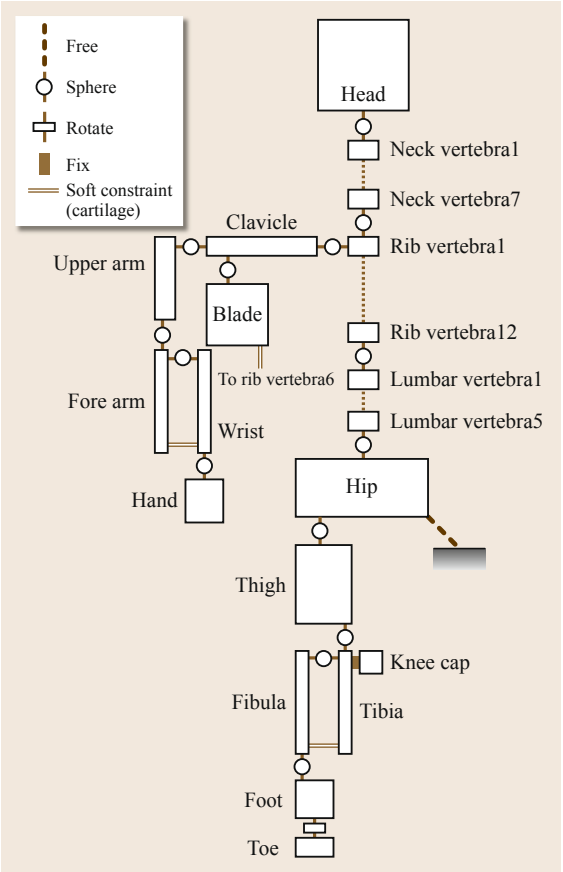


Fig. 68.2 Detailed human skeleton model

Even if we use simplified skeleton models, there is still an issue of obtaining accurate model parameters. As mentioned above, many joints do not have a fixed axis as in mechanical joints, which means that the kinematic parameters such as link length may not be constant as in mechanical systems. It is also difficult to estimate the inertial parameters because we cannot directly measure the individual link parameters, and because the human body is flexible and some of the soft tissue would move between links. We will discuss the parameter identification issue in Sect. 68.2.4.

Kinematics

Inverse and forward kinematics computations are used to analyze the kinematic quantities of measured motions such as positions, velocities, and accelerations of the links.

Inverse kinematics computation is critical to data obtained by optical motion capture because we have to convert the marker positions to skeleton joint angles. Typical inverse kinematics problem in robotics (Chap. 2) is to compute the joint angles given the

end-effector position and/or orientation. Inverse kinematics for motion capture data processing needs a slightly different problem setting because many markers are attached throughout the body, often resulting in an over-constrained system. Furthermore, measured marker positions would not perfectly fit an articulated rigid body model because human kinematics parameters may change during the motion, as discussed above.

A simple method for computing the joint angles, often used as the default method in commercial optical motion capture systems, is to first compute the joint centers by interpolating the positions of a few strategically placed markers, and then computing the joint angles from the joint centers. The advantage of this method is that it is easily customizable by, for example, simple scripts and therefore suitable for commercial systems.

Another method is to apply numerical optimization. For example, the cost function can be the sum of squared distances between the marker positions from the measurement and those computed by the kinematics model. Another approach is to *drive* the model toward the measured marker positions [68.7]. In either case, the resulting joint angle will evenly distribute the error caused by measurement error and model inaccuracy to all markers.

Figure 68.3 shows an example of inverse kinematics using a skeleton model.

Forward kinematics is performed to obtain the Cartesian positions, velocities, and accelerations of the links from the corresponding joint variables. These quantities are often used for the dynamics computation later.

Dynamics

Similarly to kinematics computation, we can consider both inverse and forward dynamics.

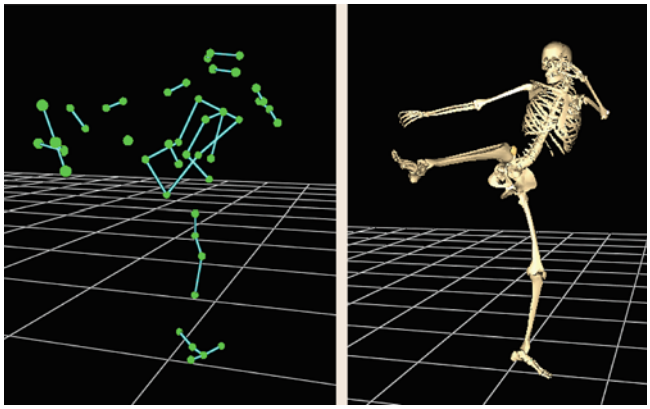


Fig. 68.3 Inverse kinematics computation example

Inverse dynamics computes the joint torques and possibly contact forces required to perform the measured motion. Any algorithm for inverse dynamics computation of articulated rigid bodies, such as the Newton–Euler formulation [68.8], can be applied. Chapter 3 gives the details on dynamics algorithms for general kinematic chains.

A major difference from manipulator inverse dynamics is that the human body is a floating-base system where the root joint can move freely in the six-dimensional (6-D) space but is not actuated. Generic inverse dynamics algorithms for manipulators give the 6-D force and torque for the root joint but they are not actually available. The equivalent force must be supplied instead by the contact forces, which have to be explicitly considered because they affect the joint torques.

The equation of motion of an n-DOF articulated rigid body system is written as

$$\boldsymbol{\tau}_{\mathrm{G}} = \mathbf{M}(\boldsymbol{\theta})\,\ddot{\boldsymbol{\theta}} + \boldsymbol{c}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right) + \boldsymbol{g}(\boldsymbol{\theta})\,, \tag{68.1}$$

where $\boldsymbol{\tau}_{\mathrm{G}} \in \mathbb{R}^n$ is the generalized force, $\boldsymbol{\theta} \in \mathbb{R}^n$ is the generalized coordinate, $\mathbf{M} \in \mathbb{R}^{n\times n}$ is the joint-space inertia matrix, $\boldsymbol{c} \in \mathbb{R}^n$ represents the centrifugal and Coriolis forces, and $\boldsymbol{g} \in \mathbb{R}^n$ is the gravitational force. In human body models subject to m contact constraints, $\boldsymbol{\tau}_{\mathrm{G}}$ is computed from active joint torques $\boldsymbol{\tau} \in \mathbb{R}^{n-6}$ and contact forces $\boldsymbol{f}_{\mathrm{c}} \in \mathbb{R}^m$ as

$$\boldsymbol{\tau}_{\mathrm{G}} = \mathbf{S}^{\mathrm{T}}\boldsymbol{\tau} + \mathbf{J}_{\mathrm{c}}^{\mathrm{T}}\boldsymbol{f}_{\mathrm{c}}\,, \tag{68.2}$$

where $\mathbf{J}_{\mathrm{c}} \in \mathbb{R}^{m\times n}$ is the Jacobian matrix of the contact constraints and $\mathbf{S}^{\mathrm{T}} \in \mathbb{R}^{n\times(n-6)}$ is the matrix that converts the active joint torques to generalized forces. Without losing generality, we assume that the root joint corresponds to the first six elements of the generalized coordinate. Then $\mathbf{S}^{\mathrm{T}}$ takes the form

$$\mathbf{S}^{\mathrm{T}} = \begin{pmatrix} \mathbf{O}_{6\times(n-6)} \\ \mathbf{I}_{(n-6)\times(n-6)} \end{pmatrix} \tag{68.3}$$

where $\mathbf{O}_*$ and $\mathbf{I}_*$ are the zero and identity matrices of the sizes given by the subscripts.

The number of constraints m changes depending on the contact condition. If, for example, both feet are in flat contact with the ground, m would be 12 because the position and orientation of each foot are constrained.

Equations (68.2) and (68.3) indicate that only $\boldsymbol{f}_{\mathrm{c}}$ contributes to the first six elements of $\boldsymbol{\tau}_{\mathrm{G}}$. This property corresponds to the well-known fact that the whole-body momentum can be changed only by applying external forces such as contact forces. We can utilize this fact to compute $\boldsymbol{f}_{\mathrm{c}}$ independently of the joint torques [68.6].

We can then compute the joint torques using the bottom $(n-6)$ equations of (68.2).

Forward dynamics is used for dynamics simulation. The simulation in this context usually involves a set of controllers that attempts to make the human model track the measured motion. This computation is often performed to simulate the effects of external disturbances and physical disabilities. However, developing controller for human body simulation is very difficult because the root joint is not actuated and the contact forces are subject to unilateral constraints. Although this topic is extensively discussed in humanoid robotics, graphics, and biomechanics, there is no consensus on how to build general controllers for human models.

68.2.3 Musculoskeletal Model and Computation

Motion analysis based on skeleton models is not sufficient for applications that require anatomical information such as muscle lengths and tensions. In biomechanics, for example, researchers are often interested in computing the muscle tensions required to perform a motion. Also in robotics, many muscle-like actuators are developed and used to design more human-like robotic systems.

Musculoskeletal models are used for motion analysis involving muscles. In addition to a skeleton model discussed in the previous section, a musculoskeletal model includes a musculotendon network model that describes where the muscles are attached to and how they are routed around the bones. Similarly to skeleton models, musculotendon network models include a number of simplifications due to the complexity of the real human body. A muscle is usually modeled as a line segment that produces a uniform tensile force. Interactions of muscles with bones and skin are not usually modeled except for some of the major muscle–bone interactions.

Apart from these simplifications, the force interaction between the muscles and bones are described as accurately as possible. A simple muscle–tendon pair connects two bones and produces a torque at the joint between them (Fig. 68.4a). More complex muscles may pass through one or more points fixed on the bone surface (Fig. 68.4b), or may have branches and therefore connects three or more bones. Wide muscles, such as pennate muscles, are modeled by several parallel muscles.

Figure 68.5 shows a detailed musculoskeletal model consisting of 955 muscles [68.6].

Kinematics computation on a musculoskeletal model gives the muscle lengths. The length and its velocity may then be used for various analysis. For example, we can use Hill–Stroeve muscle model [68.9, 10] to compute the maximum muscle tension.

The active joint torques $\boldsymbol{\tau}$ are now produced by muscle tensions $\boldsymbol{f}_{\mathrm{m}}$. Applying d'Alembert's principle, we can derive the relationship between $\boldsymbol{\tau}$ and $\boldsymbol{f}_{\mathrm{m}}$ as

$$\boldsymbol{\tau} = \mathbf{J}_{\mathrm{m}}^{\mathrm{T}} \boldsymbol{f}_{\mathrm{m}} \,, \tag{68.4}$$

where $\mathbf{J}_{\mathrm{m}}$ is the Jacobian matrix of the muscle lengths with respect to the generalized coordinates.

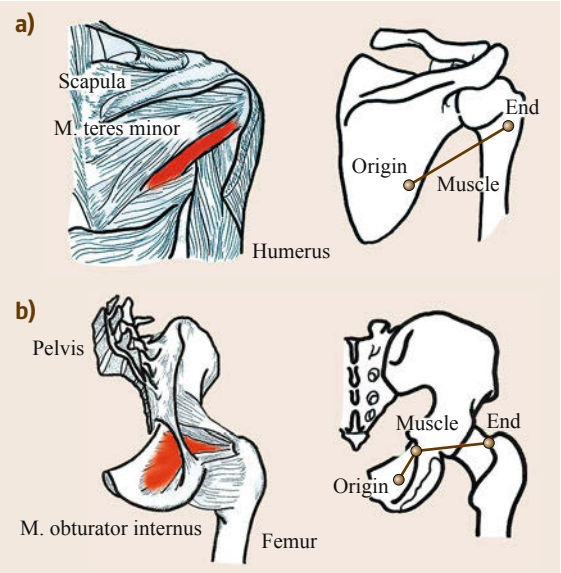


Fig.68.4a,b Examples of muscle models; (a) simple wire connecting two bones, (b) muscle with via point

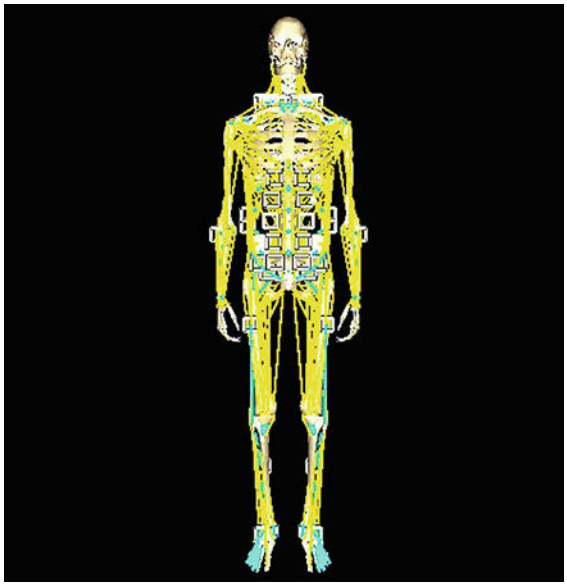


Fig. 68.5 A detailed musculoskeletal model

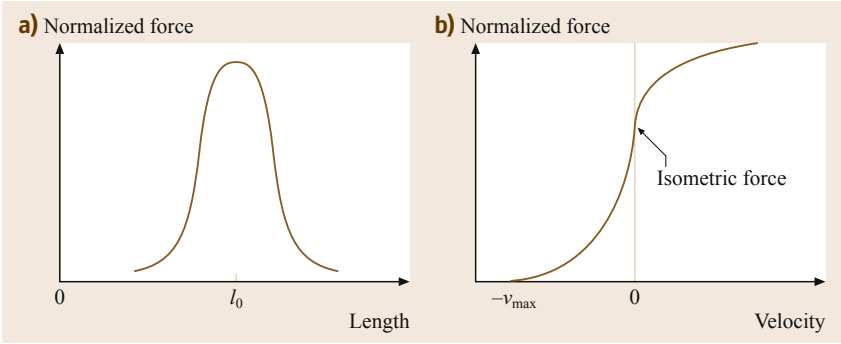


Fig.68.6a,b Relationship between maximum muscle tension and muscle fiber length. **(a)** Length–tension relationship, **(b)** velocity–tension relationship

The inverse dynamics computation for musculoskeletal models is summarized as follows:

1. Compute the joint angles, velocities, and accelerations of the motion capture data.
2. Compute the joint torques $\boldsymbol{\tau}$ of the skeleton model.
3. Map the joint torques to muscle tensions using (68.4) (VIDEO 763).

Because most musculoskeletal models contain more muscles than the skeleton DOF, the final step does not have a unique solution. The muscle tensions also have to be constrained to be tensile. A common approach for this problem is to apply numerical optimization techniques [68.6, 11]. However, actual muscle tensions may not be optimal because of the phenomena called co-contraction, where antagonistic muscles are activated at the same time. Such effect can be incorporated by measuring the muscle activity through electromyography (EMG) and consider the data in optimization [68.12].

Another method for computing muscle tensions is to utilize physiological muscle models. The most commonly used model was originally proposed by *Hill* [68.9] and later formulated by *Stroeve* [68.10].

Muscle tension f is characterized by three components: muscle activity a, muscle length l, and change of muscle length $\dot{l}$ as follows:

$$f = a\, F_{\mathrm{l}}(l) F_{\mathrm{v}}(\dot{l}) F_{\max}\,, \tag{68.5}$$

where $F_{\max}$ is the muscle-specific maximum voluntary force, $F_{\mathrm{l}}(*)$ and $F_{\mathrm{v}}(*)$ are the functions that represent length–tension and velocity–tension relationship respectively (Fig. 68.6). The muscle activity a changes according to the following differential equation,

$$\dot{a} = \frac{u-a}{T}\,, \tag{68.6}$$

where T is a time constant and u is the normalized input from the motor neuron which can be measured by EMG.

68.2.4 Model Identification

Obtaining accurate human model parameters is difficult but very important for accurate motion and force estimation. For robots, we can build a model based on CAD (computer-aided design) models or even measurements of the individual link mass, but both are impossible for humans. On the other hand, personalized models are important to make the analysis results useful for individuals.

Identification of Kinematic Parameters

In human motion analysis, the set of joints constructing a human model is usually given. Kinematic parameters include distance between joints and direction of the joint axes.

We can apply some of the kinematic parameter identification techniques for robots [68.13] using motion capture to provide the data necessary for identification. Another approach is to use a human dimension database [68.14] and use a few measurements from the subject to estimate the rest.

Identification of Inertial Parameters

Conventional methods for manipulator inertia parameter identification (Chap. 6) use joint torque data, and therefore cannot be applied to the human body.

However, *Ayusawa* et al. [68.15] proved that contact force data give enough information to estimate the same set of parameters for floating-base models. As shown in Chap. 6, the generalized force $\boldsymbol{\tau}_{\mathrm{G}i}$ required to perform a given motion described by the generalized coordinates $\boldsymbol{\theta}_i$, velocities $\dot{\boldsymbol{\theta}}_i$, and accelerations $\ddot{\boldsymbol{\theta}}_i$ is written as a linear equation of the identifiable inertial parameters $\boldsymbol{p}$ as

$$\mathbf{I}_i\left(\boldsymbol{\theta}_i, \dot{\boldsymbol{\theta}}_i, \ddot{\boldsymbol{\theta}}_i\right)\boldsymbol{p} = \boldsymbol{\tau}_{\mathrm{G}i}\,. \tag{68.7}$$

We can obtain a number of these samples from motion capture data. In manipulators with joint torque sensors, all elements of $\mathbf{I}$ and $\boldsymbol{\tau}_{\mathrm{G}}$ are available and therefore it is trivial to solve (68.7).

As shown in (68.2), the generalized forces for human body models are produced by active joint torques and contact forces. From (68.3), (68.7) can be divided into the first six rows and the remaining $(n-6)$ rows as

$$\mathbf{I}_{fi}(\boldsymbol{\theta}_i, \dot{\boldsymbol{\theta}}_i, \ddot{\boldsymbol{\theta}}_i)\boldsymbol{p} = \mathbf{J}_{cfi}^{\mathrm{T}} \boldsymbol{f}_{ci} \,, \tag{68.8}$$

$$\mathbf{I}_{ai}(\boldsymbol{\theta}_i, \dot{\boldsymbol{\theta}}_i, \ddot{\boldsymbol{\theta}}_i)\boldsymbol{p} = \boldsymbol{\tau}_i + \mathbf{J}_{cai}^{\mathrm{T}} \boldsymbol{f}_{ci} \,. \tag{68.9}$$

For human body, joint torque data $\boldsymbol{\tau}_i$ are not available. However, we can obtain the contact force data from force sensors embedded in the floor. In this case, we have all the information in (68.8). *Ayusawa* et al. [68.15] proved that this equation is enough for identifying the same set of parameters as using the full set of (68.7).

68.3 Reconstruction for Understanding

Human motion data can be used for motion and task learning for humanoid robots, as represented by imitation learning or learning by demonstration [68.1].

68.3.1 Segmentation of Human Motion

Human motion data are converted to a sequence of configurations, such as joint angles or link positions over the whole of the robot's body, and the sequence is encoded into a set of model parameters as the motion primitive. This encoding should follow the motion segmentation; a chunk of motion should be obtained by finding the frames where the motion starts and terminates in the seamless motion data. Manually selected motion segments are intuitive, reliable, and easily assigned their relevant motion labels. However, manual segmentation is limited by its lack of scalability for the large numbers of human motions required for training. Human motion should be segmented so that a humanoid robot can observe human actions and incrementally memorize the motion primitives by segmenting and encoding unknown motions.

In the supervised segmentation method, the relation between each motion frame and its boundary score from the training dataset is learned, and the boundary in the motion observation is identified. For the training dataset, human subjects have to detect the motion boundaries. Supervised segmentation has the same drawback in scalability to large motion datasets as manual segmentation.

There are studies of unsupervised segmentation based on various methods: stopped moment, motion clustering, and motion prediction. Motion segmentation based on the stopped moments is very simple. This segmentation assumes that a motion pattern transits to another motion via the stopped moment. The simplest strategy only has to find a pose when either joint stops as a boundary candidate. The extensive approach approximates a temporal sequence of the boundary candidates by a periodic function as a reference for the motion boundaries. These methods may work for rhythmic movements, such as dance, but it is not clear whether they would be useful for segmenting daily human actions [68.16, 17].

Stochastic graphical models, typified by hidden Markov models (HMM), are used to model the stream of motion data [68.18]. An optimal node in the graphical model that is the most likely to generate the current motion observation can be calculated. The observations classified into the same node consequently are considered to be the same motion. In other words, the transition between two nodes corresponds to the boundary of the motions [68.19].

In natural language processing, word segmentation is critical, especially for speech recognition, in which words must be selected from a stream of phonemes. Predictability strategy hypothesizes that it is difficult to predict the character at the beginning of the word. A recurrent network based on this strategy has been applied to the segmentation of words [68.20]. The network learns the sequence of characters, and consequently predicts the character from its preceding one. This predictor detects a boundary where the error between the predicted and actual character becomes large, and subsequently selects words. Similar to this approach, during motion learning motion is converted to a sequence of short movements, each of which is represented by a binary vector equivalent to a word, and this sequence is trained as the correlation matrix. This matrix can be easily used to predict the short movement by multiplying this matrix by the vector of the preceding movement [68.21]. Movements with a large prediction uncertainty can be detected as the motion boundary.

68.3.2 Categorization of Human Motion

Human motion data is encoded into a set of parameters in the dynamical systems [68.22, 23] or stochastic models [68.24, 25]. The dynamical system representing state transitions by nonlinear differential equations is a popular way to encode human motions and subsequently synthesize robot motion adapted to various environments. The dynamics in the human motions is

defined as a vector field, and each motion pattern is expressed by a closed curve in this field, which is constructed so as to attract robot's motion to the reference motion [68.26]. This attractor field will stabilize the robot's motion even when the robot is perturbed by the external environment. A popular approach based on the dynamical system is a framework of the dynamic movement primitives (DMPs) [68.27]. This framework learns a single trajectory by using a spring-damping model including a forcing term which makes it easy to control a robot motion reaching a goal configuration. This framework has been extended to learn multiple human motions such that the robot can perform various motions and demonstrate the highly scalable nature [68.28]. These dynamical systems mainly focus on synthesize and control robot motions adaptively to the environment. While they can predict motions and classify the observation into its specific motion category, but they are not directly used for motion classification. Another popular approach based on the dynamical system is a recurrent neural network (RNN). The RNN generates the human posture following the input current one, and is optimized such that error between the generated posture and measured one can be minimized. The resultant RNN can be useful for the motion predictor. The RNN has been extended to a modified structure with a layer of the small number of parametric biases, into which the dynamics in the human motion can be encoded [68.29]. This structure has an advantage of not only generating motions from perception, but also recognize the perception as a category defined by these biases. Due to the complexity of the model structure, it is difficult to tune the parameters from a large number of training data.

Stochastic framework, typified by HMMs, are widely used for categorization of human motions. A compact notation $\lambda = \{\boldsymbol{\Pi}, \mathbf{A}, \boldsymbol{B}\}$ represents an HMM. $\boldsymbol{\Pi}$ is a set of initial node probabilities $\boldsymbol{\Pi} = \{\pi_i\}$, $\mathbf{A}$ is a transition matrix $\mathbf{A} = \{a_{ij}\}$ whose entries a_{ij} is the probability of transitioning from the i-th node to the j-th node, and $\boldsymbol{B}$ is a set of output distributions $\boldsymbol{B} = \{\boldsymbol{b}_i(\boldsymbol{\theta})\}$. Figure 68.7 shows the overview of the HMM. Human motion is expressed by a sequence of joint angles or link positions, $\boldsymbol{x} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_T\}$. The parameters of the HMM are optimized by Baum–Welch algorithm, which is one of expectation maximization algorithms, such that the likelihood $P(\boldsymbol{x}|\lambda)$ of the human motion $\boldsymbol{x}$ being generated by its relevant HMM λ is maximized [68.30]. The HMM represents spatiotemporal features of the human motions, and is hereafter referred to as motion primitive.

Human motion $\boldsymbol{x}$ can be classified into a motion category represented by the motion primitive. The classification finds an HMM $\lambda_{\mathcal{R}}$ which is the most likely to

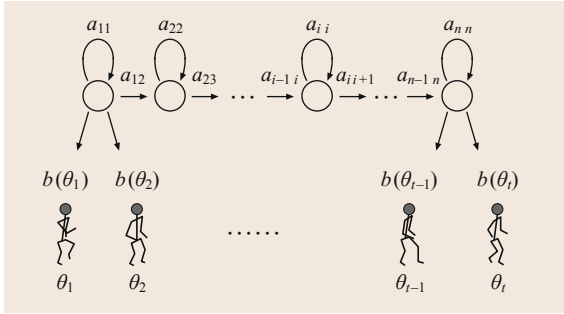


Fig. 68.7 The overview of an HMM. An HMM is the graphical model. A node extracts the spatio feature and an edge extract the temporal feature

generate the human motion as follows

$$\lambda_{\mathcal{R}} = \arg\max_{\lambda} P(\boldsymbol{x}|\lambda) \,. \tag{68.10}$$

The likelihood $P(\boldsymbol{x}|\lambda)$ can be efficiently computed in the recursive manner by the forward algorithm

$$\alpha_t(j) = \sum_i \alpha_{t-1}(i) a_{ij} b_j(\boldsymbol{\theta}_t) \,, \tag{68.11}$$

$$P(\boldsymbol{x}|\lambda) = \sum_i \alpha_T(i) \,, \tag{68.12}$$

where $\alpha_t(j)$ is the probability of generating a sequence $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_t\}$ and staying at the j-th node at the time t.

68.3.3 Temporal Sequence of Human Motion

The representations of not only motion segments but also the temporal sequence of them are important for the robots to accurately understand human actions. *Motion graph* is the popular approach to represent the transition between human postures [68.31]. It is possible to transfer between two postures that are similar to one another. This framework creates the edges among the human postures based on their spatial similarities.

The motion segmentation and motion categorization can be extended to the representation of temporal sequence of human motions. Human motion is expressed by a series of motion segments $\{\ldots, \boldsymbol{x}(k-1), \boldsymbol{x}(k), \boldsymbol{x}(k+1), \ldots\}$ by the segmentation technique, and the series of motion segments is converted to a temporal series of motion primitives $\{\ldots, \lambda(k-1), \lambda(k), \lambda(k+1), \ldots, \}$ by the motion categorization technique. The transition structure of the motion primitives can be extracted by deriving a motion N-gram model from the series of motion primitives. The

motion N-gram model assumes that the motion primitive depends only on its $N-1$ pieces of the motion primitives. The simplest structure is the motion bigram (2-gram) model. The probability of transitioning from the motion λ_i to the motion λ_j can be estimated as

$$P(\lambda_i|\lambda_j) = \frac{\sum_k C[\lambda(k-1)=\lambda_j, \lambda(k)=\lambda_i]}{\sum_k C[\lambda(k)=\lambda_j]} , \tag{68.13}$$

where $C(\lambda)$ is the frequency of the motion λ appearing in the observation of human motions, and this transition probability maximizes the probability of the observation being generated by the motion bigram model.

This transition structure can be used for several applications such as editing animation character, detecting abnormal actions, or predicting human actions [68.32]. In the case of the action prediction, current human motion is classified into the specific motion $\lambda(0)$. The sequence of the motions $\boldsymbol{\lambda} = \{\lambda(1), \lambda(2), \lambda(K)\}$, which is the most likely to be generated by the motion bigram model, can be searched for by Dijkstra's algorithm

$$\boldsymbol{\lambda}_{\mathcal{P}} = \arg\max_{\boldsymbol{\lambda}} P[\lambda(K)|\lambda(K-1)] \times P[\lambda(K-1)|\lambda(K-2)] \cdots P[\lambda(1)|\lambda(0)] . \tag{68.14}$$

The selected sequence of motions is considered to be motions predicted from the current motion (VIDEO 764).

68.3.4 Linguistic Interpretation of Human Motion

Integration of the human motions with the natural language leads to not only understanding the human motions in the form of linguistic expressions but also generating robot's motions from the queries of language commands. A framework of RNN with parametric bias has been applied to the integration of the motions and language. One RNN represents the dynamics in the motions, and the other RNN represents the dynamics of language, more specifically sequences of words. The parametric biases shared by these two RNNs join together RNNs [68.33]. Given sentences, the parametric biases for the sentences are estimated in the RNN for the language, the RNN for the motion generates motions using the parametric biases derived from the language RNN. The robot can generate the motions corresponding to the sentences. This conversion from the language to the motions is reversed, and it allows for the conversion from motions to their descriptive sentences [68.34].

Another integration of the human motions with language is stochastical approach. Human action is represented as a sequence of motion primitives $\{\lambda(1), \lambda(2), \ldots, \lambda(k)\}$. A sequence of verbs $\omega(1)$, $\omega(2), \ldots, \omega(l)$ is also manually assigned to the same human action, where $\omega(i)$ is the verb assigned to the manually segmented motion at the i-th position. The problem of extracting the references between the human motions and the verbs can be considered as a translation problem since the human motion is expressed by a sequence of symbols (Fig. 68.8). The IBM translation model has been widely used in the natural language processing [68.35], and can be applied to integrating the human motions with their relevant verbs [68.36]. This translation model consists of translation probability $t(\lambda|\omega)$ of the motion λ being generated by the verb ω, and alignment probability $a(i|j,l,k)$ that the i-th position in a sequence of the verbs can align to the j-th position in a sequence of the motions. These parameters are optimized such that the following probability that the sequence of verbs generates the sequence of motions is maximized by the expectation-maximization (EM) algorithm.

$$\Phi = P(\lambda(1), \lambda(2), \ldots, \lambda(k)|\omega(1), \omega(2), \ldots, \omega(l)) . \tag{68.15}$$

Additionally, the verb bigram model extracts the transition between two verbs as the verb transition probability $P(\omega_i|\omega_j)$ from the sequences of verbs attached to the human motions. A sequence of verbs $\boldsymbol{\omega} = \{\omega(1), \ldots, \omega(l)\}$ which is the most likely to describe the human motion is

$$\boldsymbol{\omega}_{\mathcal{R}} = \arg\max_{\boldsymbol{\omega}} P(\boldsymbol{\omega}|\boldsymbol{\lambda}) = \arg\max_{\boldsymbol{\omega}} P(\boldsymbol{\lambda}|\boldsymbol{\omega}) P(\boldsymbol{\omega}) . \tag{68.16}$$

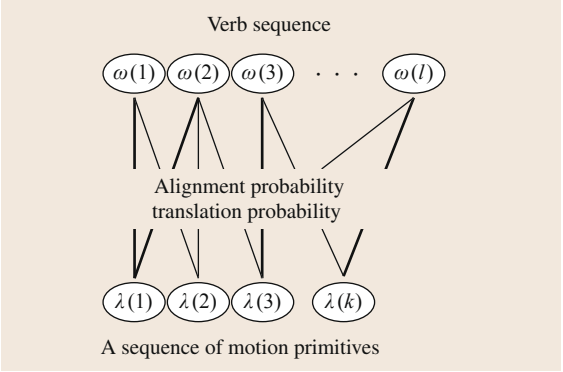


Fig. 68.8 Mapping between sequences of verbs and motion primitives

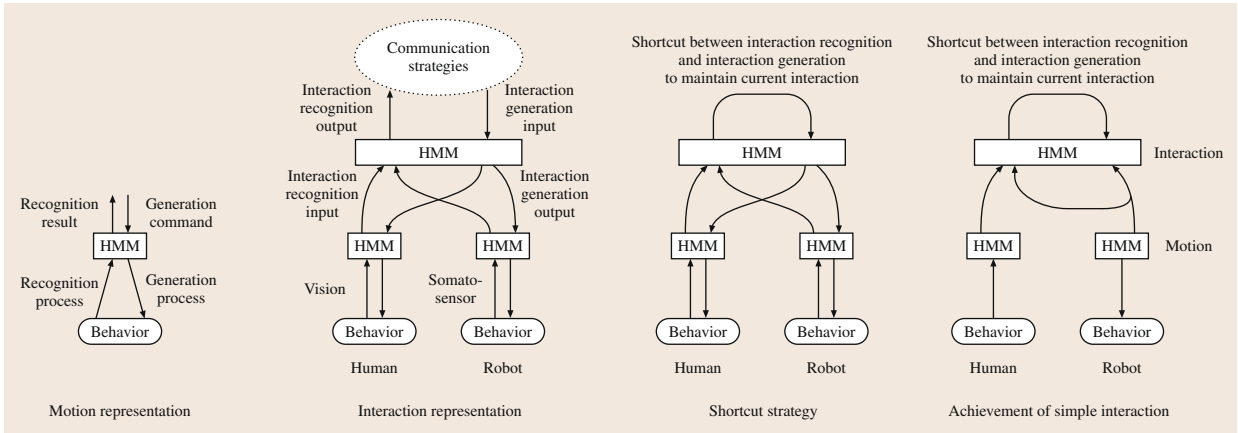


Fig. 68.9 Modeling of the behavioral interaction

$P(\boldsymbol{\lambda}|\boldsymbol{\omega})$ can be computed using the translation probability and alignment probability, and $P(\boldsymbol{\omega})$ can be computed using the verb transition probability. The human motion is interpreted as the resultant verb sequence $\boldsymbol{\omega}_{\mathcal{R}}$. This framework has been extended to handling not only verbs but also nouns, adverbs such that human motion can be converted to the descriptive sentences [68.37] (VIDEO 766).

68.3.5 Theory of Communication

HMMs has been extended to hierarchical HMMs (HHMM) or coupled HMMs (CHMM) to represent complicated spatiotemporal features. Statistical model is embedded into each node of the HMM in HHMM while commonly used HMM has an Gaussian distribution function in the node [68.38]. CHMM, in which multiple HMMs interact, is a method for representing mutual dependence between two such as a human and a robot, or a motion and a manipulated object. These approaches may be applied to the interaction that requires the complex representation [68.39]. However, the large number of parameters in these frameworks limits their application, and it is difficult to use these frameworks for human–robot interaction.

One efficient approach to using the representation of human motions as HMM is to design hierarchy for motion primitives and interaction primitives. HMMs representing human whole body motions can be useful to recognize motion observation. The HMMs can be reused to generate human-like motions for humanoid robot. A node is sampled according to the node transition probabilities, and a configuration is sampled according to the output probabilities on the selected node. The generation of the robot motion by the sampling manner can be done in the same framework. It is fundamental for the humanoid robot not only to recognize human motions but also to generate human-like motions, and this framework has been applied to the interaction between a human and a humanoid robot as shown in Fig. 68.9.

The behavioral interaction is defined by the exchange of the motion primitives between the two [68.37]. Observation of two interacting humans is converted to a sequence of pairs of motion primitives by recognizing their motions using the HMMs. HMMs in the higher level represent these interactions. These HMMs are more abstract analogy of representation of the motions, and useful for recognition and generation of the interaction patterns. A hypothesis is then introduced for the simple commutation, more specifically the communication is established by recognizing the relationship of the two and subsequently maintaining it.

The two recognition output of the human and robot using the HMMs in the lower level becomes the input

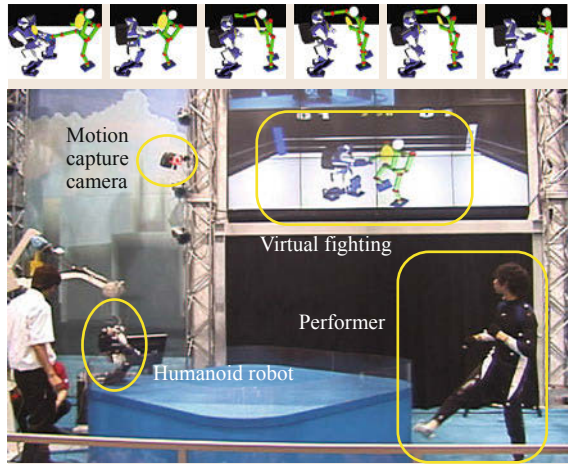


Fig. 68.10 Human–robot interaction realized by a symbolically hierarchical communication model

of the HMMs in the higher level, and the interaction can be recognized as a higher HMM. The current node can also be estimated in the selected higher HMM. The generation output from this node are a pair of two motion primitives for the human and robot, and they become the input of the lower HMMs. One motion primitive for the human becomes the prediction of human behavior, and another motion primitive for the robot becomes the command for the motion generation. The recognition and generation by the higher HMMs imply the estimated state of the interaction and the control strategy for the interaction, respectively. The importance for the naturally drifting interaction is to represent the process of the control strategy based on the estimated interaction state as a shortcut between the recognition and generation [68.40]. The experiment of interaction between the human and robot in a virtual world demonstrated its validity as shown in Fig. 68.10, and is further applied to the human–robot interaction with compliant physical contacts [68.41].

68.4 Reconstruction for Robots

This section reviews some examples of synthesizing human-like motions of robots or virtual humans based on human motions and/or physiological model.

68.4.1 Reconstruction for Virtual Humans

Instead of using inverse kinematics as described in Sect. 68.2.2, *Demerican* et al. [68.42] developed a method for reconstructing human motions by directly tracking marker trajectories with a physically simulated skeleton model. Figure 68.11 shows the set of markers tracked as well as the skeleton model used in the simulation. Examples of measured and simulated marker trajectories are shown in Fig. 68.12, which demonstrates the smoothness of the reconstructed human motions.

68.4.2 Controller Optimization from Physiological Model

Recorded human motion is an outcome of the human motor control. Instead of using the outcome, i. e., the motion, *Wang* et al. [68.44] optimized controllers based on physiological human model that includes eight major leg muscles, including both uniarticular (drive single joint) and biarticular (drive two joints) (Fig. 68.13). The method optimizes the controller parameters based on simulation considering the physiological muscle properties [68.9]. The results demonstrate good match with human measurements in terms of both joint trajectories and torques.

68.4.3 Reconstruction for Physical Robots

Tracking human motions with a physical humanoid robot is not as straightforward as it may appear even if the robot has human-like body topology, due to various differences between the human and robot bodies.

First, there is always some difference in the kinematic properties such as dimensions and proportion. In manipulation, for example, simply copying the joint angles will not result in the same hand positions if the upper and lower arm lengths are different.

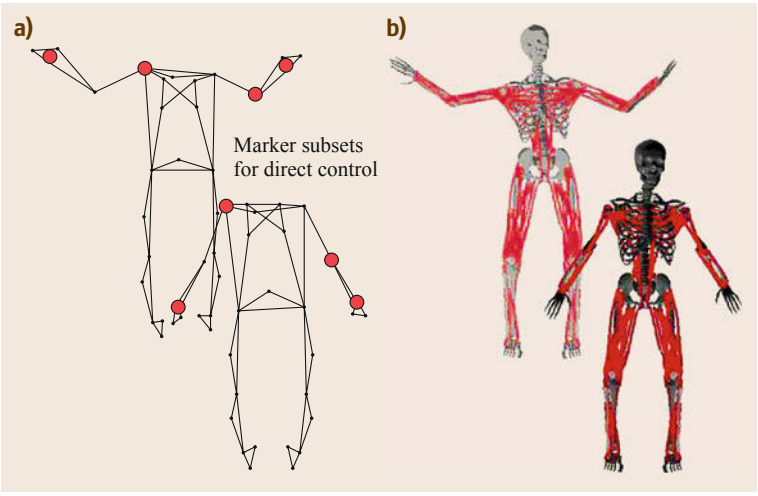


Fig.68.11a,b Marker set and skeletal model used in (after [68.42]). **(a)** Motion capture data. **(b)** Skeleton model in simulation and active interfaces (SAI) (after [68.43])

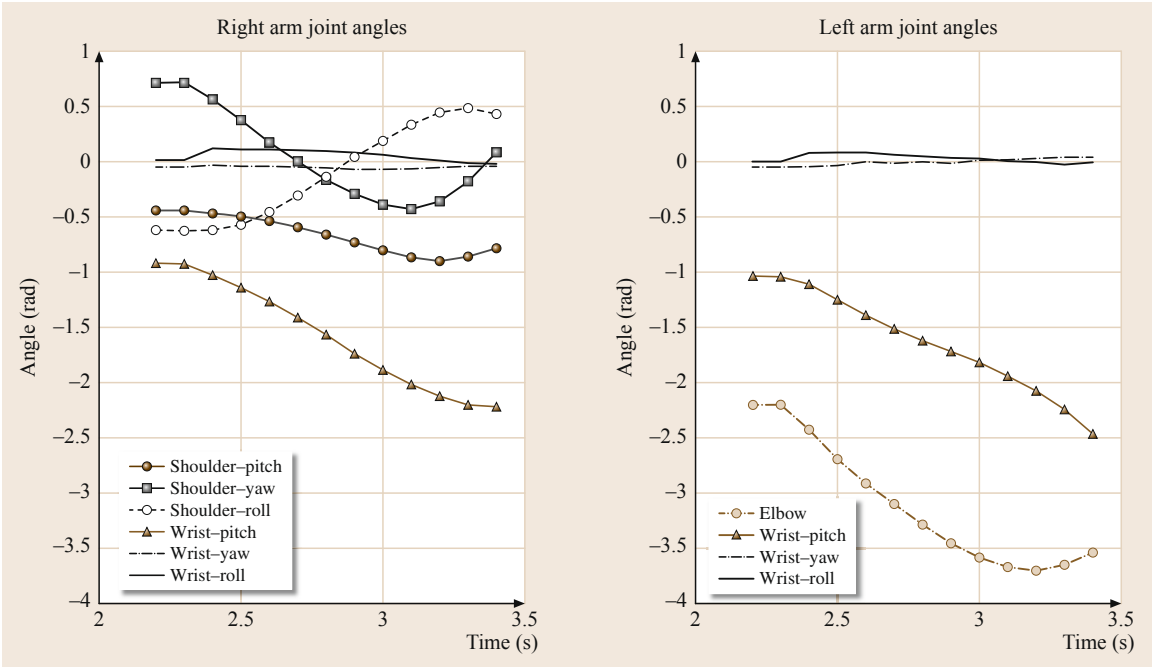


Fig.68.12a,b Joint angles computed from direct marker tracking (after [68.42]). **(a)** Right arm joint angles. **(b)** Left arm joint angles

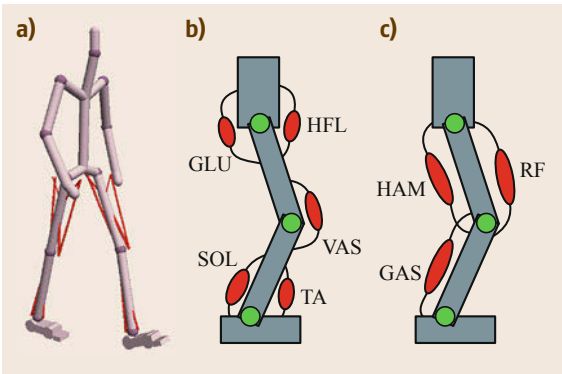


Fig.68.13a–c The musculoskeletal model used in [68.44] **(a)** skeleton and muscles (*red lines*), **(b)** five uniarticular muscles, and **(c)** three biarticular muscles

The kinematic difference comes not only from different dimensions, but also from flexibility of the human body that is difficult to replicate with mechanical systems. For example, the human spine has much more degrees of freedom than the torso of typical humanoid robots.

Secondly, robot hardware is generally much more restrictive in terms of joint motion range as well as velocity, acceleration, and torque limits. As a result, robots cannot always perform the same range of motion as humans. We would therefore have to modify the original motions such that they are within the hardware limits.

Finally, the inertial properties such as mass, inertia, and center of mass locations are different. In motions that involve balancing, the different inertial properties may make the motion unbalanced and cause a fall.

While motion capture data have been frequently applied to virtual human characters in simulated environments [68.45–47], much less work has been done on directly controlling humanoid robots with human motion capture data. *Nakaoka* et al. [68.2] divided captured dancing motion into multiple tasks and optimized each task to be feasible for the robot dynamics. *Miura* et al. [68.48] computed feasbile center of mass trajectory based on the center of pressure trajectory determined from the footsteps in the motion data, and then computed the joint motions by inverse kinematics.

Ott et al. [68.49] and *Yamane* and *Hodgins* [68.50, 51], on the other hand, combined online balance control and tracking control to address the problem of different dynamics. Figure 68.14 depicts some snapshots from the hardware experiment of tracking a human motion capture sequence with a floating-base humanoid robot [68.51] (VIDEO 765).

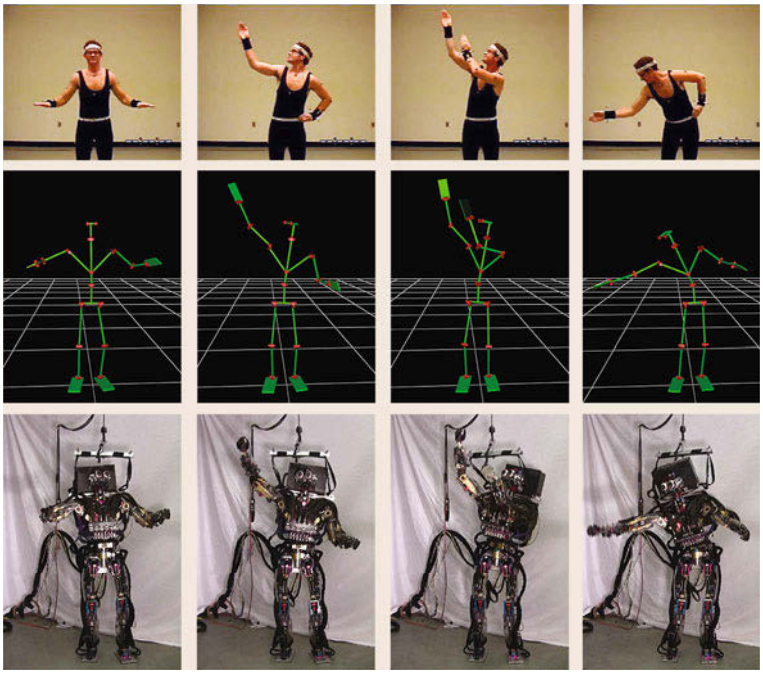


Fig. 68.14 Snapshots from the hardware experiment

Video-References

VIDEO 762 Example of optical motion capture data converted to joint angle data
available from http://handbookofrobotics.org/view-chapter/65/videodetails/762

VIDEO 763 Example of muscle tensions computed from motion capture data
available from http://handbookofrobotics.org/view-chapter/65/videodetails/763

VIDEO 764 The Crystal Ball: Predicting future motions
available from http://handbookofrobotics.org/view-chapter/65/videodetails/764

VIDEO 765 Human motion mapped to a humanoid robot
available from http://handbookofrobotics.org/view-chapter/65/videodetails/765

VIDEO 766 Converting human motion to sentences
available from http://handbookofrobotics.org/view-chapter/65/videodetails/766

References

68.1 M.J. Mataric: Getting humaniods to move and imitate, IEEE Intell. Syst. **15**(4), 18–24 (2000)

68.2 S. Nakaoka, A. Nakazawa, F. Kanehiro, K. Kaneko, M. Morisawa, H. Hirukawa, K. Ikeuchi: Learning from observation paradigm: Leg task models for enabling a biped humanoid robot to imitate human dances, Int. J. Robotics Res. **26**(8), 829–844 (2010)

68.3 Organic Motion, Inc.: OpenStage, http://organicmotion.com/mocap-for-animation

68.4 Microsoft: Kinect for Xbox One, http://www.xbox.com/en-US/xbox-one/accessories/kinect-for-xbox-one

68.5 M. Vondrak, L. Sigal, J. Hodgins, O. Jenkins: Video-based 3D motion capture through biped control, ACM Trans. Graph. **31**(4), 24 (2012)

68.6 Y. Nakamura, K. Yamane, Y. Fujita, I. Suzuki: somatosensory computation for man-machine interface from motion capture data and musculoskeletal human model, IEEE Trans. Robotics **21**(1), 58–66 (2005)

68.7 K. Yamane, Y. Nakamura: Natural motion animation through constraining and deconstraining at will, IEEE Trans. Vis. Comput. Graph. **9**(3), 352–360 (2003)

68.8 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Addison-Wesley, Reading 1986)

68.9 A.V. Hill: The heat of shortening and the dynamic constants of muscle, Proc. R. Soc. Lond. B **126**, 136–195 (1938)

68.10 S. Stroeve: Impedance chracteristics of a neuromusculoskeletal model of the human arm I: Posture control, J. Biol. Cyberneics **81**, 475–494 (1999)

68.11 J. Rasmussen, M. Damsgaard, M. Voigt: Muscle recruitment by the min/max criterion—a comparative study, J. Biomech. **34**(3), 409–415 (2001)

68.12 K. Yamane, Y. Fujita, Y. Nakamura: Estimation of physically and physiologically valid somatosensory information, Proc. IEEE/RSJ Int. Conf. Robotics Autom. (ICRA) (2005) pp. 2635–2641

68.13 B.W. Mooring, Z.S. Roth, M.R. Driels: *Fundamentals of Manipulator Calibration* (Wiley, New York 1991)

68.14 Digital Human Research Center, AIST: Human Body Properties Database, https://www.dh.aist.go.jp/database/properties/index-e.html

68.15 K. Ayusawa, G. Venture, Y. Nakamura: Identification of humanoid robots dynamics using minimal set of sensors, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 2854–2859

68.16 T. Kim, S.I. Park, S.Y. Shin: Nonmetric individual differences multidimensional scaling: An alternating least squres method with optimal scaling features, ACM Trans. Graph. **22**(3), 392–401 (2003)

68.17 T. Shiratori, A. Nakazawa, K. Ikeuchi: Detecting dance motion structure through music analysisg, Proc. 6th IEEE Int. Conf. Autom. Face Gesture Recognit. (2004) pp. 857–862

68.18 J. Kohlmorgen, S. Lemm: A dynamic HMM for online segmentation of sequential data, Proc. Conf. Neural Inf. Process. Syst. (2002) pp. 793–800

68.19 D. Kulic, W. Takano, Y. Nakamura: Online segmentation and clustering from continuous observation of whole body motions, IEEE Trans. Robotics **25**(5), 1158–1166 (2009)

68.20 J.L. Elman: Finding structure in time, Cogn. Sci. **14**, 179–211 (1990)

68.21 W. Takano, Y. Nakamura: Humanoid robot's autonomous acquisition of proto-symbols through motion segmentation, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2006) pp. 425–431

68.22 A.J. Ijspeert, J. Nakanishi, S. Shaal: Learning control policies for movement imitation and movement recognition, Neural Inf. Process. Syst. **15**, 1547–1554 (2003)

68.23 M. Haruno, D. Wolpert, M. Kawato: MOSAIC model for sensorimotor learning and control, Neural Comput. **13**, 2201–2220 (2001)

68.24 T. Inamura, I. Toshima, H. Tanie, Y. Nakamura: Embodied symbol emergence based on mimesis theory, Int. J. Robotics Res. **23**(4), 363–377 (2004)

68.25 A. Billard, S. Calinon, F. Guenter: Discriminative and adaptive imitation in uni-manual and bi-manual tasks, Robotics Auton. Syst. **54**, 370–384 (2006)

68.26 M. Okada, K. Tatani, Y. Nakamura: Polynomial design of the nonlinear dynamics for the brain-like information processing of whole body motion, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 1410–1415

68.27 A.J. Ijispeert, J. Nakanishi, T. Shibata, S. Schaal: Nonlinear dynamical systems for imitation with humanoid robots, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2001)

68.28 T. Matsubara, S.H. Hyon, J. Morimoto: Learning parametric dynamic movement primitives from multiple demonstrations, Neural Netw. **24**(5), 493–500 (2011)

68.29 J. Tani, M. Ito: Self-organization of behavioral primitives as multiple attractor dynamics: A robot experiment, IEEE Trans. Syst. Man Cybern. A **33**(4), 481–488 (2003)

68.30 L. Rabiner: A Tutorial on hidden Markov models and selected applications in speech recognition, Proceedings IEEE (1989) pp. 257–286

68.31 L. Kovar, M. Gleicher, F. Pighin: Motion graphs, ACM Trans. Graph. **21**(3), 473–482 (2002)

68.32 W. Takano, H. Imagawa, D. Kulic, Y. Nakamura: What do you expect from a robot that tells your future? The crystal ball, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Taipei (2008) pp. 1780–1785

68.33 Y. Sugita, J. Tani: Learning semantic combinatoriality from the interaction between linguistic and behavioral processes, Adapt. Behav. **3**(1), 33–52 (2005)

68.34 T. Ogata, M. Murase, J. Tani, K. Komatani, H.G. Okuno: Two-way translation of compound sentences and arm motions by recurrent neural networks, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2007) pp. 1858–1863

68.35 P.F. Brown, S.A.D. Pietra, V.J.D. Pietra, R.L. Mercer: The mathematics of statistical machine translation: Parameter estimation, Comput. Linguist. **19**(2), 263–311 (1993)

68.36 W. Takano, Y. Nakamura: Construction of a space of motion labels from their mapping to full-body motion symbols, Adv. Robotics **29**(2), 115–126 (2015)

68.37 W. Takano, Y. Nakamura: Statistical mutual conversion between whole body motion primitives and linguistic sentences for human motions, Int. J. Robot. Res. **34**(10), 1314–1328 (2015)

68.38 S. Fine, Y. Singer, N. Tishby: The hierarchical hidden markov model: Analysis and application, Mach. Learn. **32**, 41–62 (1998)

68.39 M. Brand, N. Oliver, A. Pentland: Coupled hidden Markov models for complex action recognition, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (1999) pp. 994–999

68.40 W. Takano, K. Yamane, T. Sugihara, K. Yamamoto, Y. Nakamura: Primitive communication based on motion recognition and generation with hierarchical mimesis model, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 3602–3609

68.41 D. Lee, C. Ott, Y. Nakamura: Mimetic communication model with compliant physical contact in human-humanoid interaction, Int. J. Robotics Res. **29**(13), 1684–1704 (2004)

68.42 E. Demerican, L. Sentis, V. De Sapio, O. Khatib: Human motion reconstruction by direct control of marker trajectories. In: *Advances in Robot Kinematics: Analysis and Design*, ed. by J. Lenar-

cic, P. Wenger (Springer, Berlin, Heidelberg 2008) pp. 263–272
68.43 O. Khatib, O. Brock, K. Chang, F. Conti, D. Ruspini, L. Sentis: Robotics and interactive simulation, Commun. ACM **45**(3), 46–51 (2002)
68.44 J.M. Wang, S.R. Hamner, S.L. Delp, V. Koltun: Optimizing locomotion controllers using biologically-based actuators and objectives, ACM Trans. Robotics **31**(4), 25 (2012)
68.45 V.B. Zordan, J.K. Hodgins: Motion capture-driven simulations that hit and react, Proc. ACM SIGGRAPH Symp. Comput. Animat. (2002) pp. 89–96
68.46 K.W. Sok, M.M. Kim, J.H. Lee: Simulating biped behaviors from human motion data, ACM Trans. Graph. **26**(3), 107 (2007)
68.47 M. Da Silva, Y. Abe, J. Popović: Interactive simulation of stylized human locomotion, ACM Trans. Graph. **27**(3), 82 (2008)
68.48 K. Miura, M. Morisawa, F. Kanehiro, S. Kajia, K. Kaneko, K. Yokoi: Human-like walking with toe supporting for humanoids, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 4428–4435
68.49 C. Ott, D.H. Lee, Y. Nakamura: Motion capture based human motion recognition and imitation by direct marker control, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2008) pp. 399–405
68.50 K. Yamane, J.K. Hodgins: Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data, Proc. IEEE/RSJ Int. Conf. Intell. Robot Syst. (IROS) (2009) pp. 2510–2517
68.51 K. Yamane, S.O. Anderson, J.K. Hodgins: Controlling humanoid robots with human motion data: Experimental validation, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2010) pp. 504–510

Multimedia Contents

69. Physical Human–Robot Interaction

Sami Haddadin, Elizabeth Croft

Over the last two decades, the foundations for physical human–robot interaction (pHRI) have evolved from successful developments in mechatronics, control, and planning, leading toward safer lightweight robot designs and interaction control schemes that advance beyond the current capacities of existing high-payload and high-precision position-controlled industrial robots. Based on their ability to sense physical interaction, render compliant behavior along the robot structure, plan motions that respect human preferences, and generate interaction plans for collaboration and coaction with humans, these novel robots have opened up novel and unforeseen application domains, and have advanced the field of human safety in robotics.

This chapter gives an overview on the state of the art in pHRI. First, the advances in human safety are outlined, addressing topics in human injury analysis in robotics and safety standards for pHRI. Then, the foundations of human-friendly robot design, including the development of lightweight and intrinsically flexible force/torque-controlled machines together with the required perception abilities for interaction are introduced. Subsequently, motion-planning techniques for human environments, including the domains of biomechanically safe, risk-metric-based, human-aware planning are covered. Finally, the rather recent problem of interaction planning is summarized, including the issues of collaborative action planning, the definition of the interaction planning problem, and an introduction to robot reflexes and reactive control architecture for pHRI.

69.1 **Classification** 1836
69.1.1 Classification of Interaction 1838
69.2 **Human Safety** 1839
69.2.1 Human Injury in Robotics 1839
69.2.2 Safety Standards for Human–Robot Interaction 1845
69.3 **Human-Friendly Robot Design** 1847
69.3.1 Lightweight Design 1847
69.3.2 Intrinsically Flexible Design 1849
69.3.3 Perception for Interaction 1851
69.3.4 Proprioceptive Force/Torque Sensing 1852
69.3.5 Tactile Perception 1852
69.3.6 Visual Perception 1853
69.4 **Control for Physical Interaction** 1853
69.4.1 Interaction Control 1853
69.4.2 Learning and Adaptation 1854
69.4.3 Collision Handling 1856
69.4.4 Shared Manipulation Control 1858
69.5 **Motion Planning for Human Environments** 1859
69.5.1 Biomechanically Safe Motion Planning 1859
69.5.2 Risk-Metric-Based Motion Planning 1861
69.5.3 Human-Aware Motion Planning 1862
69.6 **Interaction Planning** 1862
69.6.1 Collaborative Action Planning 1862
69.6.2 Interaction Planning Problem 1864
69.6.3 Robot Reflexes 1865
69.6.4 Reactive Control Architecture 1866
69.7 **Conclusions and Challenges** 1867
Video-References 1868
References 1869

69.1 Classification

Robotics is currently undergoing a fundamental paradigm shift, both in research and real-world applications. Classically, it was dominated for the last decades by possibly dangerous position-controlled rigid robots carrying out typical automation tasks, such as positioning and path tracking in various applications. Recently, a new generation of mechatronic robots has appeared on the landscape, including novel concepts in general robot design within the soft-robotics context. This trend brings us closer to the long-term goal of safe, seamless physical human–robot interaction (pHRI) in the real domestic and professional world (Fig. 69.1).

Recent advances in physical human–robot interaction have shown the potential and feasibility of robot systems for active and safe workspace sharing and collaboration with humans. The fundamental breakthrough was the human-centered design of robot mechanics and control (soft-robotics), which also induced the novel research stream of intrinsically elastic robots (series elastic actuators (SEA) or its generalization variable impedance actuators (VIA)). By considering the physical contact of the human and the robot in the design phase, possible injuries due to unintentional contacts can be considerably mitigated. Furthermore, taking into account the human's intention and preferences will enable the realization of human-friendly motions and interaction behavior. Some of the most advanced systems that were developed are now entering into industrial markets. These technologies serve both industrial and service-oriented domains. Possible future applications of these novel devices developed for close interaction with humans are depicted in Fig. 69.2. They range from industrial coworkers and mobile servants over robots in the professional service sector, assistive devices for physically challenged individuals, to service robots for the support of general household activities. All of these applications share the common requirement of close, safe, and dependable physical interaction between human and robot in a shared workspace. Therefore, such robots need to be carefully designed for human friendliness. That is, they have to be able to safely sense, reason, learn, and act in a partially unknown world inhabited by humans. In turn, this set of requirements necessitates the design of novel solutions in various theoretical and technological developments. In contrast to the classical modular view on robotics technology, and the role humans play in this, a fundamental paradigm shift in robot development has to be pursued. While encompassing safety issues based on biomechanical human injury analysis as well as on the kinesiologic biomechanics

Fig. 69.1 The current paradigm shift in robotics induced by new target domains and robots toward the vision of close human–robot coexistence (courtesy of Keller und Knappich Augsburg (KUKA), Deutsches Zentrum für Luft- und Raumfahrt (DLR), ABB, Rethink Robotics)

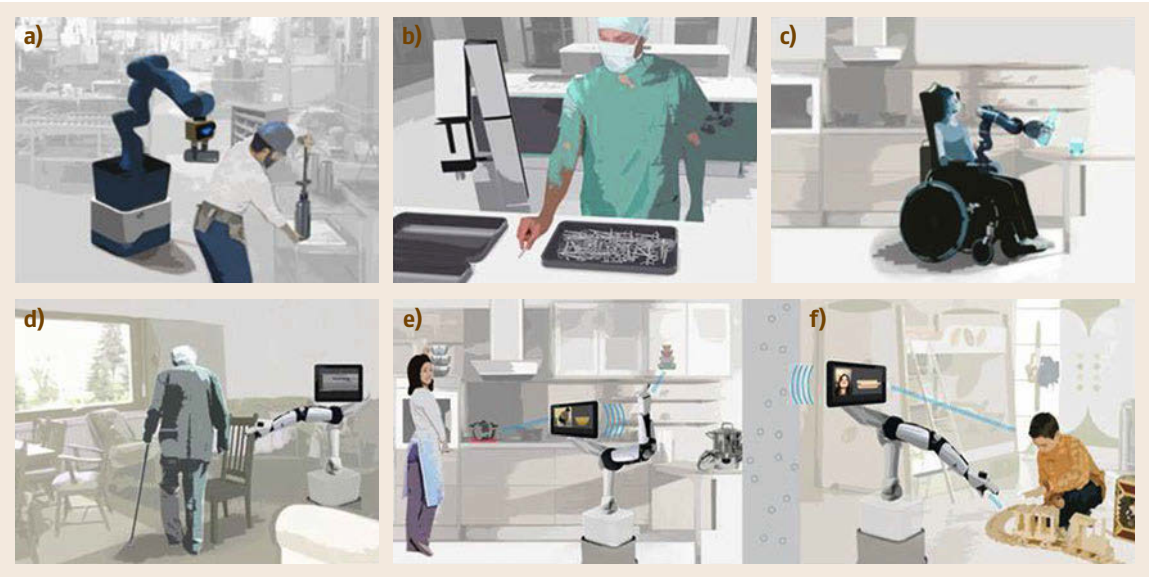


Fig.69.2a–f Application examples for pHRI, ranging from shop floor logistics and manipulation **(a,b)**, over professional service robots and assisitve devices for the disabled **(c,d)**, to service robots in domestic applications **(e,f)**

of human movements, human-friendly hardware design and interaction control strategies, learning, perceptive, and cognitive key components have to be developed and validated. These need to enable robots to track, understand, and predict human motions in real time in a weakly structured dynamic environment. Apart from developing the capabilities for interactive autonomy including self-improvement, human safety and physical interaction have to be embedded at the cognitive decisional level as well. This will enable the robots to react or physically interact with humans in a safe and autonomous way. Biomechanical knowledge, neuromechanical insights, and biologically motivated variable compliance actuators can be used to design manipulation/interaction systems of varying complexity close to human properties and performance. Further fundamental insights into novel designs of VIAs for an improved torque/mass ratio and energy efficiency with new control methods to exploit the stiffness and damping properties are required.

Planning and adapting motions and tasks of such complex systems in real-time require new concepts, including tight coupling of control, planning, and learning, which will lead to reactive behaviors capable of self-improvement. Moreover, self-explaining interaction and communication frameworks need to be developed to enhance the system usability and interpretability for humans. These should, for example, communicate whether a situation is safe or dangerous using not only verbal, but also nonverbal communication cues, such as gestures and emotional feedback. Finally, the dependability of all system components and algorithms is a major issue, the systematic treatment of which is of particular importance for subsequent industrial commercialization of the technology and also for the commercial domestic use of robots in everyday environments. Thus, the foreseeable breakthrough of the next generation of robotic systems in flexible automation in both small and medium enterprises (SMEs) and global market companies depends primarily on the pHRI development over the next years. Robotic assistance in manual processes that advantageously partner human and robot workers has an enormous unexplored potential to amplify productively in processes that previously could not be automated due to technological, cost, or efficiency reasons. Furthermore, very promising application domains of the technology are in the professional service sector (e.g., hospital support systems) and in the logistics domain (food logistics and quality inspection), which are so far to a large extent still purely manual work places. As the natural next step, systems capable of pHRI will enter the home sector as home assistants, elderly care assist, and assistive devices (VIDEO 607, VIDEO 614, VIDEO 618, VIDEO 623) for physically challenged people (VIDEO 618, VIDEO 619, VIDEO 620, VIDEO 621, VIDEO 622). First, rather basic tasks, such as fetch-and-carry or environment manipulation, will be solved followed by applications with increasing complexity.

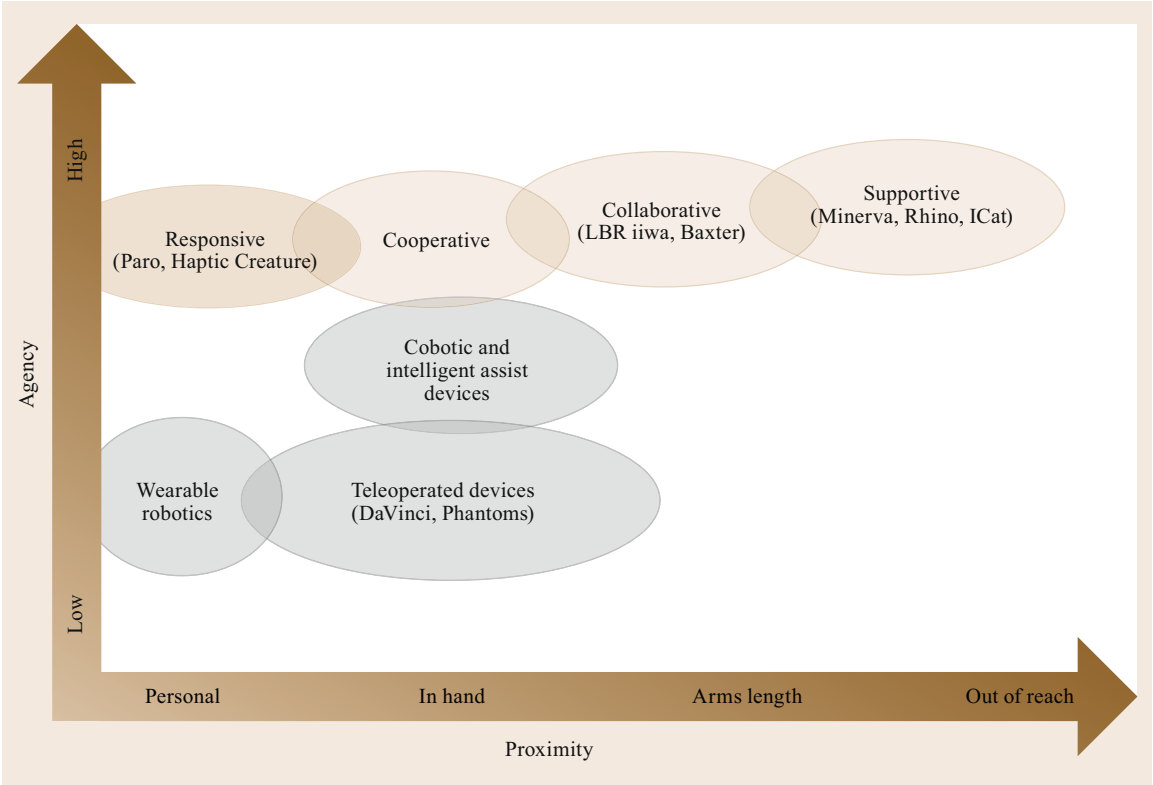


Fig. 69.3 Classification scheme for pHRI, by proximity of the interaction and agency (available autonomy) of the robot

69.1.1 Classification of Interaction

pHRI falls into the category of *proximate* interaction where humans and robots are colocated, as opposed to *remote* or teleoperated interaction (Chap. 43) [69.1]. Beyond proximity, the nature of the physical interaction can be understood in the context of the tasks and roles undertaken by the robot and human actors in a pHRI scenario. In all of these scenarios, a key feature is the available autonomy, or agency, of the robot partner for performing its portion of the task. This agency separates pHRI from Cobotic devices [69.2] and other passive robotic lift assists that require, by design, input from the operator.

Most work in pHRI can be generally classified across three broad categories of interaction: *supportive, collaborative, and cooperative*. Ordered in this way we note that these interactions are marked by increasing frequency and necessity of physical contact with the robot and level of proximity to the user (Fig. 69.3). Further categories include touch-based, personally responsive robots, for example, Paro [69.3] and the Haptic Creature [69.4], and wearable robots (Chap. 70).

In *supportive* interactions, we group interactions where the robot is not integral to the central performance of a task, but instead provides the human with the tools, materials, and information to optimize the human's task performance or objectives, for example, museum tour guide robots, shopping assistant robots for aiding seniors [69.5], and homecare robots (Chaps. 65 and 73). In this context, pHRI is typically concerned with safety, that is, preventing and mitigating the effect of unexpected contacts or collisions, and performing appropriate proxemic behavior. When required, physical interaction is infrequent and transitory in nature – typically limited to handoffs or other infrequent transactional exchanges. To support safety, as well as these limited physical interactions, well-structured human–robot communication (Chap. 71) is essential. For example, recent work [69.6–11] has demonstrated the importance of bi-lateral gesture cues in performing turn-taking, information sharing, close proximity activities, and precontact handover operations.

In *collaborative* interactions (VIDEO 609), the human and robot both work on the task, with the labor divided between the robot and human, each separately completing the parts of the task best suited to their

abilities, but more frequently interacting through turn-taking and part/tool passing [69.12, 13] (VIDEO 716), or haptically enabled mode switching where contact is used to switch the robot's interactive behavior [69.14] (VIDEO 717, VIDEO 632). In these scenarios, the human completes task elements requiring human dexterity or decision making, while the robot completes elements not well suited to direct human involvement, for example, repetitive or high-force applications, chemical deposition, or precision placement. In both supportive and collaborative interactions, physical space is often shared but planned physical interactions, although much more frequent, are still *transactional* in nature.

Cooperative interactions refer to the extension of cooperative manipulation (see also Chaps. 39 and 70) to include force interactions with humans. This type of interaction is differentiated from Cobots in that the robot operates as an independent agent, rather than a passive assist. That is, the human and the robot work in direct physical contact, or indirect contact through a common object, with continuous and cooperative shared control of the task. Cooperative interactions encompass tasks, such as cooperative lifting and carrying [69.15–17] (VIDEO 613, VIDEO 820), kinesthetic teaching [69.18] (VIDEO 627), coordinated material handling (e.g., managing long and flexible objects), and rehabilitation therapy (Chap. 64).

69.2 Human Safety

Providing safety in pHRI is a multifaceted challenge and requires an analysis on various levels of abstraction. pHRI aims at the coexistence of humans and robots in a common workspace and at extending their communication modes by physical means. This spatial proximity leads to a variety of *potential threats*, determined by the current state of the system of interest, which consists of the human(s), the robot(s), and their surrounding environment. Understanding the respective threats, in particular regarding potential human injury originating from physical robot–human contacts, and embedding the insights accordingly into safety standards/regulations is one of the major challenges of nowadays robotics (note that this chapter does not cover functional safety or robot dependability).

69.2.1 Human Injury in Robotics

Impact Scenarios

In order to quantify human injury that may occur in the context of pHRI, one needs to understand how mechanical contacts may cause injury in principle. Figure 69.4 depicts relevant robot–human impact scenarios. These may involve unconstrained impacts, clamping in the robot structure, constrained impacts, partially constrained impacts, and resulting secondary

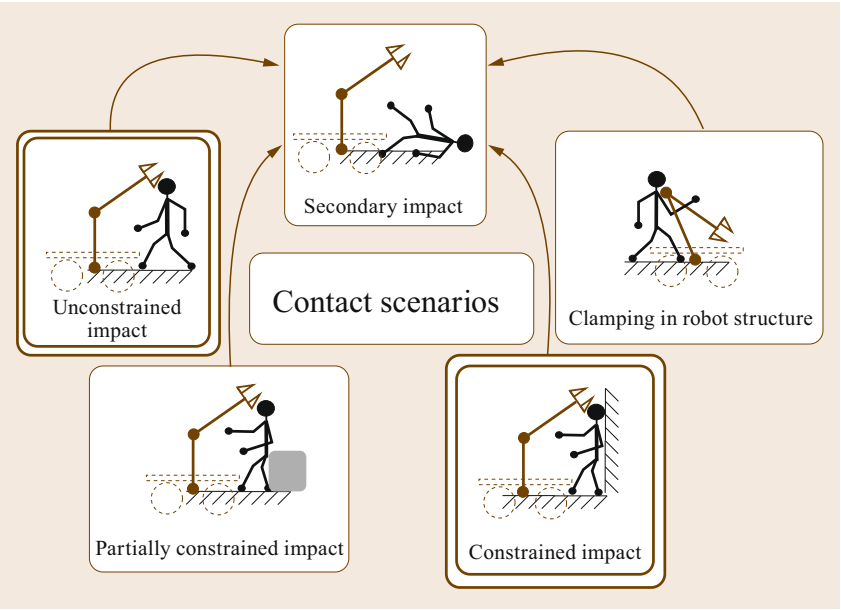


Fig. 69.4 Robot–human impact scenario classes. Unconstrained and constrained impacts are considered the two main scenarios

impacts [69.19]. Apart from such situational definitions, the most urgent question is how to quantify the human injury level that might occur due to a collision between human and robot. The understanding of human injury has been treated in the fields of injury biomechanics and forensics for several decades and the respective studies served for the early work on human injury in robotics. In fact, various injury measures from biomechanics and forensics were applied to human injury analysis in robotics [69.19–25]. An overview on the most important existing injury classification metrics and biomechanical injury measures can be found in [69.26] (VIDEO 608). The most important results from the biomechanics, forensics, and robotics literature are briefly reviewed now.

Overview Biomechanics Literature

In order to derive the injury characteristics of different body parts for direct collisions with an impactor, which is the most relevant case for robotics, countless experiments and publications have been produced over the last 50 years. The investigated impactors used in robotics and biomechanics experiments vary significantly in size and shape. However, from the test setups, one can identify and cluster principal geometric primitives. The main primitives and their parameters are depicted in Fig. 69.5. The z-axis of the coordinate frame associated with each primitive defines the direction of impact $\boldsymbol{u}$.

Numerous relevant impact experiments with cadavers, volunteers, crash test dummies, and biological tissue for the head, neck, and chest were generated (Tables 69.1–69.4). There for all selected experimental campaigns, the collision scenario, impacted body part, impact parameters according to Fig. 69.5, subject, and impact velocity are listed. For describing the collision scenario, we use the following abbreviations: D: dynamic, QS: quasi-static, U: unconstrained, C: constrained, PC: partially constrained. A collision experiment denoted by DU is, thus, dynamic unconstrained, while quasi-static constrained impacts are labeled QSC (Sect. 69.2.1, *Synopsis*). The respective impactor type and parameters are listed for comparison.

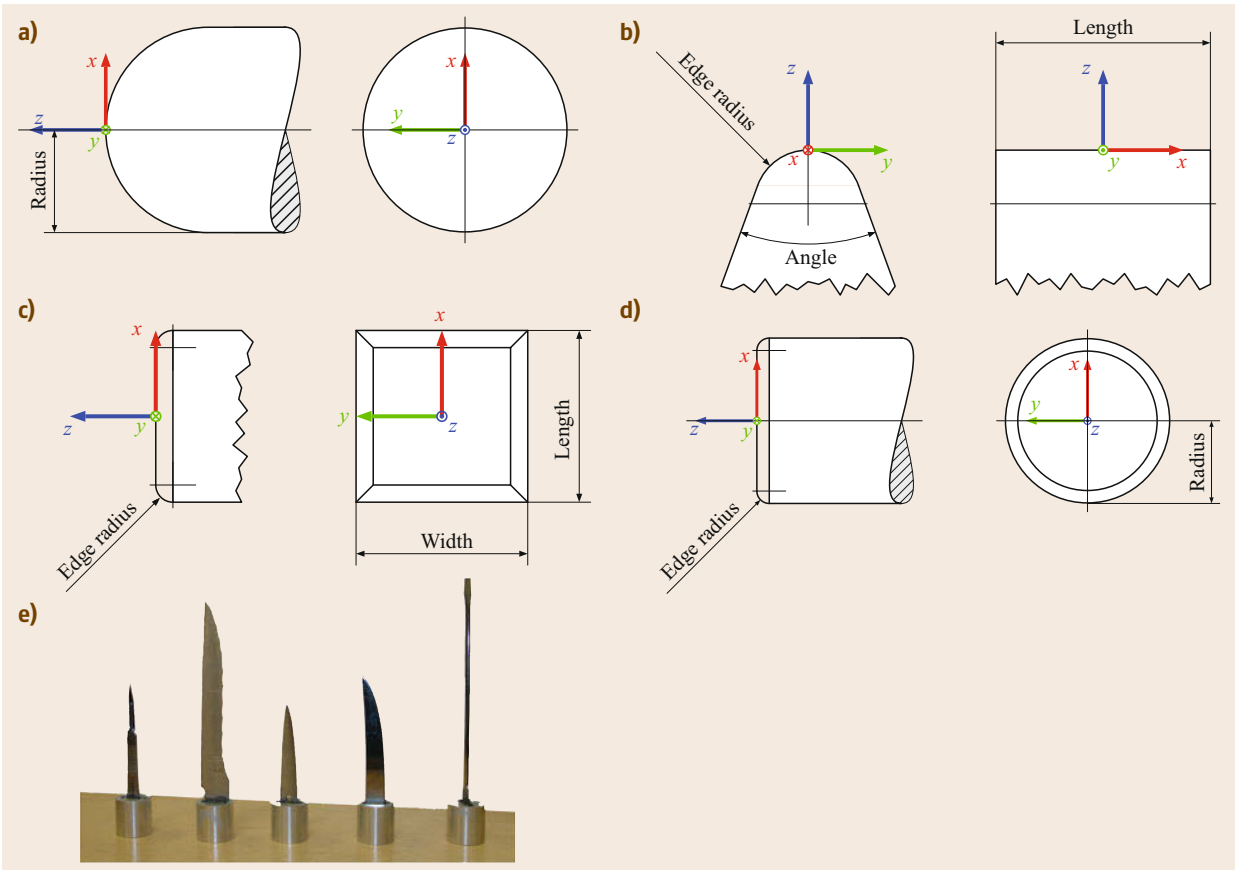


Fig.69.5a–e Typical impactor primitives with according parameters. **(a)** Sphere, **(b)** edge, **(c)** cuboid, **(d)** flat circular, **(e)** sharp tools

Table 69.1 Overview of selected impact experiments from biomechanics and robotics literature. Body part: Head

Impactor type	Impactor parameters	Collision case	Subject	Mass (kg)	Velocity (m/s)	References
Flat circular						
Maxilla, zygoma, frontal, temporo-parietal, mandible	14.3 mm radius	dynamic constrained (DC)	Cadaver	1.08–3.82	2.99–5.97	[69.27, 28]
Temporo-Parietal	12.7 mm radius	DC	Cadaver	10.6	2.7	[69.29]
Nose	14.3 mm radius	DC	Cadaver	3.2	1.56–3.16	[69.30]
Frontal	35 mm radius	DU	Cadaver	14.3	3.37–6.99	[69.31]
Edge						
Nose	12.5 mm radius	DU	Cadaver	32, 64	2.77–6.83	[69.32]
Maxilla, zygoma, frontal	10 mm radius	DC	Cadaver	14.5	2.4–4.2	[69.33]
Frontal	12.7 mm radius	dynamic partially constrained (DPC)	Cadaver	∞ (human falling on impactor)	2.23–3.14	[69.34]
Cuboid						
Temporo-parietal	50 mm length, 100 mm width	DC	Cadaver	12	4.3	[69.29]
Frontal	Size not specified, padded	DPC	Cadaver	5.31–5.97	3.56–9.6	[69.35]
Frontal	size not specified	DPC	Cadaver	∞ (human falling on impactor)	2.23–3.87	[69.34]
Sphere						
Frontal	120 mm radius	DU, QSC, DPC	Hybrid III dummy	4, 67, 1980	0.2–4.2	[69.36, 37]
Frontal	203.2, 76.2 mm radius	DPC	Cadaver	∞ (human falling on impactor)	2.87–3.5	[69.34]

Table 69.2 Overview of selected impact experiments from biomechanics and robotics literature. Body part: Torso

Impactor type	Impactor parameters	Collision case	Subject	Mass (kg)	Velocity (m/s)	References
Flat circular						
Thorax	76.2 mm radius, 12.77 mm edge radius	DU, DC	Cadaver	1.6–23.6	4.34–14.5	[69.38, 39]
Thorax	76 mm radius, rubber padded	DU	Volunteer	10	2.4–4.6	[69.40]
Thorax	76.2 mm radius, 12.77 mm edge radius	DU	Cadaver	19.27	4.0–10.6	[69.41]
Abdomen	12.7 mm radius	DU	Cadaver	32, 64	4.9–13.0	[69.42]
Sphere						
Thorax	120 mm radius	DU, QSC	Hybrid III dummy	4, 67, 1980	0.2–4.2	[69.36, 37]
Abdomen	5, 12.5 mm radius	DC	Pig tissue	2–10	0.5–4.0	[69.25]
Edge						
Abdomen	45° angle, 200 mm length, 0.2 mm edge radius	DC	Pig tissue	2–10	0.5–4.0	[69.25]

Table 69.3 Overview of selected impact experiments from biomechanics and robotics literature. Body part: Upper extremities

Impactor type	Impactor parameters	Collision case	Subject	Mass (kg)	Velocity (m/s)	References
Edge						
Forearm	12.5 mm radius, angle 0°	DC	Cadaver	9.48	3.63	[69.43]
Forearm	size not specified	DC	Cadaver	9.75	2.44, 4.23	[69.44]
Shoulder, upper arm, forearm	5 mm edge radius, 30° angle	DC	Volunteer	4.16, 8.65	0.45–1.25	
Flat circular						
Forearm, hand	size not specified	QSC	Cadaver	∞ (velocity control)	25 mm/min	[69.45]

Table 69.4 Overview of selected impact experiments from biomechanics and robotics literature. Body part: Lower extremities

Impactor type	Impactor parameters	Collision case	Subject	Mass (kg)	Velocity (m/s)	References
Sharp	Fig. 69.5	DC	Pig tissue, volunteer	4	0.16–0.8	[69.24]

Next, some essential characteristics of human–robot impacts are elaborated for a more general understanding of the underlying dynamics.

Robot–Human Impacts

Let us assume that of a serial chain rigid robot consisting of n joints, there is at most a single link involved in a collision. Let

$$\dot{\boldsymbol{x}}_{\mathrm{c}} = \begin{pmatrix} \boldsymbol{v}_{\mathrm{c}} \\ \boldsymbol{\omega}_{\mathrm{c}} \end{pmatrix} = \begin{pmatrix} \mathbf{J}_{\mathrm{c,lin}}(\boldsymbol{q}) \\ \mathbf{J}_{\mathrm{c,ang}}(\boldsymbol{q}) \end{pmatrix} \dot{\boldsymbol{q}} = \mathbf{J}_{\mathrm{c}}(\boldsymbol{q})\dot{\boldsymbol{q}} \in \mathbb{R}^6 \quad (69.1)$$

be the stacked (screw) vector of linear velocity at the contact point and the angular velocity of the associated robot link, with an associated (geometric) contact Jacobian $\mathbf{J}_{\mathrm{c}}(\boldsymbol{q})$ that is a function of the joint angle $\boldsymbol{q}$. Accordingly, the Cartesian collision wrench is denoted by

$$\boldsymbol{\mathcal{F}}_{\mathrm{ext}} = \begin{pmatrix} \boldsymbol{f}_{\mathrm{ext}} \\ \boldsymbol{m}_{\mathrm{ext}} \end{pmatrix} \in \mathbb{R}^6 \,. \quad (69.2)$$

Robot Collision Modeling. When such a collision occurs, the robot dynamics becomes

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{\tau}_{\mathrm{F}} = \boldsymbol{\tau} + \boldsymbol{\tau}_{\mathrm{ext}} \,, \quad (69.3)$$

where $\mathbf{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is the symmetric and positive definite joint space inertia matrix, $\mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^n$ is the centripetal and Coriolis vector, and $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^n$ is the gravity vector; $\boldsymbol{\tau} \in \mathbb{R}^n$ is the motor torque, and $\boldsymbol{\tau}_{\mathrm{F}} \in \mathbb{R}^n$ is the dissipative friction torque; $\boldsymbol{\tau}_{\mathrm{ext}} \in \mathbb{R}^n$ is the typically unknown external joint torque given by

$$\boldsymbol{\tau}_{\mathrm{ext}} = \mathbf{J}_{\mathrm{c}}^{\mathrm{T}}(\boldsymbol{q})\,\boldsymbol{\mathcal{F}}_{\mathrm{ext}} \,. \quad (69.4)$$

The effective mass $m_{\boldsymbol{u}}$ of a robot acting in the instantaneous collision direction $\boldsymbol{u}$, which has to be consistent to $\mathbf{J}_{\mathrm{c}}(\boldsymbol{q})$, can be deduced from $\mathbf{M}(\boldsymbol{q})$ via the Cartesian kinetic energy matrix $\boldsymbol{\Lambda}(\boldsymbol{q})$. This is defined as

$$\boldsymbol{\Lambda}(\boldsymbol{q}) = \left[\mathbf{J}_{\mathrm{c}}(\boldsymbol{q})\mathbf{M}(\boldsymbol{q})^{-1}\mathbf{J}_{\mathrm{c}}(\boldsymbol{q})^{\mathrm{T}}\right]^{-1} \,, \quad (69.5)$$

where the inverse of $\boldsymbol{\Lambda}(\boldsymbol{q})$ is based on the decomposition of the kinetic energy matrix

$$\boldsymbol{\Lambda}(\boldsymbol{q})^{-1} = \begin{bmatrix} \boldsymbol{\Lambda}_v(\boldsymbol{q})^{-1} & \overline{\boldsymbol{\Lambda}}_{v\omega}(\boldsymbol{q}) \\ \overline{\boldsymbol{\Lambda}}_{v\omega}(\boldsymbol{q})^{\mathrm{T}} & \boldsymbol{\Lambda}_{\omega}(\boldsymbol{q})^{-1} \end{bmatrix} , \quad (69.6)$$

with $\overline{\boldsymbol{\Lambda}}_{v\omega}(\boldsymbol{q}) = \mathbf{J}_{\mathrm{c,lin}}(\boldsymbol{q})\mathbf{M}(\boldsymbol{q})^{-1}\mathbf{J}_{\mathrm{c,ang}}(\boldsymbol{q})^{\mathrm{T}}$. Finally, $m_{\boldsymbol{u}}$ is found to be

$$m_{\boldsymbol{u}} = [\boldsymbol{u}^{\mathrm{T}}\boldsymbol{\Lambda}_v(\boldsymbol{q})^{-1}\boldsymbol{u}]^{-1} \,. \quad (69.7)$$

It should be noted that the Jacobian has to be the *center-of-mass-Jacobian*. Otherwise, the entire inverse of the Cartesian inertia tensor has to be used, and not just its translational component block. More details can be found in [69.46]. We assume the local impact curvature in the $\boldsymbol{u}$-direction to be denoted by $c_{\boldsymbol{u}}$.

Characteristic Robot–Human Impact Force Profile. A physical collision between robot and human is typically characterized by a distinct force profile that is composed by two consecutive phases (note that for unconstrained soft-tissue collisions these two phases can simplify into a single *Phase I* impact) (Fig. 69.6):

1. Phase I is characterized by a very short impact, governed by the robot- and human-reflected dynamics.
2. Phase II is characterized by a quasistatic contact event. Without clamping, this is a pushing force, whereas if the human is clamped it is a crushing force.

Phase I can be treated from a pure impact physics, almost open-loop point of view, that is, it is determined by the reflected inertia, velocity, and impact curvature $c_{\boldsymbol{u}}$ of the robot together with the characteristics of the respective body part that is being struck. The maximum contact force is denoted F_{I}.

Phase II, on the other hand, has to be further subdivided into either *clamping* or *no clamping* incident. In the case of no clamping, the maximum force is F_{IIA}, whereas for clamping, the maximum force is F_{IIB}. In particular, *Phase II* is highly robot control and design dependent and is especially important in the case of clamping:

- *Phase IIA*: *No clamping*. Typically, for free impacts at robot velocities $> 0.3\,\mathrm{m/s}$, F_{IIA} is significantly smaller than F_{I}. Otherwise, F_{I} is smaller than F_{IIA} and is governed by the robot actuator torques (active quasistatic pushing) and the reaction of the human body that is mainly governed by its reflected impedance.
- *Phase IIB*: *Clamping*. In the case of clamping, the final maximum force F_{IIB} is limited by the maximum motor torques $\boldsymbol{\tau}_{\max}$ of the robot via $\boldsymbol{\mathcal{F}}_{\mathrm{ext}} = \mathbf{J}_{\mathrm{c}}^{\mathrm{T\#}}\boldsymbol{\tau}_{\max}$, where $\mathbf{J}_{\mathrm{c}}^{\mathrm{T\#}}$ is the contact Jacobian pseudoinverse. If the robot is powerful enough to generate active contact forces that penetrate or break human tissue/structure, the contact force is, of course,

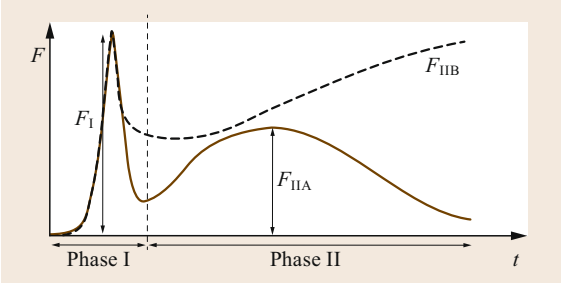


Fig. 69.6 Typical robot–human collision force profiles

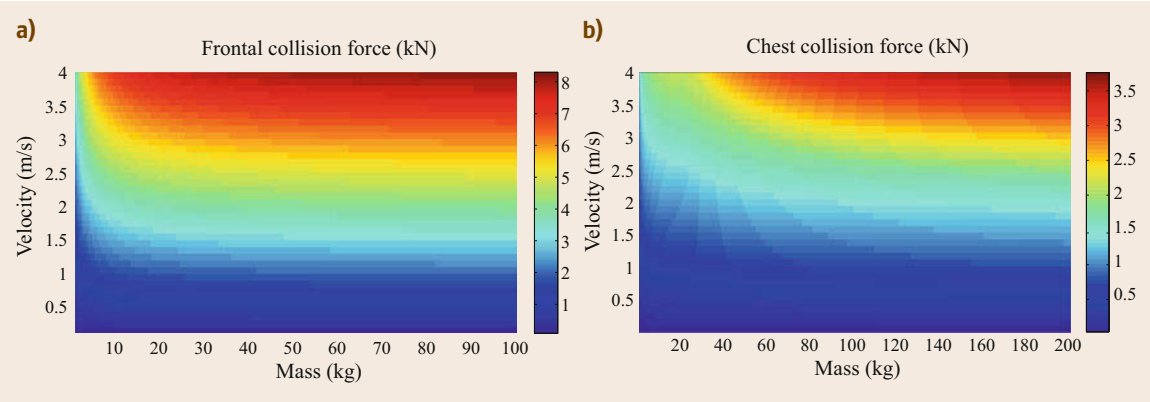


Fig.69.7a,b Mass–velocity dependency for **(a)** human head and **(b)** chest contact force. A mass–spring-mass model is used for collisions against the head, where the head mass M_{H} is 4.5 kg and the approximate contact stiffness of the frontal bone $K_{\mathrm{H}} = 1000\,\mathrm{N/mm}$ (after [69.33]). For the chest, the model proposed in [69.47] is used

limited by the human maximum tissue resistance. Please note that singularities need careful treatment, which, however, goes beyond the scope of the chapter.

Next, the influence of robot mass and velocity for the unconstrained impact are described. This analysis is particularly important to understand Phase I.

Influence of Robot Mass and Velocity. Assume a simple mass–spring-mass model for the impact between human and robot. M_{H} is the reflected inertia of the human. K_{H} is the contact stiffness, which is in the case of a rigid robot mainly the effective stiffness of the human contact area. $\dot{x}^0_{\mathrm{re}}$ is the relative impact velocity between the robot and human. Solving the corresponding differential equation leads to the maximum contact force

$$\mathcal{F}^{\max}_{\mathrm{ext}} = \sqrt{\frac{m_u M_{\mathrm{H}}}{m_u + M_{\mathrm{H}}}}\sqrt{K_{\mathrm{H}}}\dot{x}^0_{\mathrm{re}} \,. \tag{69.8}$$

Assume a simplifying decoupling of the head from the torso, which holds for the short duration of the impact. For the post-impact phase, neck stiffness and body inertia have to be considered, which complicates the analysis considerably. The dependency of frontal bone contact force on the robot mass and velocity is depicted in Fig. 69.7a. It can be observed that collision force (which is a well-known bone fracture indicator) generally increases with velocity. For increasing mass, however, a saturation effect takes place. After a certain robot mass has been reached ($m_u \approx 20\,\mathrm{kg}$ in Fig. 69.7), additional weight has only negligible influence on collision force. This inertial saturation effect can also be observed for other impact locations, such as contacts with the chest (Fig. 69.7b).

If the robot mass is significantly larger than the human head mass, that is, $m_u \gg M_{\mathrm{H}}$, (69.8) reduces to

$$\mathcal{F}^{\max}_{\mathrm{ext}}(m_u \gg M_{\mathrm{H}}) = \sqrt{K_{\mathrm{H}} M_{\mathrm{H}}}\dot{x}^0_{\mathrm{re}} \,. \tag{69.9}$$

This shows that for a robot with significantly larger reflected inertia than the human head, only the contact stiffness, the impact velocity, and the mass of the human head are relevant but not the robot mass.

The behavior of human tissue during collisions is complex. Consequently, surrogates cannot reveal the entire diversity. Accordingly, the conduction of human voluntary experiments is necessary to fully understand human injury and pain dynamics in robotics.

Human–Robot Impact Voluntary Testing

The following experimental test was the first systematic analysis in this direction. The voluntary experiments were conducted with a healthy young adult in the year 2011. The collision experiments were performed with the KUKA/DLR lightweight robot (LWR) and the following approaches to injury and pain analysis were carried out: *injury severity analysis according to AO* (Arbeitsgemeinschaft für Ostheosynthesefragen), *biomechanical analysis*, *pain*, and *imaging methods*. The setup and experiment steps are depicted in Fig. 69.8. The robotic system allows to conduct controlled robot–human collisions in order to analyze input parameters and their effect on output parameters, such as pain and injury. Measured impact characteristics and quantities included impact force, impact area, tissue displacement, tissue stiffness, stress, impact velocity, kinetic energy, and energy density. The reflected inertia

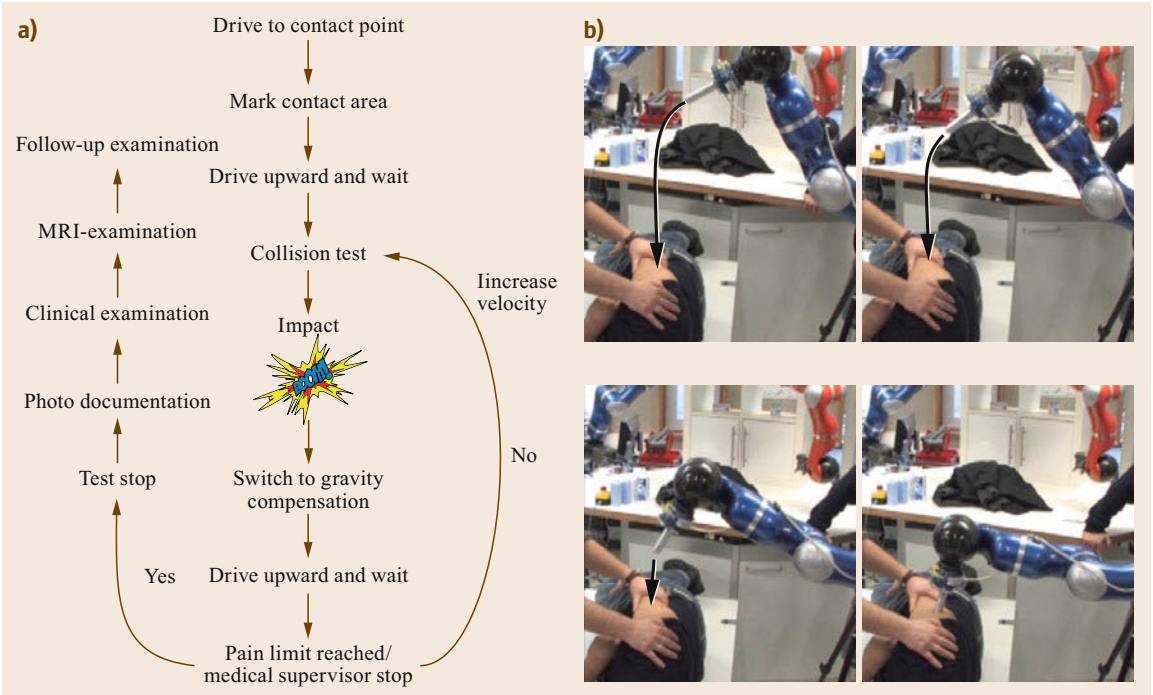


Fig. 69.8 **(a)** Flow chart depicting the basic experimental steps. **(b)** Collision trajectory with subject

was kept constant at $m_u = 3.75\,\mathrm{kg}$ for every test. The used impactor for the resulting Table 69.5 was a sphere with a radius of 12.5 mm.

The injury was defined using the AO-classification [69.48] directly after each test series. Each impact series was carried out at the same location on the human body at increasing impact velocity until the participant initiated a controlled system stop during the experiment. The impact areas were then imaged with a magnetic resonance imaging (MRI) after a time interval of about 4–5 h. The remaining tissue did not show any pathological signs. Compared to an equivalent drop test in [69.25] with abdominal pig tissue (large sphere, 4.2 kg, 2.5 m/s), the voluntary experiments provide similar results in terms of injury severity. The maximum velocity of 2.55 m/s is at the border of inducing a contusion. Where there were no marks immediately after impact, a mild contusion formed at day 1. For the pain tolerance at a visual analog scale (VAS) of 6/10, an impact force of $F = 272.2\,\mathrm{N}$ was measured. The energy density appears to have the most significant correlation to pain.

Synopsis

An overview of the potential injury threats depending on the current state of the robot and the human, a classification of these mechanisms, governing factors of the particular process and possible injuries are depicted in Fig. 69.9. Physical contact can be divided into two fundamental subclasses: quasi-static and dynamic loading. Fundamental differences in injury severity and mechanisms are observed as well if a human is (partially) constrained or not, leading to the second subdivision. For the quasi-static case, it is differentiated between near-singular and nonsingular clamping as already outlined. The last differentiation separates injuries caused by blunt contact from the ones induced by tools or sharp surface elements.

Each class of injury is characterized by possible injuries (PI), worst-case factors (WCFs), and their worst-case range (WCR). WCFs are the main contributors to the worst case, such as maximum joint torque, the distance to singularity or the robot speed. The worst-case range indicates the maximum possible injury depending on the worst-case factors. In addition to the classification of injury mechanisms for each such class, suggestions for injury measures (IMs) are given as well. They are specific injury measures which are appropriate, useful for the classification and measurement of injury potentially occurring during the physical human–robot interaction. Please note that the list of injury measures is not necessarily complete, but these ones are certainly suitable to be applied to a more granular robotics injury analysis. This does not mean that criteria, such as the well-known head injury criterion (HIC), do not provide general insights; they are just not

Table 69.5 Impact data for the lateral surface of the right upper arm

Impact	Max. impact force (N)	Impact area (mm^2)	Displacement (m)	Tissue stiffness (N/m)	Stress σ (N/mm^2)	Impact velocity (m/s)	Kinetic energy (J)	Energy density (J/mm^2)	AO	VAS
1	9.5	966	0.03	316.7	0.001	0.2	0.08	0.0001	IC1MT1NV1	0
2	19	966	0.037	513.5	0.002	0.44	0.36	0.0007	IC1MT1NV1	0
3	38.1	966	0.044	865.9	0.039	0.65	0.80	0.0016	IC1MT1NV1	0
4	59.6	966	0.055	1083.6	0.062	0.88	1.45	0.003	IC1MT1NV1	0
5	81.4	966	0.058	1403.4	0.084	1.11	2.31	0.005	IC1MT1NV1	1
6	103.5	966	0.060	1725	0.107	1.34	3.37	0.007	IC1MT1NV1	1.5
7	128.1	966	0.064	2001.6	0.133	1.55	4.50	0.009	IC1MT1NV1	2
8	154.1	966	0.069	2233.3	0.16	1.76	5.81	0.012	IC1MT1NV1	3
9	186.4	966	0.069	2701.4	0.193	2.03	7.73	0.016	IC1MT1NV1	3
10	224.5	966	0.069	3253.6	0.253	2.24	9.41	0.019	IC1MT1NV1	4
11	272.2	966	0.077	3535.1	0.282	2.55	12.2	0.025	IC1MT1NV1	6

necessarily optimal to understand injury on a more differentiated lower-injury scale.

For example ① in Fig. 69.9 represents blunt clamping in the near-singular configuration (Fig. 69.9). Even for low-inertia robots, this situation could become dangerous and is, therefore, a possible serious threat with almost any robot on a fixed base within a (partially) confined workspace. Possible injuries are fractures and secondary injuries, for example, caused by penetrating bone structures or an injured neck if the trunk is clamped but the head is free. This would mean that the robot pushes the head further, while the trunk remains in its position. Another possible threat is shearing off a locally clamped human along an edge. Appropriate indices are, for example, the contact force and the compression criterion (CC) [69.49]. ③ in Fig. 69.9 represents the clamped blunt impact in nonsingular configuration. The injury potential is defined by the maximum actuation torque $\boldsymbol{\tau}_{\max}$ and can range from no injury to severe injury or even death for high-inertia (and torque) robots. The robot stiffness does not contribute to the worst case since a robot without collision detection would simply increase the motor torque to follow the desired trajectory. Therefore, robot stiffness only contributes to the detection mechanism by enlarging the detection time. Also, the contact force and CC are well suited to predict occurring injury. ⑧ in Fig. 69.9 denotes the unconstrained impact which was the first injury mechanism investigated in the robotics literature. This process is governed by the impact velocity and (up to a saturation value) by the robot mass. As shown in [69.22], even a robot of arbitrary mass cannot severely injure a human head by means of impact-related criteria from the automobile industry like the head injury criterion (HIC). However, fractures, for example, of facial bones are likely to occur but not all would be classified as a serious injury. Laceration by means of crushes and gashes are worth evaluating, especially with respect to service robotics. The contact force and CC are well-suited severity criteria for this class. In order to evaluate lacerations the energy density has to be considered.

The preceding overview is intended as a worst-case analysis for the described contact cases. The next step is to ask which actions can be taken against each particular threat [69.19]. At this point, however, it shall be noted that instead of quantifying injury in terms of a measurable injury criterion, injury evaluation by a medical expert, for example, via the AO-classification can always be applied and would presumably result in a more exhaustive and precise judgement.

69.2.2 Safety Standards for Human–Robot Interaction

Robotics standardization made significant progress to establish the underlying regulations for co-working cells in the real world. Safety for industrial robots is addressed in a variety of general standards [69.50–52]. The most important industrial robotics standards is the International Organization for Standardization (ISO) 10218. It was established in the recognition of the particular hazards that industrial robots and industrial robot systems may pose. The machinery concerned and the extent to which hazards, hazardous situations, and events are covered are indicated in the scope of ISO 10218. In recognition of the variable nature of hazards with different uses of industrial robots, ISO 10218 is divided into two parts. It provides a detailed analysis of mechanical hazards, such as impacts, crushing, shearing, entanglement, drawing-in or tapping, cutting or severing, and contact of persons with live parts (direct contact) [69.53]. In particular, the introduction of collaborative robots has been a ma-

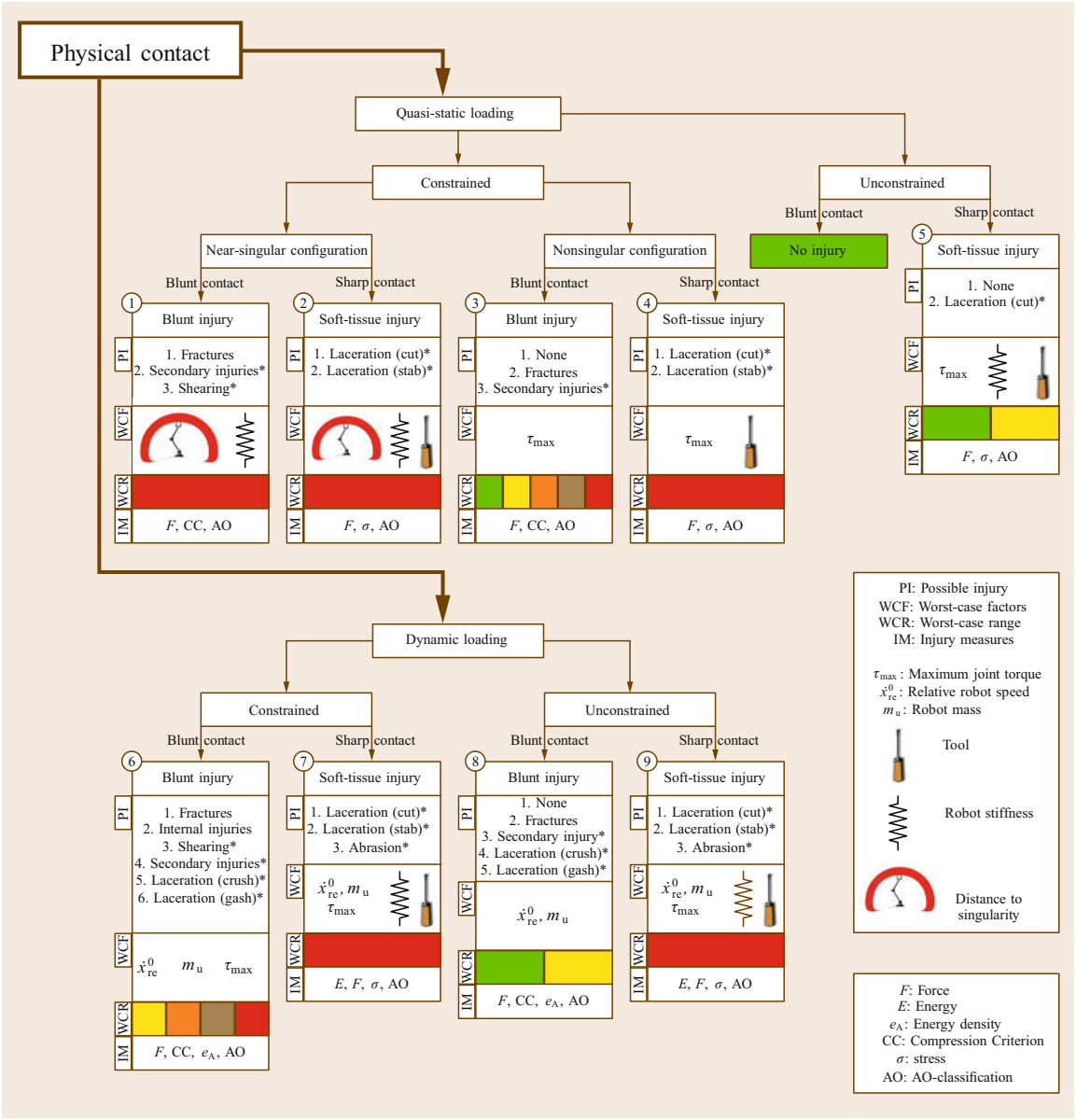


Fig. 69.9 Safety tree showing possible injury (PI), major worst-case factors (WCF) and the possible worst-case range (WCR). * indicates still ongoing topics of research. Additionally, relevant injury criteria are given for the head, chest, and soft-tissue injuries

jor acknowledgment to the advances made in robotics research in pHRI over the last decade. The recent updates to ISO 10218 (safety requirements for industrial robots) led to the development of the new technical specification (TS) 15066. It is regarded as a complementary information that concretizes the content of ISO 10218. Generally, ISO/TS 15066 provides guidance for collaborative robot operation where a robot and a person share the same workspace. It considers collaborative modes and requirements, such as minimum separation distances, safety-rated monitored stops, speed and separation monitoring, and power and force limiting. In collaborative operations, the integrity of the safety-related control system is of major importance, particularly when process parameters, such as speed and force, are being controlled. A comprehensive risk assessment is required to assess not only the robot system itself, but also the environment in which it is

placed, that is, in the workplace. A key process in the elimination of hazards and reduction of risks is the design of the collaborative robot system and the associated cell layout. Various considerations about the access and clearance of the collaborative workspace are provided. During the design of a robotic system, the maximum space and the restrictions of the collaborative robot system have to be considered. Furthermore, the need for clearances around obstacles and the accessibility for operators should influence the design. The intended contact(s) between parts of the robot system and an operator plays a major role toward a possibly intrinsically safe design. In order to identify the risks resulting from the collaborative action, an appropriate set of collision incidents that can occur during the collaborative work activities and foreseeable misuse has to be determined. This has to include affected body regions and the involved collision areas of the robot. The limit values that may not be exceeded during the collision incident depend on the affected body regions. The geometry of the involved areas of the robot and the biomechanical properties of the affected body regions influence the forces occurring during the collision incident. Therefore, the ISO/TS 15066 describes injury severity criteria that consist of maximum allowable limit values on individual body regions. These limit values are established to prevent the occurrence of skin/tissue penetrations that are accompanied by bleeding wounds, fractures, or other skeletal damage [69.54].

In addition to the industrial standardization efforts in the pHRI domain, the ISO 13482 [69.55] is the first nonindustrial robot safety standard that allows/regulates close pHRI. This international standard specifies requirements and guidelines for the inherent safe design, protective measures, and information for the use of so called *personal care robots*. It focuses on three types of personal care robots (mobile servant robots, physical assistant robots, and person carrier robots). These robots typically perform tasks to improve the quality of life of intended users irrespective of age or capability. The standard describes hazards associated with the use of these robots and provides requirements to eliminate or reduce the risks associated with these hazards to an acceptable level. Significant hazards are presented and this standard describes how they are to be dealt with for each personal care robot type. Robotic devices used in personal care applications are also covered by this standard and are to be treated as personal care robot.

69.3 Human-Friendly Robot Design

Designing robots for interaction has become a challenging subdomain in pHRI, leading to novel devices that have one thing in common: active and/or passive compliance together with lightweight design being the central design paradigms. A number of research-focused robots have been designed specifically for pHRI. The most important design guidelines, representatives, and modeling basics are outlined in this section. Apart from robots that are designed to act as general purpose co-workers, large robot assists with balanced, inertia reducing, cable-driven gantry systems can be used to create large payload robots that may be operated next to human workers [69.56].

69.3.1 Lightweight Design

In the process of making robots inherently suitable for close physical interaction with humans or only partially known environments, a design paradigm shift moving away from heavy, stiff, and rigid designs toward lightweight and highly integrated mechatronics designs has taken place. Low inertia and high (active) compliance have become desirable features, as has the use of redundant sensing principles on the proprioceptive level (position, velocity, and torque).

General Characteristics

Generally, two major design approaches for lightweight robots have proven successful over the last years [69.57], namely the *mechatronics approach* and the *tendon-based approach*, respectively. Their commonalities are as listed:

- *Lightweight structures*: Lightweight, high-strength metals, or composite materials for the robot links. Moreover, the design of the entire system (controllers, power supply) is optimized for weight reduction to enable mobility.
- *Low power consumption*: This is mainly achieved by small moving inertias and accordingly designed motors.

Typically, *mechatronic robots* integrate electronics into the joint structure for allowing highly modular units. Such a design enables the assembly of different kinematics with increasing complexity, while keeping the respective joint philosophy. In terms of actuation,

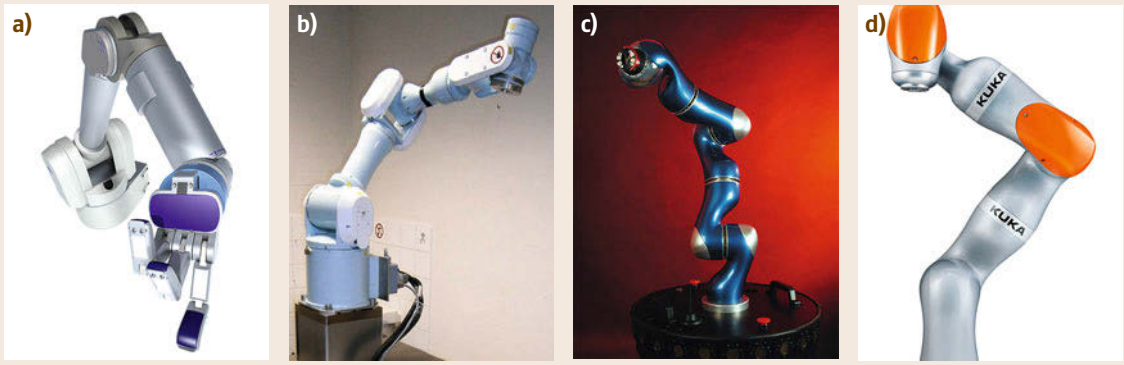


Fig. 69.10 **(a)** Barrett arm (after [69.58]), **(b)** Mitsubishi PA10 arm, **(c)** DLR lightweight robot III (after [69.59]), **(d)** KUKA LBR iiwa (after [69.60]) (courtesy of Barret Technology Inc., DLR, KUKA)

the combination of high power/torque motors with high transmission ratio gears is usually applied. From the proprioceptive sensing side, these systems are typically equipped with additional sensors, such as joint torque, force, and current sensing in addition to basic motor side position and current sensing only (Sect. 69.3.3).

Tendon-based robots have three major characteristics. First, they are typically equipped with remote direct drives. This reduces the overall weight of moving parts for a fixed base manipulator as the actuators are located in the robot base. For placing the actuators remotely from the base, cable-pulley systems are usually applied. Finally, low reduction ratios are used for keeping the system backdrivable. In turn, larger motors have to be selected, which adds additional total weight.

Another interesting class of compliant actuators is of note that implements compliance quite differently. They use rheological fluids such that they can alter their characteristic properties under the influence of either a magnetic or an electric field. A clutch between link and motor is operated in such a way that the output torque can be controlled. These actuators were also motivated and discussed from a human-safety perspective in [69.61].

Lightweight Robotic Systems

The most prominent robots that fall into the category of lightweight robots are depicted in Figure 69.10. The *Barrett arm* is a classic example of a *tendon-based design*, where the actuators are placed in the manipulator base and the joints are backdrivable due to the low reduction ratio. The *Mitsubishi PA10* arm was a commercially available lightweight redundant arm, with a weight of 38 kg and a payload of 10 kg. The fully torque-controlled *KUKA LBR iiwa* is based on the DLR lightweight robot technology [69.60]. Its third generation DLR LWR-III weighs 13.5 kg and is able to handle loads up to 15 kg, so an approximate unitary payload-to-weight ratio is achieved. The robot is equipped with joint torque sensors in each joint and has redundant position measurement (on motor and link side) [69.62].

Apart from single-arm robots as described above, various lightweight designs were also successfully integrated into research and commercial humanoid systems

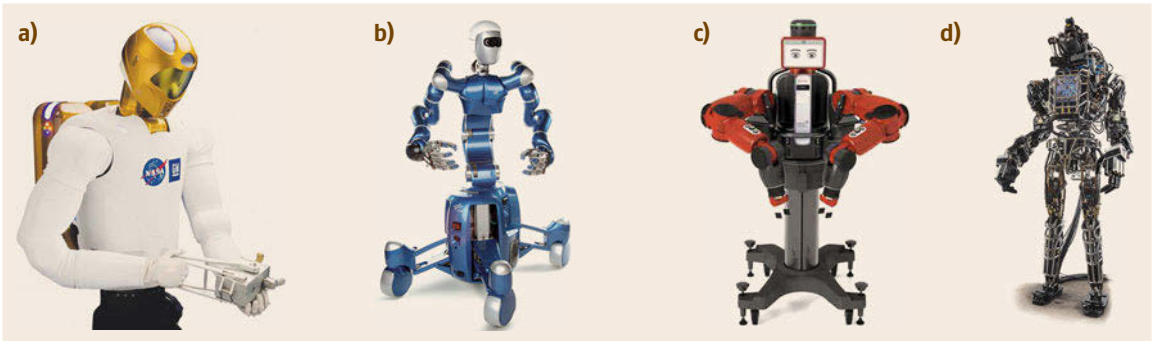


Fig. 69.11 **(a)** NASA *Robonaut 2*, **(b)** DLR *Rollin' Justin*, **(c)** Rethink Robotics *Baxter* and **(d)** Boston Dynamics *Atlas* (courtesy of NASA, DLR, Rethink Robotics Inc., Boston Dynamics)

(Fig. 69.11). The NASA (National Aeronautics and Space Administration) *Robonaut 2* [69.63], equipped with SEA, was originally designed for teleoperation and exploration in space [69.64]. The fully torque-controlled humanoid robot TORO [69.65], based on the DLR lightweight robot technology, has its origins in the upper body bimanual system *Justin* [69.66] (VIDEO 626). Rethink Robotics *Baxter* is a commercial two-arm upper body system for pick and place tasks that is equipped with SEA for torque measurement purposes. The hydraulically actuated humanoid, *Atlas* from Boston Dynamics, is one of the systems that entered the *DARPA (Defense Advanced Research Projects Agency) Robotics Challenge*.

Modeling Lightweight Robots

For a robot with n viscoelastic joints, the so-called reduced model of *Spong* [69.67] has become the standard way of modeling lightweight robots. When including also the presence of joint torques due to contact forces (on the link dynamics), we shall consider the following dynamic model of robots with viscoelastic joints:

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}_{\mathrm{J}} + \boldsymbol{\tau}_{\mathrm{ext}}\,, \tag{69.10}$$

$$\mathbf{B}\ddot{\boldsymbol{\theta}} + \boldsymbol{\tau}_{\mathrm{J}} = \boldsymbol{\tau}\,. \tag{69.11}$$

The generalized coordinates are doubled since there will be a dynamic displacement $\boldsymbol{\delta} = \boldsymbol{\theta} - \boldsymbol{q}$ between the motor positions $\boldsymbol{\theta} \in \mathbb{R}^n$, as reflected through the gear ratios, and the link positions $\boldsymbol{q}$. The matrix $\mathbf{B} = \mathrm{diag}(B_i) \in \mathbb{R}^{n\times n}$ is the diagonal, positive definite motor inertia matrix. We define the *elastic joint torque* transmitted through the joints and coupling (69.10) and (69.11)

$$\boldsymbol{\tau}_{\mathrm{J}} = \mathbf{K}_{\mathrm{J}}(\boldsymbol{\theta} - \boldsymbol{q}) + \mathbf{D}_{\mathrm{J}}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}})\,, \tag{69.12}$$

where $\mathbf{K}_{\mathrm{J}} = \mathrm{diag}(K_{\mathrm{J},i}) \in \mathbb{R}^{n\times n}$ is the diagonal, positive definite joint stiffness matrix, and $\mathbf{D}_{\mathrm{J}} = \mathrm{diag}(D_{\mathrm{J},i}) \in \mathbb{R}^{n\times n}$ is the diagonal, positive semidefinite joint damping matrix. The quantity $\boldsymbol{\tau}_{\mathrm{J}}$ in (69.12) is also the output of joint torque-sensing devices, when available. In many practical cases, the mechanical design of the transmission/reduction elements is such that one can neglect the joint damping, that is, $\mathbf{D}_{\mathrm{J}} \simeq \mathbf{0}$. Joint elasticity has long been addressed for lightweight robot systems, however, more as an undesired consequence that the control has to handle [69.68]. This requires advanced control techniques in order to obtain accurate, performant motion. For a complete overview on the properties of robots with joint elasticity effects, please refer to Chap. 11.

69.3.2 Intrinsically Flexible Design

Recently, the class of intrinsically flexible actuators and robots have become increasingly popular. Inspired by the flexible properties in biological muscles, compliant joints are designed with the aim of imitating human or animal motions during various tasks. The main idea of intrinsically flexible actuation is to come closer to human capabilities in terms of speed and shock absorption. This is not realizable with todays rigid industrial robots, when assuming approximately the same torque range or weight as humans have. By storing and releasing energy in the joints, one aims for improvement in tasks, such as running or throwing [69.69–72] where humans still clearly outperform robots. In Chap. 21 and [69.73], a more in depth review of intrinsically flexible actuation is presented. In this chapter, we review only the relevant insights for placing the technology in the pHRI context.

General Characteristics

Roughly speaking, intrinsically flexible actuators can be divided into two categories:

- Actuators with fixed mechanical impedance, where the effective joint impedance is altered via active control. The most well-known example is the series elastic actuator (SEA) [69.74], whose acronym became a generic term for this class of actuators.
- Actuators where the impedance can be adjusted by altering mechanical joint properties, such as stiffness and damping. There exist various categories like variable stiffness actuators (VSAs) that allow stiffness changes or variable impedance ac-

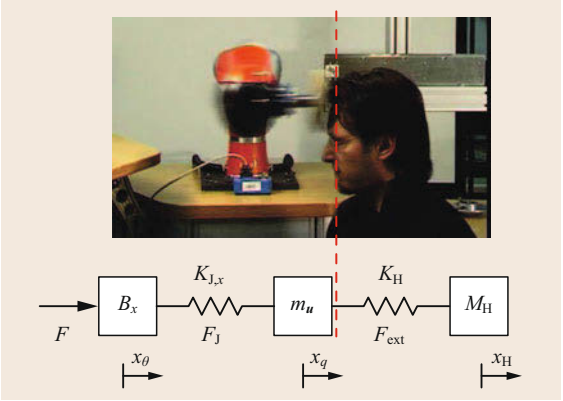


Fig. 69.12 Human–robot collision in operational space, which is defined by the reflected flexible dynamics of the robot and the local contact stiffness/mass properties of the human head

tuators (VIAs) that enable more general impedance changes, including damping adjustment.

The original motivation for introducing VSA and VIA, for example, in [69.20], was to make robots safer during unforeseen collisions due to dynamic decoupling of the motor and link-side inertia from each other. This effect reduces the collision danger by alleviating the impacting robot inertia. In [69.20], it is, for example, shown that HIC could be reduced by introducing elasticity in the joint. This idea was generalized and systematically analyzed in [69.75, 76], and is summarized here.

In order to simplify the human robot collision analysis basic models in operational space coordinates were used for this analysis (Fig. 69.12). For this, operational space coordinates and reflected inertias/stiffnesses along an arbitrary instantaneous impact direction $\boldsymbol{u}$ are considered [69.46]. The masses $M_{\mathrm{H}}, m_{\boldsymbol{u}}, B_x \in \mathbb{R}^+$ are the reflected human, link, and motor mass. $K_{\mathrm{H}}, K_{\mathrm{J},x} \in \mathbb{R}^+$ are the reflected (human) contact stiffness and joint/structural elasticity. The projected human, motor, and link impact position are denoted by x_{H}, x_θ, and $x_q \in \mathbb{R}$, respectively. $x_\theta = p_{\boldsymbol{u}}(T(\boldsymbol{\theta}))$ and $x_q = p_{\boldsymbol{u}}(T(\boldsymbol{q}))$ are defined via the respective forward kinematics maps projected in the $\boldsymbol{u}$-direction.

As pointed out in [69.20], a robot with quite low reflected link inertia $m_{\boldsymbol{u}} = 0.1\,\mathrm{kg}$ is able to reduce the impact forces significantly if a contact stiffness of $K_{\mathrm{H}} = 5\,\mathrm{kN/m}$ is assumed. Similar to the work in [69.21], it was shown that a decrease in joint stiffness can significantly reduce the impact characteristics and, thus, is a powerful countermeasure against large contact forces. In [69.77], it was deduced that for the case of a 2-DOF (degree of freedom) planar intrinsically compliant robot, already slightly touching a rigid wall with its second link, the compliant mechanism can limit the maximum static force/torque effectively if the motor torque is slowly increased. It is, of course, unquestionable that joint elasticity decouples the motor from the link. However, as was indicated in [69.22], a reduction in joint stiffness cannot reduce the impact characteristics during very rigid, fast, and blunt crash-test dummy impacts for a lightweight system, such as the LWR-III (which is basically an SEAs type robot from a modeling perspective). This was proven by measuring the decoupling of motor and link inertia via the integrated joint torque sensors and the additionally recorded external contact force.

Figure 69.13 depicts the experimental evidence for a collision at 1 m/s between a DLR LWR-III and a stationary unconstrained Hybrid III dummy (HIII) head on the frontal area [69.22]. The contact force f_{ext} was measured with a high-speed force sensor $f_{\mathrm{ext,fs}}$ and for consistency check with a triaxial accelerometer $f_{\mathrm{ext,as}}$. As one can see, the simulation and experimental signals show very good consistency. Clearly, the projected elastic joint force f_{J} reacts delayed to the collision, thus proving the desired decoupling property already for such an intrinsically very stiff robot. For the impact simulation, a reflected stiffness of $6.72 \times 10^4\,\mathrm{N/m}$ is chosen, which represents a realistic joint stiffness and structural elasticity value. The reflected motor inertia is $B_x = 13\,\mathrm{kg}$ and the reflected link side inertia $m_{\boldsymbol{u}} = 2\,\mathrm{kg}$. The dummy head mass is $M_{\mathrm{H}} = 4.5\,\mathrm{kg}$ and the contact stiffness is $K_{\mathrm{H}} = 3.2 \times 10^5\,\mathrm{N/m}$.

This result showed that already the compliances of the built in gear and the joint torque sensor is sufficient to decouple the motor from the link, making it entirely unnecessary to further reduce joint stiffness for the given robot for this purpose. There are two main aspects, which have to be considered to fully understand this result. On the one hand, the contact stiffness of the crash-test dummy used is significantly larger than the reflected elasticity of the DLR LWR-III; however, it is also realistic for the human frontal area [69.27]. Furthermore, the reflected motor and link inertia for the DLR LWR-III are the same order of magnitude as for the human head mass. This is quite reasonable for a full-scale lightweight robot arm.

Two further benefits of inherent compliance are as follows. First, the robot itself is able to resemble compliant behavior without the inherent need for high-performance force/torque feedback. Please note that it is not possible to display arbitrary Cartesian stiffness behavior for a robot with diagonal stiffness matrix only [69.79]. Second, the joint elasticity acts in combination with the link inertia as a mechanical low-

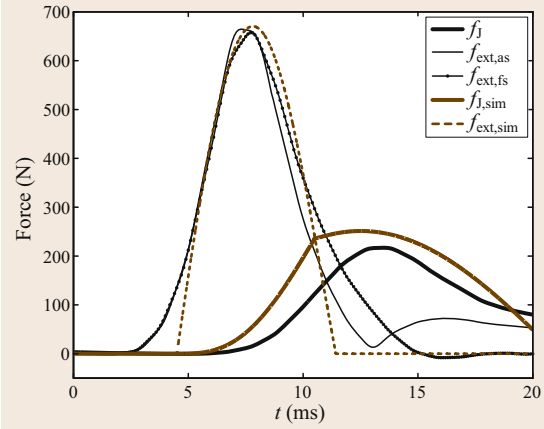


Fig. 69.13 Impact experiment and simulation of a LWR-III with a Hybrid III Dummy. The experimental contact force is measured for consistency check with a high-speed acceleration (as) and force sensor (fs) simultaneously

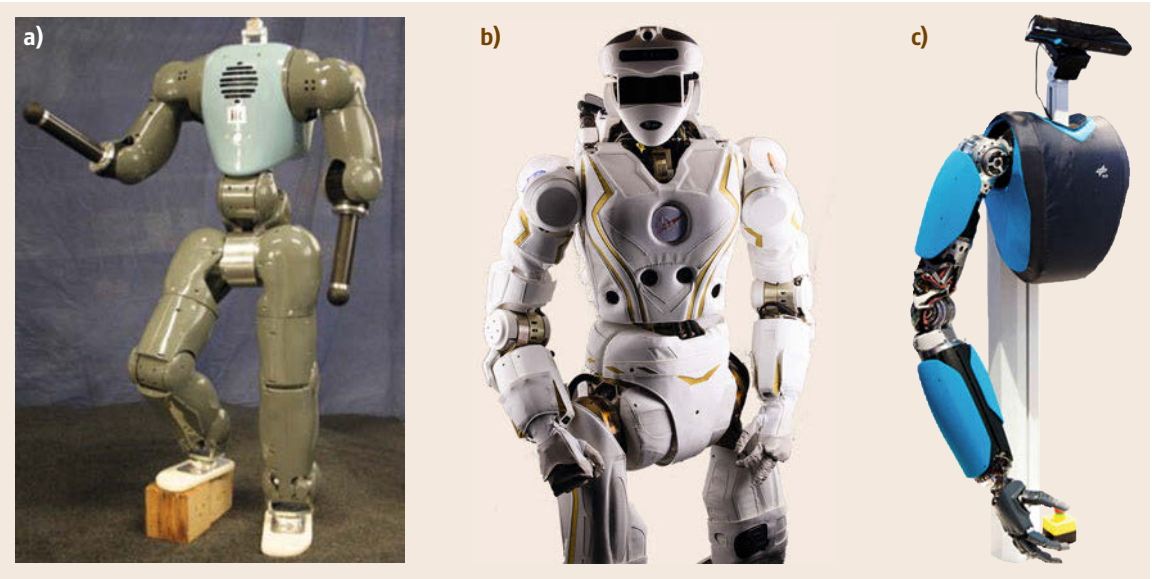


Fig.69.14a–c Passively flexible anthropomorphic robots: compliant humanoid platform (COMAN, **(a)**) from IIT (after [69.78]), Valkyrie **(b)** from NASA, and hand arm system (HASY, **(c)**) from DLR (courtesy of IIT, DARPA, DLR)

pass filter, thus protecting the drive train from impact shocks; that is, it makes the system inherently more robust [69.71, 80].

Another interesting use of elasticities in the drive train of a manipulator relates to the fact that it can serve as a storage mechanism for potential energy, and be used to increase a robot's link speed beyond the maximum motor velocity [69.70, 71, 80–82]. This ability strongly influences the safety of intrinsically elastic systems, as a robot's impact speed determines its inherent collision danger to a large extent. However, we refer to [69.75] for further details on this matter, where the effect of a robot's elastic speed increase on potential danger in terms of HIC is derived.

Intrinsically Flexible Robotic Systems

Figure 69.14 shows some of the most prominent recent anthropomorphic/humanoid devices that employ passively compliant actuators. The COMAN system of IIT (VIDEO 624) is a full-scale SEA robot, as is NASA's Valkyrie. Both system's ability to measure torques results from the ability to flex around their joint axes and then relate this elastic deflection to the joint torque. The DLR hand-arm system [69.72] is fully equipped with VSA in every joint, while having approximately human size, strength, and dexterity.

Modeling Passively Flexible Robots

From a formal modeling point of view, lightweight robots and robots with fixed mechanical impedance are very similar. The main difference lies in the respective torque sensor measurement principle. The joint elasticity in lightweight robots is rather high and allows only for small deflections between the motor and link position. The stiffness in compliant actuators, however, is intentional and typically at least an order of magnitude lower, thus enabling significantly larger deflections. Consider the elastic joint model (69.10) and (69.11) from Sect. 69.3.1, which under certain mild conditions also holds for intrinsically elastic robots. The main difference is that for lightweight robots, $K_{\mathrm{J},i}$ is large compared to *SEA-like* systems. For many VSA systems, the transmitted torque can be expressed as

$$\boldsymbol{\tau}_{\mathrm{J}} = \boldsymbol{f}(\boldsymbol{\delta}, \boldsymbol{\sigma}) \,, \tag{69.13}$$

where $\boldsymbol{\sigma}$ denotes a stiffness adjustment control input (typically the position of a second motor). The torque–displacement curve $\boldsymbol{f}(\boldsymbol{\delta}, \boldsymbol{\sigma})$ may exhibit different characteristics, such as being strictly increasing $\partial \boldsymbol{f}(\boldsymbol{\delta}, \boldsymbol{\sigma})/\partial \boldsymbol{\delta} > \mathbf{0}$ and convex $\partial^2 \boldsymbol{f}(\boldsymbol{\delta}, \boldsymbol{\sigma})/\partial \boldsymbol{\delta}^2 > \mathbf{0}$. More details on implementations can be found in [69.73].

69.3.3 Perception for Interaction

The recent advances in pHRI benefit greatly from previous achievements in contact and noncontact sensors and sensing techniques. In the current chapter, we shortly review the essential sensing techniques that had strong

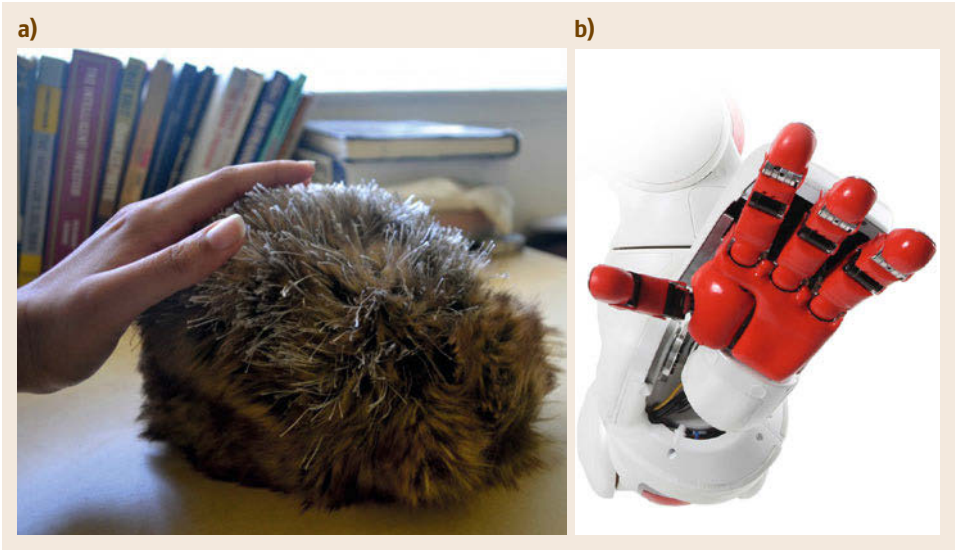


Fig. 69.15 **(a)** Furry robot *skin* utilizing piezo-resistive fabric and conductive sensors provides a low cost design concept with high tactile gesture recognitions rates (after [69.83]). **(b)** The hand of TWENDY-ONE with integrated distributed pressure sensors (courtesy of Karon MacLean, UBC, Shigeki Sugano Lab., Waseda University)

impact on pHRI and focus on the underlying main concepts. Chapters 28 and 32 give a full view and can be consulted for further details.

69.3.4 Proprioceptive Force/ Torque Sensing

Many robot arms have the option to include a six-axis force/torque sensor at the wrist, enabling not only force control-based manipulation, but also symbolic haptic interaction [69.84, 85], possibly also based on joint torque sensing. More recently, torque sensors have been integrated into the joints of commercial robots, for example, into the KUKA LWR [69.86]. This allows for contact sensing along the entire robot structure with measurements of contact magnitude, direction, and knowledge of which link was contacted [69.87].

Essentially, two major sensing principles for joint torque measurement exist: either by directly measuring the torque which is typically done via strain gauges, or by measuring it implicitly via the deflections δ between link and motor position. For this, it is assumed that δ directly relates to torque typically by a constant multiplier, the joint stiffness $K_{\mathrm{J},i}$. For the latter, both link- and motor-side position sensors need to be mounted. The first works that aimed at developing and minimizing torque-sensing devices can be found in [69.88–90]. The torque-sensing principle has also a strong relation to the design classification in Sect. 69.3.

69.3.5 Tactile Perception

Several tactile skins have been developed over the years such as, for example, [69.91–94]. In the latter work, for example, the human skin served as a design metaphor. Therein, a solution of the goal conflict between the desired high sensitivity and the required mechanical robustness is in the focus. Furthermore, beyond providing the required sensory capability, the mechanical deformation and damping properties necessary for the operation of the robotic system are considered as well. Using whole-arm tactile sensing across the manipulator surface explicitly allows the sensing of multiple contacts; for example, in [69.95], the authors operate a robot in a cluttered environment in order to successfully reach a goal that is occluded by obstacles using a model predictive controller. Other recent examples of localized sensors include touch pads placed at key points on the robot [69.96], and force-sensitive resistors distributed over the robot [69.97]. The latter example is used in a novel haptic robot designed to both read and display emotion mediated through cues sensed primarily through physical interaction. These methods have the advantage of using reliable and fairly precise sensors, but are capable of measuring contact only at discrete locations. For more extensive contact sensing, several researchers have developed various types of robot skin; that is a distributed sensor network that covers extensive areas and measures diverse qualities of contact events, including location, magnitude, orientation, and temperature. Hard robot skins or shells have been used to

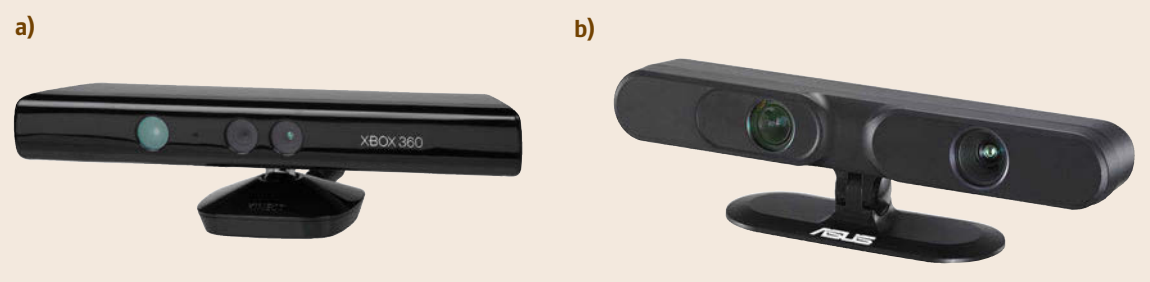


Fig. 69.16 Low cost 3-D-RGB-D cameras: **(a)** Microsoft Kinect (Wikipedia Commons), **(b)** Asus Xtion (courtesy Asus)

detect collisions [69.98], while soft robot skins that can conform to different robot shapes have been used for more complex, often social or affective, human–robot interaction, for example, [69.99]. While robot skin can add significant sensing capability to robots, it may also add to the complexity and cost, and thus compromises the robustness of the robot. However, recently a furry *creature* skin (Fig. 69.15) was created [69.83] using a combination of low-resolution piezo-resistive fabric, and conductive fur sensors to create a low-cost robot covering with 90% interaction gesture (stroking, patting, poking, hitting) recognition rates when trained on an individual and 68–80% recognition rates when trained on groups (VIDEO 615). A detailed review of tactile sensing technology for human–robot interaction can be found in [69.100]. Other examples can be found in Chap 28.

69.3.6 Visual Perception

Visual (noncontact) sensing is very important for preparing for pHRI. Tracking and planning for the location of a human partner and predicting, for example, the motion of the partner's hand is an important precursor to a successful handover operations [69.12]. While well-known marker-based systems, such as Vicon and Optotrak systems, provide very high resolution tracking systems, they are impractical for day-to-day use. The development of low-cost 3-D-RGB-D (three-dimensional red green blue depth) cameras (Fig. 69.16) permits the 3-D tracking of full body models in large and partially occluded spaces, with ongoing improvements toward robustness in body pose tracking and for tracking of hand gestures [69.101, 102].

69.4 Control for Physical Interaction

For soft and safe pHRI, the question arises how to gently handle physical contact in robotics from a control point of view. As impedance control [69.103] became the most popular interaction control paradigm in the pHRI world, this particular scheme will be one focus of this section. Its generalization to multipriority impedance control laws allows the realization of sophisticated robot compliance with multiple objectives via active control. A major advantage of impedance control is that discontinuities like contact–noncontact do not create stability problems as they occur, for example, with hybrid force control [69.104]. Its extension to impedance and feed-forward learning and adaptation, for which the first works can be found in [69.105, 106], is discussed after introducing the concept of multipriority impedance control. Apart from nominal interaction control, a robot sharing its workspace with humans and physically interacting with its environment should be able to quickly detect collisions and safely react to them. In the absence of external sensing, relative motions between robot and environment/human are unpredictable and unexpected collisions may occur at any location along the robot arm. The state-of-the-art schemes for collision detection and reflex reaction are introduced. Finally, the last part of the section deals with the shared manipulation problem, one of the standard application examples of pHRI.

69.4.1 Interaction Control

Originally developed for robust and compliant object manipulation, impedance and the related admittance control (VIDEO 610) form a paradigm to treat robotic systems from an energetic point of view such that motion and force can be controlled in a unified manner. They offer the advantage over hybrid force-motion controllers to provide a framework independent from kinematic work space constraints. These control types popularized by [69.103] are also especially advantageous in terms of uncertainties and disturbances in

unknown environments due to their inherently robust nature [69.107]. The terms impedance and admittance, are derived from electrical system theory where they describe the relationship between voltage and current as input/output pairs. To generalize impedance and admittance such pairs can be defined domain-independently as effort and flow variables. For robotics, the mechanical analogies, that is, mechanical impedance and admittance are of particular interest.

More details on the conceptual basics of impedance and the dual admittance control can, for example, be found in [69.108, 109]. Furthermore, Chap. 9 gives a more thorough basis on force control strategies in general and impedance control in particular.

The mostly used version of impedance control is to impose a second-order dynamics of a mass–spring–damper system (so-called target impedance [69.103]) on the closed-loop equations. Typically, the control objective is expressed in operational space coordinates $\boldsymbol{x}$ as

$$\mathbf{M}_x\ddot{\tilde{\boldsymbol{x}}} + \mathbf{D}_x\dot{\tilde{\boldsymbol{x}}} + \mathbf{K}_x\tilde{\boldsymbol{x}} = \boldsymbol{\mathcal{F}}_{\text{ext}}\,, \tag{69.14}$$

where $\tilde{\boldsymbol{x}} := \boldsymbol{x} - \boldsymbol{x}_{\text{d}}$ is the position error and $\boldsymbol{x}_{\text{d}}$ is called equilibrium position. $\mathbf{M}_x$ denotes the desired inertia, while $\mathbf{D}_x$ and $\mathbf{K}_x$ are the according closed-loop damping and stiffness matrices in operational space. $\boldsymbol{\mathcal{F}}_{\text{ext}}$ denotes the external wrench acting at the end-effector of the robot. Assuming rigid body dynamics, the control law to obtain the aforementioned behavior is

$$\begin{aligned}\boldsymbol{\tau}_{\text{C}^*} = {} & \boldsymbol{g}(\boldsymbol{q}) + \mathbf{J}(\boldsymbol{q})^{\text{T}}[\boldsymbol{\Lambda}(\boldsymbol{q})\ddot{\boldsymbol{x}}_{\text{d}} + \boldsymbol{\mu}(\dot{\boldsymbol{x}}, \boldsymbol{x})] \\ & - \mathbf{J}(\boldsymbol{q})^{\text{T}}\left[\boldsymbol{\Lambda}(\boldsymbol{q})\mathbf{M}_x^{-1}(\mathbf{K}_x\tilde{\boldsymbol{x}} + \mathbf{D}_x\dot{\tilde{\boldsymbol{x}}})\right] \\ & + \mathbf{J}(\boldsymbol{q})^{\text{T}}\left[\boldsymbol{\Lambda}(\boldsymbol{q})\mathbf{M}_x^{-1} - \mathbf{I}\right]\boldsymbol{\mathcal{F}}_{\text{ext}}\,. \end{aligned} \tag{69.15}$$

In order to fully implement this scheme, a wrist force torque sensor is necessary for the inertia shaping part. In [69.110], a modified impedance controller was designed that uses angle/axis representations for the rotational components of the operational space. For its derivation, energy contributions with physical interpretation are considered and the end-effector orientation displacement representation is chosen to be in terms of a unit quaternion to avoid singularities.

For redundant robots, it is typically desired to also control the nullspace behavior in order to embed other control objectives $\boldsymbol{\tau}_{N,i}$ into a stacked hierarchy of tasks (Chaps. 17 and 36). For the case of a single nullspace controller $\boldsymbol{\tau}_N$, this torque has to be projected via the nullspace projector matrix $\mathbf{N}(\boldsymbol{q})$ into the nullspace of the task, leading to the overall control law

$$\boldsymbol{\tau}_{\text{C}} = \boldsymbol{\tau}_{\text{C}^*} + \mathbf{N}(\boldsymbol{q})^{\text{T}}\boldsymbol{\tau}_N\,. \tag{69.16}$$

The nullspace projection matrix can be chosen in different ways. The simplest case is $\mathbf{N}(\boldsymbol{q}) = \mathbf{I} - \mathbf{J}(\boldsymbol{q})^{\#}\mathbf{J}(\boldsymbol{q})$, where $\mathbf{J}(\boldsymbol{q})^{\#}$ denotes the Moore–Penrose pseudoinverse. Alternatively, one may chose the *dynamically consistent* generalized pseudoinverse

$$\mathbf{J}(\boldsymbol{q})^{\#} = \mathbf{M}(\boldsymbol{q})^{-1}\mathbf{J}(\boldsymbol{q})^{\text{T}}\boldsymbol{\Lambda}(\boldsymbol{q})\,. \tag{69.17}$$

In particular, in the pHRI domain, a multitude of different subtasks $\boldsymbol{\tau}_{N,i}$ are meaningful to be executed simultaneously. These may, for example, involve:

- Safety (collision anticipation and avoidance, self-collision avoidance, ...)
- Physical constraints (joint limits, geometric task constraints)
- Task execution (tracking control, ...)
- Posture primitives (in particular for humanoids).

To realize consistent behaviors, task hierarchies are constructed such that certain tasks are prioritized over others [69.111]. In [69.112], a hierarchy is realized by null space projection techniques, which also prevents discontinuities concerning unilateral constraints by smoothing out transitions.

Extensions to the basic schemes for flexible joint dynamics [69.113–115] and for the SEA case [69.116] were developed as well. Furthermore, Cartesian impedance control has been applied to grasping and multiple-arm robotic systems in [69.117]. For a more in depth treatment of multipriority impedance control, please also refer to Chap. 67.

69.4.2 Learning and Adaptation

Close physical interaction between human and robot is a complex evolving process with high uncertainty and is hard to be modeled explicitly. Therefore, several learning and adaptation approaches were proposed to enhance a robot's capability and account for the inherently present uncertainty and unpredictability. In this sense, the extension of impedance control toward adaptive controllers that are able to learn and/or adapt the controller's impedance and feed forward torques is a challenging and rather recent research problem. Obviously, also the learning of the desired trajectories, for example, in terms of motion patterns became an important problem in this context. Generally, also collaborative tasks (Sect. 69.4.4) require learning either the impedance properties and/or certain trajectories such that the robot gains the ability to adapt its behavior to the human counterpart, or even guides it to guarantee successful collaboration. An important aspect of learning physical interactions is how to choose the right task coordinates and (meta-)parameters. This is essential to reformulate the otherwise high-dimensional problem spaces in a tractable form.

Since impedance control has already shown to be a valuable technique in cluttered and complex manipulation tasks, recent research focuses on the adaption of the impedance characteristics to further improve the robot capabilities during interaction. Iterative learning control techniques belong to the first methods that have been investigated to tackle difficult manipulation problems [69.118, 119]. However, they do not only require the exact same repetitive motion of the manipulator, but also to account for unforeseen changes in the environment due to force inconsistencies which are particularly present in pHRI. Other early approaches include, for example, the use of neural networks in impedance control to counteract disturbances and environmental uncertainties [69.120]. An approach to adapt force, trajectory, and impedance simultaneously has been presented in [69.121] as a biomimetic controller. It is constructed based on studies in neuroscience that show that humans adapt feedfoward and feedback forces as well as their impedance in order to learn unstable tasks in daily life. According to [69.122], the principles of motor learning are:

1. Motor commands to perform a desired action are composed of both feedforward commands, defined as the component of the motor command learned by repeating an activity, and feedback commands.
2. Learning is performed in muscle space.
3. Feedforward increases with the muscle stretch in previous trial.
4. Feedforward also increases with antagonist muscle stretch.
5. Feedforward decreases when the error is small.

Mathematically, these principles can be expressed as follows [69.105]. Generally, the central nervous system tends to minimize the motion error as well as the metabolic cost such that no extra effort will be spent on the learned impedance and feedforward torques. This can be expressed by minimizing the joint-level cost function

$$V(t) = \frac{1}{2}\int_{t-T}^{t} \tilde{\boldsymbol{\Phi}}^{\mathrm{T}}(\sigma)\mathbf{Q}^{-1}\tilde{\boldsymbol{\Phi}}(\sigma)\mathrm{d}\sigma + \frac{1}{2}\boldsymbol{\epsilon}(t)^{\mathrm{T}}\mathbf{M}(\boldsymbol{q})\boldsymbol{\epsilon}(t). \tag{69.18}$$

Therein, $\mathbf{Q}$ is a positive definite weighting matrix corresponding to the learning rates; $\tilde{\boldsymbol{\Phi}}$ is defined as the difference between the instantaneous value

$$\boldsymbol{\Phi} = \left[\mathrm{vec}(\mathbf{K}_q(t))^{\mathrm{T}}, \mathrm{vec}(\mathbf{D}_q(t))^{\mathrm{T}}, \boldsymbol{\tau}_{\mathrm{ff}}(t)^{\mathrm{T}}\right]^{\mathrm{T}} \tag{69.19}$$

and the required optimal value $\boldsymbol{\Phi}^*$, that is, $\tilde{\boldsymbol{\Phi}} = \boldsymbol{\Phi} - \boldsymbol{\Phi}^*$. The quantities $\boldsymbol{\epsilon}$, $\mathbf{K}_q$, $\mathbf{D}_q$, and $\boldsymbol{\tau}_{\mathrm{ff}}$ denote the tracking control error [69.123] and the values to be learned, namely, the closed loop joint space stiffness, damping, and feed forward motor torques. For the joint stiffness adaption law, this leads for example to

$$\begin{aligned}\delta\mathbf{K}_q(t) &= \mathbf{K}_q(t) - \mathbf{K}_q(t-T)\\ &= \mathbf{Q}_{\mathbf{K}_q}\left[\boldsymbol{\epsilon}(t)\boldsymbol{e}(t)^{\mathrm{T}} - \gamma(t)\mathbf{K}_q(t)\right],\end{aligned} \tag{69.20}$$

where $\gamma(t) > 0$ is a constant forgetting factor and $\mathbf{Q}_K$, which is contained in $\mathbf{Q}$, is a symmetric positive definite matrix corresponding to the learning rate of the stiffness. T is a time constant to denote task: periodicity. In an analog manner, adaption for feedforward torque and damping can be deduced, which results in the following control law,

$$\begin{aligned}\boldsymbol{\tau}_{\mathrm{C}}(t) &= \boldsymbol{\tau}_{\mathrm{ff}}(t) - \mathbf{K}_q(t)\boldsymbol{e}(t) - \mathbf{D}_q(t)\dot{\boldsymbol{e}}(t)\\ &\quad - \mathbf{L}(t)\boldsymbol{\epsilon}(t) + \boldsymbol{\tau}_{\mathrm{r}}(t)\end{aligned} \tag{69.21}$$

$$\boldsymbol{\epsilon} = \dot{\boldsymbol{e}}(t) + \kappa\boldsymbol{e}(t)\,, \tag{69.22}$$

where $\boldsymbol{\tau}_{\mathrm{r}}(t)$ compensates for robot/arm dynamics and bounded noise. The term $\mathbf{L}(t)\boldsymbol{\epsilon}(t)$ ensures a certain stability margin. The overall control law flow chart is depicted in Fig. 69.17. A stability proof of this approach can be found in [69.105]. An open problem with this method that remains to be unsolved is how to automatically select the meta parameters, such as the forgetting factor γ. A similar approach was developed in [69.124], where impedance adaptation laws were tested on a two-robot collaboration task. The robots lift together a beam after a task model was learned. A different approach based on the perturbations of the robot imposed kinesthetically by a human teacher such that the stiffness of the robot is adapted is investigated in [69.125]. Finally, the so-called teleimpedance paradigm [69.126] aims to remotely controlling the robot via the human arm reference position and impedance.

Learning task trajectories has been a rather well-studied field for several years now. For example, in [69.127], the authors examined the task of learning collaborative lifting of an object using hidden Markov models (HMM) and Gaussian mixture regression. Task demonstration was done via a haptic interface that controlled the robot hand. HMMs are also used in [69.128] in order to learn semantic task structures during a joint task execution. A semantic label of recognized task segments is acquired from a human partner by using speech recognition. Tasks in pHRI where transitions from contact to noncontact situations arise have been examined in [69.129]. Therein, marker data is used to obtain motion primitives. These are then updated with contact time information, comprising a low-level layer of learning. In addition, there is a high-level layer, which selects

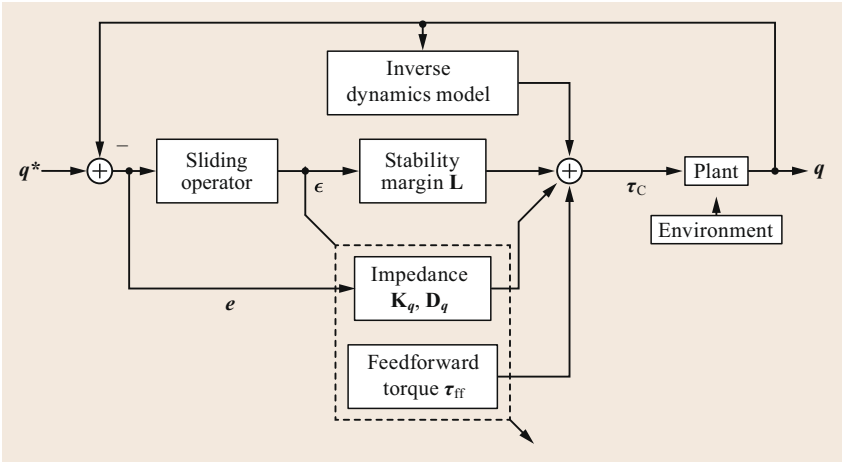


Fig. 69.17 Control diagram of the biomimetic adaptive impedance controller from (after [69.105])

a motion primitive and an end-effector reference position. Again, both layers are encoded via HMMs to provide a suitable abstraction, which is a key when dealing with high-dimensional spaces.

In [69.17], dimensionality reduction for learning was employed to realize assistive tasks, such as helping a robot to stand–up or walk. This principal component analysis (PCA)-based reduction lets the learning take place in a low-dimensional joint manifold instead of the original high-dimensional space. It reduces the complexity of the learning domain for multidegree-of-freedom systems such as humanoids. A recent approach to learning trajectories based on human preferences was proposed in [69.130], where machine-learning techniques were applied to derive the optimal behavior from human action ratings during runtime. Figure 69.18 shows how the robot learns to handle a knife safer by receiving low rewards from the human if the dangerous tool is moved in undesired orientations and/or trajectories.

69.4.3 Collision Handling

One of the core problems in pHRI is the handling of collisions between robots and humans, with the primary motivation of limiting possible human injury due to physical contacts. Various *monitoring signals* can be used to gather context independent information about the event.

The *collision detection phase*, whose binary output denotes whether a robot collision occurred or not, is characterized by the transmission of contact wrenches, often for very short impact durations. The occurrence of a collision, which may happen anywhere along the robot structure, shall be detected as fast as possible. A major practical problem is the selection of a threshold on the monitoring signals, so as to avoid false positives and achieve high sensitivity at the same time. A heuristic approach is to monitor the measured currents in robot electrical drives, looking for fast transients possibly caused by a collision [69.131, 132]. Another proposed scheme compares the actual commanded torques (or motor currents) with the nominal model-based control law (i. e., the instantaneous torque expected in the absence of collision), with any difference being attributed to a collision [69.133]. This idea has been refined by considering the use of an adaptive compliance control [69.134, 135]. However, tuning of collision detection thresholds in these schemes is difficult because of the highly varying dynamic characteristics of the control torques.

Knowing which robot part (e.g., which link of a serial manipulator) is involved in the collision is an important information that can be exploited for robot

Fig. 69.18 The robot Baxter learns to safely handle a knife ([69.130]; courtesy of Saxena's Robot Learning Lab)

reaction. *Collision isolation* aims at localizing the contact point $\boldsymbol{x}_c$, or at least which link i_c out of the n-body robot collided. One way to obtain both collision detection and isolation is to use sensitive skins [69.91–94]. However, it would obviously be more practical and reliable to detect and possibly isolate a collision without the need of additional tactile sensors. On the other hand, the previously mentioned monitoring signals used in [69.131–135] are in general not able to achieve reliable collision isolation (even when robot dynamics is perfectly known). In fact, they either rely on computations based only on the nominal desired trajectory, or compute joint accelerations by inverting the mass matrix and thus spreading the dynamic effects of collision on a single link, or use acceleration estimates for torque prediction and comparison, which inherently introduces noise (due to double numerical differentiation of position data) and intrinsic delays. The common drawback of these methods is that the effect of a collision on a link propagates to other link variables or joint commands due to robot dynamic couplings, thus affecting the isolation property.

Other relevant quantities about a collision that are deduced during the *collision identification phase* are the directional information and the intensity of the generalized collision force, either in terms of the acting Cartesian wrench $\boldsymbol{\mathcal{F}}_{\text{ext}}(t)$ at the contact, or of the resulting joint torque $\boldsymbol{\tau}_{\text{ext}}(t)$ during the entire physical interaction event. This information characterizes (in some cases, completely) the collision event. The first method that simultaneously achieved collision detection, isolation, and identification was proposed in [69.136]. The basic idea was to view collisions as faulty behaviors of the robot actuating system, while the detector design took advantage of the decoupling property of the robot *generalized momentum* $\boldsymbol{p} = \mathbf{M}(\boldsymbol{q})\dot{\boldsymbol{q}}$ [69.137, 138].

During the *collision reaction phase*, the robot should react purposefully in response to a collision event, that is, taking into account available contextual information. Because of the fast dynamics and high uncertainty of the problem, the robot reaction should be embedded in the lowest control level. For instance, the simplest reaction to a collision is to stop the robot. However, this may possibly lead to inconvenient situations, where the robot is unnaturally constraining or blocking the human [69.19]. To define better reaction strategies, information from collision isolation, identification and classification phases should be used. Some examples of successful collision reaction strategies have been given in [69.87, 139].

Collision Detection and Identification

A recent overview on standard techniques to estimate $\boldsymbol{\tau}_{\text{ext}}$ can be found in [69.26]. In this chapter, we focus on the main method, namely the monitoring method based on the observation of the generalized momentum that was introduced in [69.136]. The scheme, which is regarded as the standard algorithm, was motivated by the desire of avoiding the inversion of the robot inertia matrix, decoupling the estimation result, and also eliminating the need of an estimate of joint accelerations. The according disturbance observer-based estimator for the rigid case is defined as

$$\boldsymbol{r}(t) = \mathbf{K}_{\text{O}} \left\{ \hat{\boldsymbol{p}}(t) - \int_0^t [\boldsymbol{\tau} - \hat{\boldsymbol{\beta}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{r}] \mathrm{d}s - \hat{\boldsymbol{p}}(0) \right\} , \tag{69.23}$$

with $\hat{\boldsymbol{p}} = \hat{\mathbf{M}}(\boldsymbol{q})\dot{\boldsymbol{q}}$, $\hat{\boldsymbol{\beta}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \hat{\boldsymbol{g}}(\boldsymbol{q}) + \hat{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \dot{\hat{\mathbf{M}}}(\boldsymbol{q})\dot{\boldsymbol{q}}$, and $\mathbf{K}_{\text{O}} = \text{diag}(k_{\text{O},i}) > 0$ being the diagonal gain matrix of the observer. Under ideal conditions, $\hat{\mathbf{M}} = \mathbf{M}$ and $\hat{\boldsymbol{\beta}} = \boldsymbol{\beta}$, the dynamic relation between the external torque $\boldsymbol{\tau}_{\text{ext}}$ and $\boldsymbol{r}$ is

$$\dot{\boldsymbol{r}} = \mathbf{K}_{\text{O}}(\boldsymbol{\tau}_{\text{ext}} - \boldsymbol{r}) . \tag{69.24}$$

In other words, $\boldsymbol{r}$ is a stable, linear, decoupled, first-order estimation of the external collision torque $\boldsymbol{\tau}_{\text{ext}}$. Large values of $k_{\text{O},i}$ give small time constants $T_{\text{O},i} = 1/k_{\text{O},i}$ in the transient response of that component of $\boldsymbol{r}$, which is associated with the same component of the external joint torque $\boldsymbol{\tau}_{\text{ext}}$. In the limit, we obtain

$$\mathbf{K}_{\text{O}} \to \infty \quad \Rightarrow \quad \boldsymbol{r} \approx \boldsymbol{\tau}_{\text{ext}} . \tag{69.25}$$

Collision Reflex Reactions

After a collision has been detected, suitable collision reflex reaction is needed. Four basic context-independent joint-level collision reflexes are discussed next. They lead to significantly different reflex behavior after a contact was detected. In the third and fourth schemes, the directional information on contact torques provided by suitable identification schemes such as (69.23) may be used to safely drive the robot away from the collision location.

Robot Stop. The most obvious strategy to react to a collision is to stop the robot. This behavior can, for example, be obtained by setting $\boldsymbol{q}_{\text{d}} = \boldsymbol{q}(t_c)$, where t_c is the instant of collision detection or by simply engaging the robot's brakes. More elaborate braking strategies can be found in [69.140].

Torque Control with Gravity Compensation. One may also react to a collision by switching the controllers. Typically, prior to the collision incident, the

robot moves along a desired trajectory with a position reference-based controller (e.g., position or impedance control). After detection, the control mode is switched to a compliance-based controller that ignores the previous task trajectory. A particularly useful variant is to switch to torque control mode with gravity compensation $\boldsymbol{\tau} = \hat{\boldsymbol{g}}(\boldsymbol{q})$ (VIDEO 611). Note that this strategy does not explicitly take into account any information about $\boldsymbol{\tau}_{\text{ext}}$.

Torque Reflex. This strategy extends the torque control-based strategy by explicitly incorporating the estimation or measurement of $\boldsymbol{\tau}_{\text{ext}}$ into the motor torque $\boldsymbol{\tau}$ via

$$\boldsymbol{\tau} = \hat{\boldsymbol{g}}(\boldsymbol{q}) + (\mathbf{I} - \mathbf{K}_{\text{r}})\boldsymbol{\tau}_{\text{ext}} \,, \tag{69.26}$$

where $\mathbf{K}_{\text{r}} = \text{diag}\{k_{\text{r},i}\} > 1$. It can be shown that, under sufficiently accurate estimates or measurements, such a law is equivalent to scaling of the robot dynamics by $\mathbf{K}_r^{-1}$. The closed-loop dynamics become

$$\underbrace{\mathbf{K}_r^{-1}\mathbf{M}(\boldsymbol{q})}_{\mathbf{M}'(\boldsymbol{q})}\ddot{\boldsymbol{q}} + \mathbf{K}_r^{-1}\mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{\text{ext}} = \mathbf{0} \,, \tag{69.27}$$

where $\mathbf{M}(\boldsymbol{q}) > \mathbf{M}'(\boldsymbol{q})$ holds component-wise.

Admittance Reflex. Reference trajectory modification via an admittance-type strategy that uses the measurement or estimation of $\boldsymbol{\tau}_{\text{ext}}$ can easily, for example, be realized via

$$\boldsymbol{q}_{\text{d}}(t) = -\int_{t_{\text{c}}}^{\text{T}} \mathbf{K}_{\text{a}}\boldsymbol{\tau}_{\text{ext}}\,\text{d}t \,, \tag{69.28}$$

where $\mathbf{K}_{\text{a}} = \text{diag}\{k_{\text{a},i}\} > 1$. With this scheme that requires no control switching the robot quickly drives away from the external torque source and decreases the contact forces till they decay to zero.

69.4.4 Shared Manipulation Control

Collaborative carrying, particularly of a long, large, heavy or flexible object, is a common scenario in pHRI research (Fig. 69.19). As discussed earlier, Cobots represent the parallel case where passive robotic devices control the path along which a shared load will be transported, but give their operator full control of load motion along that path (Chaps. 39 and 70; VIDEO 821). Most shared manipulation schemes utilize some form of impedance control [69.142]. In an early work, the authors of [69.143, 144] proposed an impedance controller with speed-dependent damping coefficients. The authors of [69.145] employed a similar approach (with fixed virtual impedance) to control the horizontal movement of their Mobile Robot Helper's mobile base in response to applied forces of a human user on the other end of a cooperatively carried load (VIDEO 606). The authors of [69.144] also described a lifting controller employing a pair of cascaded second-order virtual admittance controllers. Due to the raising/lowering of a cooperatively carried load the applied torques on an admittance-controlled, high-stiffness wrist generate a wrist deflection that translated (via a fixed gain) into a virtual vertical force. This force raised or lowered the robot end effector via a second admittance controller. In [69.141], an admittance controller was presented that was tuned for human preference, which is typically slightly underdamped. In [69.146], an admittance strategy was used to translate user force input into robot steering commands within a constrained trajectory.

Ideally, in cooperative manipulation, robotic and human partners will naturally take turns with leading and following roles depending on the state of a shared task. A switching model for haptically linked human–robot pairs that allows the robot to continuously vary its behavior from *completely following* to *completely leading* was introduced in [69.16]. In [69.147], the authors presented a mathematical treatment of the cooperative load manipulation problem to allow one or more robots to carry a load with a human user along a desired trajectory. They also incorporated variable leading behavior in the robots, enabling them to just steer the load (following) or completely control the load's axial motion (leading).

More recently, Evrard's homotopy approach was applied to load lifting, enabling the robot assistant to vary its behavior between leading and following based on its confidence in its predictions of the human user's

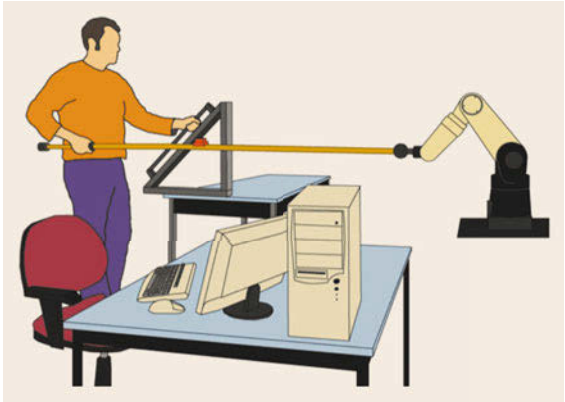


Fig. 69.19 Collaborative lifting experiment with a long object (after [69.141])

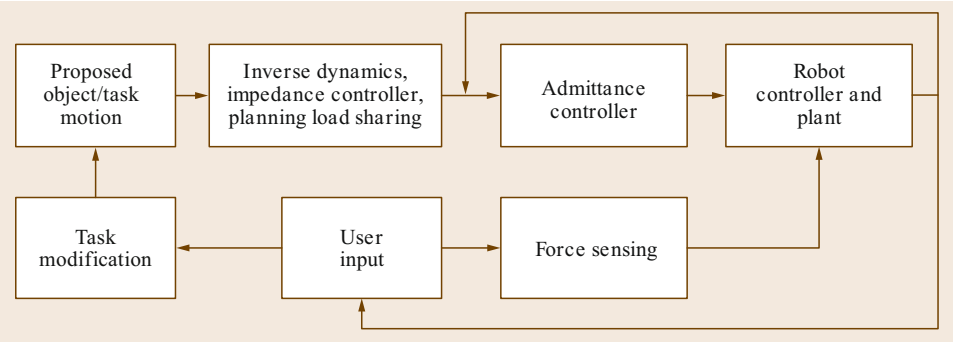


Fig. 69.20 Generalized schematic for collaborative task control

intentions [69.15] (VIDEO 617). These are monitored through the kinematics of the shared load. Recently, an admittance control law that guarantees the stability of the robot during constrained motion and also provides a quite intuitive human interaction was designed in [69.148]. The admittance law uses the time derivative of the contact force between the human and the robot to estimate human intent and an online estimate of the interaction stiffness resulting in very accurate shared-control of the robot system. In [69.149], the authors developed and user tested an interactive controller for a collaborative carrying task, utilizing a strategy to model effort sharing between the robot and the user. A generalized schematic for admittance-based collaborative task control is depicted in Fig. 69.20. On the motion planning side, minimum jerk trajectories [69.150] have been used to generate robot trajectories that are well matched to the motion of the human partner, and require the human to use less energy in the interaction, for example, [69.151]. Results from [69.152] indicate that a simpler quintic trajectory was also suitable for this purpose.

69.5 Motion Planning for Human Environments

The definition and quantification of injury, pain, or general risk are essential to express what safe behavior really means. The according insights can also be applied to generate safer robot motions such that injury and risk prevention are explicitly taken into account at this level. Two main branches of motion planning algorithms for danger reduction were developed, which are described hereafter.

69.5.1 Biomechanically Safe Motion Planning

As mentioned in the beginning of the chapter, human collision safety has to be ensured from an injury biomechanics perspective. Safety could, for example, be defined as *ensuring that only mild contusions may occur in worst case scenarios*. The natural question that arises is now how to formally respect such a metric, that is, how should a robot be controlled so that an unforeseen contact remains subcritical according to the underlying injury biomechanics or pain data. For this the authors of [69.25] gave the schema, termed safe motion unit (SMU), to link basic data to intrinsically safe robot velocity. The basic idea is to represent any data or general insight in a (possibly purely data driven) functional relation, linking impact properties to human injury or pain. With this, it is possible to calculate the instantaneous safe robot velocity given the robot's inertial properties, surface curvature, and possible human body parts it might collide with

$$\text{(mass, velocity, geometry, body part)} \rightarrow \text{observed injury/pain}\,. \tag{69.29}$$

For the injury/pain metric, multiple international classifications exist from which one may choose from. In [69.25], for example, the AO-classification was used. Thereafter, a careful set of experiments, or if possible simulations, can be generated to deduce the relation of interest. Each experiment should be designed such that the impacting mass, its velocity, and curvature are known in addition to any supplementary sensory readings of interest. Furthermore, the observed, measured, or calculated injury and/or pain level of the involved body part has to be quantified according to the selected scale. To restrict the amount of experiments to a feasible number, the impactor geometry is assumed to be composed of different basic geometric primitives (spheres, cuboids, corners). Thus, a finite set of so-called *safety curves* can be constructed, which represent fitted threshold curves for a given injury indicator in

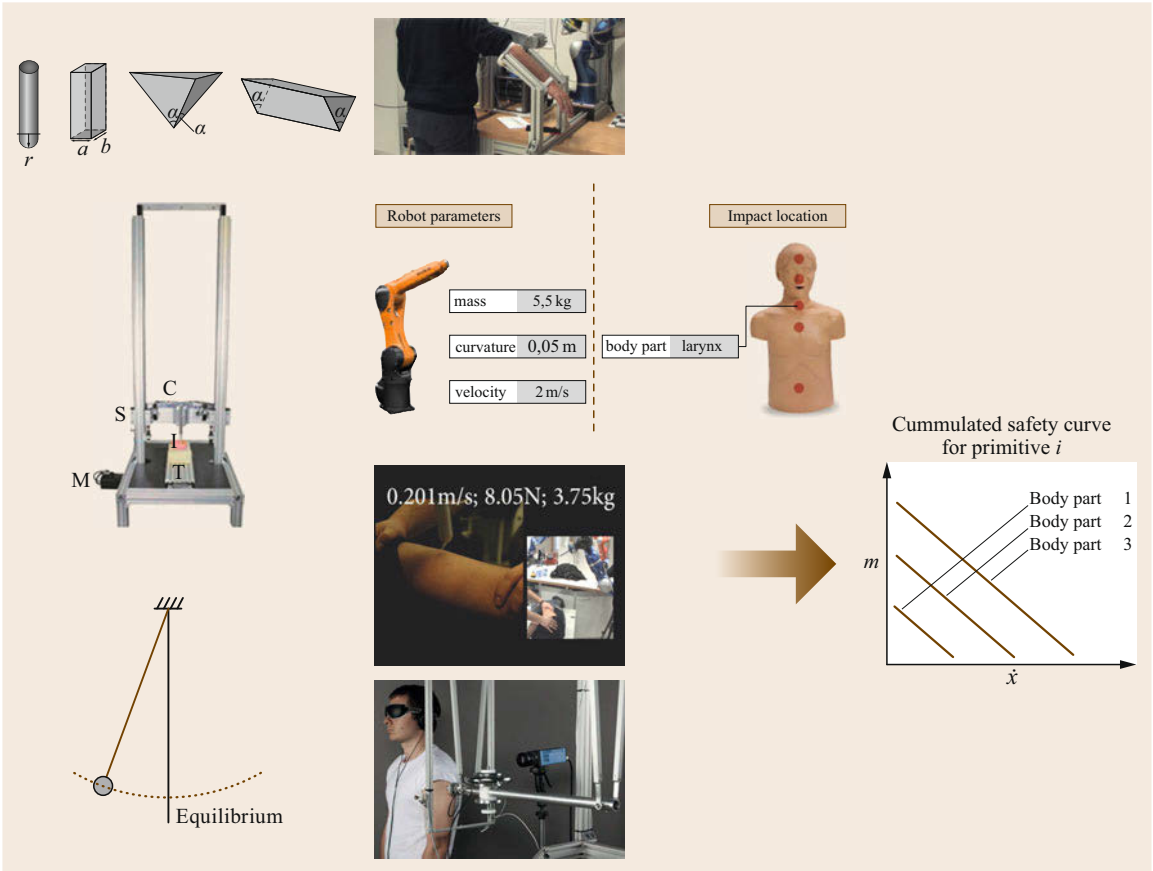


Fig. 69.21 A unified view: From collision experiment/simulation to cumulative *safety curves* for biomechanically safe robot velocity

the mass–velocity space parameterized by the body part (Fig. 69.21).

This idea can now be applied to a robot for safe velocity generation (Fig. 69.22). The concept is to scale any desired velocity command $\dot{\boldsymbol{q}}_\mathrm{d}$ by α to a biomechanically safe value $\dot{\boldsymbol{q}}_\mathrm{d}^*$. The scalar α is computed from evaluating the instantaneous robot inertial and local surface properties with respect to their injury potential. For this, the effective mass m_POI in the respective $\boldsymbol{u}$-directions of the relevant points of interest (POIs) are

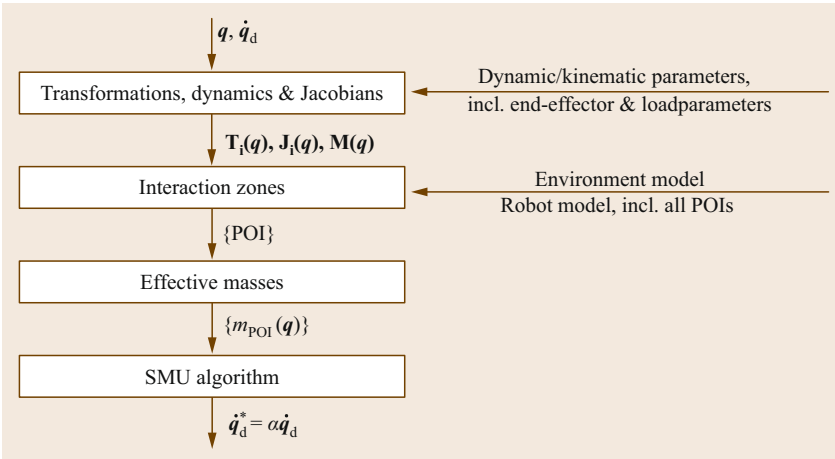


Fig. 69.22 Pipeline for creating biomechanically safe velocities based on the SMU algorithm (after [69.25])

evaluated via (69.7). Each POI is associated with a surface geometry primitive that is then used to link the respective POI to some underlying biomechanical injury, pain, or risk *safety curve*.

69.5.2 Risk-Metric-Based Motion Planning

As discussed in Sect. 69.4.3, there are numerous techniques for detecting impacts or potential collisions during pHRI. Collision monitoring methods can be used to ensure that forces and torques, or more simply the energy, of the robot system are limited during a collision event. To ensure safe and human-friendly interaction in unstructured environments, additional safety measures, utilizing system control, and planning are required.

Since physical interaction itself is a collision, a key problem of human-safe planning and control methods is to identify when human safety is actually threatened. In [69.154], a danger evaluation method was developed using the potential impact force as an evaluation measure. In this work, several danger indices were proposed based on the design properties of the robot. This danger index can be defined as a product of factors which affect the potential impact force between the robot and the human, such as distance, relative velocity, robot inertia, and robot stiffness. The authors applied this index as an objective function for improved mechanical design, control, and motion planning. A danger index based on estimated impact forces for a potential collision between a robot and a human was computed in [69.155] and used as an input to a real-time trajectory generation system, which balances the goal seeking with potential danger. In [69.153], the same authors combined their system with a vision based and physiological monitoring system, which allows the robot to respond to the user's real-time position and attentiveness. During the interaction, the user is monitored to asses his level of approval of the robot's actions while the trajectory planner monitors safety factors, such as robot velocity and user intent. Finally, the safety control module provides a real-time response to short-term horizon factors evaluated by a *safety measure estimation module* (Fig. 69.23).

For mobile service robots, the authors of [69.156] proposed a collision avoidance controller based on estimates of user behavior, using a *social force* model to determine whether the user intends to avoid the collision or not so that the robot can respond accordingly. In [69.157], the authors introduced a method using so-called kinetostatic danger fields as a metric for the danger which the current posture and velocity of a robot pose to objects in its environment. The work presented in [69.121] shows an algorithm which maximizes the productivity of an industrial robotic manipulator while guaranteeing a safe human–robot distance. In addition to the relative human–robot distance, it takes also dynamic and control characteristics into account. In [69.158], the authors proposed a method based on an intuitive physical interpretation, namely, an impedance-like second-order motion generation. It is designed to provide safe motion in complex environments, taking into account both proximity to objects, and external forces. A method that achieves collision avoidance for a redundant robot while permitting the end-effector task to continue is presented in [69.159]. Using data from a 3-D-RGB-D camera, this method is purely based on computations of distances between the robot body and dynamic obstacles in the workspace (e.g., the user) that are processed through a *risk function* to adjust the joint

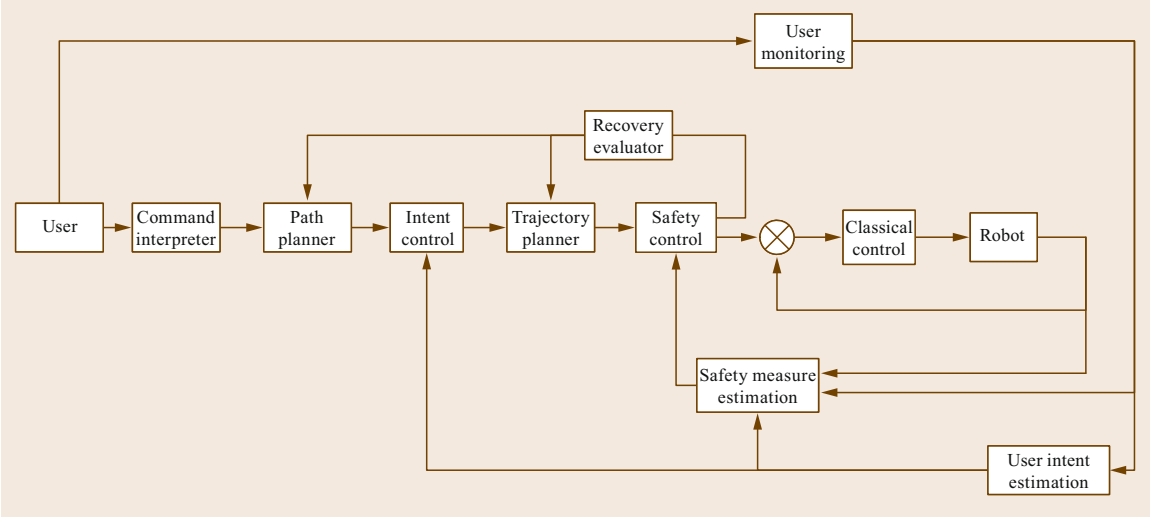


Fig. 69.23 Schematic of motion planning system with user intent and safety monitoring input (after [69.153])

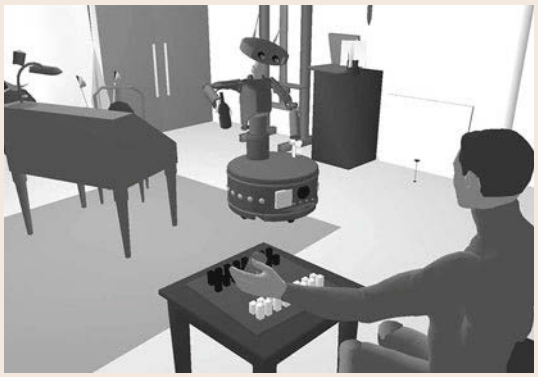


Fig. 69.24 Human-aware motion planner of Laboratoire d'analyse et d'architectures des systèmes (LAAS) (after [69.160]; courtesy of Rachid Alami)

velocities. The motion task of the end effector is modified by an artificial potential field-type method.

69.5.3 Human-Aware Motion Planning

Applying and extending classical motion planning techniques to the problem of human-robot interaction was originally done in [69.160]. The authors developed a human aware mobile robot motion planner, which incorporates humans accessibility, their vision field, and their preferences in terms of relative human–robot placement (Fig. 69.24). Human dynamics were integrated into the algorithm.

Extensions of the original work toward a motion planner for pHRI scenarios were elaborated in [69.161]. Therein, certain constraints, such as distance, visibility, and comfort are taken into account to generate safer motions and were demonstrated within a handover scenario. The algorithm has been extended to cluttered environments in [69.162], where a randomized cost-based exploration method provides an initial path that is relevant with respect to pHRI and workspace constraints.

69.6 Interaction Planning

In order to profit from the collaboration of human and robot by combining the flexibility, knowledge and sensory skills of a human with the efficiency, strength, endurance, and accuracy of a robot, according interaction planners need to be designed to plan their joint actions for a common goal. In the interaction planning domain, the central question is how to plan robot human joint actions and reactions in a certain interaction process, involving also unexpected environmental changes, one of the most basic ones being the human entering the robot's workspace. The definition of the interaction planning problem and the integration of reflex reaction schemes that potentially lead to an abrupt deviation from nominal course have to be elaborated. On an architectural level, the incorporation of *reactivity* has to be systematically represented.

69.6.1 Collaborative Action Planning

Approaches to subproblems of the full interaction planning problem were addressed in the literature. In [69.163], for example, the system SHARY for human-aware task planning was introduced. It produces social plans of a task by implementing communication schemes to negotiate the task solution with the human partner.

In [69.164], dynamic neural fields (DNFs) were used to build a decision-making system for interaction in cooperative human–robot tasks. The main idea is to let the robot imitate human behavior in order to make the cooperation between robot and human appear rather intuitive. A structural overview of the system is depicted in Fig. 69.25. A vision system observes the scenery and recognizes task-related objects and gestures of the human coactor. Object positions are stored in the *object memory layer*. The *action observation layer* decides which type of known action was executed by the coactor. The expected outcome of this action is then simulated by the *action simulation layer* and thereafter passed to the *intention layer*. Here, the intention of the coactor is determined. The knowledge about the coactor's intention and the expected outcome of his actions is used by the *common subgoal layer*. It contains prior knowledge about the task structure and the subgoals to be achieved for task completion. The *common subgoal layer* is responsible for enabling actions that lead to the fulfillment of a currently achievable subtask only. Based on the information coming from the *intention layer* and the *object memory layer*, the *action execution layer* decides to perform an action, which supports the human in the intended subtask or leads to the completion of an independent subtask and does not interfere with the current human intention. Each layer of the system is formalized by one or more DNFs. The activity $u_i(x, t)$ at time t of a neuron x is described by the differential

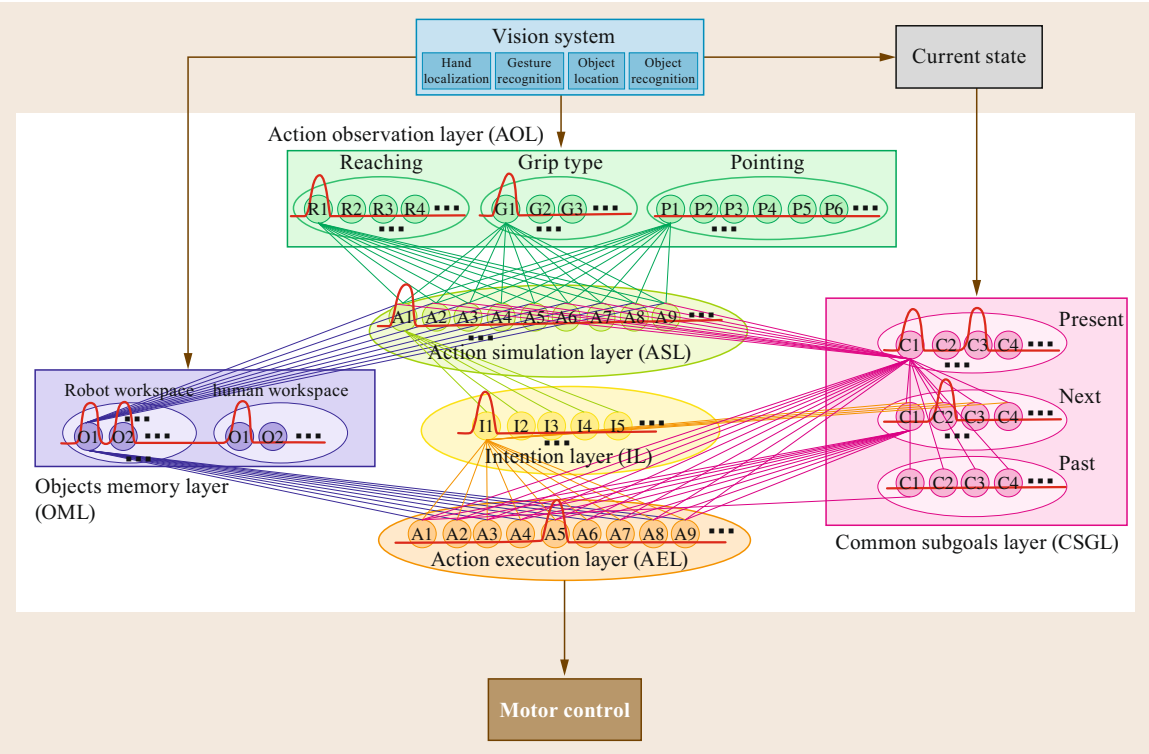


Fig. 69.25 Structural overview of the decision-making system from (after [69.164]; courtesy of Estella Bicho)

equation

$$\tau_i \frac{\delta u_i(x,t)}{\delta t} = -u_i(x,t) + S_i(x,t) + \int w_i(x-x') f_i(u_i(x',t))\mathrm{d}x' - h_i \,, \tag{69.30}$$

where $\tau_i > 0$ and $h_i > 0$ denote the timescale and the resting level of the field dynamics, respectively. S_i is the summed input to a local population. The output function f_i is chosen to be a sigmoid function and the interaction strength $w_i(x-x')$ is a Gaussian curve, which depends only on the distance between x and x' (Fig. 69.26).

In [69.129], a mimetic communication model for pHRI is introduced, where the according motion primitives are taught by human demonstration via marker control. These are subsequently encoded into hidden Markov models (HMMs) and allow the robot to execute them and even to recognize the motion primitives executed by the human partner. Based on the motion primitives, interaction primitives are then learned as chains of actions and reactions. The flow of interaction primitive learning is depicted in Fig. 69.27a. First, the robot executes a motion primitive, followed by observing the human's respective reaction in order to learn the correct interaction pattern sequence. After this step, the motion primitives are updated and the process is repeated. The underlying scheme of interaction is shown in Fig. 69.27b. The learned motion primitives are encoded as continuous HMMs (CHMMs) and form the middle layer of the system. They are then used to recognize the motion primitive executed by

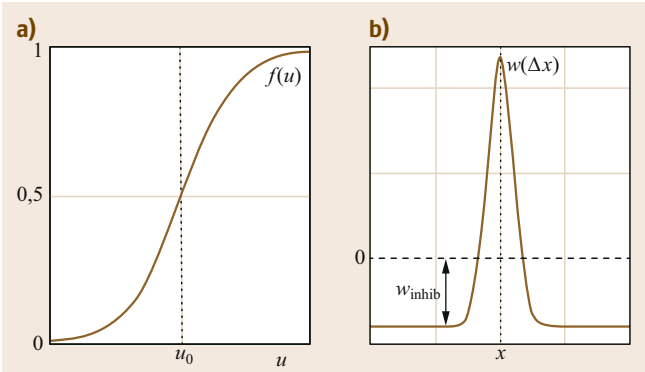


Fig. 69.26 **(a)** Nonlinear threshold output function that maps activity to a sigmoidal with threshold u_0. **(b)** Synaptic weight function for modeling the interaction strength between any two neurons x and x' in (69.30) (after [69.164]; courtesy of Estella Bicho)

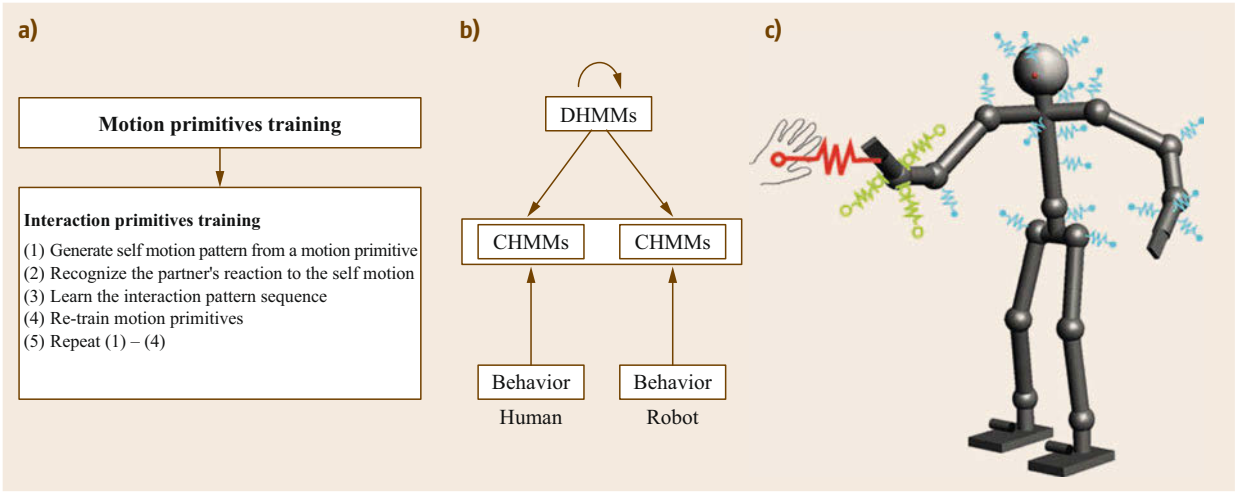


Fig. 69.27 (a) Learning of interaction primitives. (b) Scheme of interaction. (c) Adaption of robot motion via virtual spring (after [69.129]; courtesy of Dongheui Lee)

the human and to generate the motion of the robot according to the interaction primitives encoded as discrete HMMs (DHMMs) at the top. The robot motion (more precisely its behavior in the context of the authors' work) may then be modified to adapt to human motion in the real world (expressed by the thin horizontal arrow). The bold horizontal arrow represents the assumed adaption of the human to the robot behavior. The adaptation strategy of the robot movement is depicted in Fig. 69.27c. For this, an impedance controller is used in combination with a virtual spring connected to the robot hand to attract the device into the correct position. Figure 69.28 depicts an application of the interaction scheme, where the robot high- or low-fives a human coactor with one or two hands, respectively.

69.6.2 Interaction Planning Problem

In order to formalize the interaction planning problem, one needs to be able to describe the entire scenario, the system state of the robot, the state of the human(s) (possibly including future behavior prediction of both), the environment state including all relevant objects and the overall abstract task state. This information about the world state comprises the concept of an *interaction world*, which builds the basis for formulating the general *interaction planning problem*.

Definitions

The set of *world states* $\boldsymbol{WS}$ is defined as

$$\boldsymbol{WS} = \boldsymbol{RS} \times \boldsymbol{HS}^n \times \boldsymbol{OBS}^m \times \boldsymbol{TS} \,, \tag{69.31}$$

where $\boldsymbol{RS}, \boldsymbol{HS}, \boldsymbol{OBS}, \boldsymbol{TS}$ denote the set of the robot states, n humans, m obstacles, and an overall task state, respectively.

The set of *robot states*

$$\boldsymbol{RS} = \boldsymbol{S} \times \boldsymbol{RA}^n \times \boldsymbol{IR} \tag{69.32}$$

contains information about the internal state $s \in \boldsymbol{S}$ of the robot, the robot awareness $ra \in \boldsymbol{RA}$, and the interaction state $ir \in \boldsymbol{IR}$. The interaction state of the robot may, for example, take the values *autonomous*, *collaborative*, or *cooperative*. The robot awareness indicates its ability to predict human actions. The set of internal states $\boldsymbol{S} = \boldsymbol{S}_{\mathrm{ac}} \times \boldsymbol{S}_{\mathrm{b}} \times \boldsymbol{S}_{\mathrm{p}}$ contains information about the set of actions carried out by the robot $\boldsymbol{S}_{\mathrm{ac}}$. Furthermore, the behavior set $\boldsymbol{S}_{\mathrm{b}}$, which includes, for example, controller choices and respective parameterizations, reflex reactions, and type or parameterizations of trajectory

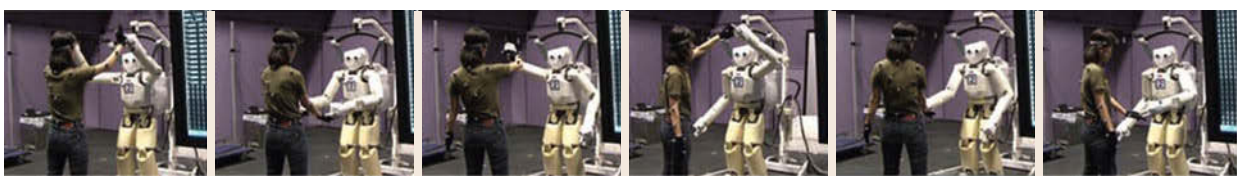


Fig. 69.28 pHRI using the mimetic communication model (after [69.129]; VIDEO 625; courtesy of Dongheui Lee)

planners. Such a reflex reaction which can be thought of as an analogy to human reflexes, intends to bring the robot to a safe state or even a reflex cascade $\boldsymbol{R}$ (Sect. 69.6.3). Finally, the physical state $sp \in \boldsymbol{S}_\mathrm{p}$ contains, for example, the robot's instantaneous position, velocity, and momentum.

The set of *human states*

$$\boldsymbol{HS} = \boldsymbol{PSH} \times \boldsymbol{PA} \times \boldsymbol{HA} \times \boldsymbol{IH} \times \boldsymbol{D} \tag{69.33}$$

combines the sets of physical state $\boldsymbol{PSH}$ (position, velocity, etc.), personal attributes $\boldsymbol{PA}$ (fitness, age, experience, etc.), human awareness $\boldsymbol{HA}$ (the ability of the human to predict the robots actions), interaction states of the human $\boldsymbol{IH}$ (for example, *waiting for robot*, *work with robot* or *work without robot*), and the distances $\boldsymbol{D}$ between human and robot. In its most simple nontrivial case, $\boldsymbol{D}$ could, for example, consist of the elements *in workspace*, *in perception*, *out of perception*, and *lost perception*.

The set of *task states*

$$\boldsymbol{TS} = \boldsymbol{A} \times \boldsymbol{TC} \times \boldsymbol{IS} \tag{69.34}$$

contains $\boldsymbol{A}$, $\boldsymbol{TC}$, $\boldsymbol{IS}$ that denote the sets of possible actions, task criticality, and expected interaction between human and robot, respectively. The task criticality specifies the effect of potential task failure on human safety. A task consists of the meaningful composition of actions and/or more complex skills such as *grasp object* or simply *move to pose* according to a particular creation process.

The Interaction Planning Problem

The *interaction planning problem* then denotes the problem of selecting a suitable, and if possible optimal, robot action and associated behavior in every time step based on the information contained in the world state, the history of the process $\boldsymbol{HIS}$, and available (dynamic) knowledge stored in a knowledge base $\boldsymbol{KB}$ (for example, safety knowledge, object properties, world grounding rules, ...). Formally, this can be expressed by the *select action* mapping

$$sa : \boldsymbol{WS} \times \boldsymbol{HIS} \times \boldsymbol{KB} \rightarrow \boldsymbol{S}_\mathrm{ac} \times \boldsymbol{S}_\mathrm{b} \,. \tag{69.35}$$

Optimality could, for example, be expressed in the following *reinforcement learning* sense, where the selected behavior shall lead to a maximum reward. This is obtained as follows:

1. A *task evaluator* evaluates the suitability of a particular action/behavior for fulfilling the desired task.
2. A *safety evaluator* evaluates the overall safety of the situation by taking into account different biomechanical injury criteria, the expected interaction between human and robot, human–robot–related geometric quantities, such as minimal distance, as well as the task criticality.
3. Finally, additional rewards may be given by the human in order to capture human-friendly behavior.
4. The overall reward is then a suitable combination of the basic rewards.

69.6.3 Robot Reflexes

Human reflexes are involuntary reactive body movements in response to a perceived input, the so-called stimulus, that is, one does not even have to *think about what to do*. In humans, reflexes aim for body protection. Based on built-in heuristics, they automatically protect the human body from injury. However, the exact separation in terms of hybrid system-like theory is not yet clear. Thus, the line between discrete reflex states and according systemic responses on both electrical and mechanical levels is somewhat blurred. In robotics, on the other hand, it is possible to separate these levels and extend the concept of basic *nominal robot actions* by *reflex reactions* [69.9, 85, 139]. These concepts systematically incorporate the possibilities one gains from the already described collision detection and reflex reaction schemes (Sect. 69.4.3). Other work in similar directions for developing human-like withdrawal reflexes can be found in [69.165].

In contrast to the classical planning and execution pipeline, the activation of reflexes that are able to override the nominal task plan due to environmental or internal conditions that do not comply with nominal task behavior have to be considered. In contrast to humans, these reflexes do not necessarily have to be fixed (or slowly time-varying over learning cycles). On the contrary, they may be tailored to every nominal action such that local context dependency (in particular instantaneous sensory input) may be taken into account. After instantaneous reactions are executed, the main problem is to decide what to do next. Either local re-entry into the previous plan or entire re-planning have to be executed. Figure 69.29 depicts the overall concept also in the context of task and interaction planning.

Formally, a robot reflex is associated with a suitable activation signal. Typically, this represents either the indication of a certain stimulus or a fault. Stimuli are general perception inputs, whereas faults are detected either by processed stimuli (observation of external torques, proximity information, etc.) or general system malfunctions as, e.g., communication collapse or runtime violations. Even rather complex reflex patterns can be activated that may be represented as directed reflex graphs, i. e., a decisional component in the inner most

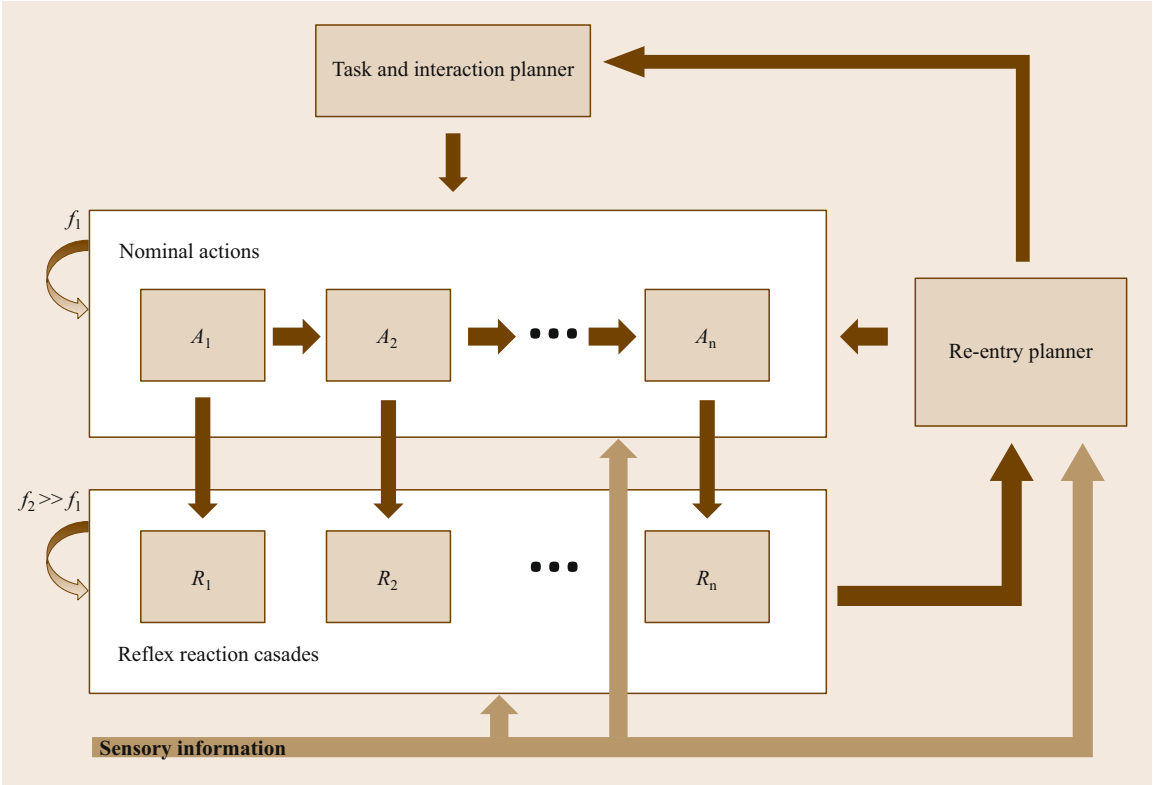


Fig. 69.29 Conceptual sketch of robot reflexes; f_1 and f_2 indicate the update rates, or alternatively the system bandwith of the nominal planning and execution as well as the reflex reaction level. Typically, the former acts at a much lower frequency than the latter one (after [69.85, 139])

control loop of the system. A still relatively open research problem is how to resolve potential failures after reflexes were triggered by planning suitable re-entering and continuation of the task.

69.6.4 Reactive Control Architecture

Obviously, the diverse methods and complex requirements in the pHRI domain and more specifically from interaction control, reflex planning, human-centered motion planning, and interaction planning make new architectural concepts necessary. The central requirements to an according control architecture become rather different from the classical ones (Chap. 12) due to the demands for very responsive behavior not only on control but also on planning level. Figure 69.30 depicts such a reactive control framework [69.166] (VIDEO 616). One of the main questions to address is how to adapt dynamically, safe, and task consistent, while keeping the overall plan and the respective context in mind. The control framework is composed of three layers of abstraction operating at different time scales:

1. On the highest level of abstraction a *global task planning* module builds global task plans that contain temporo-logical concatenations of robot skills, which are derived from any suitable task planning language. Typically, this module runs either offline, in particular when building an entire nominal task plan that contains interaction schemata, or at very slow update rates of several seconds or minutes, depending on the complexity and novelty of the respective task. Skills are topologically invariant capability structures, which instantiations can be modified in terms of skill parameter vectors. The most basic skill is the atomic robot action that represents the formal interface to the robot real-time control core and mechatronics. More complex skills can, e.g., be *grasp an object* or *hand over an object*. Each skill itself is aware of the current task state and which atomic action is currently being executed. This information is then sent to the *dynamic planning layer*.
2. The second layer of abstraction, the *dynamic planning level* is capable of executing and/or modifying plans dynamically, which means that it shall run at least in the range of $\approx 1-10$ Hz. It is responsible

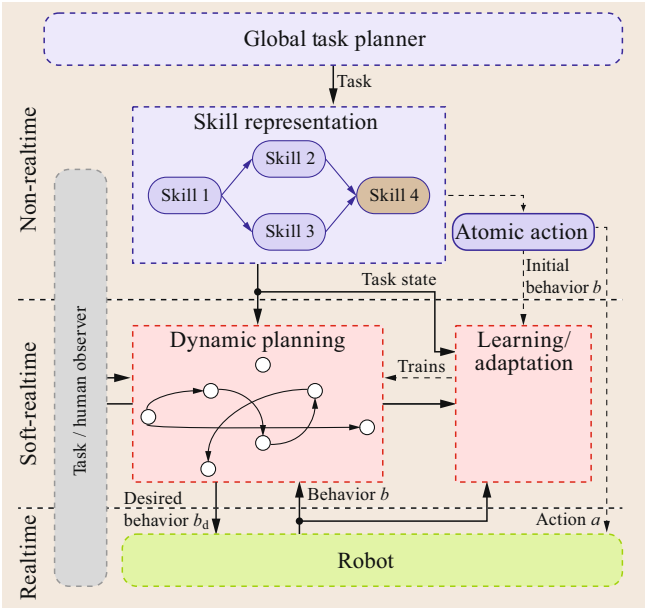


Fig. 69.30 Reactive and human-friendly control framework for acting in partially known environments and interaction ►

for selecting the best action under the premise of the current task state, human state and behavior, as well as the environmental state. In a *learning and adaptation* unit the global task knowledge is translated into the respective dynamic planning domain language, i. e., global knowledge and plans are encoded into a dynamic level such that dynamic adaptation can make use of global context and the respective mission. The according reactive planning unit is then able to (re-)plan the robot's desired nominal actions such that safe, yet task consistent actions, are executed if possible. In particular, instantaneous perception may cause alterations of the original global plans. A major task on this abstraction level is to change the preplanned course of action safely and task consistently. However, finding its way back into the task is at least equally challenging and requires careful treatment (re-entry planning).

3. The lowest architectural layer is the *low-level (real-time) control layer*. Typically, this is subdivided into multiple hierarchical layers, involving also the reflex machine (Fig. 69.29). However, for sake of clarity, we consider it to be a single consistent representation that is accessible via a desired action/behavior complex (a_d, b_d) that is sent to the system for execution. As the desired behavior may alter due to reflex behaviors in case of accidental events such as unforeseen collisions, the control layer feeds back the currently active action/behavior pair (a, b), such that the *dynamic planning layer* is able to react accordingly.

Fundamental to the aforementioned architecture is the capacity to observe human actions on various levels of abstraction. In this sense, a *human observer* that gathers all relevant information and knowledge about human agents is essential. In particular, it provides human-related information that can be of further use.

69.7 Conclusions and Challenges

pHRI has become a central discipline in robotics over the last decade. This is due to the significant progress made in the fields of mechatronics, interaction control, motion planning, and 3-D sensing toward highly integrated and sensorized lightweight systems that are able to physically interact with their surrounding. In direct consequence the learning, planning, and execution of safe and legible interactions has become a widely taken research direction. Clearly, the rise of a new generation of commercial robots capable of physical interaction has also contributed to the large interest in the field. The robotics research and industrial community expects these systems to open up new markets and to push robotics further toward domestic applications that may also involve even more complex and possibly mobile manipulators.

However, despite this recent success in research and also in the commercialization of assistance robots, there are many open research questions that need to be tackled before this class of systems can become a commodity not only in early adopter industrial applications but also on a broader scale: Continuing the road toward safe robotics by tightly coupling injury biomechanics and safe interaction control with lightweight and compliant robot design will further push the boundaries and build the foundation of pHRI. On the other hand, learning interaction controllers and planning intuitive and safe interactions are still very young fields, however, they are the key to solving the long-term physical interaction problem. Furthermore, the current application of assistance robots in real-world problems will bring further novel research questions. The recent

use of these robots clearly underlines that the programming models and paradigms of interaction and soft manipulation are very different from classical industrial robot programming. In particular, they go beyond simple pick and place models toward models of *force based programming*, an interesting research question currently being investigated by various researchers worldwide.

Video-References

VIDEO 606 Mobile robot helper – Mr. Helper
available from http://handbookofrobotics.org/view-chapter/69/videodetails/606

VIDEO 607 Generation of human care behaviors by human-interactive robot RI-MAN
available from http://handbookofrobotics.org/view-chapter/69/videodetails/607

VIDEO 608 Injury evaluation of human-robot impacts
available from http://handbookofrobotics.org/view-chapter/69/videodetails/608

VIDEO 609 Safe physical human-robot collaboration
available from http://handbookofrobotics.org/view-chapter/69/videodetails/609

VIDEO 610 Admittance control of a human centered 3 DOF robotic arm using differential elastic actuators
available from http://handbookofrobotics.org/view-chapter/69/videodetails/610

VIDEO 611 A control strategy for human-friendly robots
available from http://handbookofrobotics.org/view-chapter/69/videodetails/611

VIDEO 613 Human–robot interactions
available from http://handbookofrobotics.org/view-chapter/69/videodetails/613

VIDEO 614 ISAC: A demonstration
available from http://handbookofrobotics.org/view-chapter/69/videodetails/614

VIDEO 615 Smart fur
available from http://handbookofrobotics.org/view-chapter/69/videodetails/615

VIDEO 616 Human–robot interaction planning
available from http://handbookofrobotics.org/view-chapter/69/videodetails/616

VIDEO 617 The power of prediction: Robots that read intentions
available from http://handbookofrobotics.org/view-chapter/69/videodetails/617

VIDEO 618 Reach and grasp by people with tetraplegia using a neurally controlled robotic arm
available from http://handbookofrobotics.org/view-chapter/69/videodetails/618

VIDEO 619 An assistive decision and control architecture for force-sensitive hand–arm systems driven via human–machine interfaces (MM1)
available from http://handbookofrobotics.org/view-chapter/69/videodetails/619

VIDEO 620 An assistive decision and control architecture for force-sensitive hand–arm systems driven via human–machine interfaces (MM2)
available from http://handbookofrobotics.org/view-chapter/69/videodetails/620

VIDEO 621 An assistive decision-and-control architecture for force-sensitive hand–arm systems driven by human–machine interfaces (MM3)
available from http://handbookofrobotics.org/view-chapter/69/videodetails/621

VIDEO 622 An assistive decision-and-control architecture for force-sensitive hand–arm systems driven by human–machine interfaces (MM4)
available from http://handbookofrobotics.org/view-chapter/69/videodetails/622

VIDEO 623 Twendy One demo
available from http://handbookofrobotics.org/view-chapter/69/videodetails/623

VIDEO 624 Full body compliant humanoid COMAN
available from http://handbookofrobotics.org/view-chapter/69/videodetails/624

VIDEO 625 Physical human–robot interaction in imitation learning
available from http://handbookofrobotics.org/view-chapter/69/videodetails/625

VIDEO 626 Justin: A humanoid upper body system for two-handed manipulation experiments
available from http://handbookofrobotics.org/view-chapter/69/videodetails/626

VIDEO 627 Torque control for teaching peg in hole via physical human–robot interaction
available from http://handbookofrobotics.org/view-chapter/69/videodetails/627

VIDEO 632 Flexible robot gripper for KUKA Light Weight Robot (LWR): Collaboration between human and robot
available from http://handbookofrobotics.org/view-chapter/69/videodetails/632

VIDEO 716 Human–robot handover
available from http://handbookofrobotics.org/view-chapter/69/videodetails/716

VIDEO 717 Collaborative human-focused robotics for manufacturing
available from http://handbookofrobotics.org/view-chapter/69/videodetails/717
VIDEO 820 Dancing with Juliet
available from http://handbookofrobotics.org/view-chapter/69/videodetails/820
VIDEO 821 A cobot in automobile assembly
available from http://handbookofrobotics.org/view-chapter/69/videodetails/821

References

69.1 M.A. Goodrich, A.C. Schultz: Human-robot interaction: A survey, Found. Trends Hum.-Comput. Interact. **1**(3), 203–275 (2007)
69.2 M.A. Peshkin, J.E. Colgate, W. Wannasuphoprasit, C.A. Moore, R.B. Gillespie, P. Akella: Cobot architecture, IEEE Trans. Robotics Autom. **17**(4), 377–390 (2001)
69.3 T. Shibata, T. Mitsui, K. Wada, A. Touda: Mental commit robot and its application to therapy of children, IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (2001) pp. 1053–1058
69.4 S. Yohanan, K.E. MacLean: The role of affective touch in human-robot interaction: Human intent and expectations in touching the haptic creature, Int. J. Soc. Robotics **4**(2), 163–180 (2011)
69.5 Y. Iwamura, M. Shiomi, T. Kanda, H. Ishiguro, N. Hagita: Do elderly people prefer a conversational humanoid as a shopping assistant partner in supermarkets?, ACM Int. Conf. Hum.-Robot Interact. (2011) p. 449
69.6 T. Ende, S. Haddadin, S. Parusel, W. Tilo, M. Hassenzahl, A. Albu-Schäffer: A human-centered approach to robot gesture based communication within collaborative working processes, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 3367–3374
69.7 A.L. Thomaz, C. Chao: Turn-taking based on information flow for fluent human-robot interaction, AI Mag. **32**(4), 53–63 (2011)
69.8 B. Gleeson, K. MacLean, A. Haddadi, E. Croft, J. Alcazar: Gestures for industry Intuitive human-robot communication from human observation, ACM/IEEE Int. Conf. Hum.-Robot Interact. (2013) pp. 349–356
69.9 S. Haddadin, S. Parusel, R. Belder, A. Albu-Schäffer: It is (almost) all about human safety: A novel paradigm for robot design, control, and planning, Lect. Notes Comput. Sci. **8153**, 202–215 (2013)
69.10 J. Mainprice, D. Berenson: Human-robot collaborative manipulation planning using early prediction of human motion, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 299–306
69.11 M. Cakmak, S.S. Srinivasa, M.K. Lee, S. Kiesler, J. Forlizzi: Using spatial and temporal contrast for fluent robot-human hand-overs, ACM/IEEE Int. Conf. Hum.-Robot Interact. (2011) pp. 489–496
69.12 E.A. Sisbot, R. Alami: A human-aware manipulation planner, IEEE Trans. Robotics **28**(5), 1045–1057 (2012)
69.13 W.P. Chan, C.A.C. Parker, H.F.M. Van der Loos, E.A. Croft: A human-inspired object handover controller, Int. J. Robotics Res. **32**(8), 972–984 (2013)
69.14 M.S. Erden, T. Tomiyama: Human-intent detection and physically interactive control of a robot without force sensors, IEEE Trans. Robotics **26**(2), 370–382 (2010)
69.15 A. Thobbi, Y. Gu, W. Sheng: Using human motion estimation for human-robot cooperative manipulation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 2873–2878
69.16 P. Evrard, E. Gribovskaya, S. Calinon, A. Billard, A. Kheddar: Teaching physical collaborative tasks: Object-lifting case study with a humanoid, IEEE-RAS Int. Conf. Humanoid Robots (2009) pp. 399–404
69.17 S. Ikemoto, H. Ben Amor, T. Minato, H. Ishiguro, B. Jung: Mutual learning and adaptation in physical human-robot interaction, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 24–335
69.18 D. Lee, C. Ott: Incremental kinesthetic teaching of motion primitives using the motion refinement tube, Auton. Robots **31**(2–3), 115–131 (2011)
69.19 S. Haddadin, A. Albu-Schäffer, G. Hirzinger: Requirements for safe robots: Measurements, analysis & new insights, Int. J. Robotics Res. **28**(11–12), 1507–1527 (2009)
69.20 A. Bicchi, G. Tonietti: Fast and soft arm tactics: Dealing with the safety-performance trade-off in robot arms design and control, IEEE Robotics Autom. Mag. **11**, 22–33 (2004)
69.21 M. Zinn, O. Khatib, B. Roth: A new actuation approach for human friendly robot design, Int. J. Robotics Res. **23**, 379–398 (2004)
69.22 S. Haddadin, A. Albu-Schäffer, G. Hirzinger: Safety evaluation of physical human-robot interaction via crash-testing, Proc. Robotics Sci. Syst. Conf. (2007) pp. 217–224
69.23 S. Oberer, R.-D. Schraft: Robot-dummy crash tests for robot safety assessment, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 2934–2939
69.24 S. Haddadin, A. Albu-Schäffer, F. Haddadin, J. Roßmann, G. Hirzinger: Study on soft-tissue injury in robotics, IEEE Robotics Autom. Mag. **18**(4), 20–34 (2011)
69.25 S. Haddadin, S. Haddadin, A. Khoury, T. Rokahr, S. Parusel, R. Burgkart, A. Bicchi, A. Albu-Schäffer: On making robots understand safety: Embedding injury knowledge into control, Int. J. Robotics Res. **31**, 1578–1602 (2012)

69.26 S. Haddadin: Towards safe robots – Approaching Asimov's 1st law, Springer Tracts Adv. Robotics **90**, 1–343 (2014)

69.27 D.C. Schneider, A.M. Nahum: Impact studies of facial bones and skull, Proc. 16th Stapp Car Crash Conf. (1972) pp. 186–204

69.28 A.M. Nahum, J.D. Gatts, C.W. Gadd, J. Danforth: Impact tolerance of the skull and face, Proc. Stapp Car Crash Conf. (1968)

69.29 D. Allsop, T.R. Perl, C. Warner: Force/deflection and fracture characteristics of the temporoparietal region of the human head, SAE Transactions (1991) pp. 2009–2018

69.30 J. Cormier, S. Manoogian, J. Bisplinghoff, S. Rowson, A. Santago, C. McNally, S. Duma, J.I.V. Bolte: The tolerance of the nasal bone to blunt impact, Ann. Adv. Automot. Med (2010) p. 3

69.31 H. Delye, P. Verschueren, B. Depreitere, I. Verpoest, D. Berckmans, J. Vander Sloten, G. Van Der Perre, J. Goffin: Biomechanics of frontal skull fracture, J. Neurotrauma **24**(10), 1576–1586 (2007)

69.32 G.W. Nyquist, J.M. Cavanaugh, S.J. Goldberg, A.I. King: Facial impact tolerance and response, Proc. 30th Stapp Car Crash Conf. (1986) pp. 733–754

69.33 D.L. Allsop, C.Y. Warner, M.G. Wille, D.C. Schneider, A.M. Nahum: Facial Impact response – A comparison of the hybrid III dummy and human cadaver, Proc. Stapp Car Crash Conf. (1988) pp. 781–797

69.34 V.R. Hodgson, L.M. Thomas: Comparison of head acceleration injury indices in cadaver skull fracture, Proc. Stapp Car Crash Conf. (1971) pp. 299–307

69.35 A.M. Nahum, R.W. Smith: An experimental model for closed head impact injury, Proc. Stapp Car Crash Conf. (1976)

69.36 S. Haddadin, A. Albu-Schäffer, M. Frommberger, J. Rossmann, G. Hirzinger: The *DLR Crash Report*: Towards a standard crash-testing protocol for robot safety – Part I: Results, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 272–279

69.37 S. Haddadin, A. Albu-Schäffer, M. Frommberger, J. Rossmann, G. Hirzinger: The "DLR Crash Report": Towards a standard crash-testing protocol for robot safety – Part II: Discussions, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 280–287

69.38 C.K. Kroell, D.C. Schneider, A.M. Nahum: Impact tolerance and response of the human thorax I, Proc. Stapp Car Crash Conf. (1971)

69.39 C.K. Kroell, D.C. Scheider, A.M. Nahum: Impact tolerance and response of the human thorax II, Proc. Stapp Car Crash Conf. (1974) pp. 383–457

69.40 L.M. Patrick: Impact force deflection of the human thorax, Proc. 25th Stapp Car Crash Conf. (1981) pp. 471–496

69.41 A.M. Nahum, C.W. Gadd, D.C. Schneider, C. Kroell: Deflection of the human thorax under sternal impact, Int. Automot. Saf. Conf. (1970)

69.42 J.M. Cavanaugh, G.W. Nyquist, S.J. Goldberg, A.I. King: Lower abdominal impact tolerance and response, Proc. Stapp Car Crash Conf. (1986)

69.43 S.M. Duma, P. Schreiber, J. McMaster, J. Crandall, C. Bass, W. Pilkey: Dynamic injury tolerances for long bones of the female upper extremity, Int. Res. Council Biomech. Inj. (IRCOBI) (1998) pp. 189–201

69.44 S.M. Duma, J.R. Crandall, S.R. Hurwitz, W.D. Pilkey: Small female upper extremity interaction with the deploying side air bag, Proc. Stapp Car Crash Conf. (1998) pp. 47–63

69.45 J.A. Spadaro, F.W. Werner, R.A. Brenner, M.D. Fortino, L.A. Fay, W.T. Edwards: Cortical and trabecular bone contribute strength to the osteopenic distal radius, J. Orthop. Res. **12**, 211–218 (1994)

69.46 O. Khatib: Inertial properties in robotic manipulation: An object-level framework, Int. J. Robotics Res. **14**(1), 19–36 (1995)

69.47 T.E. Lobdell, C.K. Kroell, D.C. Scheider, W.E. Hering: Impact response of the human thorax, Symp. Hum. Impact Response (1972) pp. 201–245

69.48 T.P. Ruedi, W.M. Murphy: *AO Principles of Fracture Management*, Vol. 1 (Thieme, Stuttgart 2007)

69.49 I.V. Lau, D.C. Viano: Role of impact velocity and chest compression in thoraic injury, Avia. Space Environ. Med. **56**, 16–21 (1983)

69.50 ISO: *ISO12100:2010: Safety of Machinery – General Principles for Design – Risk Assessment and Risk Reductions* (Int. Organization for Standardization, Geneva 2010)

69.51 ISO: *ISO13849-1:2006: Safety of Machinery – Safety-Related Parts of Control Systems – Part 1: General Principles for Design* (Int. Organization for Standardization, Geneva 2006)

69.52 ISO: *ISO13855:2010: Safety of Machinery – Positioning of Safeguards With Respect to the Approach Speeds of Parts of the Human Body* (Int. Organization for Standardization, Geneva 2010)

69.53 ISO: *ISO10218-1:2011: Robots and Robotic Devices – Safety Requirements for Industrial Robots – Part 1: Robots* (Int. Organization for Standardization, Geneva 2011)

69.54 ISO/DTS 15066: *Robots and Robotic Devices – Safety Requirements for Industrial Robots – Collaborative operation* (Int. Organization for Standardization, Geneva) under development

69.55 ISO: *ISO13482:2014: Robots and Robotic Devices – Safety Requirements for Personal Care Robots* (Int. Organization for Standardization, Geneva 2014)

69.56 C. Gosselin, T. Laliberte, B. Mayer-St-Onge, S. Foucault, A. Lecours, V. Duchaine, N. Paradis, D. Gao, R. Menassa: A friendly beast of burden: A human-assistive robot for handling large payloads, IEEE Robotics Autom. Mag. **20**(4), 139–147 (2013)

69.57 G. Hirzinger, A. Albu-Schäffer: Lightweight robots, Scholarpedia **3**, 3889 (2008)

69.58 W.T. Townsend, J.K. Salisbury: Mechanical design for whole-arm manipulation, Proc. NATO Adv. Workshop Robots Biol. Syst, ed. by P. Dario, G. Sandini, P. Aebischer (1993) pp. 153–164

69.59 A. Albu-Schäffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck, G. Hirzinger: The DLR lightweight

robot – Lightweight design and soft robotics control concepts for robots in human environments, Ind. Robot J. **34**(5), 376–385 (2007)
69.60 KUKA Roboter GmbH: http://www.kuka-lbr-iiwa.com (2015)
69.61 A.S. Shafer, M.R. Kermani: Design and validation of a magneto-rheological clutch for practical control applications in human-friendly manipulation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 4266–4271
69.62 G. Hirzinger, J. Butterfaß, M. Fischer, M. Grebenstein, M. Hähnle, H. Liu, I. Schaefer, N. Sporer: A mechatronics approach to the design of lightweight arms and multi-fingered hands, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000)
69.63 M.A. Diftler, J.S. Mehling, M.E. Abdallah, N.A. Radford, L.B. Bridgwater, A.M. Sanders, R.S. Askew, D.M. Linn, J.D. Yamokoski, F.A. Permenter, B.K. Hargrave, R. Piatt, R.T. Savely, R.O. Ambrose: Robonaut 2 – The first humanoid robot in space, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 2178–2183
69.64 M. Bluethmann, R. Ambrose, R. Askew, M. Goza, C. Lovechik, D. Magruder, M.A. Differ, F. Rehnmark: Robonaut: A robotic astronaut's assistant, Int. Conf. Adv. Robotics (2001)
69.65 C. Ott, B. Henze, D. Lee: Kinesthetic teaching of humanoid motion based on whole-body compliance control with interaction-aware balancing, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 4615–4621
69.66 C. Ott, O. Eiberger, W. Friedl, B. Bauml, U. Hillenbrand, C. Borst, A. Albu-Schäffer, B. Brunner, H. Hirschmuller, S. Kielhofer, S. Kielhofer, R. Konietschke, M. Suppa, T. Wimbock, F. Zacharias, G. Hirzinger: A humanoid two-arm system for dexterous manipulation, IEEE-RAS Int. Conf. Humanoid Robots (2006) pp. 276–283
69.67 M.W. Spong: Modeling and control of elastic joint robots, ASME J. Dyn. Syst. Meas. Control **109**(4), 310–319 (1987)
69.68 G. Hirzinger, N. Sporer, M. Schedl, J. Butterfaß, M. Grebenstein: Torque-controlled lightweight arms and articulated hands: Do we reach technological limits now?, Int. J. Robotics Res. **23**(4/5), 331–340 (2004)
69.69 B. Vanderborght, B. Verrelst, R.V. Ham, M.V. Damme, D. Lefeber, B.M.Y. Duran, P. Beyl: Exploiting natural dynamics to reduce energy consumption by controlling the compliance of soft actuators, Int. J. Robotics Res. **25**(4), 343–358 (2006)
69.70 S. Haddadin, M. Weis, A. Albu-Schäffer, S. Wolf: Optimal control for maximizing link velocity of robotic variable stiffness joints, IFAC World Congr. (2011) pp. 3175–3182
69.71 S. Haddadin, T. Laue, U. Frese, S. Wolf, A. Albu-Schäffer, G. Hirzinger: Kick it like a safe robot: Requirements for 2050, Robotics Auton. Syst. **57**, 761–775 (2009)
69.72 A. Albu-Schäffer, O. Eiberger, M. Grebenstein, S. Haddadin, C. Ott, T. Wimböck, S. Wolf, G. Hirzinger: Soft robotics: From torque feedback controlled lightweight robots to intrinsically compliant systems, IEEE Robotics Autom. Mag. **15**(3), 20–30 (2008)
69.73 B. Vanderborght, A. Albu-Schäffer, A. Bicchi, E. Burdet, D.G. Caldwell, R. Carloni, M.G. Catalano, O. Eiberger, W. Friedl, G. Ganesh, M. Garabini, M. Grebenstein, G. Grioli, S. Haddadin, H. Hoppner, A. Jafari, M. Laffranchi, D. Lefeber, F. Petit, S. Stramigioli, N.G. Tsagarakis, M.V. Damme, R.V. Ham, L.C. Visser, S. Wolf: Variable impedance actuators: A review, Robotics Auton. Syst. **61**(12), 1601–1614 (2013)
69.74 G.A. Pratt, M. Williamson: Series elastics actuators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1995) pp. 399–406
69.75 S. Haddadin, A. Albu-Schäffer, O. Eiberger, G. Hirzinger: New insights concerning intrinsic joint elasticity for safety, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 2181–2187
69.76 S. Haddadin, K. Krieger, N. Mansfeld, A. Albu-Schaffer: On impact decoupling properties of elastic robots and time optimal velocity maximization on joint level, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 5089–5096
69.77 J.-J. Park, H.-S. Kim, J.-B. Song: Safe robot arm with safe joint mechanism using nonlinear spring system for collision safety, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2009) pp. 3371–3376
69.78 N.G. Tsagarakis, S. Morfey, G. Medrano Cerda, L. Zhibin, D.G. Caldwell: Compliant humanoid coman: Optimal joint stiffness tuning for modal frequency control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2013) pp. 673–678
69.79 A. Albu-Schäffer, M. Fischer, G. Schreiber, F. Schoeppe, G. Hirzinger: Soft robotics: What cartesian stiffness can we obtain with passively compliant, uncoupled joints?, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 3295–3301
69.80 M. Garabini, A. Passaglia, F. Belo, P. Salaris, A. Bicchi: Optimality principles in variable stiffness control: The VSA hammer, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 3770–3775
69.81 D. Braun, M. Howard, S. Vijayakumar: Exploiting variable stiffness in explosive movement tasks, Robotics Sci. Syst. (2011)
69.82 U. Mettin, A. Shiriaev: Ball-pitching challenge with an underactuated two-link robot arm, IFAC World Congr. (2011) pp. 1–6
69.83 A. Flagg, K. Maclean: Affective touch gesture recognition for a furry zoomorphic machine, Int. Conf. Tangible Embed. Embodied Interact. (2013) pp. 1–4
69.84 R.M. Voyles, P.K. Khosla: Tactile gestures for human/robot interaction, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1995) pp. 7–13
69.85 S. Haddadin, M. Suppa, S. Fuchs, T. Bodenmüller, A. Albu-Schäffer, G. Hirzinger: Towards the robotic co-worker, Int. Symp. Robotics Res. Lucerne (2009)

69.86 R. Bischoff, J. Kurth, G. Schreiber, R. Koeppe, A. Albu-Schäffer, A. Beyer, O. Eiberger, S. Haddadin, A. Stemmer, G. Grunwald, G. Hirzinger: The KUKA-DLR lightweight robot arm: A new reference platform for robotics research and manufacturing, Int. Symp. Robotics (2010) pp. 1–10

69.87 S. Haddadin, A. Albu-Schäffer, A. De Luca, G. Hirzinger: Collision detection & reaction: A contribution to safe physical human-robot interaction, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 3356–3363

69.88 L.E. Pfeffer, O. Khatib, J. Hake: Joint torque sensory feedback in the control of a PUMA manipulator, IEEE Trans. Robotics Autom. **5**(4), 418–425 (1989)

69.89 G. Plank, G. Hirzinger: Controlling a robot's motion speed by a force-torque-sensor for deburring problems, IFAC Inf. Control Probl. Manuf. Technol. (1982) pp. 97–102

69.90 G. Hirzinger, U. Brunet: Fast and self-improving compliance using digital force-torque control, 4th Int. Conf. Assembly Autom. (1983) pp. 268–281

69.91 V.J. Lumelsky, E. Cheung: Real-time collision avoidance in teleoperated whole-sensitive robot arm manipulators, IEEE Trans. Syst. Man Cybern. **23**(1), 194–203 (1993)

69.92 G. De Maria, C. Natale, S. Pirozzi: Force/tactile sensor for robotic applications, Sens. Actuators A **175**, 60–72 (2012)

69.93 R.S. Dahiya, P. Mittendorfer, M. Valle, G. Cheng, V.J. Lumelsky: Directions toward effective utilization of tactile skin: A review, IEEE Sens. J. **13**(11), 4121–4138 (2013)

69.94 M. Strohmayr: Artificial Skin in Robotics, Ph.D. Thesis (Karlsruhe Institute of Technology, Karlsruhe 2012)

69.95 A. Jain, M.D. Killpack, A. Edsinger, C.C. Kemp: Manipulation in clutter with whole-arm tactile sensing, Int. J. Robotics Res. **32**(4), 458–482 (2013)

69.96 A.J. Schmid, M. Hoffmann, H. Worn: A tactile language for intuitive human-robot communication, IEEE-RAS Int. Conf. Humanoid Robots (2007) pp. 569–576

69.97 S. Yohanan, J.P. Hall, K.E. MacLean, E.A. Croft, H.F.M. Van Der Loos, M.A. Baumann, J. Chang, D. Nielsen, S. Zoghbi, G. Jih Shiang Chang: Affect-driven emotional expression with the haptic creature, Proc. User Interface Softw. Technol. (UIST) (2009) p. 2

69.98 M. Frigola, A. Casals, J. Amat: Human-robot interaction based on a sensitive bumper skin, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 283–287

69.99 T. Mukai, M. Onishi, T. Odashima, S. Hirano: Development of the tactile sensor system of a human-interactive robot, IEEE Trans. Robotics **24**(2), 505–512 (2008)

69.100 B.D. Argall, A.G. Billard: A survey of tactile human-robot interactions, Robotics Auton. Syst. **58**(10), 1159–1176 (2010)

69.101 M. Van den Bergh, D. Carton, R. De Nijs, N. Mitsou, C. Landsiedel, K. Kuehnlenz, D. Wollherr, L. Van Gool, M. Buss: Real-time 3D hand gesture interaction with a robot for understanding directions from humans, IEEE Int. Symp. Robot Hum. Interact. Commun. (2011) pp. 357–362

69.102 M. Sigalas, M. Pateraki, I. Oikonomidis, P. Trahanias: Robust model-based 3D torso pose estimation in RGB-D sequences, IEEE Int. Conf. Computer Vis. Work. (2013) pp. 315–322

69.103 N. Hogan: Impedance Control: An Approach to Manipulation: Part I – Theory, Part II – Implementation, Part III – Applications, J. Dyn. Syst. Meas. Control **107**, 1–24 (1985)

69.104 J. Craig, M. Raibert: A systematic method for hybrid position/force control of a manipulator, IEEE Computer Softw. Appl. Conf. (1979) pp. 446–451

69.105 C. Yang, G. Gowrishankar, S. Haddadin, S. Parusel, A. Albu-Schäffer, E. Burdet: Human like adaptation of force and impedance in stable and unstable interactions, IEEE Trans. Robotics **27**(5), 918–930 (2010)

69.106 A. Stemmer, A. Albu-Schäffer, G. Hirzinger: An analytical method for the planning of robust assembly tasks of complex shaped planar parts, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2007) pp. 317–323

69.107 N. Hogan: On the stability of manipulators performing contact tasks, IEEE Int. Conf. Robotics Autom. **4**(6), 677–686 (1988)

69.108 C. Ott, R. Mukherjee, Y. Nakamura: Unified impedance and admittance control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010) pp. 554–561

69.109 T.R. Kurfess: *Robotics and Automation Handbook* (CRC, Boca Raton 2010)

69.110 F. Caccavale, C. Natale, B. Siciliano, L. Villani: Six-DOF impedance control based on angle/axis representations, IEEE Trans. Robotics Autom. **15**(2), 289–300 (1999)

69.111 L. Sentis, O. Khatib: Synthesis of whole-body behaviors through hierarchical control of behavioral primitives, Int. J. Humanoid Robotic **s**, 505–518 (2005)

69.112 A. Dietrich, T. Wimböck, A. Albu-Schäffer: Dynamic whole-body mobile manipulation with a torque controlled humanoid robot via impedance control laws, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 3199–3206

69.113 A. Albu-Schäffer, C. Ott, U. Frese, G. Hirzinger: Cartesian impedance control of redundant robots: Recent results with the DLR-light-weight-arms, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 3704–3709

69.114 A. Albu-Schäffer, C. Ott, G. Hirzinger: A unified passivity-based control framework for position, torque and impedance control of flexible joint robots, Int. J. Robotics Res. **26**, 23–39 (2007)

69.115 L. Zollo, B. Siciliano, A. De Luca, E. Guglielmelli, P. Dario: Compliance control for an anthropomorphic robot with elastic joints: Theory and experiments, J. Dyn. Syst. Meas. Control **127**(3), 321–328 (2005)

69.116 R. Platt Jr., M. Abdallah, C. Wampler: Multiple-priority impedance control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 6033–6038

69.117 S. Stramigioli: *Modeling and IPC Control of Interactive Mechanical Systems: A Coordinate-Free Approach* (Springer, New York 2001)

69.118 C.-C. Cheah, D. Wang: Learning impedance control for robotic manipulators, IEEE Trans. Robotics Autom. **14**(3), 452–465 (1998)

69.119 Y. Li, S. Sam Ge, C. Yang: Learning impedance control for physical robot–environment interaction, Int. J. Control **85**(2), 182–193 (2012)

69.120 S. Jung, T.C. Hsia: Neural network impedance force control of robot manipulator, IEEE Trans. Ind. Electron. **45**(3), 451–461 (1998)

69.121 A.M. Zanchettin, P. Rocco: Path-consistent safety in mixed human-robot collaborative manufacturing environments, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2013) pp. 1131–1136

69.122 D.W. Franklin, E. Burdet, K.P. Tee, R. Osu, C.-M. Chew, T.E. Milner, M. Kawato: CNS learns stable, accurate, and efficient movements using a simple algorithm, J. Neurosci. **28**(44), 11165–11173 (2008)

69.123 J.-J.E. Slotine, W. Li: *Applied Nonlinear Control* (Prentice Hall, Englewood Cliffs 1991)

69.124 E. Gribovskaya, A. Kheddar, A. Billard: Motion learning and adaptive impedance for robot control during physical interaction with humans, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 4326–4332

69.125 K. Kronander, A. Billard: Learning compliant manipulation through kinesthetic and tactile human-robot interaction, IEEE Trans. Haptics **7**(3), 367–380 (2014)

69.126 A. Ajoudani, N.G. Tsagarakis, A. Bicchi: Tele-impedance: Towards transferring human impedance regulation skills to robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 382–388

69.127 S. Calinon, P. Evrard, E. Gribovskaya, A. Billard, A. Kheddar: Learning collaborative manipulation tasks by demonstration using a haptic interface, Int. Conf. Adv. Robotics (2009) pp. 1–6

69.128 J.R. Medina, M. Lawitzky, A. Mörtl, D. Lee, S. Hirche: An experience-driven robotic assistant acquiring human knowledge to improve haptic cooperation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2011) pp. 2416–2422

69.129 D. Lee, C. Ott, Y. Nakamura: Mimetic communication model with compliant physical contact in human-humanoid interaction, Int. J. Robotics Res. **29**(13), 1684–1704 (2010)

69.130 A. Jain, B. Wojcik, T. Joachims, A. Saxena: Learning trajectory preferences for manipulators via iterative improvement, Adv. Neural Inf. Process. Syst. (2013) pp. 575–583

69.131 K. Suita, Y. Yamada, N. Tsuchida, K. Imai, H. Ikeda, N. Sugimoto: A failure-to-safety *kyozon* system with simple contact detection and stop capabilities for safe human – Autonomous robot coexistence, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 3089–3096

69.132 Y. Yamada, Y. Hirasawa, S. Huang, Y. Umetani, K. Suita: Human-robot contact in the safeguarding space, IEEE/ASME Trans. Mechatron. **2**(4), 230–236 (1997)

69.133 S. Takakura, T. Murakami, K. Ohnishi: An approach to collision detection and recovery motion in industrial robot, Annual Conf. IEEE Ind. Electron. Soc. (1989) pp. 421–426

69.134 S. Morinaga, K. Kosuge: Collision detection system for manipulator based on adaptive impedance control law, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 1080–1085

69.135 K. Kosuge, T. Matsumoto, S. Morinaga: Collision detection system for manipulator based on adaptive control scheme, Trans. Soc. Instrum. Control Eng. **39**, 552–558 (2003)

69.136 A. De Luca, A. Albu-Schäffer, S. Haddadin, G. Hirzinger: Collision detection and safe reaction with the DLR-III lightweight manipulator arm, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 1623–1630

69.137 A. De Luca, R. Mattone: Actuator fault detection and isolation using generalized momenta, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2003) pp. 634–639

69.138 H.-B. Kuntze, C.W. Frey, K. Giesen, G. Milighetti: Fault tolerant supervisory control of human interactive robots, IFAC Workshop Adv. Control Diagn. (2003) pp. 55–60

69.139 S. Parusel, S. Haddadin, A. Albu-Schäffer: Modular state-based behavior control for safe human-robot interaction: A lightweight control architecture for a lightweight robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 4298–4305

69.140 N. Mansfeld, S. Haddadin: Reaching desired states time-optimally from equilibrium and vice versa for visco-elastic joint robots with limited elastic deflection, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2014) pp. 3904–3911

69.141 C.A.C. Parker, E.A. Croft: Design & personalization of a cooperative carrying robot controller, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 3916–3921

69.142 N. Hogan, S.P. Buerger: Impedance and interaction control. In: *Robotics and Automation Handbook*, ed. by T.R. Kurfess (CRC, Boca Raton 2005) pp. 19-1–19-24

69.143 R. Ikeura, H. Inooka: Variable impedance control of a robot for cooperation with a human, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1995) pp. 3097–3102

69.144 R. Ikeura, T. Moriguchi, K. Mizutani: Optimal Variable Impedance Control for a Robot and Its Application To Lifting an Object with a Human, IEEE Int. Workshop Robot Hum. Interact. Commun. (2002) pp. 500–505

69.145 K. Kosuge, N. Kazamura: Mobile robot helper, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 583–588

69.146 N. Nejatbakhsh, K. Kosuge: Adaptive guidance for the elderly based on user intent and physical im-

pairment, IEEE Int. Symp. Robot Hum. Interact. Commun. (2006) pp. 510–514
69.147 M. Lawitzky, A. Mörtl, S. Hirche: Load sharing in human-robot cooperative manipulation, IEEE Int. Symp. Robot Human Interact. Commun. (2010) pp. 185–191
69.148 V. Duchaine, B. Mayer St.-Onge, C. Gosselin: Stable and intuitive control of an intelligent assist device, IEEE Trans. on Haptics **5**(2), 148–159 (2012)
69.149 A. Mörtl, M. Lawitzky, A. Kucukyilmaz, M. Sezgin, C. Basdogan, S. Hirche: The role of roles: Physical cooperation between humans and robots, Int. J. Robotics Res. **31**(13), 1656–1674 (2012)
69.150 T. Flash, N. Hogan: The coordination of arm movements: Mathematical model, J. Neurosci. **5**(7), 1688–1703 (1985)
69.151 Y. Maeda, T. Hara, T. Arai: Human-robot cooperative manipulation with motion estimation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 2240–2245
69.152 S. Miossec, A. Kheddar: Human motion in cooperative tasks: Moving object case study, Robotics Biomim. (2009) pp. 1509–1514
69.153 D. Kulić, E. Croft: Pre-collision strategies for human robot interaction, Auton. Robots **22**(2), 149–164 (2007)
69.154 K. Ikuta, H. Ishii, M. Nokata: Safety evaluation method of design and control for human-care robots, Int. J. Robotics Res. **22**(5), 281–298 (2003)
69.155 D. Kulić, E.A. Croft: Real-time safety for human-robot interaction, Robotics Auton. Syst. **54**(1), 1–12 (2006)
69.156 Y. Tamura, T. Fukuzawa, H. Asama: Smooth collision avoidance in human-robot coexisting environment, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 3887–3892
69.157 B. Lacevic, P. Rocco, A.M. Zanchettin: Safety assessment and control of robotic manipulators using danger field, IEEE Trans. Robotics **29**(5), 1257–1270 (2013)
69.158 S. Haddadin, H. Urbanek, S. Parusel, D. Burschka, J. Roßmann, A. Albu-Schäffer, G. Hirzinger: Real-time reactive motion generation based on variable attractor dynamics and shaped velocities, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 3109–3116
69.159 F. Flacco, T. Kroger, A. De Luca, O. Khatib: A depth space approach to human-robot collision avoidance, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2012) pp. 338–345
69.160 E.A. Sisbot, L.F. Marin-urias, R. Alami, T. Siméon: A human aware mobile robot motion planner, IEEE Trans. Robotics **23**(5), 874–883 (2007)
69.161 J. Mainprice, E.A. Sisbot, T. Siméon, R. Alami: Planning safe and legible hand-over motions for human-robot interaction, IARP Workshop Tech. Chall. Dependable Robots Hum. Environ. (2010) p. 7
69.162 J. Mainprice, E.A. Sisbot, L. Jaillet, J. Cortes, R. Alami, T. Simeon: Planning human-aware motions using a sampling-based costmap planner, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 5012–5017
69.163 A. Clodic, H. Cao, S. Alili, V. Montreuil, R. Alami, R. Chatila: SHARY: A supervision system adapted to human-robot interaction, Springer Tracts Adv. Robotics **54**, 229–238 (2009)
69.164 E. Bicho, W. Erlhagen, L. Louro, E. Costa e Silva: Neuro-cognitive mechanisms of decision making in joint action: A human–robot interaction study, Hum. Mov. Sci. **30**(5), 846–868 (2011)
69.165 T.S. Dahl, A. Paraschos: A force-distance model of humanoid arm withdrawal reflexes, Lect. Notes Comput. Sci. **7429**, 13–24 (2012)
69.166 S. Parusel, H. Widmoser, S. Golz, T. Ende, N. Blodow, M. Saveriano, K. Krieger, A. Maldonado, I. Kresse, R. Weitschat, D. Lee, M. Beetz, S. Haddadin: Human-Robot interaction Planning, AAAI Video Competition, http://www.aaaivideos.org/2014/15_hri_planning/ (2014)

Multimedia Contents

70. Human–Robot Augmentation

Massimo Bergamasco, Hugh Herr

The development of robotic systems capable of sharing with humans the load of heavy tasks has been one of the primary objectives in robotics research. At present, in order to fulfil such an objective, a strong interest in the robotics community is collected by the so-called wearable robots, a class of robotics systems that are worn and directly controlled by the human operator. Wearable robots, together with powered orthoses that exploit robotic components and control strategies, can represent an immediate resource also for allowing humans to restore manipulation and/or walking functionalities.

The present chapter deals with wearable robotics systems capable of providing different levels of functional and/or operational augmentation to the human beings for specific functions or tasks. Prostheses, powered orthoses, and exoskeletons are described for upper limb, lower limb, and whole body structures. State-of-the-art devices together with their functionalities and main components are presented for each class of wearable system. Critical design issues and open research aspects are reported.

70.1 **Concept and Definitions** 1876

70.2 **Upper Limb Wearable Systems** 1877
- 70.2.1 Location of Placement 1877
- 70.2.2 Degrees of Freedom 1877
- 70.2.3 Kinematic Architecture and Implementation 1878
- 70.2.4 Actuation and Power Transmission Methods 1878
- 70.2.5 Type of Sensorization 1880
- 70.2.6 Hand Exoskeletons 1880

70.3 **Lower Limb Wearable Systems** 1882
- 70.3.1 Lower Extremity Autonomous Wearable Robotics 1882
- 70.3.2 Neuromechanical Basis for Wearable-Robotic Leg Design 1882
- 70.3.3 Design Principles for Wearable Leg Robotics 1882
- 70.3.4 Series and Parallel-Elastic Actuation for Power Amplification 1883
- 70.3.5 Series-Elastic Actuation for Minimizing Motor Work..... 1885
- 70.3.6 Neuromuscular-Based Reflexive Control 1885
- 70.3.7 Leg Exoskeletons 1887

70.4 **Whole Body Wearable Systems**..... 1889
- 70.4.1 General Issues..... 1889
- 70.4.2 Specific Design 1890

70.5 **Control of Human–Robot Augmentation Systems**..... 1892
- 70.5.1 A Historical Perspective 1892
- 70.5.2 Power Augmentation through Proprioceptive Sensors .. 1895
- 70.5.3 EMG Force Control and Power Augmentation 1896
- 70.5.4 Power Augmentation through Exteroceptive Sensors.... 1897
- 70.5.5 Control Schemes..... 1898

70.6 **Conclusions and Further Developments** 1902

Video-References..... 1902

References..... 1902

70.1 Concept and Definitions

Since early developments, one of the primary objectives of technology has been that of relieving humans from heavy tasks [70.1]. The concept of human–robot augmentation refers to the use of robotics systems to increase human functionalities in different operational contexts [70.2]. In general, the augmentation of humans' performances through the use of robotic systems refers solely to the increase in mechanical capabilities involving motion control and generation of scaled levels of forces. Although complex operations involve and exploit perceptual capabilities, in this chapter the concept of human–robot augmentation is not referred to perceptual functionalities only. Examples of working conditions in which human–robot augmentation can play an essential role for the completion of the task can be considered, e.g., operations for debris removal after an earthquake in which human operators, wearing such systems, can move and operate in a more flexible and accurate way with respect to machines like cranes or excavators.

Other scenarios are represented by tasks in which the number of repetitions, instead of level of loads to be operated, is high and becomes the critical factor for decreasing fatigue, thus maintaining long-term levels of humans' safety.

By definition, wearable robots follow the human operator's movements and in this way can inherently and fully exploit human's dexterity, thus assuring a higher level of flexibility in addressing a specific task with respect to usual loading machines. A first and fundamental characteristic, and at the same time a strict design constraint, of wearable robots is the condition of possessing a physical structure which is always in contact with the human operator's body during operation. As a consequence, wearable robots design must take into account both biomechanics and human motor control aspects.

Research on wearable robots has received a strong contribution from the development of haptic interfaces, i. e., robotic systems that are able to generate forces on the human body (force feedback) while maintaining a high level of naturalness of user's movements. The development of haptic technologies in the last two decades has focused on the importance of designing robotic systems able to generate forces at the level of the hand or other parts of the human body. As a consequence, an increased attention of the scientific robotic community toward wearable robots has been recorded. In fact, most of the recent developments on robotic exoskeletons derive from their application in neurorehabilitation [70.3], where the robotic system is used to kinetically assist the patient's limb during the performance of specific exercises.

The augmentation of human performances can be achieved with different types of robotic systems, e.g. from teleoperated robots to self-standing robotic systems amplifying human actions. Examples of robotic systems directly operated by the human operator are the Cobot system [70.4] and the intelligent autonomous system (IAS) [70.5]. However, in the framework of the present review work, the analysis is performed by considering only robotic systems that can be worn by the user.

At present, few definitions of wearable robotic systems have been considered. Herr makes a first classification between serial and parallel system [70.6].

Serial systems are considered those that increase the length and reachability of a limb, while parallel structures contribute to the effective augmentation of torques and forces generated by the human limb. Serial systems perform their action by exploiting elastic components integrated in their structure, thus not requiring a specific computer-based control system. Other classes of wearable robotic systems can be considered the so-called *hybrid* wearable systems, i. e., parallel robotic structures worn by the human operator in which the command for their motion is achieved by FES (functional electric stimulation). In this way, the control of the robotic structure is limited and does not allow a complete force control by the user [70.7]. Hybrid systems are not considered in this chapter.

Wearable robotics systems for human augmentation can be divided into three different categories: (i) prostheses, for the functional replacement of human limbs; (ii) powered orthoses, whose function is to actively operate in parallel with unhealthy human joints; and (iii) robotic exoskeletons that operate in parallel with human limbs [70.8].

The design of the above three types of robotic systems shows different approaches whether they are referred to upper, lower limbs, or to the whole body, respectively. In the following, the analysis of wearable robotic systems is given by considering present achievements in terms of devices, robotic components utilized for the actuation, control systems architectures, and devised application domains in which they are presently utilized.

70.2 Upper Limb Wearable Systems

Upper limb exoskeletons (ULEs) are wearable robots that are placed in parallel and work in concert with the human arm. Wearer's motions that fall within ULEs coverage are shoulder elevation/depression and retraction/protraction which are provided by the human's sterno-clavicular articulation; shoulder flexion/extension, abduction/adduction and medial/lateral rotation which are provided by the human's gleno-humeral articulation; elbow flexion/extension which is provided by the human's humero-ulnar articulation; forearm prono/supination which is provided by the human's radio-ulnar articulation; wrist flexion/extension and radial/ulnar deviation which are provided by the human's radio-carpal, mid-carpal, and intra-carpal articulations; and finger abduction/adduction and flexion/extension which are provided by the human's metacarpo-phalangeal and inter-phalangeal joints.

Proposed usages of ULEs are as rehabilitation devices, motion assists, human power/force amplifiers, and haptic interfaces. Depending on the specific application, the available ULEs differ in location of placement, number of degrees of freedom (DOFs) and their subdivision between active and passive DOFs, type of kinematic architecture and implementation, type of actuators and power transmission methods, types of sensorization, and number and place of connection points between ULE and wearer.

70.2.1 Location of Placement

With regard to location of placement, a number of systems have been developed which differ from the type of segments they can cover; for instance shoulder–elbow–wrist–finger complex [70.10, 11], shoulder–elbow–finger complex [70.9] (Fig. 70.1), shoulder–elbow–wrist complex [70.12–16] (Fig. 70.2), shoulder–elbow–forearm complex [70.17–20] (Fig. 70.3), shoulder–elbow complex [70.21–27] (Fig. 70.4), elbow–wrist complex [70.28, 29], wrist–finger complex [70.30], and the hand (fingers only) [70.31–33].

Fig. 70.1 Upper limb exoskeletons for the shoulder–elbow–wrist–finger complex (PERCRO-II, L-Exos, after [70.9])

70.2.2 Degrees of Freedom

With regard to DOFs, developed systems largely differ depending on the specific location of placement. In particular, exoskeletons with up to 9-DOFs have been developed for the shoulder–elbow–wrist–finger complex [70.16], whereas exoskeletons with up to 18-DOFs have been developed for the wrist–finger complex [70.30].

Focusing on ULEs covering at least the shoulder–elbow–forearm complex, the developed systems typically feature [70.9–13, 15–20] the following: three active DOFs for shoulder flexion/extension, abduction/adduction and medial/lateral rotation; one active DOF for elbow flexion/extension; and one DOF, either active or passive, for forearm prono/supination.

Fig.70.2a–c Upper limb exoskeletons for the shoulder–elbow–wrist complex: **(a)** RUPERT (after [70.14]); **(b)** ARMIN (after [70.15]); **(c)** SUEFUL-7 (after [70.16])

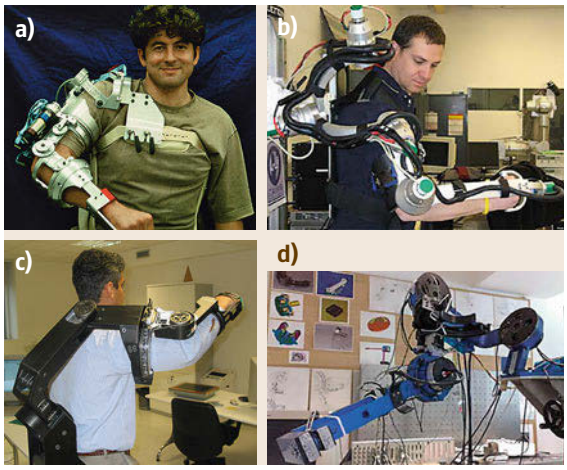


Fig.70.3a–d Upper limb exoskeletons for the shoulder–elbow–forearm complex: **(a)** PERCRO-I (after [70.34]); **(b)** MGA (after [70.19]); **(c)** PERCRO-III (after [70.9]), **(d)** RehabExos (after [70.20])

Some of these systems are also provided with a wrist having two active DOFs for flexion/extension and radial/ulnar deviation [70.10, 12, 13, 16]; an active [70.11, 15, 19, 23] or passive [70.26] DOF for shoulder elevation/depression (some systems also exist in which shoulder elevation/depression is coupled to shoulder abduction/adduction [70.15, 16, 35]); and an active [70.23] or passive [70.11, 26] DOF for shoulder retraction/protraction. Although the availability of ergonomic evaluation data is still limited, the inclusion of shoulder elevation/depression is deemed necessary for large-range-of-motion ULEs with arm elevation angles larger than 60° [70.15].

70.2.3 Kinematic Architecture and Implementation

Despite a few ULEs [70.10, 12, 26], the majority of the developed systems feature a kinematic architecture that is isomorphic to that of the wearer's upper limb (which means that the rotation axes of ULE joints are coaxial to those of the wearer's articulations). This choice usually reduces the chances of wearer–exoskeleton interference and enables the wearer to be connected to the ULE at different points over multiple links. With regard to implementation, different solutions have been proposed depending on the considered ULE articulation. In particular, three shoulder rotations have been implemented in the following different ways: (1) via three standard revolute pairs with either nonintersecting axes [70.10, 26, 26, 36] or with axes intersecting away from the center of the wearer's gleno-humeral articulation [70.12], which leads to system designs that are not isomorphic with the wearer's limb kinematics; (2) via three standard revolute pairs with axes intersecting at the center of the wearer's gleno-humeral articulation [70.19, 23], which usually leads to ULEs with either limited range of motion or excessive encumbrance; (3) as in (2) but with the last revolute joint being a circular guide [70.9, 11, 13, 16, 20, 37], which enables for a more compact design that is, however, affected by a limited singularity-free workspace; (4) as in (2) but with the second revolute joint being a circular guide [70.38], which provides both a very compact design and a rather wide singularity-free workspace. Regarding the additional DOFs at the shoulder, both elevation/depression and retraction/protraction motions have been implemented via either revolute [70.19, 23] or prismatic [70.11, 15, 26] pairs. Elbow flexion/extension has been implemented almost univocally via a standard revolute pair. As for the wrist, flexion/extension and radial/ulnar deviation have been implemented by using two revolute joints with either intersecting axes [70.10, 12, 13] or slightly off-set axes [70.16]. In fact, in the human wrist, the radial/ulnar deviation axis passes distally with respect to the flexion/extension axis by approximately 5 mm.

70.2.4 Actuation and Power Transmission Methods

Actuation methods have also been quite variegate. Besides passive systems employing springs (or elastic bands) [70.22] and hydraulic disk brakes [70.26], ULEs have been equipped with hydraulic motors [70.10], pneumatic cylinders [70.24], pneumatic muscles [70.12, 14], and electric motors. Hydraulic motors provide very large power-to-weight ratio, high stiffness, and direct-drive operation; however, they feature low energetic efficiency and require bulky, high-pressure fluid components (power supply, valves, and tubing). Pneumatic cylinders and muscles provide large power-to-weight ratio and direct-drive operation; however they feature low energetic-efficiency, low stiffness, slow and nonlinear response, and require bulky and noisy fluid components (power supply, valves and tubing). In addition, pneumatic muscles can generate forces in one direction only, and thus they need to be used in an agonist–antagonist arrangement [70.12]. Electric motors are very efficient, easy to control, and have good power-to-weight ratio; however, they work better at high speeds. In the existing ULEs, electric motors are rarely used in direct-drive operation, but rather they are combined with either standard gearboxes (with spur [70.16] or planetary [70.9] gear-heads) or harmonic drives [70.19, 20, 27].

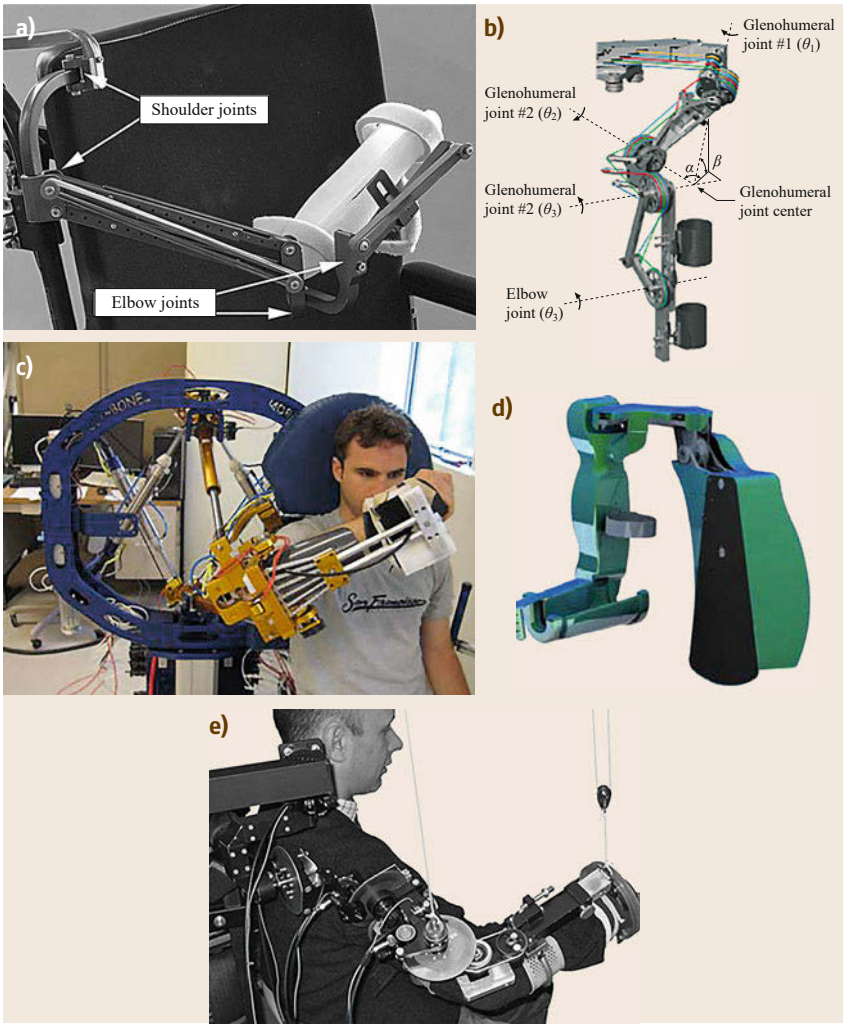


Fig.70.4a–e Upper limb exoskeletons for the shoulder–elbow complex: **(a)** WREX (after [70.22]); **(b)** MEDARM (after [70.23]); **(c)** BONES (after [70.24]); **(d)** ABLE (after [70.25]); **(e)** DAMPACE (after [70.26])

In the existing ULEs, hydraulic motors and pneumatic muscles have been located directly across single exoskeleton joints [70.10, 12, 14], whereas pneumatic cylinders have been used in an in-parallel arrangement to activate multiple joints [70.24]. Instead, electric motors have been placed either at the joints [70.19, 20] or in a remote location [70.9, 13, 23] (usually in the ULE base frame). This second choice has been made primarily in order to reduce the weight and inertia of moving exoskeleton parts, which enables the use of smaller motors and minimizes the disturbances on the wearer's natural motions.

ULEs with remotely placed actuators require appropriate transmission systems to transfer the power from the actuators to the joints. Except from specific cases [70.15, 23, 35] in which transmission linkages and timing belts have been used, the common power transmission solution for ULEs has consisted in cable-drive systems, either closed-ended [70.9, 13] or open-ended [70.23]. Closed-ended transmissions require long cables, whereas open-ended transmissions require more motors than the number of actuated DOFs. Both open- and closed-ended cable-drive systems are lightweight, highly efficient, and back-drivable; thus they are very good solutions to transmit mechanical power over long distances. In addition, cable drives enable the easy implementation of speed-reduction stages, which should be placed as close as possible to the joint in order to maximize overall transmission stiffness [70.9]. Moreover, cable-drive transmissions spanning multiple joints make it possible to distribute loads across several cables, which can be used to minimize the torque requirements of the actuators [70.23].

70.2.5 Type of Sensorization

Apart from few passive systems only conceived as weight compensation devices [70.39], all the developed ULEs are equipped with position sensors (potentiometers or encoders), which are placed at the motors and/or at the joints so as to make it possible to estimate/control the overall system configuration. Besides, the considered ULEs largely differ from the point of view of force/torque sensorization. Thanks to the intrinsic low friction and high back-drivability, exoskeletons with cable-drives have been designed to operate without the need of force/torque sensors [70.9, 13, 23]. All the other ULEs are instead equipped with this kind of sensorization, which can be either concentrated at the points of connection with the wearer [70.11, 15, 16, 19] or distributed at every ULE joint [70.20]. The use of torque sensors distributed at every ULE joint enables for multicontact force control at multiple points distributed over multiple links and, additionally, makes it possible to design very robust force-feedback controllers that are rather insensitive to exoskeleton link variable inertia, transmission compliance, friction losses, actuator torque ripples, and gear teeth wedging actions [70.20]. The use of multi-DOF force/torque sensors concentrated at the points of wearer–ULE interaction makes the use of the exoskeleton less flexible and the controller less robust, but enables for a more accurate estimation and control of the endpoint forces that are exchanged with the wearer. In general, ULEs equipped with force/torque sensorization provide better haptic feedback than those relying on cable-drive transmissions only.

70.2.6 Hand Exoskeletons

The functionality of hand exoskeletons (HE) consists in exerting programmable forces on human fingers. Hand exoskeletons can be classified in two functional categories depending on the capability of exerting forces on a single phalanx (usually the distal phalanx correspondent to the fingertip) or on multiple phalanges (Fig. 70.5).

Multiphalanx hand exoskeletons (MPHE) are devices able to exert different forces on at least two of the phalanges of the same finger: usually the force can be applied on a fixed direction normal to the phalanx axis and belonging to the medial plane of the finger.

Single-phalanx hand exoskeletons (SPHE) are able to exert forces on one phalanx of the finger, usually the distal one only; some devices are able to generate a force only along a fixed direction but more commonly they can exert forces with any desired orientation.

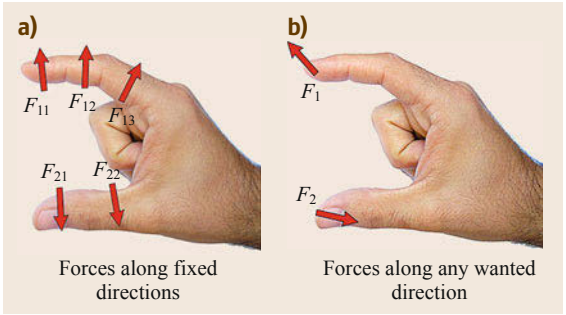


Fig. 70.5 (a) Multiphalanx and (b) single-phalanx scheme for hand exoskeletons

Another taxonomy of hand exoskeletons considers two other categories:

- Anthropomorphic devices: the kinematics of the HE is morphologically similar to the human fingers
- Nonanthropomorphic devices: the kinematics is morphologically different from the human fingers.

The scientific literature in the last 30 years reports many works dealing with different HE designs mainly addressing three application fields: (1) teleoperation and virtual environments in which the HE are utilized as master controllers providing force feedback or for generating illusory forces to the hand of the human operator; (2) neurorehabilitation, in which HE are used for restoring hand functionalities after strokes or other neural impairments; (3) space applications in which HE are employed as active devices for extending grasping forces of the astronauts in extra vehicular activities.

The first known example of HE was introduced in the field of teleoperation by *Zarudiansky* back in 1981 [70.40]. The inventor has registered a patent for a tele-manipulation system equipped with a master device able to provide force feedback on different phalanges of the human hand. At the Jet Propulsion Laboratory Pasadena (JPL) in 1988, *Jau* [70.41] built a complete teleoperated master/slave system comprehending a hand exoskeleton device for four fingers. In the same years *Burdea* et al. developed a pneumatic actuated hand force feedback device [70.42], called Rutgers Master, able to exert a single force to the distal phalanx along a direction dependent on the kinematics of the device and on the postion of the finger. *Bergamasco* [70.37] developed a four-finger MPHE. A nonanthropomorphic device was realized by *Koyama* et al. at Keyo University; this device is a three-finger exoskeleton for thumb, index, and middle finger with passive clutches actuation [70.43]. Such a device is anchored at the user's wrist and is able to exert forces in every direction belonging to the sagittal plane

of the fingers. *Frisoli* et al. [70.9] developed another nonanthropomorphic HE, called Pure Form Hand-Exos for virtual environment (VE) applications that features a very complex steel cable transmission studied for an optimal use of motor torque. *Nakagawara* et al. [70.44] built a device integrating the concept of encountered haptics: the device possesses 1-DOF for each finger and it is able to track the finger without any contact (even at the fingertip) through a noncontact sensor placed in correspondence of the nail. When contact is detected in the remote/virtual environment, a plate is moved against the user's fingertip. Another example of anthropomorphic hand exoskeleton for VE applications has been developed by *Fontana* et al. [70.32] with the aim of maximizing force accuracy and resolution (Fig. 70.6).

Other examples of HEs have been developed for medical applications. In this field, HEs are employed for rehabilitation procedures or as power-assisting devices (orthoses). In rehabilitation procedures a HE is commonly used for exerting forces (or trajectories) and, at the same time, measuring velocities and positions (or force) of the fingers. *Gomez* et al. [70.45] have integrated the Rutgers Master in a VE rehabilitation system. *Wege* et al. [70.46] developed a 4-DOF HE with a modular structure, actuated by Bowden cables. *Mulas* et al. [70.47] realize a HE that makes use of electromyography (EMG) signals for predicting the movements of the user finger. *Ito* et al. [70.48] built a complex HE for rehabilitation purposes with 4-DOF for the index, medium, and ring fingers and 5-DOF for the thumb, able to exert controlled torque on each phalanx.

HEs can implement the functionality of supporting people with permanent disabilities to perform common activities for daily living (ADL). Such devices, called hand orthoses or assistive HE, are controlled as power extenders allowing the user to exert only a fraction of the force required to accomplish the task. The first orthosis for the finger was developed by *Brown* et al. [70.49]. The realized device was a HE composed by three rings attached on the phalanges that were actuated with pulling cables. Other examples of hand orthoses are described in [70.4, 39]. Another field where HEs have proved their usefulness is in assisting astronauts in performing EVA (extra-vehicular activities). Due to the pressure difference, the suits that the astronauts wear during EVA require high force to be bended and represent a major constraint especially for manipulative tasks. For this reason, custom-designed HEs

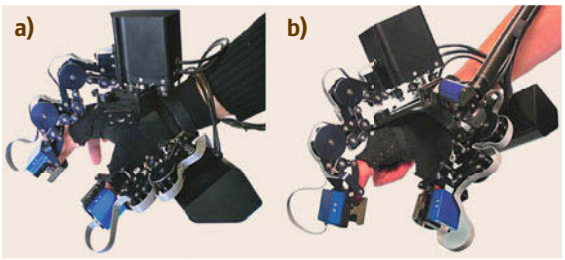


Fig.70.6a,b Hand exoskeleton (after [70.32]) **(a)** without and **(b)** with tracking system

are used as an active tool to overcome the stiffness of the gloves. *Shields* et al. [70.50] developed the first HE for space applications. Such a device was an anthropomorphic HE with three active fingers; the thumb was excluded and the little finger was connected together with the ring finger. The HE has two DOFs for each finger and exploits a special mechanisms for the implementation of the finger flexion joints. The anthropomorphic kinematics is obtained through remote center of rotation joints. In 2001, *Yamada* et al. [70.51] developed, for the same space applications, another simplified HE called Skil Mate in which only the first flexion DOF of each finger is actuated. The HE developed by *Wang* et al. [70.52] is still utilized for the simulation and evaluation of different space gloves. The device features a kinematics implementation that is close to the HE developed by Shields but in this case the device has three DOFs coupled in a way that only one actuator is needed for each finger. Also in this case, flexion joints are implemented through remote center of rotations allowing the HE joints to closely correspond with the natural joints (metacarpophalangeal MCP, interphalangeal IP, distal D) of the human hand.

In general, it can be observed that HEs realized for VE and teleoperation have several mechanical features that are not present in the devices conceived for rehabilitation, ADL or EVA applications. VE and teleoperation application require in fact the simulation of fine haptic interaction and generally more demanding performances are needed in terms of mechanical features. In VE and teloperation, the design of HE requires to minimize friction, inertia, and weight. On the other side, in other applications, a higher level of friction can be tolerated, allowing the use of Bowden cables for delocalizing the motors, or a simplified kinematic structure can be acceptable allowing exerting forces with less accuracy.

70.3 Lower Limb Wearable Systems

Lower extremity autonomous wearable robotics is a general classification covering actuated leg prostheses, orthoses, and exoskeletons.

70.3.1 Lower Extremity Autonomous Wearable Robotics

These robots are defined as active mechanical devices that are worn on a wearer's legs, and work in concert with the wearer's movements, either as permanent assistive devices for persons suffering from leg pathology, or augmentative devices for persons having *normal* physiological leg function. In general, the classification *active leg orthosis* is typically used to describe a device that is used to increase the capacity of a person suffering from leg pathology to ambulate, whereas the classification *leg exoskeleton* is used to describe a device that augments the locomotory performance of an able-bodied wearer. However, the classification *leg exoskeleton* has also been used to describe certain assistive devices that span in parallel across the entirety of the human leg [70.6]. Autonomous wearable leg robots are used as assistive interventions in the treatment of leg disabilities caused by amputation, stroke, cerebral palsy, and multiple sclerosis. In the realm of human augmentation, leg exoskeletons have been constructed for load-carrying augmentation, to improve the economy of walking or running, or to decrease musculoskeletal stress [70.6].

Since robots within this classification are worn by the wearer, it is critical that their design resemble biological leg function, both morphologically and neurologically. For this discussion on lower extremity wearable robotics, we use the term *bionic* to refer to an electromechanical structure designed to emulate, or extend, normal physiological function. A long-standing goal in robotic science is to apply neuromechanical principles of human movement to the development of highly functional wearable leg robotics [70.53]. Critical to this effort is the development of actuator technologies that behave like muscle, device architectures that resemble the body's own musculoskeletal leg design, and control methodologies that exploit principles of biological leg movement. In the next section, we present a computational model of human walking that unifies muscle and joint biomechanics with whole-body metabolism for level-ground walking at self-selected speed. The model underscores critical design principles for wearable leg robotics, including the importance of tendon-like compliance and reflexive control on locomotory performance – discussed in detail in Sect. 70.3.6. These design principles are discussed in detail and then, by way of example, used to analyze the design of a bionic transtibial prosthesis. The chapter concludes with a discussion on critical areas for future research.

70.3.2 Neuromechanical Basis for Wearable-Robotic Leg Design

By understanding the underlying mechanisms that govern individual muscle–tendon behaviors in human locomotory function, roboticists can better design wearable robots that seamlessly integrate dynamically with the biological leg. To this end, in this section we present a computational model of human walking [70.54] that unifies muscle and joint biomechanics with whole-body metabolism for level-ground walking at self-selected speed. The model, shown in Fig. 70.7, comprises ankle, knee, and hip joints connected by rigid bodies representing the human trunk and two, three-segment legs. In the model, muscle–tendon units that dorsiflex the ankle, and flex and extend the knee, are assumed to act as linear springs upon neural activation; each muscle–tendon is modeled as a tendon spring in series with an isometric force source, or a controllable clutch. To provide the mechanical power for step-to-step gait transitions, a Hill-type soleus muscle is modeled to actively plantar flex the ankle throughout mid to terminal stance using muscle state and force as reflex feedback signals. Finally, to stabilize the trunk during stance, and to protract and retract each leg throughout the swing phase, two mono-articular Hill-type muscles actuate the model's hip joint. State transitions of a finite machine were facilitated by the walking model, and its interactions with a modeled ground surface, and each muscle was engaged or activated according to state [70.54]. Following a forward dynamic optimization procedure described in [70.54], the walking model is shown to predict joint biomechanics, as well as whole-body metabolism, as shown in Fig. 70.8, supporting the idea that the preponderance of leg muscle–tendons operates isometrically, affording the relatively high metabolic walking economy of humans.

70.3.3 Design Principles for Wearable Leg Robotics

Neuromechanical models of human locomotory function such as the *Endo* and *Herr* model [70.54] provide critical insights into wearable robotic leg design. Biophysical models that describe the morphology and neural control of human limbs can motivate robotic designs that are quiet, low mass, economical, and stable;

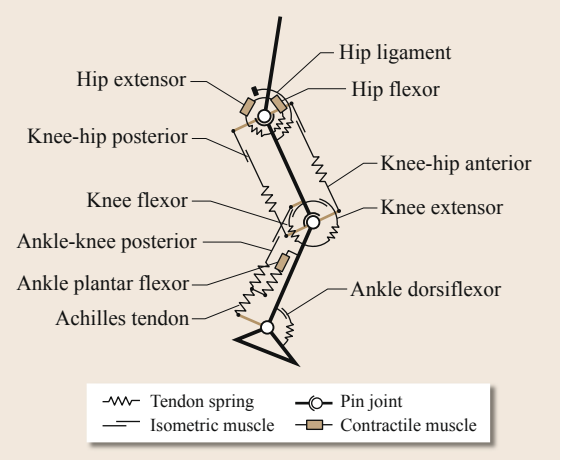


Fig. 70.7 The musculoskeletal walking model of *Endo* and *Herr* [70.54]. Only three contractile muscles act about the model's ankle and hip joints capable of performing nonconservative positive work, namely the ankle plantar flexor, representing the soleus muscle, and two mono-articular, hip flexor/extensor muscles. All remaining muscle–tendon units of the leg are modeled as an isometric muscle in series with a compliant linear tendon spring. The hip joint also includes a unidirectional, linear torsional spring representing the dominant ligaments that tend to flex that joint

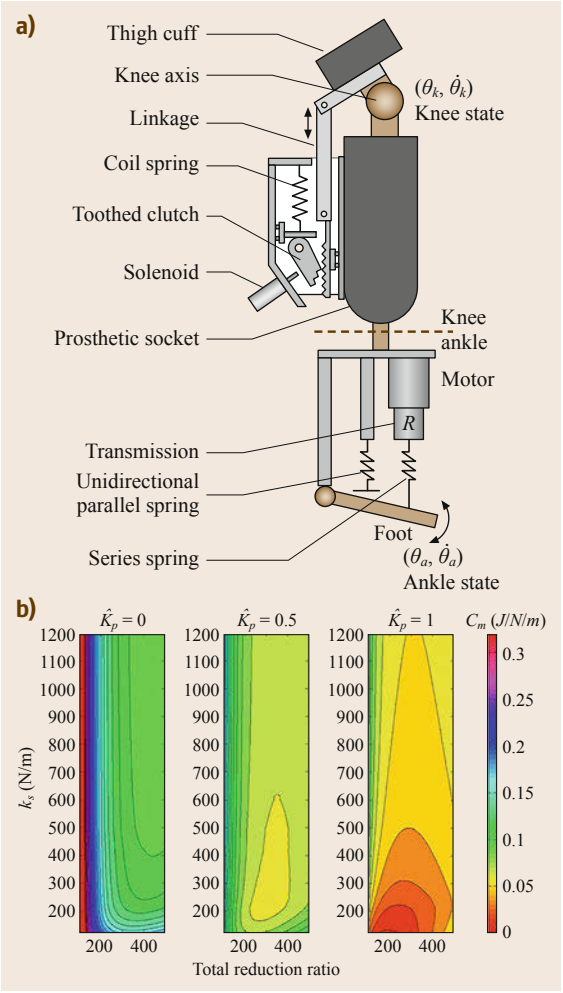


Fig. 70.8 Biomechanical and energetic predictions from the walking model shown in Fig. 70.7. Plotted is the maximum cross-correlation coefficient R, summed across three joint angles and normalized by 3, versus the metabolic cost of transport (COT), or the metabolic energy required to transport unit body weight unit distance. $R = 1$ indicates perfect agreement with biological ankle, knee and hip angle data throughout one gait cycle, whereas $R = 0$ indicates no agreement. Each closed circle is one forward dynamic model solution that can walk for at least 20 s without falling down. The dashed line and shaded area represent mean ± one standard deviation for human metabolic COT data from the literature [70.55]. The open diamond is the model's optimal solution with a maximal R value, and energetically within one standard deviation of the average human walking metabolism

robots that move like, and feel like, their biological counterparts. In the subsequent sections, we discuss key design features critical for a robotic prosthetic, orthotic, or exoskeletal leg to behave like a healthy, normal biological leg. Each design principle is first discussed, and then applied to a leg prosthesis for a transtibial amputee.

70.3.4 Series and Parallel-Elastic Actuation for Power Amplification

Perhaps the most paramount design feature of a wearable robotic leg is a biomimetic actuator architecture designed to achieve biologic levels of powered plantar flexion at the ankle during terminal stance. The biological calf muscles generate nearly 80% of the mechanical work required to complete each gait cycle during mid to late stance plantar flexion, and typically performs greater positive than negative ankle work during each stance period of level-ground walking [70.36, 56–61]. Ankle-powered plantar flexion in the trailing leg is critical for powering the transition from one single support phase to the next single support phase, minimizing impact of the leading leg [70.56, 58]. For this purpose, in the walking model of [70.54] shown in Fig. 70.7, a Hill-type soleus muscle, or ankle plantar flexor, is modeled to actively plantar flex the model's ankle from mid to late stance. The muscle is in series with a tendon spring, representing the Achilles tendon. Further, the

bi-articular gastrocnemius muscle, or the ankle–knee posterior in Fig. 70.7, spanning the ankle and knee, is modeled as an isometric force source, or clutch, in series with a tendon spring.

Such an architecture is critical for powering ankle plantar flexion while still achieving efficient, low-mass robotic actuation. Since the ankle–knee posterior muscle in Fig. 70.7 is shown to transfer little mechanical energy between the knee and ankle throughout the walking gait cycle [70.54], it is reasonable to model this muscle using mono-articular actuators about the knee and ankle. In such a framework, the effect of the ankle–knee posterior on these joints is represented as effective monoarticular torque-producing elements. Such an architecture is shown in Fig. 70.9a showing the schematic of a bionic transtibial prosthesis. Here the ankle is modeled as a series-elastic actuator (SEA) in parallel with a unidirectional parallel spring. The SEA represents the ankle-plantar flexor in Fig. 70.7, and the unidirectional spring represents the torque contribution of the ankle–knee posterior muscle on the ankle joint.

In Fig. 70.9b, the results of a compliant transmission optimization are presented, showing the effects of ankle total reduction ratio, series stiffness, and parallel stiffness on the ankle prosthesis COT. The prosthesis COT is equal to the amount of electrical energy consumed by the prosthesis throughout one gait cycle, divided by the product of half body weight and the distance traveled in one gait cycle. The half body weight in the denominator assumes that each transtibial prosthesis has its own power supply. Without parallel stiffness ($\hat{K}p = 0$), the energy consumption of the prosthesis is minimal at a total reduction ratio greater than 400, and a series spring stiffness greater than $400\,\mathrm{kN\,m^{-1}}$. As parallel spring stiffness increases ($\hat{K}p > 0$), the prosthesis COT can attain a lower minimum value while also requiring a smaller total reduction ratio and series spring stiffness. Using a lower reduction ratio allows the system to have a larger bandwidth, a faster intermittent response, and a lower acoustic noise output. Further, a lower series stiffness improves the prosthesis COT as shown in Fig. 70.9a, shock tolerance upon impact loading at heel-strike [70.62], and motor power amplification at the ankle joint output [70.63]. For further details on compliant ankle transmission design, see [70.62, 63]. In the design of bionic ankle–foot wearable robots, both motor-series and parallel elasticity are of paramount importance.

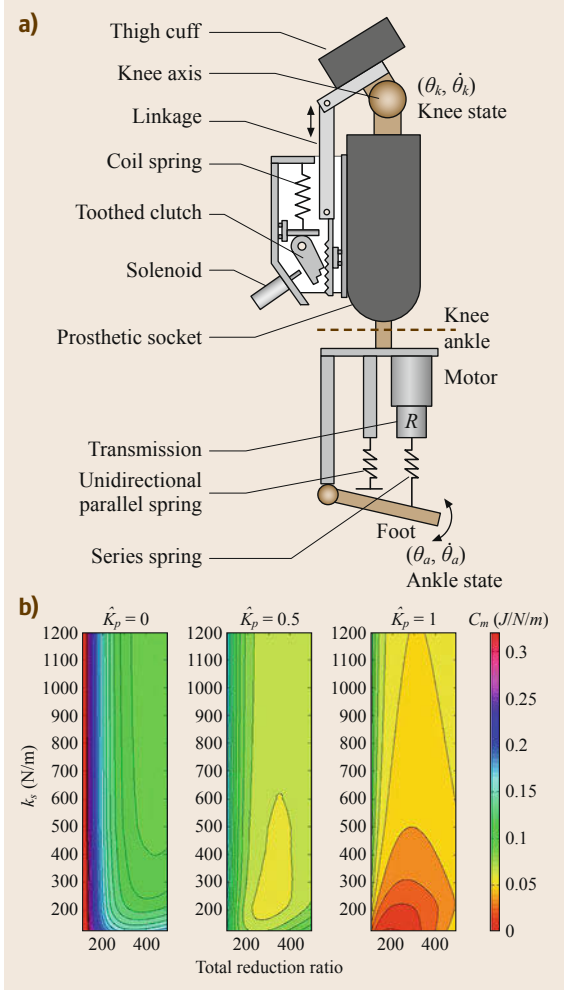


Fig.70.9a,b Shown in (a) is a bionic transtibial prosthesis represented as an electromechanical schematic including ankle motor, ankle transmission ratio R, ankle series spring k_s, and ankle unidirectional parallel spring k_p. The parallel spring is unidirectional, engaging only at ankle angles less than 90° but completely disengaged for angles greater than 90°. In addition, spanning the knee joint are elements of a series-elastic clutch mechanism. Plotted in (b) are the results of a compliant ankle transmission optimization showing the effects of ankle reduction ratio, series stiffness k_s and normalized unidirectional parallel stiffness $\hat{K}p$ on ankle prosthesis cost of transport, C_m. Parallel stiffness, normalized by the human-controlled dorsiflexion stiffness of $630\,\mathrm{Nm\,rad^{-1}}$ is noted as $\hat{K}p$ on the upper portion of each graph. The total reduction ratio is equal to the SEA moment arm r multiplied by transmission ratio R. The compliant ankle assembly shown in (a) was constrained to track biological ankle torque for a human subject with a mass of 78 kg walking at $1.25\,\mathrm{m\,s^{-1}}$ with a stride distance of 1.4 m. Further details of the compliant transmission optimization can be found in [70.62]. ►

70.3.5 Series-Elastic Actuation for Minimizing Motor Work

It has been hypothesized that the preponderance of leg muscle–tendons generates force at low muscle fascicle speed, allowing economical force generation in level-ground walking [70.54]. In the walking model shown in Fig. 70.7, six of the nine leg muscles act in a purely isometric manner, where a tuned, series-tendon compliance enables full energy absorption and delivery, eliminating muscle work and lowering metabolic demands in walking. Specifically, all muscles spanning the knee, as well as the ankle dorsiflexor, are assumed to act isometrically upon neural activation when ambulating across a level-ground surface.

For wearable robotic actuation in level-ground walking, such behaviors can be achieved using a series-elastic clutch mechanism with a tuned series stiffness. However, for a more diverse set of locomotory tasks such as ascending and descending slopes and steps, the ideal actuator architecture is a series-elastic actuator with a tuned series-spring stiffness where the actuator is designed to lock or clutch at lower power consumption. As an example, for level-ground ambulation using an electric actuator, a clutch acting on the motor shaft can be used to lock the shaft during those times in the gait cycle where a spring response is sought, eliminating the high power consumption required if the motor itself were to clutch the shaft. For other movement tasks outside of level-ground walking, the motor–shaft clutch could then be disengaged, and the motor could apply the necessary torques and impedances required for the task using standard series-elastic actuation strategies [70.64]. Such an architecture allows wearable robotic legs to move economically across level-ground surfaces, while still allowing a great deal of versatility for irregular-ground ambulation, sit-to-stand maneuvers, and general disturbance rejections.

This design principle of using a tuned series stiffness to minimize the need for actuator work has been used in the design of a transtibial prosthesis in [70.65] (Fig. 70.9a). Since the gastrocnemius was modeled in [70.54] as an isometric force source in series with a tendon spring, in the study of [70.65] a series-elastic clutch was mounted at the knee brace to provide the knee flexion action of the gastrocnemius during level-ground walking. The mechanism consisted of a toothed clutch at the free end of a coil spring (spring stiffness of 66 500 N/m) that acted as a knee flexor on a polycentric knee brace. The brace was integrated into a prosthetic socket connected to the ankle–foot prosthesis, as is shown in Fig. 70.9a. When the clutch was engaged via solenoid action, the free end of the spring locked with respect to the socket and the spring stretched as the knee straightened. The force developed by the spring produced a flexion torque at the knee joint. Conversely, when the clutch disengaged no torque was applied at the knee joint. The spring acted on the knee joint with a moment arm that varied between 0.02 and 0.03 m as a function of knee angle. This moment-arm function was designed so that the apparent knee stiffness matched that of the biological knee during level-ground walking [70.66].

70.3.6 Neuromuscular-Based Reflexive Control

We present a sensing and control scheme for producing biomimetic positions, torques and impedances at the hip, knee, and ankle joints of a powered leg prosthesis, orthosis, or exoskeleton during walking and running gaits. In this paradigm, sensory data are collected using intrinsic and/or extrinsic sensors. Here intrinsic sensing refers to information collected from sensors located on the wearable robotic device, and extrinsic sensing refers to all information collected from sensors located externally to the wearable device. As an example, in the case of a leg prosthesis, a surface electrode for the measurement of the electromyographic signal from residual limb muscles would be an extrinsic sensor, and an inertial measurement unit located on the device itself would be classified as an intrinsic sensor. Intrinsic sensors measure device positions, motions, forces, torques, pressures, and temperatures, whereas extrinsic sensors might comprise such mechanical and temperature sensors positioned external to the wearable device, as well as neural sensors for the determination of user motor intent.

In the neuromuscular-based control framework, such sensory data are passed to a neuromuscular model of human locomotion, such as the *Endo* and *Herr* model [70.54] presented in Fig. 70.7, which is used to compute appropriate joint dynamics for the device to provide to the user. The model-based control scheme, depicted in Fig. 70.10, relies on data collected from both intrinsic and extrinsic mechanical sensors, and extrinsic neural sensors used to infer the motor intent of the user, volitionally and/or nonvolitionally. The neuromuscular model used to compute desired joint dynamics may include muscles modeled in a variety of ways. These muscle models (muscle model block in Fig. 70.10) may include, for example, a bilinear muscle model or a Hill-type muscle model [70.67]. The measured states of the robotic joints are used to determine the internal state (length, velocity) of each of the virtual muscle–tendon units of the neuromuscular model using morphological information of the muscle moment arms about each modeled joint. This geometrical transforma-

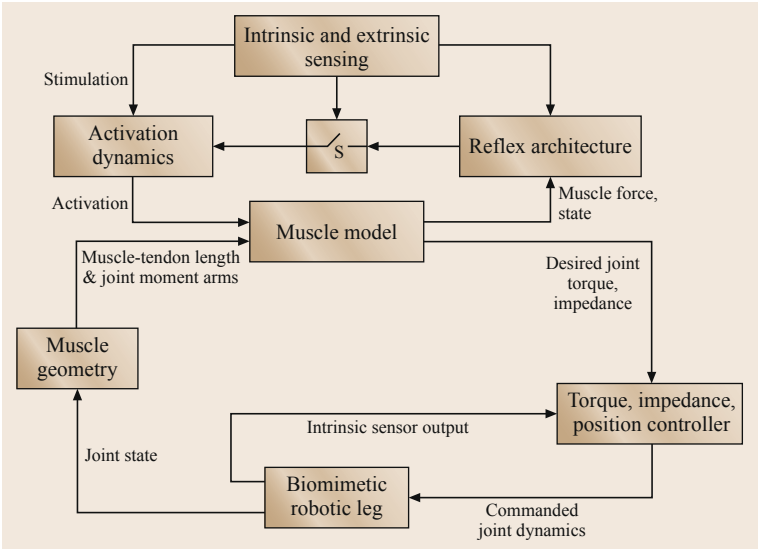


Fig. 70.10 Block diagram of a neuromuscular model-based control architecture for a bionic wearable robotic leg

tion occurs within the muscle geometry block shown in Fig. 70.10.

The impedance and force of each virtual muscle are additionally governed by the muscle activation, which may be determined from a local reflex loop, an external source, or a combination thereof. In the reflex case, a feedback loop is implemented where virtual muscle force and state are used to produce muscle stimulation, which is then filtered to produce muscle activation (activation dynamics block in Fig. 70.10). This feedback-based control scheme is designed to emulate the force feedback and stretch reflex of an intact human muscle. This reflexive feedback loop can be a linear or nonlinear function of virtual muscle force, length, and velocity. For example, the reflexive feedback loop can be nonlinear, comprising a threshold prestim parameter, as well as force and state gains and exponents, or

$$u(t) = x + y_F\,[F(t-\Delta t_F)]^{z_F} + y_l\,[l(t-\Delta t_l)]^{z_l} + y_v\,[v(t-\Delta t_v)]^{z_v}\ , \tag{70.1}$$

where x is the prestim parameter, y_i are force and state gains, z_i are force and state exponents, and t_i are corresponding delays.

In the case where muscle activation is determined solely by an external source, a neural sensor might be used to provide some estimate of motor intent, which is then input to the activation dynamics block where a muscle activation is estimated as an input to the muscle model. Typically, such measurements of motor intent would comprise one or more peripheral neural sensors from implants interfacing with nerves and/or muscles, but in the most general case, such motor intent commands may additionally be measured from central brain implants.

In the combination case, the framework of Fig. 70.10 describes a procedure where the reflexive parameters are modulated by the controller either within a single gait cycle, and/or from gait cycle to gait cycle in an updating manner, based upon detected variations in gait speed and terrain. Extrinsic efferent neural signals from muscles and/or peripheral nerves might be used to modulate reflexive parameters, such as force and state gains and exponents. For example, measured calf muscle EMG in a transtibial amputee has been used to modulate the gain of a positive torque feedback during the stance period of a walking gait cycle, providing the amputee direct volitional control over powered plantar flexion during terminal stance in walking [70.68]. Clearly, one day it may be possible to stimulate through a nerve implant to reflect intrinsic/extrinsic mechanical sensory data as an afferent feedback signal to allow the user of the wearable robot to better modulate efferent neural motor commands for a desired wearable robotic dynamical response.

Once the force of each virtual muscle spanning a joint is determined (using the implemented muscle model), each muscle force is multiplied by its biologically realistic muscle moment arm and then all muscle torque contributions are summed around the joint to produce a net torque and impedance estimate. The model estimates are then sent to the controller (Fig. 70.10) as the desired net torque and impedance for each robotic joint. The controller tracks these desired values at each joint to produce human-like joint forces and impedances. In the case where the human

user's motor intent is to control device joint position, the controller would integrate the desired joint torque to achieve a joint position estimate, and then modulate device joint position to achieve that desired position.

By way of example, we present a neuromuscular-based control of a transtibial prosthesis (Fig. 70.11). For this example, the model-based control framework of Fig. 70.10 is used to control the prosthetic apparatus. In the study of [70.65], ankle and knee joint states of the apparatus were measured and used to provide real-time input to a neuromuscular model simulated by an on-board microcontroller. The resulting torque command from the neuromuscular model was used to produce ankle torque while a knee controller adjusted the torque produced by the knee brace. This configuration, shown in Fig. 70.11, was set up to enable the prosthetic apparatus to behave as if it were a human lower leg with reflex-controlled muscles acting at the ankle (Fig. 70.12).

70.3.7 Leg Exoskeletons

Research has lead to the developement of leg exoskeletons with the prime intention to allow otherwise healthy individuals to perform difficult tasks more easily or enable them to perform tasks that are otherwise impossible using purely human strength or skill [70.69]. Early developments in leg exoskeletons consisted of long bow/leaf springs operating in parallel to the legs and were intended to augment running and jumping [70.70]. Each leg spring was engaged during the foot contact to effectively transfer the body's weight to the ground and to reduce the forces borne by the stance leg. During the aerial phase, the parallel leg spring was designed to disengage in order to allow the biological leg to freely flex and to enable the foot to clear the ground. In 1963, *Zaroodny* published a technical report detailing his work on a *powered orthopedic supplement* intended to augment the load-carrying abilities of an able-bodied wearer such as a soldier [70.71]. Other early implementations of exoskeletons, including legs, were attempted by General Electric Research with the *Hardiman* project [70.72], by Moore at the Los Alamos National Laboratory, and *Rosheim* [70.73]. However, a major impetus for the recent work in performance augmenting exoskeletons has come from the DARPA (Defense Advanced Research Projects Agency) program *Exoskeletons for Human Performance Augmentation* (EHPA) [70.74], the outcomes of which were different working exoskeletons.

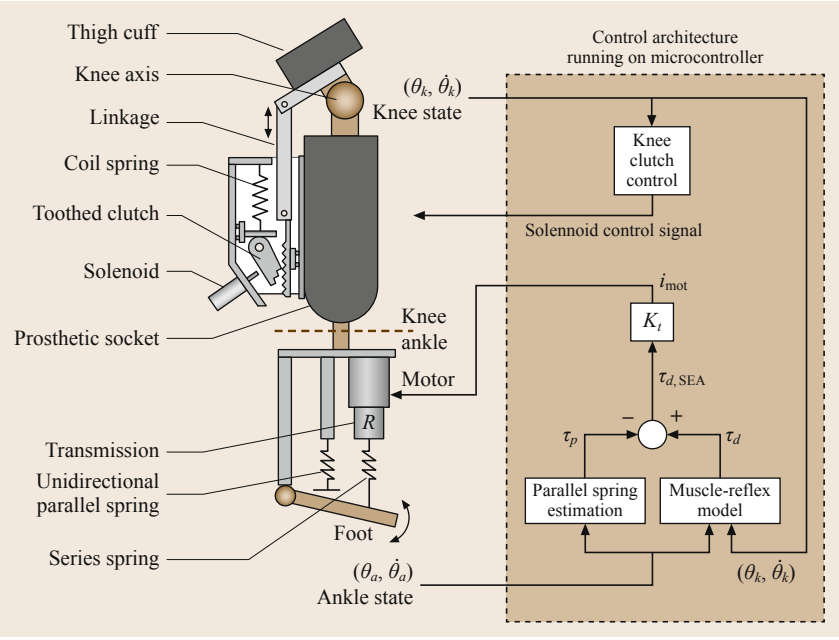


Fig. 70.11 Labeled schematic and control architecture for the prosthetic apparatus. Rotary elements in the ankle-foot prosthesis are shown as linear equivalents in the model schematic to improve clarity. In the control schematic, the parallel spring contribution to prosthesis ankle torque, τ_p, was subtracted from the desired ankle torque command from the neuromuscular model, τ_d, to obtain the desired SEA torque, τ_d. A motor current command i_{mot} was obtained by dividing the desired SEA torque by the motor torque constant, K_t. The knee clutch was engaged via the solenoid depending on knee state as obtained from the knee potentiometer. For further details on the mechatronic and control design, see [70.65]

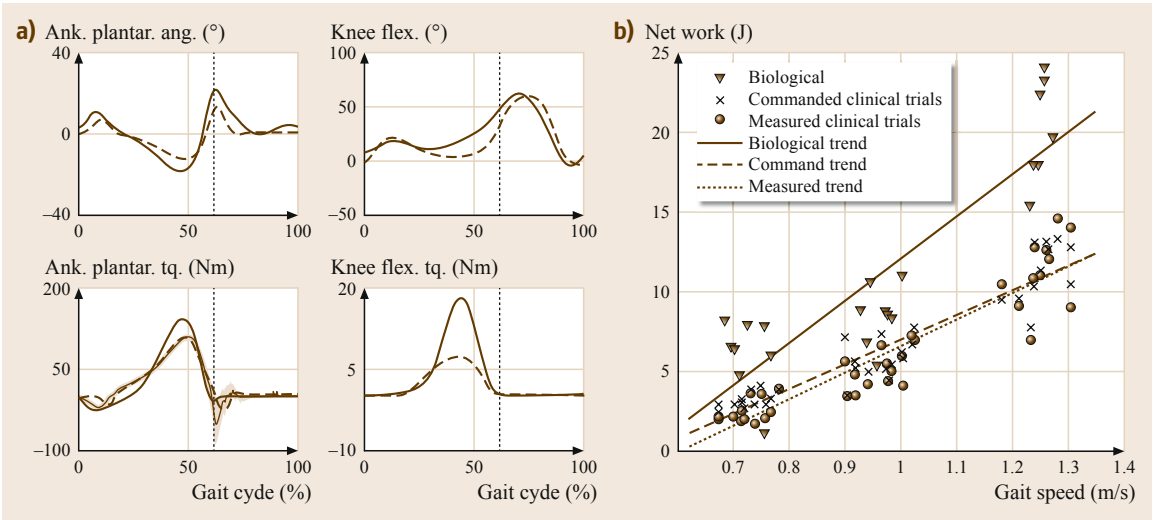


Fig. 70.12 **(a)** Comparison of prosthesis ankle and knee angles and torques during the clinical trials (measured) with those from a height- and weight-matched subject with intact limbs (biological). The biological values are the thick solid lines (with shaded errors) in each plot while the dashed lines are the values measured on the prosthesis. In the ankle torque plot the commanded torque is shown as a thinner solid line, again with shaded error bars. The knee torque plot compares the torque provided by the clutch-spring mechanism to that provided by the natural gastrocnemius in simulation. The vertical line indicates toe-off in each plot. **(b)** Energy output of the ankle across gait speed. Shown are biological data, net work as commanded by the ankle–foot prosthesis during clinical trials, and measured net work during the clinical trials. For further details on experimental methodology and results, see [70.65]

In this framework, the Berkely Exoskeleton (BLEEX) was the first example of an energetically autonomous system [70.75]. BLEEX features 3-DOF at the hip, 1 at the knee, and 3 at the ankle. Of these, four are actuated: hip flexion/extension, hip abduction/adduction, knee flexion/extension, and ankle inversion/eversion. Of the unactuated joints, the ankle inversion/eversion and hip rotation joints are spring loaded, and the ankle rotation joint is free-spinning [70.76]. The BLEEX exoskeleton [70.77] was designed with linear hydraulic actuators since they were the *smallest actuation option available* based on their *high specific power (ratio of actuator power to actuator weight)* [70.78, 79]. However, a further study determined that electric motor actuation significantly decreased power consumption during level walking in comparison to hydraulic actuators [70.80, 81]. The weight of the implementation of the electrically actuated joint, however, was approximately twice that of their hydraulically actuated joint (4.1 versus 2.1 kg). Significant effort was made in developing a hybrid hydraulic–electric portable power supply [70.82]. In terms of performance, users wearing BLEEX can reportedly support a load of up to 75 kg while walking at 0.9 m/s and can walk at speeds of up to 1.3 m/s without the load. A second generation of the Berkeley exoskeleton is currently in testing. A laboratory spin-off company called Berkeley Bionics (Berkeley, CA) has been created in order to market exoskeleton technology.

A quasi-passive exoskeleton concept has been advanced in the Biomechatronics Group at the Massachusetts Institute of Technology (MIT) Media Laboratory that seeks to exploit the passive dynamics of human walking in order to create lighter and more efficient exoskeleton devices. The MIT exoskeleton employs a quasi-passive design that does not use any actuators for adding power at the joints. Instead, the design relies completely on the controlled release of energy stored in springs during the (negative power) phases of the walking gait [70.83]. The 3-DOF hip employs a spring loaded joint in the flexion/extension direction that stores energy during extension that is released during flexion. The hip abduction/adduction direction is also spring loaded, but only to counter the moment induced by the backpack load. Additionally, a cam mechanism was incorporated into the hip to compensate for the relative change in length between the thigh of the exoskeleton and the user due to the joint offset during abduction/adduction. Additionally, spring loaded hip rotation and ankle rotation joints were included to allow nonsagittal plane limb movements. The knee of the MIT exoskeleton consists of a magneto-rheological variable damper that is controlled to dissipate energy at appropriate levels throughout the gait cycle. For the

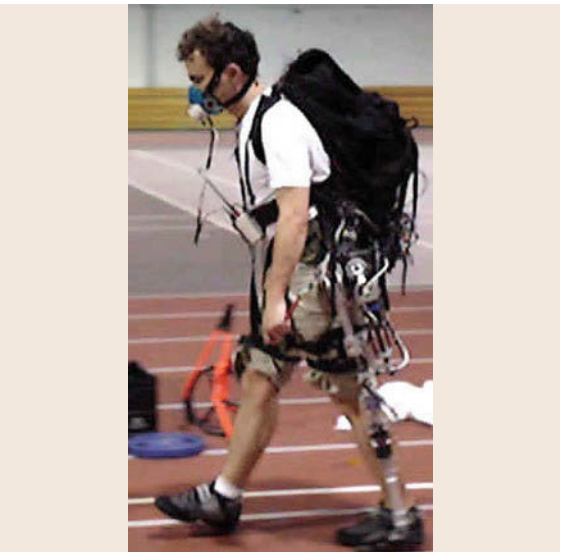


Fig. 70.13 MIT exoskeleton during metabolic testing

ankle, separate springs for dorsi and plantar flexion are implemented in order to capture different behaviors during these two stages of motion, and store/release the optimum amount of energy. Without a payload, the exoskeleton weighs 11.7 kg and requires only 2 W of electrical power during loaded walking. This power is used mainly to control the variable damper at the knee. The first studies on the metabolic cost associated with walking under the aid of an exoskeleton have been reported by using the MIT exoskeleton [70.84] (Fig. 70.13).

Further experimental work with the MIT quasi-passive exoskeleton showed a significant reduction in metabolic cost of walking versus the same exoskeleton without the springs at the hip and ankle and the variable damper at the knee, demonstrating the utility of quasi-passive elements.

Other examples of leg exoskeletons comprehend the hybrid assistive leg developed at the University of Tsukuba by *Sankai* [70.85] that is targeted for both performance-augmenting and rehabilitative purposes. At the Kanagawa Institute of Technology in Japan, researchers have developed an exoskeleton for the purpose of assisting nurses during patient transfer [70.86]. Yobotics, Inc. (Cincinnati, OH) [70.87] developed a simple exoskeleton for adding power at the knee to assist in stair climbing and squatting during load-carrying tasks [70.78]. Researchers in the Departments of Mechanical Engineering and Physical Therapy at the University of Delaware have developed a passive leg orthosis that is designed to reduce the forces of gravity on the patient during walking, thus easing the effort required for locomotion [70.88]. Caldwell has developed a 10-DOF lower limb exoskeleton device [70.81]: actuation is provided to the flexion/extension directions of the hip, knee, and eankle, and abduction/adduction of the hip via pneumatic muscle actuators. Recently, few leg exoskeleton systems have been commercialized worldwide by different companies: the ReWalk lower limb exoskeleton developed by Argo Medical Technologies, Israel [70.89] for elderly/paraplegic assistance, and the system called REX developed by Rex Bionics, New Zealand [70.90].

70.4 Whole Body Wearable Systems

Despite the considerable interest and impressive advancements in the field, so far only few researchers have attempted the development of full-body EHPA (FB-EHPA), in particular fully actuated systems for heavy-duty applications, and the available scientific literature is quite limited [70.8, 83, 85, 91–99]. This section reviews the general design issues concerning electro-mechanical design of FB-EHPA. The general design issues are analyzed considering and presenting a series of specific designs.

70.4.1 General Issues

Kinematic Architecture

The kinematic scheme of FB-EHPA is a fundamental aspect that strongly affects the basic functionalities and capabilities of such machines. Just as other wearable robots, the kinematic design is strongly influenced by the close proximity of robot linkages and human limbs. Kinematics of an FB-EHPA must be designed in order to be compatible with:

- The execution of common human movements: walking, squatting, lifting, turning, etc.
- Requirements for the execution of specific tasks: raising weights, handling specifically shaped objects, running, etc.
- Encumbrance and spatial limitation
- Possible collisions with the operator body and self-interferences.

Common solutions for the existing FB-EHPA are based on anthropomorphic kinematics since this kind of solution guarantees:

- The best workspace matching with reduced encumbrances
- The most simple strategy to avoid self/body collisions
- The execution of common human movements
- Mass distribution that is compatible with human executed tasks.

A full anthropomorphic design that allows replicating all the human DOFs could be desirable; however, the complexity of such a system would be far above what is technically affordable. In this sense, a drastic simplification is required to reduce complexity without affecting the basic functional requirements of the machine. This type of simplification is assumed in many of the available FB-EHPA.

70.4.2 Specific Design

According to a historical perspective, the first technical concepts of FB-EHPA were proposed in 1956 by *Lent*, who proposed an inflatable space suit with powered joints to assist the wearer during the flexion of suit extremities stiffened by the internal pressure [70.92]. Lately, in 1966, *Mizen* proposed, although without realizing it, the first man amplifier for civilian and military applications [70.93].

The first practical investigation was instead attempted by General Electric between 1965 and 1971, by developing the Hardiman [70.94] that is briefly described in the following section. Researches on FB-EHPA were resumed in 1996, when *Snyder* and *Kazerooni* proposed a novel under-actuated material handling system with 6 passive joints (2 for the legs and 4 for the arms) and 12 electrically actuated joints (6 for the legs and 6 for the arms) under force-feedback control [70.95]. System arms and legs were implemented and tested separately.

Following this renewed interest, from 2002 to 2009, three different under-actuated FB-EHPA have been developed in Japan for medium-duty applications, such as nursing care [70.83, 97], disabled or elderly assistance [70.85], and agriculture [70.100]. The best performing and most renowned system is the hybrid assistive limb (HAL) [70.85]. Still in Japan, in 2009, Panasonic-Activelink presented the POWER LOADER, a heavy-duty FB-EHPA for material handling [70.98]. In 2010, Raytheon-Sarcos unveiled the XOS2 [70.99, 101], which is one of the most advanced and performing medium-duty hydraulically actuated FB-EHPA for military/logistics applications. One of the latest FB-EHPA is called Body Extender (BE) [70.102] and has been realized at the PERCRO Laboratory, Scuola Superiore Sant'Anna (Italy). The BE is a full electrical machine developed for heavyweight material handling. In 2012, the French company RB3D has presented HERCULE, an electrically actuated FB-EHPA with legs similar to those of HAL, but with different arm kinematics, and designed for lifting 20 kg with each arm at full horizontal extension. HERCULE has been developed for the French Ministry of Defense and detailed specifications are not available at present.

In this section, an overview of the basic features of some of the most relevant and advanced existing systems is provided.

Hardiman

Hardiman I is the first prototype of FB-EHPA and has been developed between 1965 and 1971 by General Electric Research (Schenectady, NY). The project aimed at developing a high-power machine for handling heavyweight materials in unstructured environments. The project was jointly supported by Engineering Psychology Programs-ONR, Naval Air Systems Command, and the Army Mobility Equipment Research and Development Center (Fort Belvoir).

The project results have been reported by *Fick* and *Makinson* in [70.94]. Hardiman I has a structure based on a 30-DOF kinematics including:

- 2-DOF for each end-effectors: two flexion movements of a two-phalanges finger
- 7-DOF for each arm: wrist flexion, forearm rotation, elbow, upper arm rotation, shoulder flexion, back flexion and arm abduction/adduction
- 4-DOF for each leg: hip abduction/adduction, hip flexion, knee flexion and ankle flexion
- 2-DOF for each foot: ankle inversion and ankle rotation.

The actuation system employs hydraulic actuators that require an external hydraulic circuit at 206 bar (3000 psi) with a power of 18.6 kW (25 hp). The global weight of the system is 680 kg (1500 lbs) and the exoskeleton is rated for a lift capability of 6800 N (1500 lbs). The control of the hand is completely hydromechanical (implemented with no electronic components), while the arms, legs, and feet are controlled through electrohydraulic systems. Hardiman I did not succeed to be tested in its full functionalities. In particular, the developers reported several problems in the implementation of the controller for the lower limbs, mainly also due to the lack of reliable electronic components and reduced computation power characterizing the technology at that time.

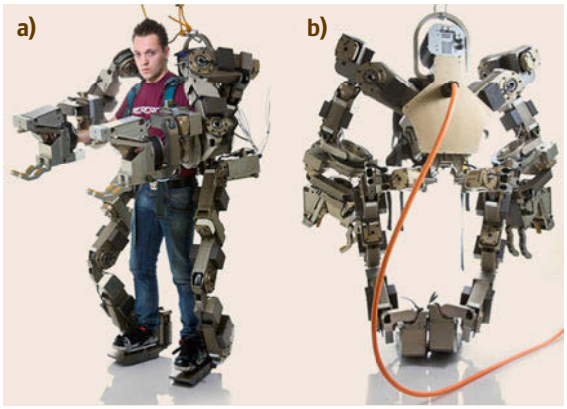


Fig.70.14a,b Picture of the Body Extender (BE): **(a)** system worn by a user; **(b)** rear view (after [70.103])

HAL: Hybrid Assistive Limb

HAL is a series of full-body exoskeleton systems, the development started in 1996 and the current version, HAL-5, is the first commercial version. HAL has been developed by *Sankai* [70.85] and his research group of the University of Tsukuba and commercialized by the company Cyberdyne, Inc. (Tsukuba, Japan) [70.103].

It is realized in two versions, the first one is for lower limbs only and the second includes the upper limbs. The exoskeleton is developed with the aim to expand, augment, and support human physical capability in the field of medical welfare, heavy work support, and entertainment. The kinematic structure features 6 passive joints (2 for the legs and 4 for the arms) and 8 electrically actuated joints (6 for the legs and 6 for the arms). HAL is commanded via a specially designed controller which takes into account the wearer's muscular activity through the employment of bioelectrical signals from EMG sensors that are disposed on wearer's skin. HAL weighs 23 kg in the full-body version, including batteries for an autonomy of 2 h and 40 min. Although the lifting capabilities are not specified, it is reported in [70.97] that laboratory tests have been conducted with a payload of 15 kg with each arm at full horizontal extension; however, it requires locking of the passive joints and an additional wrist support. HAL is currently being experimentally tested in clinics and the research group is currently working on improving the system with new control algorithms.

POWER LOADER

POWER LOADER is a full-body exoskeleton developed by Activelink Ltd, a spin-off of Panasonic Ltd, Japan. The system development was started in 2003 and the first working prototype reported in literature has been realized in 2010. POWER LOADER has been developed for rescue operation in natural disaster scenarios such as big earthquakes. The mechanical design is based on an anthropomorphic design only for the arms. The legs have an anthropomorphic structure. The knee is back rotated [70.98] in order to leave more free space around the user legs. The actuation is achieved through electric motors and drivers.

PERCRO Body Extender

One of the latest full-body exoskeleton has been realized at the PERCRO Laboratory of Scuola Superiore Sant'Anna, Italy (Fig. 70.14). The developed FB-EHPA has been named as the body extender (BE) [70.104]. The BE comprises two identical legs with 6-DOF each, whose kinematics is isomorphic to the wearer's kinematics, and two identical arms with 5-DOF each. All 22-DOF are powered by modular highly efficient actuation units; this solution allows to reduce the power consumption, as well as to minimize the costs and efforts related to its production and maintenance. The BE weighs 160 kg, can provide a maximum continuous usable power of 16.5 kW, is capable of lifting 50 kg with each arm at full horizontal extension, and is designed for transporting loads up to 100 kg at a walking speed of 0.5 m/s. Potential application of this system is in earthquake disaster relief operations, where search and rescue teams need to walk across tight spaces over debris and to remove rubble in an attempt to find, release, and transport survivors. The system has been integrated and fully tested in 2010 and it is currently being improved in terms of energy efficiency and stability on uneven terrains.

Commercial Systems (Uncovered by Scientific Literature)

In the past years, two advanced commercial systems have been unveiled. The first one is XOS II, which is the second version of a full-body exoskeleton developed by a US company, Sarcos-Raytheon [70.99, 101, 105]. The first version of the system was completed in 2008 and the second version has been presented in 2010. The XOS II has been developed aiming at logistic applications in the military field. It is based on an anthropomorphic kinematics equipped with 24-DOF actuated with hydraulic systems. It weighs 95 kg and has a lifting capability of 230 N. The company is currently working on improving the energy efficiency for the realization of an autonomous (untethered) version of the system. The second device is called HERCULES and has been presented in 2012 by RB3D (France). It is an electric-driven system that has a lot of similarities with the HAL-5 exoskeleton. The details of HERCULES have not yet been disclosed [70.106].

70.5 Control of Human–Robot Augmentation Systems

From a control point of view, the functionality of a power augmentation system should simultaneously provide (a) force amplification and (b) support to natural locomotion.

Hence, the control of exoskeleton for human power augmentation (EHPA) can be split into two major subsystems: lower- and upper-body designs. The first subsystem cooperates with a human gait, while sustaining much of the body loads, keeping overall equilibrium and following the human motion with transparent and natural reflection forces. The second subsystem enhances human manipulation abilities through the multiplication of the arm forces during grasping operations of heavy loads.

Both components have to respect the following requisites:

1. *Stability*: The stability margin of the (linearized) control system should be large enough to cope with the environment and operator uncertainties, and to avoid uncompensated oscillations entering into resonance with the human postural control.
2. *Transparency*: The system should be highly reversible (through mechanical design or sensor augmentation) in order to allow a correct perception of the overall interaction with the environment and the proper generation of the control actions.
3. *Regular force amplification*: The gain ratio of the human force amplification should be kept as regular as possible in all interaction conditions, including a large bandwidth of interaction, different postures, and different environment interactions.
4. *Fast response*: The bandwidth and the saturation values of the position/velocity/force controllers should be as close as possible to their respective ranges used in natural motions and interaction.

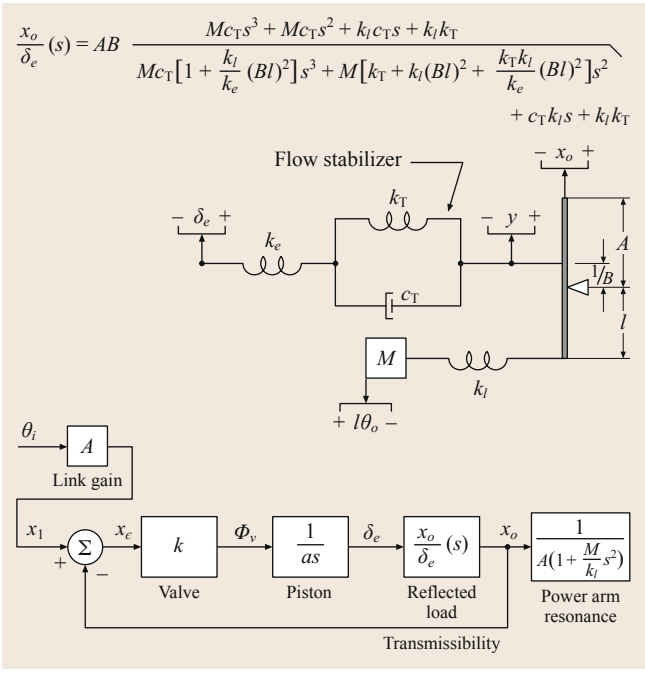


Fig. 70.15 The control of the force amplifier joint (after [70.39]). The upper scheme represents the mechanical equivalent circuit for the control of the elbow joint, and the above equation represents the associated force balance equation. The lower scheme is a block diagram that represents how the components are arranged together and how the feedback is used to control the joint actuation

70.5.1 A Historical Perspective

Yagn [70.70], first submitted a series of patents for facilitating human walking and running. At that time, the mechanism was limited to a series of springs and clutches, which altered the feet compliance during walking and used the potential energy of the springs to have a more efficient walk.

The use of active orthoses, through hydraulic servovalves was proposed by *Filippi* in 1937 [70.107]. The proposed system was a knee orthosis whose motion in the sagittal plane is powered by ahydraulic piston. At that time, there was not much advancement on electric control of hydraulic systems and the joint control was supposed being regulated mechanically. A crank located on the hip moved a cam that, on its own, winded a torsional spring on the knee. The other extremity of the spring was used to regulate the piston flux.

The first documented attempt to develop a powered suit that amplifies human force was performed by *Clark* et al. in the early 1960s [70.39]. At that time, they proposed to design a man amplifier using hydraulic amplification. In Fig. 70.15, it is represented by the device interaction model and the basic control loop they proposed. They have limited the investigation to one single joint of movement.

In Clark et al.'s work, each joint was regulated with an independent position controller that was actuated by mechanical pistons placed closed to the human body. The control was based on an admittance scheme that translates user commands into joint velocities. This scheme had no opportunity to show the reflection of contact forces.

The design and control of whole-body active exoskeletons started in 1965 with a joint Naval Army program who supported General Electrics for the design of a whole-body EHPA (Hardiman I). The power

augmentation target was fixed to an amplification factor (robot versus human) 25 : 1 having a nominal total force about 10 000 N.

The proposed device had 28 independent DOFs plus two grasping tools with one additional DOF each.

The control concept design of Hardiman was very ambitious. In the first attempt, the designer developed a set of mechanical servo-valves (as shown for the *master hand controller* in Fig. 70.16) placed in contact with the user's skin which responded to local pressure by deflecting the flux of hydraulic circuits [70.96].

The approach worked well only for simplified kinematics of the mechanical hand, while the dynamic interference between the serial kinematics of the arm link required the use of electrohydraulic circuits and that of a computer-developed dynamical model.

The project development stopped due to major limitations in the design and control of the lower leg. During the development, the authors only have tried unilateral electrohydraulic servos to track the motion of the user legs. The major development issue was related to the control of the leg for the body balance. The developers devised that the leg functionality changes between the stancing and the swinging phase, but the use of switches to detect the current leg phase demonstrated to be ineffective to validly cope with equilibrium and stability issues. Other issues were instead related to the power supply line (45 kW needed by the prototype) and to the kinematics differences between the master and the slave arm, which makes some motion unnatural, risky, and sometimes unstable for the driving user.

Few years later, *Vukobratovic* et al. [70.108] started to develop an exoskeleton to assist injured people (mostly paraplegic) to recover basic walking abilities. That work shares several features that are being implemented in present power extenders. Vukobratovic et al. were able to define the following relevant features of control and stabilization loops for walking that are still valid today:

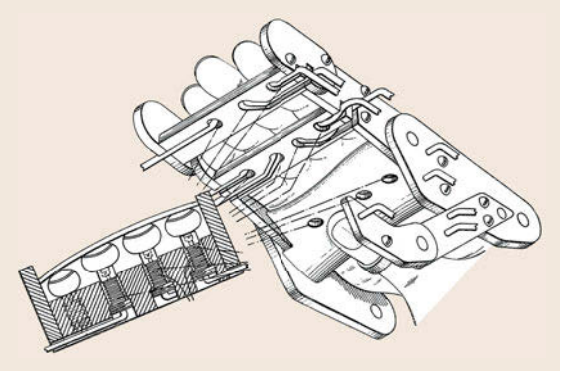


Fig. 70.16 The master hand controllers. A set of *tinklers* were moved according the grasping force of the user and drive the hydraulic circuit of the slave hand with a design amplification factor of 25:1 (after [70.96])

- A model reduction approach
- The method of prescribed synergies
- The use of zero moment point (ZMP) to derive the dynamic equation of motion
- The use of left–right symmetries and nominal gait cycle
- The design of a *model-based* analog motion compensator that provides to create motion synergies through a set of servomotors.

Motion synergies were computed using a nominal gait trajectory (step period = 1, step size coefficient = 1) in a reduced phase space as a reference to derive in advance the weight, the inertia, and the friction compensation torques that have to be exerted at each joint.

Figure 70.17 describes the high-level functional architecture of the controller. The adaptation for different gait conditions (from 1 to 5 s) was modulated on the precomputed trajectories through a couple of coordinated sequencers (time-base generators, TBG) that convert the absolute time into the loop phase required to determine compensation components.

The overall feedback also includes two additional feedback components, which use force sensors in the shoes to compute torque perturbation around the ZMP. These disturbances appear when the exoskeleton rotates around the leg in the stance phase or when a change of acceleration (step length/period) alters the gait from the nominal cycle.

In both cases, the author derived, through linearization of the dynamic model along the nominal cycle, a couple of static gains that map the sense forces into corrective torques that are added to the nominal hip torques.

Vukobratovic et al.'s exoskeleton worked well only for limit cycles which were close to the nominal gait trajectory it was designed for. Both Vukobratovic's exoskeleton and the experience derived form the Hardiman project showed that the control technology available at that time was premature to cope with the nonlinear, dynamic coupling effects that appears in walking and manipulation.

A renewed effort in research was performed by researchers at Berkeley University who in the middle of the 1980s promoted projects, in general called the *human extenders*, to develop arm exoskeletons that augment the arm force. Early extender projects were

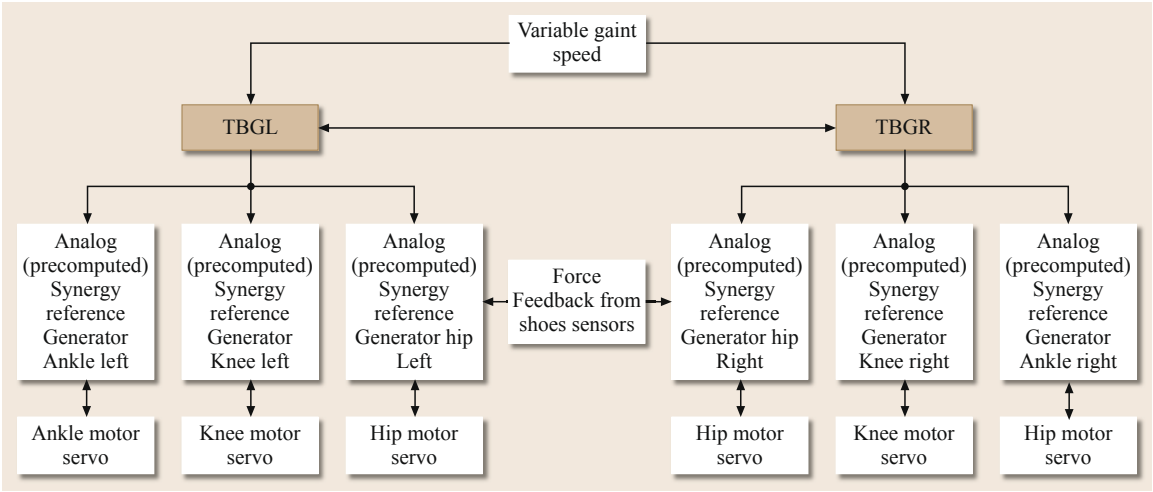


Fig. 70.17 The control structure of the 1974 *Vokubratovic* et al.'s exoskeletons (after [70.108]). The active device was powered with six pneumatic actuators whose references and feedback are determined through three different components: servo feedback, prescribed synergies, and force feedback for sagittal and lateral stabilization

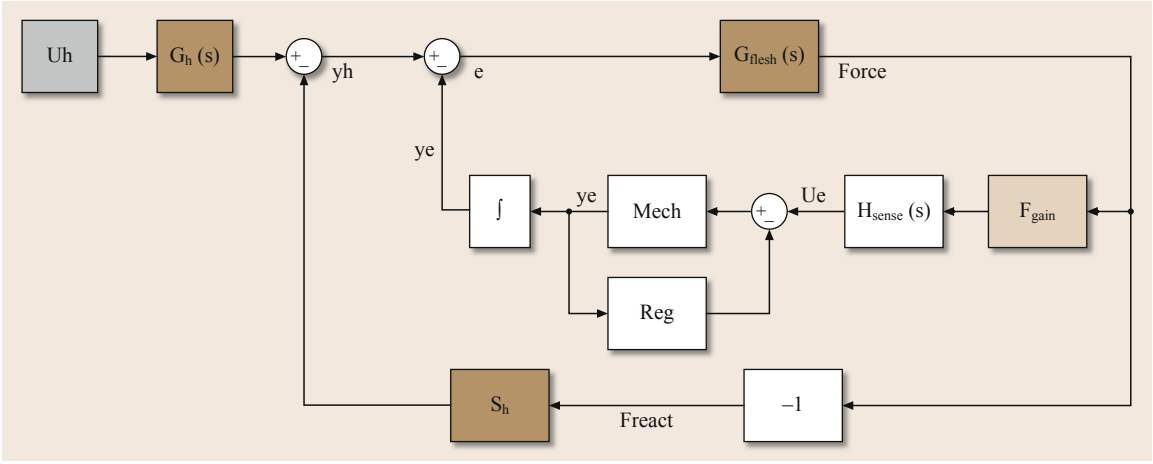


Fig. 70.18 The basic extender controller (after [70.109]). The scheme represents the extender controller as proposed by Kazerooni in 1989. The controller is based on an internal velocity controller. A given force amplifier gain F_{gain} determines the forces that are compensated dynamically with respect to the force shown to the operator

based on the combination of a robust velocity controller in feedback with a force sensor that detects user inputs.

Figure 70.18 shows the working principle of the amplification loop. With respect to previous attempts, here the use of the digital controller helped to implement more sophisticated feedback controllers that cope with the multi-input multioutput system model. The overall stability was proven through two methods: (1) a linear analysis, performed through the multivariable Nyquist criterion on the linearized model (2) and an application of the small gain theorem that provides sufficient conditions for stability in the case of the nonlinear model interacting with the unstructured environment. However, the stability conditions derived in [70.2] through the small gain theorem where highly restrictive and required the amplification gain to be less than 1. Subsequent author's work introduced different criteria (than fully unstructured interaction), such as typical operator and environment impedance ranges, and stability margins. Under such conditions [70.110] and using the μ-synthesis, the author demonstrated robust stability under higher force amplification gains.

An example of the amplification goal achievement with this technique and a 3 DOF arm was reported

in [70.111] as the maneuver bandwidth of 2 Hz, the vertical force amplification ratio 7 : 1, and the horizontal force amplification ratio 5 : 1.

70.5.2 Power Augmentation through Proprioceptive Sensors

This type of power augmentation systems use only sensors that are completely internal to the exoskeleton structures. These sensors have no direct contact with the human operator (e.g., motor or joint torque/position sensors).

One benefit of this approach is the reduced design complexity through the use of more conventional torque sensors and the improved stability of the reading even in the case of multiple contacts between the user and the exoskeleton structure.

As a side effect, there is no direct measurement of the force effectively felt by the user. The interaction controller decides the exoskeleton motion policy based on the estimation of the interaction force with the user.

This approach is used in the control of the Berkeley Lower Extremity Exoskeleton [70.5], and most of the intelligent assisting devices (IADs) [70.112–114].

Given the Lagrange formulation of the exoskeleton, the basic of this approach is to let the control providing a predefined percentage of the force/torque required for the motion

$$\begin{cases} \mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \mathbf{G}(\boldsymbol{q}) + \mathbf{G}_{\mathrm{L}}(\boldsymbol{q}) = \\ \quad \boldsymbol{Q} + \sum_{\mathrm{c}} \mathbf{J}_{\mathrm{c}}\mathbf{F}_{\mathrm{c}} \,, \\ \boldsymbol{Q} = \mathbf{J}_{\mathrm{t}}^{\mathrm{T}}(\theta) F_{\mathrm{m}} \,, \\ \dot{\boldsymbol{q}} = \mathbf{J}_{\mathrm{t}}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \,, \end{cases} \tag{70.2}$$

where $\boldsymbol{q}$ is the joint coordinates vector, $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{G}$, are the exoskeleton mass, Coriolis, and gravity matrices, respectively, $\mathbf{G}_{\mathrm{L}}$ is the manipulated load, $\boldsymbol{Q}$ is the motor-torques vector reduced to the joints $\mathbf{J}_{\mathrm{c}}, \mathbf{F}_{\mathrm{c}}$ refers to the user contact forces and the respective Jacobians computed in the points of contact, and $\mathbf{J}_{\mathrm{t}}$ and $\mathbf{F}_{\mathrm{m}}$ represent the transmission Jacobian and the motor force/torque vector, respectively, and $\boldsymbol{\theta}$ is the motor-coordinates vector.

In order to properly work, this approach also supposes to have an almost accurate model of the mechanical design ($\hat{\mathbf{M}}, \hat{\mathbf{C}}, \hat{\mathbf{G}}, \hat{\mathbf{J}}_{\mathrm{c}}, \hat{\mathbf{J}}_{\mathrm{t}}$) and the load to manipulate ($\hat{\mathbf{G}}_{\mathrm{L}}$). In these conditions, the candidate control law is

$$\begin{aligned} \mathbf{F}_{\mathrm{m}} = (\mathbf{J}_{\mathrm{t}}^{\mathrm{T}})^{-1}(\hat{\mathbf{G}}(\boldsymbol{q}) + (1-\beta^{-1})\hat{\mathbf{G}}_{\mathrm{L}} \\ + (1-\alpha^{-1})(\hat{\mathbf{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \hat{\mathbf{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}})) \,. \end{aligned} \tag{70.3}$$

By inserting (70.4) in (70.2), we have

$$\begin{cases} \alpha^{-1}(\hat{\mathbf{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \hat{\mathbf{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}) + \beta^{-1}\mathbf{G}_{\mathrm{L}}(\boldsymbol{q}) \\ \quad = \sum_{\mathrm{c}} \mathbf{J}_{\mathrm{c}}\mathbf{F}_{\mathrm{c}} + \varepsilon(\boldsymbol{q}) \\ \varepsilon(\boldsymbol{q}) = (\mathbf{M}(\boldsymbol{q}) - \hat{\mathbf{M}}(\boldsymbol{q}))\ddot{\boldsymbol{q}} + (\mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) \\ \quad - \hat{\mathbf{C}}(\boldsymbol{q},\dot{\boldsymbol{q}}))\dot{\boldsymbol{q}} + (\mathbf{G}(\boldsymbol{q}) - \hat{\mathbf{G}}(\boldsymbol{q})) \\ \quad + (\mathbf{G}_{\mathrm{L}}(\boldsymbol{q}) - \hat{\mathbf{G}}_{\mathrm{L}}(\boldsymbol{q})) \,, \end{cases} \tag{70.4}$$

where the model error is expected to converge toward zero.

Equation (70.4) shows how the contact forces perceived by the user can be reduced by a static (β) and/or a dynamic (α) amplification factor. As a matter of fact, if the simplification operates correctly ($\varepsilon \to 0$), the pilot will only feel on his own body a percentage of the carried load (β^{-1}) plus a percentage of the exoskeleton dynamic (α^{-1}).

In the case of a walking exoskeleton, the above control model is switched according to the stancing foot, or in the case of double stancing (the weight is distributed on both feet), a simplified zero-moment-point (ZMP) rule is applied [70.115]

$$\begin{cases} m_{\mathrm{TR}} x_{\mathrm{TR}} = m_{\mathrm{TL}} x_{\mathrm{TL}} \,, \\ m_{\mathrm{TR}} + m_{\mathrm{TL}} = m_L \,, \\ \dfrac{x_{\mathrm{TR}} + x_{\mathrm{TL}}}{2} = x_g \,, \end{cases} \tag{70.5}$$

where x_{TR} and x_{TL} are the stancing position of the right and left feet, respectively, x_g is the projection of the center of mass on the walking plane, m_{L} is the overall load mass, and m_{TR} and m_{TL} are the mass components to be distributed on the right and left control models, respectively.

Equation (70.5) is used as a means to separate the closed-loop kinematics into two (quasi) symmetrical equations, similar to (70.2), that only refer to the left- and right-leg dynamics. The load is consequently distributed among the two legs, thus generating two *decoupled* control models, one for each leg.

Discussion 70.1

Experiments on such a kind of controller were performed at UC Berkeley. In [70.75], the system proved to be efficient to compensate completely the weight ($\beta^{-1} = 0$) with a good dynamic scale power amplification factor ($\alpha = 10$), and that the resulting system is usable for walking at an average speed of 1.3 m/s with a transported payload of 70 kg (including the exoskeleton weight).

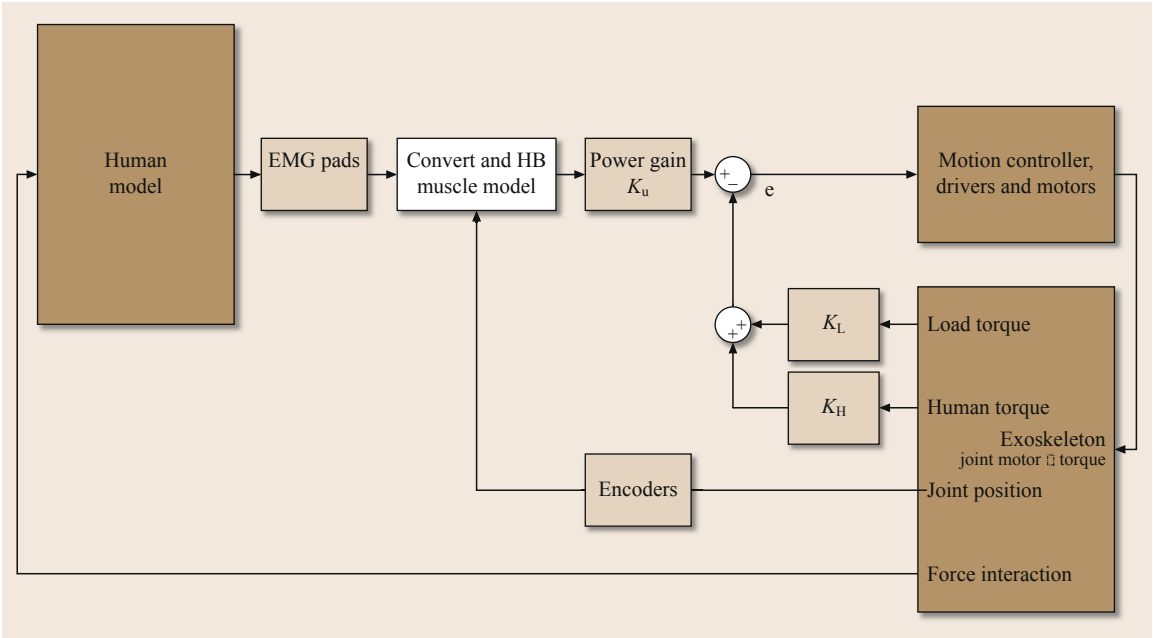


Fig. 70.19 The basic EMG controller (after [70.116]). The myo-electric signals were processed to derive a control signal through a sequence of (1) high-pass filter; (2) signal rectification (absolute value); (3) low-pass filtering; (4) normalization versus a nominal setup isometric contraction. The sign of the contraction is detected as a combination of pure agonistic–antagonistic contractions

The proposed design, however, suffers from a number of drawbacks that limit this control architecture being used in a more flexible interaction environment:

- The measurement of interaction torques made at the joint level makes it indistinguishable, whose interaction caused the reading, be it a voluntary movement provided by the user or an unvoluntary interaction provided by the environment. In both cases, the control amplifies these effects regardless of the fact that the original stimuli were provided by the user.
- The sensor and the mechanical model assumed in the discussed work only deal with a rigid mechanical design model whose structures risk to be cumbersome and dangerous when the load to be carried/manipulated is high. Flexible designs may manipulate higher loads at the cost of more complex control design [70.38].
- The robustness of the control is demanded to the accuracy of the model and the design of the compensation gains. When no gain is programmed ($\alpha = \beta = 1$), the feedback is zeroed and the user perceives all the passive effects of the dynamics. Increasing these gains lowers the stability margin in a way that is dependent on the quality of the model approximation.

70.5.3 EMG Force Control and Power Augmentation

Notwithstanding the fact that EMG-based arms were already developed in the early 1980s, power-augmented systems based on EMG signals only appeared at the beginning of the 2000s. In 2001, *Rosen* et al. [70.116] proposed a powered exoskeleton system based on the myosignal analysis (as shown in Fig 70.19).

In order to associate a map between the myo-electric signals and the amount of exerted force, Rosen first detected the level of activation, then he applied a basic muscle contraction/elongation model as described by *Hill* [70.117] during his investigation on tetanic contraction and by *Winter* [70.60] for antagonistic muscle analysis. According to Hill's model, the force correlates the combined effects of muscle activation, muscle elongation, and muscle speed to a well-known state equation

$$F = \frac{aF_0(v_0 + v)}{(vF_0 + av_0)}\,, \tag{70.6}$$

where F is the force exerted by the muscle, v is contraction velocity of the muscle, F_0 is the maximum isometric force generated by the muscle, v_0 is the muscle maximum velocity, and a is the coefficient of shortening heat.

Such a model has been applied to all involved muscles $M_f = M_f(U_{emg}, \theta_m, \dot{\theta}_m)$, to determine the moment M_f induced on the elbow joint angle θ_m, by the muscle fibers f. In addition to the muscle bio-signal, the control closed loop used two more data from sensors: (1) a measurement of the external load and (2) the overall momentum applied by the operator on the exoskeleton.

The underlying basic idea is that the muscle signal operates as an external loop that provides a reference for an internal moment controller. The three gains in the controller K_L, K_H, K_u define, respectively, the internal loop force gain K_H versus K_L, and the overall EMG signal tracking bandwidth.

In a similar effort to the Rosen elbow extender, *Kawai* et al. [70.118] proposed a 1-DOF-actuated wearable device that lowers the internal backbone strain during pick-up tasks in order to prevent from lumbago. The device senses the average back angle (measured through a potentiometer) and the EMG strain on the thigh muscles. The motor control for the device is determined through a state machine based on a typical load pick-up and release procedure. The threshold values for switching between phases were determined through a test with a nominal load.

In [70.119] and [70.82], the EMG data are employed to control the HAL-3, walking assistant, and exoskeleton. The control algorithm is a combination of a finite-state machine with a torque control. Two force sensors placed in soles of the shoes serve to determine the floor-reaction forces and to switch among phases. Each phase is then used to determine in which mode the user knee is moving (active, passive or free). The joint motors are consequently controlled using Table 70.1

In different phases of motion, the joint control adopts a computed torque approach plus two components proportional to the myo-signals of flexion and extension

$$\begin{aligned}\tau_{joints} &= K_{fl}\mathrm{EMG}_{fl} - K_{ex}\mathrm{EMG}_{ex} \\ &+ (I-M)\ddot{\boldsymbol{\theta}} + (D-B)\dot{\boldsymbol{\theta}} \\ &+ C(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + K(\boldsymbol{\theta}_0 - \boldsymbol{\theta}) \,, \end{aligned} \tag{70.7}$$

Table 70.1 The HAL dual-phase model. The phase 1 relates to the swinging leg and phase 2 relates to the stancing leg. In each phase, the table indicates the type of control to be implemented on the relative exoskeleton joint

	Phase 1		Phase 2	
Joint	Direction	Mode	Direction	Mode
Hip	Flexion	Active	Extension	Active
	Flexion		Flexion	
Knee	↓	Free	↓	Passive
	Extension		Extension	

where I and D are the inertia and viscous coefficients, respectively, as estimated through a recursive least squares auto regressive estimator (ARX) identification, C includes all Coriolis and gravity effects, M, B and K represent the target inertia, impedance to show to the operator. The term $\tau = K_{fl}\mathrm{EMG}_{fl} - K_{ex}\mathrm{EMG}_{ex}$ represents the estimated muscle torque whose conversion factors were determined through experimental tests.

The measure assistance level is close to 60% during flexion activities and to 85% during extension. In its most recent version, the HAL-5 adds contributes from accelerometers and gyroscope mounted on the backpack for better posture estimation [70.120]. HAL-5 extends the principle of operation of the lower body control to the user arm, by allowing the user to hold and carry an additional weight of 40 kg. HAL-5 extends the control with a greater number of phases (swing, landing, and support) and provides different control gains to optimize each phase [70.121].

The above EMG controllers only work fine when the developers are able to find two antagonistic muscles whose flexions and extensions can be directly related to the overall force exerted on a user joint (e.g., the bicepts and tricepts for the elbow joint). If this is not the case, *Kiguchi* et al. showed out how fuzzy controllers may estimate joint torques from several torques. In [70.122], they showed how to decode shoulder torques by combining the EMG signals recorded from eight different muscle sources. Such a fuzzy network has been coupled in the control of a 3-DOF mobile exoskeleton [70.123] to assist motion in highly impaired people.

70.5.4 Power Augmentation through Exteroceptive Sensors

High-force power augmentation systems require a different approach with respect to the proprioceptive and the EMG sensing approaches. In such devices, the flexibility of the structure and the high torques required at joints prevent the use of indirect measures of the force/torque exchanged between the device and the user. Moreover, the Hardiman approach, which integrated a master exoskeleton and an outfitting slave exoskeleton as a unique teleoperation system proved not to be functional [70.94].

Two systems have proved so far to reach this target in an operational and fully functional scenario: the Sarcos/Raytheon XOS Exoskeleton for the arms and legs [70.105] and the BE developed by PERCRO at Scuola Superiore Sant'Anna. Both of them are based on a similar working principle: the user body is directly attached to the exoskeleton in a predefined numbers of points, e.g. the two feet, the hands' handles, and a shoulder harness for the BE; and additional interac-

tion at the pelvis for the XOS. The interaction forces exchanged between the exoskeleton and the user are measured at these points and used to simultaneously control equilibrium, force amplification, and motion control.

On the other side, the two systems differ for a number of technical details: the XOS design is light, adherent to the user body and based on rotary-hydraulic actuators, the BE instead mimics the weight distribution of the human body, the exoskeleton does not touch the user body except for the predefined contact points and can automatically adapt to different human percentiles, and finally, all the actuation structure is based on a unique high-torque electrical motorization.

Very few data are documented about the operation of the XOS system and much of the information available is contained within its related patents, instead the kinematics and the control of the BE are described in several documents [70.104, 124].

70.5.5 Control Schemes

The BE control is organized as a decentralized control, where each joint is supervised by a separate electronics. Figure 70.20 shows the PERECRO-BE device. All the electronics are organized in four clusters which respectively map to the single human limbs. The interconnection between joint controllers is achieved through four separate EtherCAT-buses.

All connections converge into a central electronics placed in the trunk backpack, which manages high-level control of walking, stability, and manipulation.

A unique type of motorization as well as a unique type of low-level controller is implemented in each joint. Figure 70.21 highlights the interaction between the internal control loops and the higher control level. The internal controllers operate at a very high rate (2.5 kHz) and provide to estimate the joint velocities. At limb extremities, these controllers also acquire the information recorded by the embedded force sensors. The modularity of the architecture and the structure of the information do allow a high scalability of the system and a regular scheduling of the local and higher controllers.

The two components highlighted in Fig. 70.21 as *force control* and *feed forward control* are managed at the central level at 1 kHz basic control rate. The first generates, using the sensor force information, a velocity command that is forwarded to each limb cluster, whereas the second combines force information with posture and velocity information to define feedforward commands that can be directly added to compensate weight, loads or other gravity/inertial effects on the exoskeleton joints.

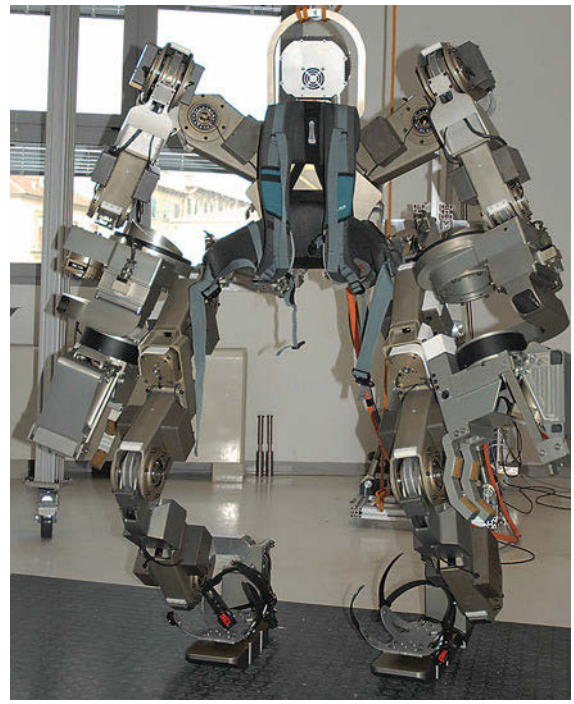


Fig. 70.20 The PERCRO-Body Extender has been already defined (BE). The BE is a whole-body exoskeleton structure designed to lift and manipulate 50 kg load per arm in the worst case conditions. The device has five attachment points with the user body: the two boots, two handles embedded in the grippers and the trunk corset. In addition to the internal sensors (position, force, and gyroscopes), force sensors embedded in the attachment connection help to determine the proper motion commands

Given its flexible structure, the central controller can operate on the joint controllers in different operational modalities; among these are: admittance controller, impedance controller, and feedback linearization controller.

Admittance Controller

Different control strategies have been implemented to move both the upper arm and the lower body part of the exoskeleton. Four basic schemes of admittance controller are implemented to manage the exoskeleton during walking and manipulation activities:

- Arm admittance
- Swinging leg
- Stancing leg
- Double stancing leg.

The basic scheme is the arm-admittance controller. In this approach, the local force sensed at the inter-

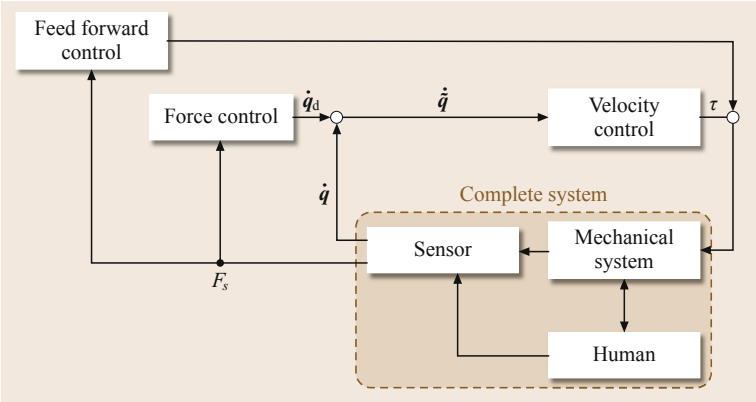


Fig. 70.21 The basic BE control structure. The internal velocity controllers are placed in the exoskeleton joints and locally perform a velocity servo loop (proportional derivative) which is enhanced by a direct feedforward torque. Both the inner-loop gain parameters and the external feedforward torque can be commanded from the higher level central control unit. This structure is repeated identically for all the exoskeleton joints

face is projected to the trunk reference system through the appropriate coordinate transformation. Its value is amplified through a gain factor and smoothed through a first-order low-pass filter that remove noisy effects. This reference signal is then back-propagated to the joint by using the inverse of the limb Jacobian.

$$\dot{\boldsymbol{q}}_{\mathrm{Ld}} = \mathbf{J}_{\mathrm{L}}^{\dagger} \frac{p_f \mathbf{K}_{\mathrm{Lv}}}{s + p_{\mathrm{f}}} \mathbf{R}_{\mathrm{h}}^{\mathrm{Tr}} \boldsymbol{F}^{\mathrm{h}} \,, \tag{70.8}$$

where $\boldsymbol{F}^{\mathrm{h}}$ is the handle force/torque vector as read by the sensor, $\mathbf{R}_{\mathrm{h}}^{\mathrm{Tr}}$ is the rotation matrix that maps to the appropriate trunk reference system, $\mathbf{K}_{\mathrm{Lv}}$ is a force/velocity gain, p_{f} is the smoothing frequency, and $\mathbf{J}_{\mathrm{L}}$ is the appropriate limb Jacobian that describes the motion of the limb with respect to the trunk.

In addition to the above velocity commands, the control system also computes in real time the gravitational effects produced by the exoskeleton joints and provide to compensate them through the feedforward input (τ_{ff})of the joint velocity controllers.

$$\tau_{\mathrm{ff}} = \tilde{\mathbf{G}}(\boldsymbol{q}) \mathbf{R}_{\mathrm{w}}^{\mathrm{Tr}} \begin{bmatrix} 0 \\ 0 \\ 9.81 \end{bmatrix} \,, \tag{70.9}$$

where $\tilde{\mathbf{G}}$ is the gravity contribution matrix (DOF $\times$ 3) that maps forces generated by the gravity to joints, $\mathbf{R}_{\mathbf{w}}^{\mathrm{Tr}}$ is the rotation matrix that describes the arm orientation with respect to a world frame having the gravity vector aligned with the z-axis.

The motion behavior generated to the programmed feedback policy can be examined through a two-step analysis. The internal position controller can be made asymptotically stable through standard robust control techniques. While the external admittance loop, which involves the human operator, exploits an intrinsically passive behavior, through the display of a pure viscous force to the operator. Hence, until the outer loop bandwidth is sufficiently smaller than the internal velocity servo, the two loops do not interfere [70.125].

The arm-admittance (70.8) scheme, also applies to floating legs which swing during walking. In such a case, the controller uses the forces measured by the booths which have been biased using offset values that ensure a stable contact among the user's foot and the exoskeleton booth.

During the stancing phase, the information provided by the force sensors in the booth is useless. Hence the force sensor between the corset and the trunk backpack (backpack force) is used as a trajectory generator for the leg motion. The reference trajectories are generated with a unique, dual step, procedure which do not change being the exoskeleton stancing on one or two legs.

In the first step, the *backpack force* has been determined. This force is filtered and amplified in the same way as for the handle admittance control. This operation generates a velocity vector $(\boldsymbol{v}_{\mathrm{b}}, \boldsymbol{\omega}_{\mathrm{b}})$ that applies to the backpack. In the second step, the central controller maps the same velocity vector to leg motion using a velocity map relationship that uses the direct kinematic between the trunk and the booths' soles. The kinematic vector, $\boldsymbol{r}_{\mathrm{l}}(q)$, that describes the booth position with respect to the *backpack force* sensor is

$$\begin{cases} \boldsymbol{v}_{\mathrm{f}} = \boldsymbol{v}_{\mathrm{b}} + \boldsymbol{\omega}_{\mathrm{b}} \wedge \boldsymbol{r}_{\mathrm{l}}(q) \,, \\ \boldsymbol{\omega}_{\mathrm{f}} = \boldsymbol{\omega}_{\mathrm{b}} \,. \end{cases} \tag{70.10}$$

The approach maps the force felt by the corset backpack to the proper leg motion regardless the knowledge of the ZMP position.

Feedback Linearization Controller

The major limitation of the admittance control is the lack of transparency. The user who wears the power augmentation exoskeleton only feels forces that are proportional to his speed, and the feedforward component

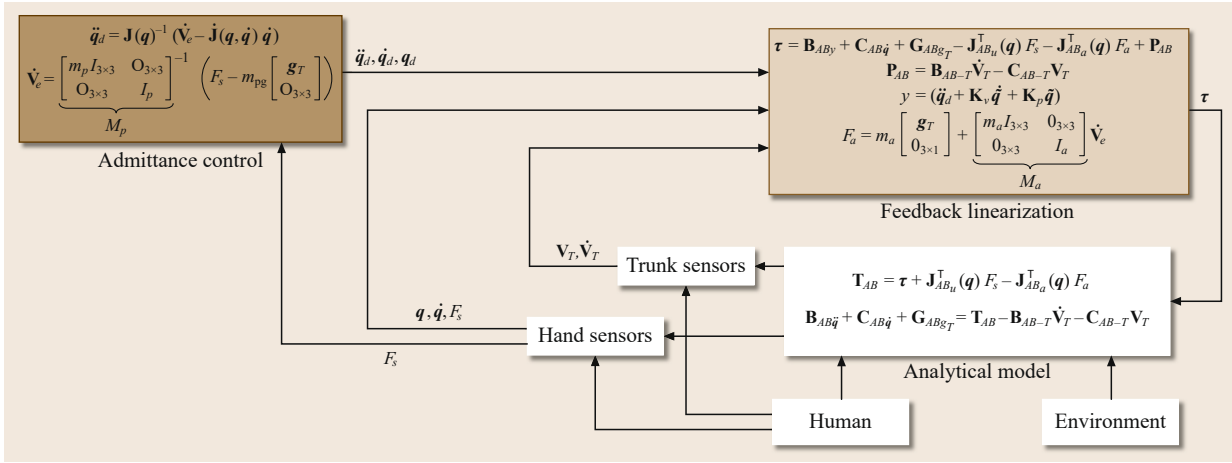


Fig. 70.22 The second-order admittance controller with feedback linearization. In this control scheme, the control input provided by the pilot generates a reference acceleration information for the limb. The feedback linearization provides to compensate the inertial/centrifugal/gravity effects of the exoskeleton and to map user input in proper acceleration information at the contact point. Moreover, the appropriate force amplification factor programmed at the sensor provide to establish an equilibrium among the environment interaction and the user commands

is only used to enlarge the control bandwidth and reduce the nonlinearity due to different postures.

However when the user touches any object, none of the contact force interferes with the exoskeleton motion creating in such a way an infinite force gain. While from the one side this is highly beneficial for joint stability, as a result it compromises the ability of the user to operate fine, scale force manipulation, tasks.

In order to recover this functionality, the BE implements on the same internal control structure two alternative control policies:

- A second-order admittance controller
- An admittance adaptive controller.

The second-order admittance controller collects user inputs to determine torque commands to the low-level joint controllers. This type of controller has been tested only on the upper body of the exoskeleton for the two relevant issues:

- The arms require greater manipulation capabilities and a better transparency
- The limited transparency of the leg motion reduces the interference between arm dynamics and trunk motion thus improving the overall stability.

The principle of operation for power augmentation is as follows (the related numerical equations have been reported in Fig. 70.22):

First, a force input is collected at each user handle. We suppose this force as being driven the interaction of a virtual mass with the given inertia properties (m_p, I_p) and a separate weight property (m_{pg}).

Hence the resulting handle (end-effector) velocity (V_e) will be determined as a direct integration of the user force as applied on this virtual mass, and the joint velocities as direct differentiation of the motion Jacobian (box admittance control 70.22).

The feedback linearization control here comprehends four components:

- The $B_{AB}y + C_{AB}\dot{q} + G_{AB}g_T$ component, which takes into account the tracking error y, and the rotated gravity vector g_T
- The effective user force F_s, considered as being equal to the sensor read, and already applied on the exoskeleton structure
- The weight and inertial effects produced on the exoskeleton by the carried load (m_a, I_a) moved at the prescribed velocity (V_e)
- And the additional inertial and Coriolis effects provided by the moving trunk ($\mathbf{P}_{AB}$).

As for the first-order admittance controller, the stability of the rigid model here can be ensured through the analysis of the linearized model. Combining the required equations, it can be derived that the robustness of the system is only affected by the amount of the inertia in the virtual mass properties (m_p, I_p). This property is unrelated from the power augmentation factor (m_a/m_p).

This is the reason why it is wise to decouple among the virtual mass (m_p, I_p) and the weight used for gravity

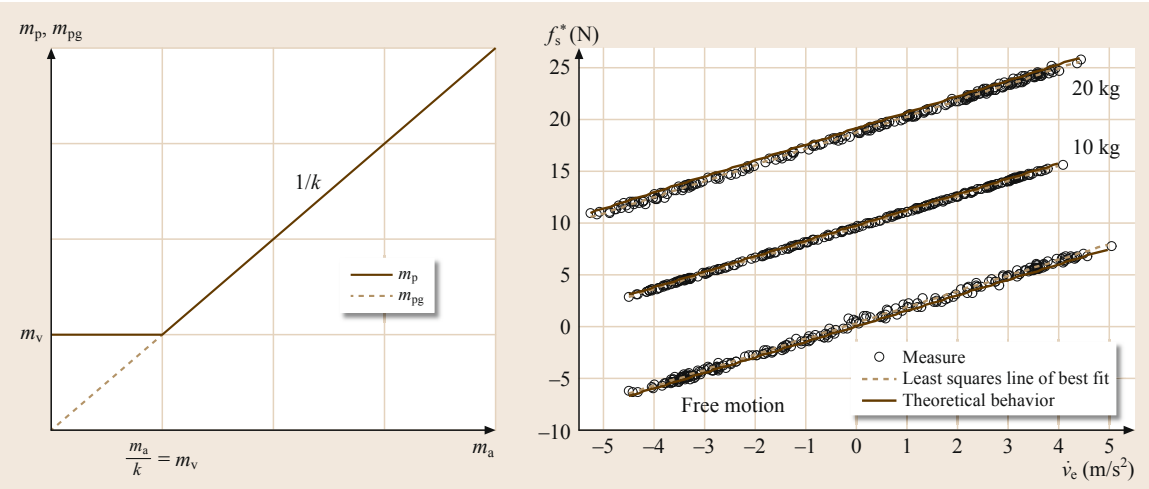


Fig. 70.23 Power augmentation profile. The power augmentation coefficient (k) is defined as the ratio of the carried load versus the perceived load. In the BE approach this ratio is represented as a piecewise linear approximation (left drawing). The quality of the control, tested on 1-DOF of the BE, has integrated with the adaptive policy and is shown on the right

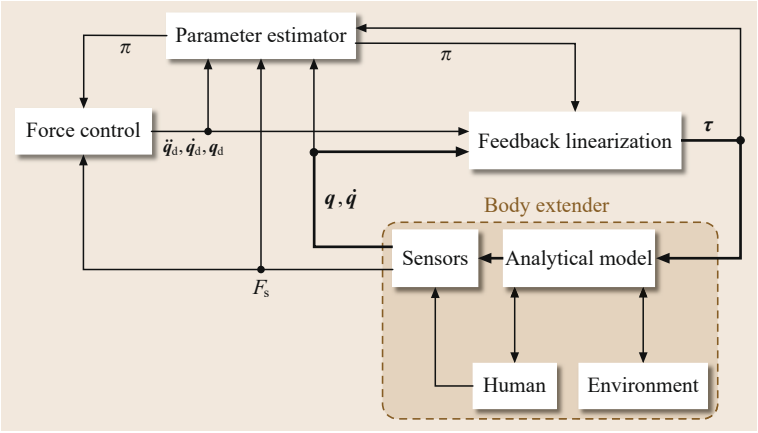


Fig. 70.24 The admittance adaptive controller with feedback linearization. A standard parameter estimator procedure has been applied to the robot structure in order to achieve online identification of the carried load. To simplify the computation and improve accuracy of estimation, the procedure assumes that the grasped object has regular mass properties and can be described with only two parameters as shown in Fig. 70.22

compensation (m_{pg}). The last not affecting the overall control stability.

Finally, the power augmentation factor has been defined as a piecewise linear approximation of the m_a versus m_p ratio which saturates for lower values of m_p in order to preserve overall stability (Fig. 70.23 left).

In order to work appropriately, the second-order admittance model requires an approximate estimation of the carried load (Fig. 70.23 right). It is supposed that this information to be subjected to the following conditions: the grasp is stable during manipulation, and the load does not change frequently.

Under these conditions, it is possible to use the motor currents as torque sensors that estimate the carried load. The principle of operation is reported in Fig. 70.24. Under the hypothesis that the load is stable and not frequently changing, the estimator can operate in lower frequencies.

Several tests to check the quality of estimation and the real-time adaptation to load change have been performed. In Fig. 70.24, the specific case of three different loads are reported. Attached to the video material of the present chapter the mass load/unload operation have been shown.

70.6 Conclusions and Further Developments

The presented review outlines some historical and state-of-the-art developments of wearable robotic systems for human–robot augmentation. Although EHPA systems are receiving an almost increasing interest from the robotics community, both from the research and from the commercial point of view, at present, the robotics technologies associated to their development are mainly derived from existing and well-established components. The lack of appropriate robotics components purposely developed for utilization on a *wearable* robotic system is mainly due, as testified by the existing research prototypes and commercial products, to a design approach that reflects the same methodologies pursued since the beginning for the design of industrial robots.

The design of a robotics system that must be worn by the human operator during a continuous operation also in unstructured environments should necessarily hamper design criteria, such as safety, fatigue, transparency, etc., that are only sometimes minimally considered for industrial robots.

From the above analysis of different typologies of EHPA systems, it comes out that the availability of adequate actuation components is fundamental to fulfill the basic functional requirements in terms of total weight, dynamics behavior, and control aspects. However, what is needed for the future research in this extraordinary interesting and challenging field of research in robotics deals primarily with a new radical design philosophy centered on biomechanical aspects of humans behavior. In particular, a novel methodology considering as primary criteria the issue of energy consumption, not only for the wearable robotic system itself but also in terms of the complex resulting system human operator-wearable robot seems a fundamental prerequisite for future developments.

As indicated in the previous sections, so far true advantages of exploiting an EHPA system in different real-application scenarios are still to be demonstrated. In fact, so far, research developments have been primarily devoted to the physical design and realization of wearable robots, without addressing the successive phase of validation and performance assessment in a true real environment. This aspect should enter as a prominent component for future developments of the research on EHPA.

The field of wearable robots is rapidly expanding and the associated challenges in terms of design methodologies and ad-hoc robotics technologies just began to be understood in their full meaning. The motivations for future developments in this field should always firmly maintain human operator's needs as the central focus of the research.

Video-References

VIDEO 146 Arm light exoskeleton (ALEx)
available from http://handbookofrobotics.org/view-chapter/70/videodetails/146

VIDEO 148 Arm-Exos
available from http://handbookofrobotics.org/view-chapter/70/videodetails/148

VIDEO 149 Body extender transversal joint
available from http://handbookofrobotics.org/view-chapter/70/videodetails/149

VIDEO 150 Hand-exoskeletons
available from http://handbookofrobotics.org/view-chapter/70/videodetails/150

VIDEO 151 Collaborative control of the Body Extender
available from http://handbookofrobotics.org/view-chapter/70/videodetails/151

VIDEO 152 Body Extender – A fully powered whole-body exoskeleton
available from http://handbookofrobotics.org/view-chapter/70/videodetails/152

VIDEO 180 L-Exos for upper-limb motor rehabilitation
available from http://handbookofrobotics.org/view-chapter/70/videodetails/180

References

70.1 M. McCullough: *Abstracting Craft: The Practiced Digital Hand* (MIT, Cambridge 1998)

70.2 H. Kazerooni: Human-robot interaction via the transfer of power and information signals, IEEE Trans. Syst. Man Cybern. **20**(2), 450–463 (1990)

70.3 A. Frisoli, F. Salsedo, M. Bergamasco, B. Rossi, M.C. Carboncini: A force-feedback exoskeleton for upper-limb rehabilitation in virtual reality, Appl. Bionics Biomech. **6**(2), 115–126 (2009)

70.4 J.E. Colgate, J. Edward, M.A. Peshkin, W. Wannasuphoprasit: Cobots: Robots for collaboration

with human operators, ASME Int. Cong. Mech. Eng. (1996) pp. 433–440
70.5 H. Kazerooni: Exoskeletons for human performance augmentation. In: *Handbook of Robotics*, ed. by B. Siciliano, O. Khatib (New York, Springer 2008) pp. 773–793
70.6 A.M. Dollar, H. Herr: Lower extremity exoskeletons and active orthoses: Challenges and state-of-the-art, IEEE Trans. Robotics **24**(1), 144–158 (2008)
70.7 R. Kobetic, C.S. To, J.R. Schnellenberger, M.L. Audu, T.C. Bulea, R. Gaudio, G. Pinault, S. Tashman, R.J. Triolo: Development of hybrid orthosis for standing, walking, stair climbing after spinal cord injury, J. Rehabil. Res. Dev. **46**(3), 447–462 (2009)
70.8 H. Herr: Exoskeletons and orthoses: Classification, design challenges and future directions, J. NeuroEng. Rehabil. **6**, 21 (2009)
70.9 A. Frisoli, F. Rocchi, S. Marcheschi, A. Dettori, F. Salsedo, M. Bergamasco: A new force-feedback arm exoskeleton for haptic interaction in virtual environments, Proc. 1st IEEE Jt. Eurohaptics Conf./Symp. Haptic Interfaces Virt. Environ. Teleoperator Syst. (2005) pp. 195–201
70.10 S.C. Jacobsen, F.M. Smith, E.K. Iversen, D.K. Backman: High performance, high dexterity, force reflective teleoperator, Proc. 38th Conf. Remote Syst. Technol., Washington (1990) pp. 180–185
70.11 Y. Ren, H.S. Park, L.Q. Zhang: Developing a whole-arm exoskeleton robot with hand opening and closing mechanism for upper limb stroke rehabilitation, IEEE Int. Conf. Rehabil. Robotics (ICORR) (2009) pp. 761–765
70.12 N.G. Tsagarakis, D.G. Caldwell: Development and control of a soft-actuated exoskeleton for use in physiotherapy and training, Auton. Robots **15**(1), 21–33 (2003)
70.13 J.C. Perry, J. Rosen, S. Burns: Upper-limb powered exoskeleton design, IEEE/ASME Trans. Mechatron. **12**(4), 408–417 (2007)
70.14 T.G. Sugar, J. He, E.J. Koeneman, J.B. Koeneman, R. Herman, H. Huang, R.S. Schultz, D.E. Herring, J. Wanberg, S. Balasubramanian, P. Swenson, J.A. Ward: Design and control of rupert: A device for robotic upper extremity repetitive therapy, IEEE Trans. Neural Syst. Rehabil. Eng. **15**(3), 336–346 (2007)
70.15 T. Nef, M. Mihelj, G. Kiefer, C. Perndl, R. Muller, R. Riener: Armin-exoskeleton for arm therapy in stroke patients, IEEE 10th Int. Conf. Rehabil. Robotics (ICORR) (2007) pp. 68–74
70.16 R. Gopura, D.S.V. Bandara, K. Kiguchi, G.K.I. Mann: Developments in hardware systems of active upper-limb exoskeleton robots: A review, J. Appl. Physiol. B **75**, 203–220 (2016)
70.17 G.M. Prisco, C.A. Avizzano, M. Calcara, S. Ciancio, S. Pinna, M. Bergamasco: A virtual environment with haptic feedback for the treatment of motor dexterity disabilities, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (1998) pp. 3721–3726
70.18 L.I. Lugo-Villeda, A. Frisoli, O. Sandoval-Gonzalez, M.A. Padilla, V. Parra-Vega, C.A. Avizzano, E. Ruffaldi, M. Bergamasco: Haptic guidance of light-exoskeleton for arm-rehabilitation tasks, 18th IEEE Int. Symp. Robot Human Interact. Commun. (RO-MAN) (2009) pp. 903–908
70.19 C. Carignan, J. Tang, S. Roderick: Development of an exoskeleton haptic interface for virtual task training, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 3697–3702
70.20 R. Vertechy, A. Frisoli, A. Dettori, M. Solazzi, M. Bergamasco: Development of a new exoskeleton for upper limb rehabilitation, IEEE Int. Conf. Rehabil. Robotics (ICORR) (2009) pp. 188–193
70.21 K. Kiguchi, T. Tanaka, K. Watanabe, T. Fukuda: Exoskeleton for human upper-limb motion support, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2003) pp. 2206–2211
70.22 T. Rahman, W. Sample, S. Jayakumar, M.M. King, J.Y. Wee, R. Seliktar, M. Alexander, M. Scavina, A. Clark: Passive exoskeletons for assisting limb movement, J. Rehabil. Res. Dev. **43**(5), 583 (2006)
70.23 S.J. Ball, I.E. Brown, S.H. Scott: MEDARM: A rehabilitation robot with 5DOF at the shoulder complex, IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (2007) pp. 1–6
70.24 J. Klein, S.J. Spencer, J. Allington, K. Minakata, E.T. Wolbrecht, R. Smith, J.E. Bobrow, D.J. Reinkensmeyer: Biomimetic orthosis for the neurorehabilitation of the elbow and shoulder (bones), 2nd IEEE/RAS/EMBS Int. Conf. Biomed. Robotics Biomechatron. (BioRob) (2008) pp. 535–541
70.25 P. Garrec, J.P. Friconneau, Y. Measson, Y. Perrot: ABLE, an innovative transparent exoskeleton for the upper-limb, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2008) pp. 1483–1488
70.26 A.M.M. Aalsma, F.C.T. van der Helm, H. van der Kooij: Dampace: Design of an exoskeleton for force-coordination training in upper-extremity rehabilitation, J. Med. Dev. **3**, 031003–31001 (2009)
70.27 M.H. Rahman, T.K. Ouimet, M. Saad, J.P. Kenne, P.S. Archambault: Development and control of a wearable robot for rehabilitation of elbow and shoulder joint movements, 36th Annu. Conf. IEEE Ind. Electron. Soc. (IECON) (2010) pp. 1506–1511
70.28 M.H. Rahman, M. Saad, J.P. Kenne, P.S. Archambault: Exoskeleton robot for rehabilitation of elbow and forearm movements, 18th Mediterr. Conf. Cont. Autom. (MED) (2010) pp. 1567–1572
70.29 A. Gupta, M.K. O'Malley: Design of a haptic arm exoskeleton for training and rehabilitation, IEEE/ASME Trans. Mechatron. **11**(3), 280–289 (2006)
70.30 H. Kawasaki, S. Ito, Y. Ishigure, Y. Nishimoto, T. Aoki, T. Mouri, H. Sakaeda, M. Abe: Development of a hand motion assist robot for rehabilitation therapy by patient self-motion control, IEEE 10th Int. Conf. Rehabil. Robotics (ICORR) (2007) pp. 234–240
70.31 P. Heo, G.M. Gu, S. Lee, K. Rhee, J. Kim: Current hand exoskeleton technologies for rehabil-

itation and assistive engineering, Int. J. Precis. Eng. Manuf. **13**(5), 807–824 (2012)
70.32 M. Fontana, S. Fabio, S. Marcheschi, M. Bergamasco: Haptic hand exoskeleton for precision grasp simulation, ASME J. Mech. Robotics **5**(4), 041014 (2013)
70.33 C.A. Avizzano, F. Bargagli, A. Frisoli, M. Bergamasco: The hand force feedback: analysis and control of a haptic device for the human-hand, IEEE Inter. Conf. Syst. Man Cybern., Vol. 2 (2000) pp. 989–994
70.34 M. Bergamasco, B. Allotta, L. Bosio, L. Ferretti, G. Parrini, G.M. Prisco, F. Salsedo, G. Sartini: An arm exoskeleton system for teleoperation and virtual environments applications, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1994) pp. 1449–1454
70.35 T. Nef, R. Riener: Shoulder actuation mechanisms for arm rehabilitation exoskeletons, 2nd IEEE/RAS/EMBS Int. Conf. Biomed. Robotics Biomechatron. (BioRob) (2008) pp. 862–868
70.36 P. DeVita, J. Helseth, T. Hortobagyi: Muscles do more positive than negative work in human locomotion, J. Exp. Biol. **210**(19), 3361–3373 (2007)
70.37 M. Bergamasco: Design of hand force feedback systems for glove-like advanced interfaces, Proc. IEEE Int. Work. Robot Human Commun. (1992) pp. 286–293
70.38 A. De Luca: Feedforward/feedback laws for the control of flexible robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 1 (2000) pp. 233–240
70.39 D.C. Clark, N.J. Deleys, C.W. Matheis: *Exploratory Investigation of the Man Amplifier Concept*, Tech. Documentary Rep. AMRL-TDR-62-89 (Cornell Aeronautical Laboratory, Buffalo 1962)
70.40 A. Zarudiansky: Remote handling devices, US Patent 430 2138 A (1981)
70.41 B.M. Jau: Anthropomorhic exoskeleton dual arm/hand telerobot controller, IEEE Int. Workshop Intell. Robots (1988) pp. 715–718
70.42 G. Burdea, J. Zhuang, E. Roskos, D. Silver, N. Langrana: A portable dextrous master with force feedback, Presence Teleoperators Virt. Environ. **1**(1), 18–28 (1992)
70.43 T. Koyama, I. Yamano, K. Takemura, T. Maeno: Multi-fingered exoskeleton haptic device using passive force feedback for dexterous teleoperation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 3 (2002) pp. 2905–2910
70.44 S. Nakagawara, H. Kajimoto, N. Kawakami, S. Tachi, I. Kawabuchi: An encounter-type multi-fingered master hand using circuitous joints, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 2667–2672
70.45 D. Gomez, G. Burdea, N. Langrana: Integration of the Rutgers Master II in a virtual reality simulation, Virt. Real. Annu. Int. Symp. (1995) pp. 198–202
70.46 A. Wege, K. Kondak, G. Hommel: Mechanical design and motion control of a hand exoskeleton for rehabilitation, IEEE Int. Conf. Mechatron. Autom., Vol. 1 (2005) pp. 155–159
70.47 M. Mulas, M. Folgheraiter, G. Gini: An emg-controlled exoskeleton for hand rehabilitation, IEEE 9th Int. Conf. Rehabil. Robotics (ICORR) (2005) pp. 371–374
70.48 S. Ito, H. Kawasaki, Y. Ishigure, M. Natsume, T. Mouri, Y. Nishimoto: A design of fine motion assist equipment for disabled hand in robotic rehabilitation system, J. Frankl. Inst. **348**(1), 79–89 (2011)
70.49 P. Brown, D. Jones, S.K. Singh, J.M. Rosen: The exoskeleton glove for control of paralyzed hands, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 642–647
70.50 B.L. Shields, J.A. Main, S.W. Peterson, A.M. Strauss: An anthropomorphic hand exoskeleton to prevent astronaut hand fatigue during extravehicular activities, IEEE Trans. Syst. Man Cybern. A **27**(5), 668–673 (1997)
70.51 Y. Yamada, T. Morizono, S. Sato, T. Shimohira, Y. Umetani, T. Yoshida, S. Aoki: Proposal of a Skilmate finger for EVA gloves, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (2001) pp. 1406–1412
70.52 J.Y. Wang, Z.W. Xie, J.D. Zhao, H.G. Fang, M.H. Jin, H. Liu: An exoskeleton system for measuring mechanical characteristics of extravehicular activity glove joint, IEEE Int. Conf. Robotics Biomim. (ROBIO) (2006) pp. 1260–1265
70.53 H. Herr, G.P. Whiteley, D. Childress: Cyborg technology – Biomimetic orthotic and prosthetic technology. In: *Biologically Inspired Intelligent Robots*, ed. by Y. Bar-Cohen, C. Breazeal (SPIE, Bellingham 2003)
70.54 K. Endo, H. Herr: A model of muscle-tendon function in human walking at self-selected speed, IEEE Trans. Neural Syst. Rehabil. Eng. **22**(2), 352–362 (2014)
70.55 H.M. Herr, A.M. Grabowski: Bionic ankle–foot prosthesis normalizes walking gait for persons with leg amputation, Proc. R. Soc. **B 279**(1728), 457–464 (2012)
70.56 J.M. Donelan, R. Kram, A.D. Kuo: Mechanical work for step-to-step transitions is a major determinant of the metabolic cost of human walking, J. Exp. Biol. **205**(23), 3717–3727 (2002)
70.57 A. Grabowski, C.T. Farley, R. Kram: Independent metabolic costs of supporting body weight and accelerating body mass during walking, J. Appl. Physiol. **98**(2), 579–583 (2005)
70.58 A.D. Kuo, J.M. Donelan, A. Ruina: Energetic consequences of walking like an inverted pendulum: Step-to-step transitions, Exerc. Sport Sci. Rev. **33**(2), 88–97 (2005)
70.59 R. Margaria: Positive and negative work performances and their efficiencies in human locomotion, Eur. J. Appl. Physiol. Occupat. Physiol. **25**(4), 339–351 (1968)
70.60 D.A. Winter: Energy generation and absorption at the ankle and knee during fast, natural, and slow cadences, Clin. Orthop. Rel. Res. **175**(175), 147 (1983)
70.61 A.H. Hansen, D.S. Childress, S.C. Miff, S.A. Gard, K.P. Mesplay: The human ankle during walk-

ing: implications for design of biomimetic ankle prostheses, J. Biomech. **37**(10), 1467–1474 (2004)
70.62 S. Au, H. Herr: Powered ankle-foot prosthesis, IEEE Robotics Autom. Mag. **15**(3), 52–59 (2008)
70.63 D. Paluska, H. Herr: The effect of series elasticity on actuator power and work output: Implications for robotic and prosthetic joint design, Robotics Auton. Syst. **54**(8), 667–673 (2006)
70.64 G. Pratt, M. Williamson, P. Dillworth, J. Pratt, A. Wright: Stiffness isn't everything, Lect. Notes Control Inf. Sci. **223**, 253–262 (1997)
70.65 J. Markowitz, P. Krishnaswamy, M.F. Eilenberg, K. Endo, C. Barnhart, H. Herr: Speed adaptation in a powered transtibial prosthesis controlled with a neuromuscular model, Philos. Trans. R. Soc. B Biol. Sci. **366**(1570), 1621–1631 (2011)
70.66 K. Endo, E. Swart, H. Herr: An artificial gastrocnemius for a transtibial prosthesis, Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC) (2009) pp. 5034–5037
70.67 H. Geyer, H. Herr: A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities, IEEE Trans. Neural Syst. Rehabil. Eng. **18**(3), 263–273 (2010)
70.68 J. Wang: EMG Control of Prosthetic Ankle Plantar Flexion, Ph.D. Thesis (MIT, Cambridge 2011)
70.69 R.A. Heinlein: *Starship Troopers* (Ace Books, New York 1987)
70.70 N. Yagn: Apparatus for facilitating walking, US Patent 42 0179 A (1890)
70.71 S.J. Zaroodny: *Bumpusher-A Powered Aid to Locomotion*, Tech. Note (Ballistic Research Laboratory, Aberdeen 1963)
70.72 R.S. Mosher: *Handyman to Hardiman*, Tech. Rep. (Society of Automotive Engineers, Warrendale 1967)
70.73 M.E. Rosheim: Man-amplifying exoskeleton, SPIE Proc. Mob. Robots IV **1195**, 402–411 (1989)
70.74 E. Garcia, J.M. Sater, J. Main: Exoskeletons for human performance augmentation (EHPA): A program summary, J. Robotics Soc. Jpn. **20**(8), 44–48 (2002)
70.75 H. Kazerooni: The berkeley lower extremity exoskeleton, Springer Tracts. Adv. Robotics **25**, 9–15 (2006)
70.76 A.B. Zoss, H. Kazerooni, A. Chu: Biomechanical design of the berkeley lower extremity exoskeleton (BLEEX), IEEE/ASME Trans. Mechatron. **11**(2), 128–138 (2006)
70.77 K. Amundson, J. Raade, N. Harding, H. Kazerooni: Hybrid hydraulic-electric power unit for field and service robots, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005) pp. 3453–3458
70.78 C.J. Walsh, K. Pasch, H. Herr: An autonomous, underactuated exoskeleton for load-carrying augmentation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 1410–1415
70.79 C.J. Walsh: Biomimetic Design of an Under-Actuated Leg Exoskeleton for Load-Carrying Augmentation, M.A. Thesis (MIT, Cambridge 2006)
70.80 C.J. Walsh, K. Endo, H. Herr: A quasi-passive leg exoskeleton for load-carrying augmentation, Int. J. Hum. Robotics **4**(03), 487–506 (2007)
70.81 N. Costa, D.G. Caldwell: Control of a biomimetic *soft-actuated* 10DoF lower body exoskeleton, 1st IEEE/RAS-EMBS Int. Conf. Biomed. Robotics Biomechatron. (BioRob) (2006) pp. 495–501
70.82 H. Kawamoto, S. Lee, S. Kanbe, Y. Sankai: Power assist method for HAL-3 using EMG-based feedback controller, IEEE Int. Conf. Syst. Man Cybern., Vol. 2 (2003) pp. 1648–1653
70.83 K. Yamamoto, K. Hyodo, M. Ishii, T. Matsuo: Development of power assisting suit for assisting nurse labor, JSME Int. J. Ser. C **45**(3), 703–711 (2002)
70.84 J.E. Pratt, B.T. Krupp, C.J. Morse, S.H. Collins: The roboknee: An exoskeleton for enhancing strength and endurance during walking, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (2004) pp. 2430–2435
70.85 Y. Sankai: Hal: Hybrid assistive limb based on cybernics. In: *Robotics Research*, ed. by M. Kaneko, Y. Nakamura (Springer, Berlin, Heidelberg 2011) pp. 25–34
70.86 US Army Research Laboratory, US Army Research Office: *ARO in Review* (USRL/USRO, Adelphi 2006)
70.87 J.E. Pratt, B.T. Krupp, C.J. Morse, S.H. Collins: The RoboKnee: An exoskeleton for enhancing strength and endurance during walking, Proc. ICRA IEEE Int. Conf. Robotics Autom., Vol. 3 (2001) pp. 2430–2435
70.88 C.J. Walsh, D. Paluska, K. Pasch, W. Grand, A. Valiente, H. Herr: Development of a lightweight, underactuated exoskeleton for load-carrying augmentation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 3485–3491
70.89 A. Goffer: Gait-locomotor apparatus, US Patent 7 153 242 (2006)
70.90 R. Little, R.A. Irving: Self contained powered exoskeleton walker for a disabled user, US Patent 201 1006 6088 A1 (2011)
70.91 R. Bogue: Exoskeletons and robotic prosthetics: A review of recent developments, Ind. Robot Int. J. **36**(5), 421–427 (2009)
70.92 C.P. Lent: Mobile space suit, US Patent 303 4131 A (1962)
70.93 N.J. Mizen: Powered exoskeletal apparatus for amplifying human strength in response to normal body movements, US Patent 344 9769 A (1969)
70.94 B.R. Fick, J.B. Makinson: *Hardiman I Prototype for Machine Augmentation of Human Strength and Endurance: Final Report*, Tech. Rep. S-71-1056 (General Electric Comp., Schenectady 1971)
70.95 T.J. Snyder, H. Kazerooni: A novel material handling system, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 2 (1996) pp. 1147–1152
70.96 B.R. Fick: Cutaneous stimuli sensor and transmission network, US Patent 353 5711 (1970)
70.97 S.R. Taal, Y. Sankai: Exoskeletal spine and shoulders for full body exoskeletons in health care, Adv. Appl. Sci. Res. **2**(6), 270–286 (2011)
70.98 T. Ishida, T. Kiyama, K. Osuka, G. Shirogauchi, R. Oya, H. Fujimoto: Movement analysis of power-

assistive machinery with high strength-amplification, Proc. SICE Annu. Conf. (2010) pp. 2022–2025
70.99 S. Jacobsen, M. Olivier: Contact displacement actuator system, Patent WO 20 0809 4191 A3 (2008)
70.100 S. Toyama, G. Yamamoto: Development of wearable-agri-robot mechanism for agricultural work, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 5801–5806
70.101 S. Jacobsen, M. Olivier, B. Maclean: Method of sizing actuators for a biomimetic mechanical joint, Patent WO 20 1002 5419 A3 (2010)
70.102 S. Marcheschi, F. Salsedo, M. Fontana, M. Bergamasco: Body extender: whole body exoskeleton for human power augmentation, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 611–616
70.103 M. Fontana, R. Vertechy, S. Marcheschi, F. Salsedo, M. Bergamasco: The body extender: A full-body exoskeleton for the transport and handling of heavy loads, IEEE Robotics Autom. Mag. **21**(4), 34–44 (2014)
70.104 G.P. Rosati Papini, C.A. Avizzano: Transparent force control for body extender, IEEE Int. Symp. Robot Human Interact. Commun. (RO-MAN) (2012) pp. 138–143
70.105 G. Mone: Building the real iron man, Pop. Sci. **4**, 1–6 (2008)
70.106 D. Bertrand: Lexosquelette hercule, le futur à nos portes, http://www.defense.gouv.fr/actualites/economie-et-technologie/l-exosquelette-hercule-le-futur-a-nos-portes (2011)
70.107 P. Filippi: Device for the automatic control of the articulation of the knee, US Patent 230 5291 (1937)
70.108 M. Vukobratovic, D. Hristic, Z. Stojiljkovic: Development of active anthropomorphic exoskeletons, Med. Biol. Eng. Comput. **12**(1), 66–80 (1974)
70.109 K. Kazerooni: On the robot compliant motion control, J. Dyn. Syst. Meas. Control. **111**(3), 416–425 (1989)
70.110 Y. Uchimura, H. Kazerooni: A μ-synthesis based control for compliant manoeuvres, IEEE Int. Conf. Syst. Man Cybern., Vol. 4 (1999) pp. 1014–1019
70.111 H. Kazerooni: The human power amplifier technology at the University of California, Robotics Auton. Syst. **19**(2), 179–187 (1996)
70.112 H. Kazerooni: Human power amplifier for lifting load including apparatus for preventing slack in lifting cable, US Patent 638 6513 A (2002)
70.113 J.E. Colgate, M. Peshkin, S.H. Klostermeyer: Intelligent assist devices in industrial applications: A review, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 3 (2003) pp. 2516–2521
70.114 D. McGee, P. Swanson: Method of controlling an intelligent assist device, US Patent 620 4620 A (2001)
70.115 H. Kazerooni: Exoskeletons for human power augmentation, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005) pp. 3459–3464
70.116 J. Rosen, M. Brand, M.B. Fuchs, M. Arcan: A myosignal-based powered exoskeleton system, IEEE Trans. Syst. Man Cybern. A **31**(3), 210–222 (2001)
70.117 A.V. Hill: The heat of shortening and the dynamic constants of muscle, Proc. R. Soc. B **126**(843), 136–195 (1938)
70.118 S. Kawai, K. Naruse, H. Yokoi, Y. Kakazu: An analysis of human motion for control of a wearable power assist system, J. Robotics Mechatron. **16**(3), 237–244 (2004)
70.119 S. Lee, Y. Sankai: Power assist control for walking aid with HAL-3 based on emg and impedance adjustment around knee joint, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vol. 2 (2002) pp. 1499–1504
70.120 E. Guizzo, H. Goldstein: The rise of the body bots [robotic exoskeletons], IEEE Spectrum **42**(10), 50–56 (2005)
70.121 K. Suzuki, G. Mito, H. Kawamoto, Y. Hasegawa, Y. Sankai: Intention-based walking support for paraplegia patients with robot suit HAL, Adv. Robotics **21**(12), 1441–1469 (2007)
70.122 K. Kiguchi, T. Tanaka, T. Fukuda: Neuro-fuzzy control of a robotic exoskeleton with EMG signals, IEEE Trans. Fuzzy Syst. **12**(4), 481–490 (2004)
70.123 K. Kiguchi, M.H. Rahman, M. Sasaki: Neuro-fuzzy based motion control of a robotic exoskeleton: considering end-effector force vectors, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 3146–3151
70.124 M. Bergamasco, A. Frisoli, C. Avizzano: Exoskeletons as man-machine interface systems for teleoperation and interaction in virtual environments, Adv. Telerobotics **31**, 61–76 (2007)
70.125 K. Zhou, J.C. Doyle, K. Glover (Eds.): *Robust and Optimal Control* (Prentice Hall, Upper Saddle River 1996)

Multimedia Contents

71. Cognitive Human–Robot Interaction

Bilge Mutlu, Nicholas Roy, Selma Šabanović

A key research challenge in robotics is to design robotic systems with the cognitive capabilities necessary to support human–robot interaction. These systems will need to have appropriate representations of the world; the task at hand; the capabilities, expectations, and actions of their human counterparts; and how their own actions might affect the world, their task, and their human partners. *Cognitive human–robot interaction* is a research area that considers human(s), robot(s), and their joint actions as a cognitive system and seeks to create models, algorithms, and design guidelines to enable the design of such systems. Core research activities in this area include the development of representations and actions that allow robots to participate in joint activities with people; a deeper understanding of human expectations and cognitive responses to robot actions; and, models of joint activity for human–robot interaction. This chapter surveys these research activities by drawing on research questions and advances from a wide range of fields including computer science, cognitive science, linguistics, and robotics.

71.1 **Human Models of Interaction** 1908
71.1.1 Mental Models of Robots 1908
71.1.2 Social Cognition 1911
71.2 **Robot Models of Interaction** 1914
71.2.1 Developmental Models 1915
71.2.2 Robot Spatial Cognition 1915
71.2.3 Symbol Grounding 1915
71.3 **Models of Human–Robot Interaction** 1916
71.3.1 Dialog-Based Models 1916
71.3.2 Simulation-Theoretic Models 1921
71.3.3 Intention- and Activity-Based Models 1922
71.3.4 Models for Action Planning 1923
71.3.5 Cognitive Models of Robot Control 1926
71.4 **Conclusion and Further Reading** 1927
Video-References 1927
References 1928

When people interact with each other, they draw on mental models of themselves, of their interaction partners, of the immediate context of the interaction, and of their broader physical, social, and cultural context. These models help them predict the actions of their interaction partners and make decisions about their own actions. To effectively interact with people, robots need similar models that help them determine their own actions and predict the actions of their users. *Cognitive human–robot interaction* (HRI) is a research area that seeks to improve interactions between robots and their users by developing cognitive models for robots and understanding human mental models of robots.

A central tenet of cognitive HRI is that humans, robots, and the context of their interaction form a complex cognitive system situated in the real world. A key research activity in the field involves the development of frameworks to represent this system [71.1–3]. This activity is informed primarily by research in cognitive science that develops frameworks to represent human cognitive systems. These frameworks include physical symbol systems [71.4], situated actions [71.5, 6], and those that combine symbolic and situated perspectives,

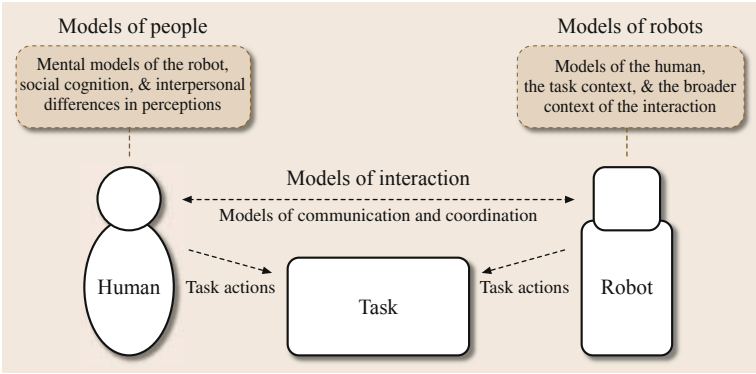


Fig. 71.1 A visual summary of the research activities in cognitive HRI

such as activity theory [71.7] and distributed cognition [71.8]. While the discussion on what framework best represents HRI as a cognitive system is ongoing, research in cognitive HRI involves the development of both symbolic and situated representations. More specifically, research activities in this area include (also illustrated in Fig. 71.1):

1. *Human models of interaction*: Building an understanding of people's mental models of robots, how people perceive robots and interpret their actions and behaviors, and how these perceptions and interpretations change across contexts and user groups.
2. *Robot models of interaction*: The development of models that enable robots to map aspects of the interaction into the physical world and develop cognitive capabilities through interaction with the social and physical environment.
3. *Models of HRI*: Creating models and mechanisms that guide human–robot communication and collaboration, action planning, and model learning.

This chapter surveys existing efforts in these research areas, drawing on research questions and advances from a wide range of fields including robotics, cognitive science, and linguistics.

71.1 Human Models of Interaction

Robots are expected to increasingly enter everyday environments – outside of factories and laboratories – including homes [71.9, 10], offices [71.11], and classrooms [71.12, 13]. In these contexts, robots will need to coexist and collaborate with a wide variety of users, such as children and the elderly, many of whom will not be technically trained. Accordingly, there is a growing research emphasis in cognitive HRI on identifying the mental models people use to make sense of emerging robotic technologies and investigating people's reactions to the appearance and behaviors of robots. This research aims not only to improve the ease of use of robots by designing them to fit human mental models, but also to gain new insights about human cognition and behavior. With the latter goal in mind, researchers also use robots to embody specific theories of human cognition that are then evaluated through HRI studies.

71.1.1 Mental Models of Robots

Research in human–computer interaction has shown that people's attitudes and behaviors toward digital technologies often follow the social rules established in human–human interaction [71.14]. It is reasonable to expect that people will similarly interpret the interactive behaviors of robots in social ways. Cognitive HRI researchers continue to investigate the extent to and conditions in which this maxim applies to HRI as they use their understanding of human social cognition to develop robotic platforms adapted to users' expectations and behaviors. In the process of evaluating such robotic platforms, researchers explore people's mental models of robots and identify areas in which users' expectations and understandings of robots may not be born out by the robot's appearance or behavior in ways detrimental to the HRI experience. Knowing which mental models people are using to interpret robot behavior not only helps roboticists to understand HRI more deeply, but also helps them in designing appropriate behaviors for the robot.

Models Ascribed to Robots

Extensive research by *Turkle* et al. [71.15, 16] examines how people, including children and older adults, make sense of their novel interactions with social robots such as Kismet, Cog, PARO, Furby, and My Real Baby.

These studies show that people apply a variety of mental models relating to animacy, sociality, affect, and consciousness to explain their experiences and emerging relationships with robots. Some research participants approached robots in a scientific-exploratory mode, interpreting a robot's actions in an emotionally detached and mechanistic manner. Others took a relational-animistic approach, investing in the interactions emotionally and treating robots as if they were living beings, such as babies or pets. The ways in which participants described the robots verbally did not always fit with the way in which they interacted with them – a person who says the robot is *only a mechanical thing* may still act toward the robot in a nurturing manner, such as soothing a crying My Real Baby [71.16, p.118]. This corresponds to previous findings in human–computer interaction (HCI) which suggest that people *mindlessly* apply social characteristics to computers [71.17]. Field studies with the seal-like robot PARO have shown that robots can also act as *evocative objects* that spark reflections on previous relationships and events (e.g., with a grandchild, spouse, or pet), which users then use to make sense of their interactions with the robots [71.18, 19].

In addition to identifying the mental models people use to interpret their experiences with robots, researchers study the effects from deliberately incorporating specific social schemas into robot design. *Anthropomorphism*, or the attribution of human characteristics to nonhuman (e.g., animal or artifact) behavior, is an interpretive schema that has been of particular interest to HRI researchers. Some scholars, such as *Nass* and *Moon* [71.17], critique anthropomorphic explanations as false and misleading. Others, including *Duffy* [71.20] and *Kiesler* et al. [71.21], suggest that the deliberate use of anthropomorphism can benefit social robot design by taking advantage of people's propensity to interpret events and other agents socially to make robot behaviors more understandable to users. This interpretation raises the question of which characteristics of the robot or the interaction are instrumental in inciting people to anthropomorphize robots and has inspired researchers to study a variety of socio-cultural cues, behaviors, and task contexts. *Kiesler* et al. [71.21] showed that people anthropomorphize a physically embodied robot more readily than an on-screen agent, and people behave in a more engaged and socially appropriate manner while interacting with the co-present robot. People also anthropomorphize robots they interact with directly more than they do with robots in general, and with robots that follow social conventions (e.g., polite robots) more than those that do not [71.22]. The personal characteristics, such as personality, of the human interaction partner can also affect their mental models of robots. For example, users with low emotional stability and extraversion scores were found to prefer mechanical-looking robots to human-like ones [71.23].

As might be expected, a robot's human-like appearance can have a positive effect on people's propensity to anthropomorphize [71.25]. Obversely, too high a level of human-likeness may place the robot in the *uncanny valley* [71.26]. The *uncanny valley* refers to a dip in the hypothetical nonlinear graph describing the relationship between a robot's human-likeness and a human's emotional response to it, suggesting that a robot with a very high degree of human-likeness coupled with some remaining nonhuman qualities will make users uncomfortable. This hypothesized effect essentially describes what happens when a person's mental model of the robot as human is not born out by its interactive capabilities. Various cognitive aspects of this hypothesis have been studied, suggesting that the construct is multidimensional [71.27, 28] rather than two-dimensional (2-D), as depicted by *Mori* [71.26]. Furthermore, research suggests that the mismatch between different dimensions, rather than any quality alone, can cause a dissonance that leads to people's discomfort with robots. *MacDorman* et al. [71.24] show that incongruencies between a robot's appearance and movement can diminish anthropomorphic attributions (Fig. 71.2). A similar result was found by *Saygin* et al. [71.29], who used functional magnetic resonance imaging (fMRI) to show that the human action perception system made distinctive responses to the mismatch between the level of human-likeness of a robot's appearance and motion but not to appearance or motion alone. Mismatches in the human- or robot-like qualities of an on-screen robot's voice and appearance were also shown to heighten people's sense of the character's eeriness – people found

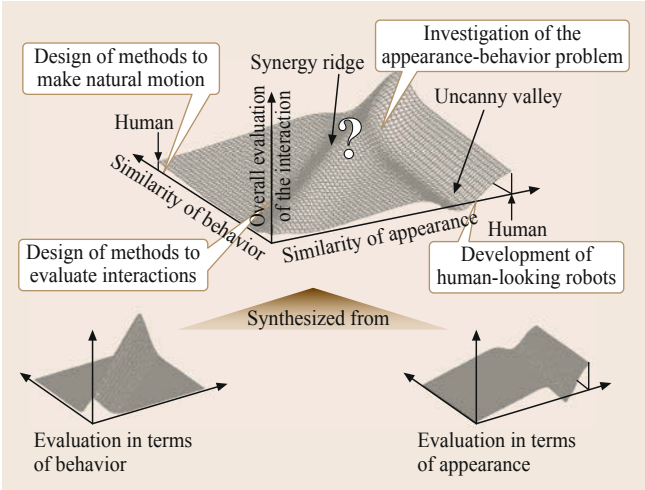


Fig. 71.2 An extended notion of the uncanny valley which includes appearance and behavior as significant variables (after [71.24])

both a robot with a human voice and a human with a robot voice to be creepy [71.30].

Mental Models in Robot Design

Researchers may deliberately include specific anthropomorphic schemas to promote user behaviors that aid robots in performing their tasks. One common example is the use of the *baby schema* – a soft round appearance, large eyes, and proto-verbal utterances – in Kismet [71.31] (Fig. 71.4), Muu [71.32] (Fig. 71.3), and Infanoid [71.33] to encourage people to anthropomorphize robots. This schema is also useful in that it can incite people to behave in a nurturing manner toward robots in the interest of scaffolding the robots' learning in a way similar to infant–parent interactions. A robot's *perceived* gender can also have an effect on people's mental models of the robot's knowledge of certain topics; for example, in one study a female robot was expected to be more knowledgeable about dating than a male robot [71.34]. While certain mental models become operational as soon as a person starts interacting with a robot (e.g., gender, age, human-likeness), people can adapt their mental models of a robot's capabilities when given additional information about the robot's personal characteristics, such as the robot's country of origin or the language it speaks [71.35]. *Goetz* et al. [71.36] showed that matching a robot's *personality* to the task it is supposed to perform can have a significant effect on its efficacy: people were more responsive to a robot that had a serious, rather than an entertaining, demeanor when its job was to motivate them to exercise. Also focusing on *task models* in HRI, *Lee* et al. [71.37] showed how people use their existing utilitarian and relational models of service to set expectations for their interactions with a service robot. These models also affected the preferred ways in which the robot should make up for any mistakes it makes in service – people with a utilitarian mental model of service preferred to receive compensation, while those with a relational model responded well to an apology.

As interactive robots are developed and used all over the world, researchers have also started exploring how *cultural models* [71.38] affect people's perceptions of and interactions with robots. Social and behavioral norms are culturally variable, so we can expect users' understanding and adoption of socially interactive robots to differ accordingly. Cross-cultural research in HRI largely supports this expectation. *Evers* et al. [71.39] showed that users from China and the US respond differently to robots. Further research by *Wang* et al. [71.40] suggests that specific cultural models regarding communication norms, particularly explicit and implicit modes of communicating information and intent to interaction partners, affect people's perceptions of a robot's trustworthiness and its in-group membership. Researchers have also shown that roboticists themselves use cultural models unintentionally in their work, including particular models of emotional display [71.41], and cultural models reflecting historical, theological, and popular perceptions of robotic technology [71.42]. Research on cultural models in HRI not only points to the importance of reflexively including such models in robot design, but also allows researchers to do systematic research on culturally situated cognition using robots as stimuli.

Research on mental models applied to interactive robots has not only shown that people use their existing mental models to make sense of these novel artifacts, but also that we may need new *ontological*

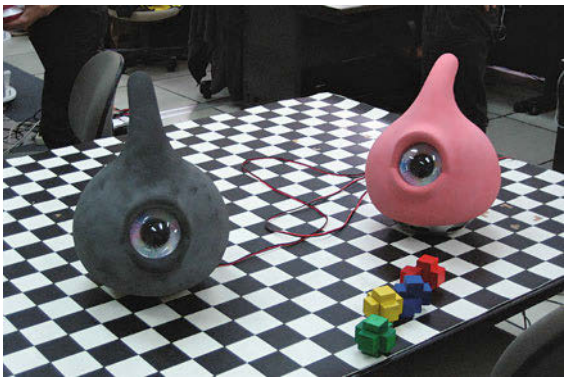


Fig. 71.3 Muu's big eyes and soft round body are designed according to the *baby schema*. Using two robots that can interact with each other instead of one suggests a relational understanding of agency (courtesy of Šabanović)

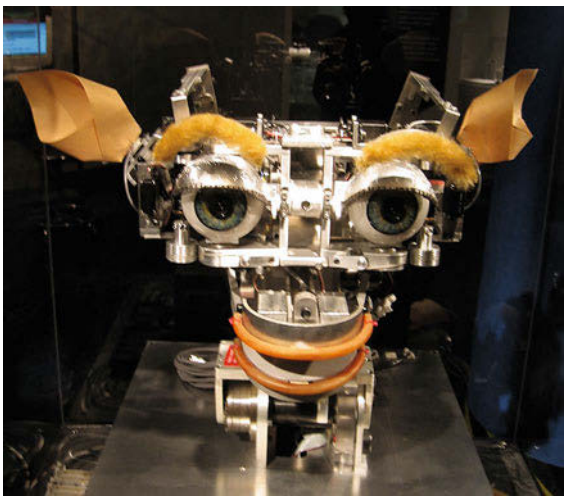


Fig. 71.4 Kismet's big round eyes and infant-like vocalizations are another example of the *baby schema* (courtesy of Šabanović)

categories to accommodate emerging mental models of these entities [71.16, 43]. *Kahn*'s et al. [71.44] studies of children's moral interpretations of interactions with an AIBO robot showed that their mental models of the robots included rationalizations and behaviors related to both inanimate and animate objects. *Turkle* [71.45] suggests that interactive robots co-opting relational feelings and responses normally reserved for animals and humans call into question the authenticity of relationships. Further, *Turkle* [71.45] suggests that a more sophisticated new notion of autonomous yet inanimate artifacts has become necessary. Both researchers have suggested that interactive robots might comprise a new ontological category, and that we also need to be conscious of the ways in which interactions with these artifacts affect our mental models of animate beings.

71.1.2 Social Cognition

The development of robots that can interact naturally with humans calls for the detailed study of social activity and the cognitive models that underly such activities. *Scassellati* [71.46] argues that robots can help us study the limits of human social cognition because they are not alive, yet they can behave in socially appropriate (or inappropriate) and evocative ways. Robots that incorporate social cues such as gaze, proximity, and facial expressions, push our *Darwinian buttons* [71.16, p. 8] and effectively coerce us into interacting with them socially. Studying which cues have these effects is an opportunity to learn more about human social cognition and improve robot design.

Researchers studying the social aspects of cognitive HRI are identifying the minimal cues robots need to evoke social responses from people, including those related to robotic embodiment, gaze, proxemic cues, and interaction rhythms. Current research is also focused on applying and evaluating different models of cognition in the context of HRI. Robots can be unprecedented experimental tools for the study of social cognition. They can be used to provide stimuli in experiments and field studies, since their actions and behaviors can be carefully controlled, finely tuned and varied, and repeated exactly and indefinitely, which is often challenging even for well-trained human confederates [71.47, 48]. Furthermore, robots do not have difficulty acting *unnaturally* (e.g., not reacting to other person's cues) or violating social norms (e.g., being rude) when needed, a source of potential stress in human researchers [71.49].

Minimal and Human-Like Cues in HRI

One approach to studying social cognition has been to try to isolate the *minimal* set of cues that evoke social responses and perceptions from human interaction partners. The creators of Muu followed a *minimal design strategy* [71.32], using cartoons and children's drawings to develop a robot that can be communicatively engaging to people without relying on overt human-likeness. *Kozima* et al.'s [71.50] Keepon was designed to include characteristics common to living beings, such as lateral symmetry and two eyes, which are assumed to be important for social interaction (Fig. 71.5). The robot also performs fundamental social behaviors, such as joint attention, eye contact, and emotional expression through bodily posture and movement and using only four degrees of freedom (Fig. 71.6). These minimal cues have been shown to be sufficient for engaging children in short-term interaction in the lab and long-term interaction in more natural environments, such as a classroom [71.50].

Studies with minimalist robots have also underscored the effect of *social context* in people's interpretations of robots. Field studies with Keepon in an elementary school showed that children incorpo-

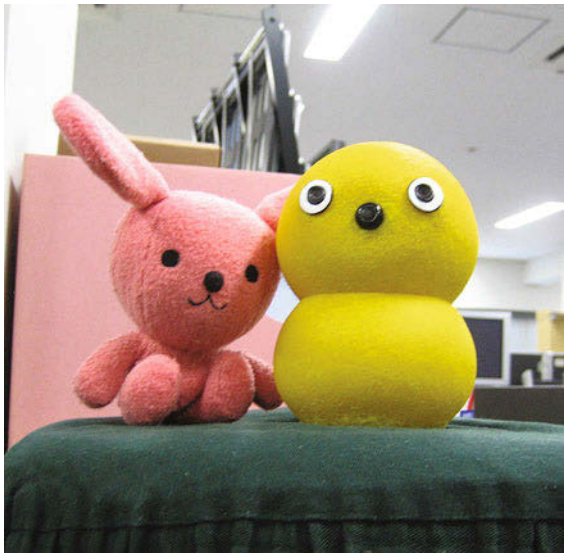


Fig. 71.5 Keepon is a simple robot used to investigate cues such as joint attention, emotive expression, and rhythmicity in HRI (courtesy of Šabanović)

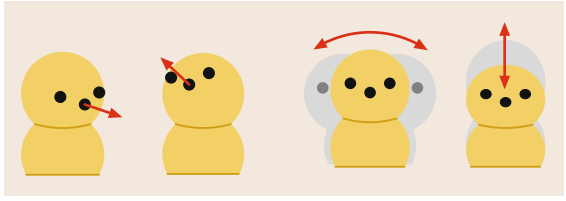


Fig. 71.6 Keepon uses four degrees of freedom to express emotive and attentional cues (after [71.50])

Fig. 71.7 An android robot fabricated by Kokoro Ltd. (courtesy of Šabanović)

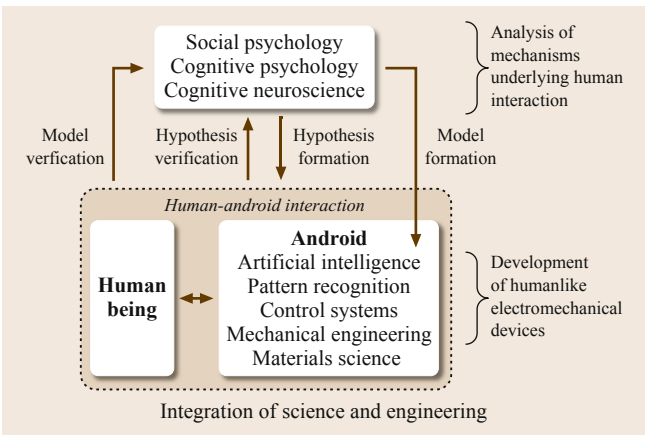


Fig. 71.8 Androids can be used to investigate human cognition analytically as well as synthetically (after [71.51])

rated the simple robot into a wide variety of interaction contexts (e.g., playing house with Keepon as a baby or pet, or treating Keepon as another student in the classroom) due not only to its interpretive flexibility, but to the richness of the social environment. This inspired children to engage with the robot over long periods of time, sometimes years, whereas they became bored after 10–15 min when interacting with Keepon in the laboratory. The above mentioned Muu's design was inspired by ecological models of cognition [71.52, 53] suggesting that a robot is inherently incomplete as a communicative device – it needs a human interaction partner to imbue its actions with meaning. Muu therefore relies on the context and the presence of other interactive agents (including people, other Muu, and objects such as blocks displayed in Fig. 71.3 for triadic interaction) to enable people to make sense of its actions and relationally ascribe social agency to the robot. The Social Trashcan project [71.54] similarly explored how minimal social cues, including contingent motion and approaching people, can be used to display the robot's intentions to children and get their assistance in trash collection. *Yamaji* et al. [71.54] also showed that robots moving together as a group – relationally – were more successful in attracting the children's attention than that moved individually.

An alternative approach to the study of social cognition through HRI focuses on *human-like realism* in appearance and behavior and is proposed by *Ishiguro* [71.47] and *MacDorman* and *Ishiguro* [71.48]. They claim that androids – robots that bear a close and sometimes *uncanny* resemblance to humans (Fig. 71.7, for example) – are unprecedented test beds for the study of social cognition. Used as stand-ins for humans in this *android science*, robots have a twofold function as experimental tools for evaluating hypotheses about human perception, cognition, and interaction, and as a testing ground for various cognitive models (Fig. 71.8). Using an android platform, *Ishiguro* [71.55] showed the importance of micro movements as a cue that incites people to attribute human-likeness to a robot in short (1–2 s) interactions. Another topic of continuing investigation is the possibility of simulating the personal presence of a remote actor in the local environment using an android platform [71.56, 57]. *Shimada* et al. [71.58] showed that people evaluated an android as more likable when it mimics them in a way similar to the *chameleon effect* that occurs when two people interact. *MacDorman* and *Ishiguro* [71.48] suggested that such androids can be used in research relating to a number of current topics of interest in cognitive science, including the mind-body problem, nature versus nurture, rationality and emotion in human reasoning, and the relationship between social interaction and internal cognitive mechanisms.

Embodied Social Cues

Embodiment separates robots from other interactive digital technologies and has been investigated through studies comparing how people interpret and act toward robots that are physically co-present with them and on-screen robots or social agents. *Wainer* et al.'s [71.59] comparison of people's interactions with an embodied robot and a simulated robot, and a co- and tele-present found that people were more engaged with, behaved more appropriately to, and anthropomorphized a co-located robot more than a tele-present robot. People interacting with an embodied robot have also shown tendencies to issue more commands than those interacting with a simulated robot [71.60]. The social effect of embodiment in HRI was further confirmed by *Bainbridge*'s et al. [71.61] study showing that people are more likely to comply with requests made by an immediately present robot rather than requests made by a remote robot communicating with them through a television screen. These converging results strongly suggest that a robot's embodied presence has a significant cognitive effect on people's social responses to the robot. The embodied nature of robots also enables the study and use of various other social cues, including proxemic behaviors, gaze, and interaction rhythms, in HRI. A more detailed review of embodied social cues is provided by *Mutlu* [71.62].

Proxemic behaviors [71.63], the study of which is enabled by the embodied nature of robots, not only have a significant effect on people's perceptions of and behaviors toward robots, but have also been used as a measure of people's perceptions of robots as social agents. *Takayama* and *Pantofaru* [71.64] found that prior experience with pets and robots decreased the distance at which people felt comfortable around robots. Individual traits such as gender and personality also affect people's preferences regarding the distance at which they are comfortable with a robot approaching them [71.64, 65]. Proxemic behavior can be related to other social cues in complex ways. For example, *Mumm*'s and *Mutlu*'s [71.66] study showed that people will compensate for the intense gaze in their direction of a robot they do not like by moving away from the robot (Fig. 71.9). While most studies of proxemic systems have been done in the laboratory, recent work is also investigating more natural interactions between humans and robots in open environments [71.67].

Gaze is an important cue in human–human interaction and is also one of the most studied nonverbal social cues in HRI (VIDEO 128). People use many such seemingly unintentional, unconscious, and automatic nonverbal cues as clues regarding the mental states and intentions of other actors, including robots. Gaze has been shown to be useful for communicating intent, modulating interaction, and even affecting participants' experience and memory of the interaction. Researchers have shown that gaze can be used to engage users [71.69, 70] and to assign them particular roles in and manage the interaction [71.71]. A robot's gaze behavior can affect the human interaction partner's gaze and speech, their comprehension of the robot's speech [71.72], and people's memory of a story narrated by the robot and perceptions of the robotic storyteller [71.73]. Researchers studying the temporal aspects of gaze in HRI found that the timing of gaze behavior provides cues to human intentions while teaching the robot the names of objects, suggesting that properly timed gaze behaviors can have a positive effect on collaborative tasks between a human and a robot [71.68] (Fig. 71.10). *Yu* et al. [71.74] have developed a data-driven approach to analyzing human gaze in the context of HRI, which can be used

Fig. 71.9 In the study by *Mumm* and *Mutlu* [71.66], participants maintain a greater distance with the unlikable robot when the robot follows them with its gaze than when its gaze avoids the participant, while their proxemic behavior is not affected by a likable robot's gaze (courtesy of Mutlu)

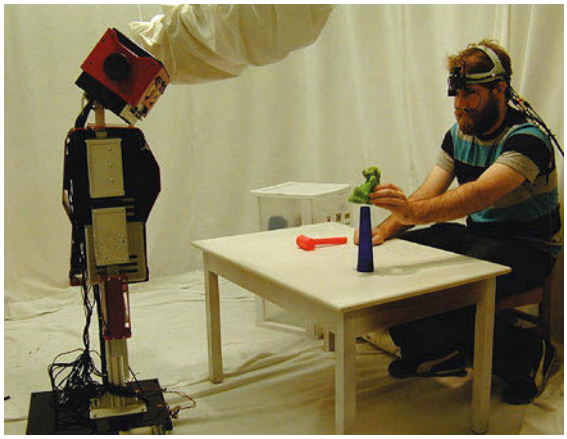


Fig. 71.10 *Yu*'s et al. [71.68] HRI studies provide data for developing models of the temporal aspects of interaction (courtesy of Yu)

to develop detailed micro-behavioral gaze models that can guide robot behaviors as well as be used to understand human intentions and behaviors in the course of collaborative activities. A recent study by *Admoni* et al. [71.75], contrary to the assumption that anthropomorphic robots engage us automatically in much the same way we are engaged by people, shows that robot gaze is not necessarily treated by people in the same automatic way that human gaze is treated; we do not, therefore, necessarily perceive robots as social in an automatic and *mindless* way.

Interaction rhythms – nonverbal and largely unconscious temporal coordination between partners in an interaction – enable the exchange of information, anticipation of the interaction partner's actions, and even positive evaluations of interaction among humans as a fundamental subscript of all human interaction [71.76–78]. The rhythmicity of interaction is therefore also a crucial factor in HRI, both in terms of developing robots that can perceive and respond to people's rhythmicity, and of understanding how people react to the temporal aspects of robot behaviors. *Michalowski* et al. [71.79] used a dancing robot to explore the rhythmic properties of social interaction and showed that children were more likely to interact with a robot that was synchronized to background music rather than one that was not, and that the children's own rhythmic behavior was influenced by the robot's rhythmicity. In further research, *Michalowski* et al. [71.80] suggest that rhythmic interaction can be used as a form of play between children and robots, and that following the robot's lead in rhythmic entrainment with music causes children to attend more closely to musical rhythm. *Avrunin* et al. [71.81] found that simple changes in a robot's rhythmic dancing behavior, such as variation of motions, flaws in the robot's synchrony with music, and coordination of behavior changes with musical dynamics, increased people's perceptions of the robot's lifelikeness. *Hoffman* and *Breazeal* [71.82] used the temporal patterns of interaction – its rhythms – to develop robotic systems that can anticipate a human partner's actions in collaborative tasks, such as AUR, a robotic desk lamp, and Shimon, a marimba playing robot [71.83] (VIDEO 236). Along with improving HRI, the use of robots in studying the rhythmic properties of interaction provides a new tool for cognitive science research on these subtle, fine grained, and unconscious social cues.

Cognitive Development in HRI

A further topic of focus in cognitive HRI has been the study of social and cognitive development through studies of typically developing and autistic children's interactions with robots. Multiple studies in educational contexts have focused on understanding how children ascribe social agency to robots [71.12, 13]. *Kozima* et al. [71.50] found that children of different ages display varying modes of interaction with the robot, which suggest different levels of comprehension of its ontological status – 0-year-old interacted with Keepon as a moving thing, 1–2-year-old interacted with the robot as an autonomous system, and children over 2 years of age treated the robot as a social agent. *Deàk* et al. [71.84] studied the mechanism of joint attention in HRI to explore the importance of contingency and find out which perceptual features infants use to achieve shared attention by modeling these in a robot. Researchers also use robots to study social deficit disorders, particularly autism. Converging results on research using robots to study social deficit disorders show that autistic children respond to robots in a social manner that they do not display with people [71.85–87], inspiring researchers to perform studies with children in the context of HRI. One aim of such research is to try to understand which aspects of a robot's behavior enable autistic children to participate in social interaction, which may clarify some of the reasons for their difficulties when interacting with humans. HRI researchers have also applied robots to various therapeutic scenarios with autistic children in an effort to provide parents and therapists with a tool to improve communication and understand the children better [71.88, 89]. Studies by *Kozima* et al. [71.90] in which the robot Keepon interacts with children with autism suggest that such minimally designed robots can be used to motivate autistic children to share their mental states with others, such as therapists or parents. This work poses a promising possibility for learning more about social deficits and development disorders such as autism, as well as providing tools for diagnosis and therapy using robotic technologies.

71.2 Robot Models of Interaction

Simon [71.91] suggests that the study of human behavior can be approached through synthesis as well as analysis and designs computer simulations as a technique for understanding and predicting the behavior of natural, social, and cognitive systems. In the spirit of Simon's synthetic approach to the study of human cognition, robotics researchers have been engineering robots as tools for developing and testing a variety of cog-

nitive, behavioral, and developmental models [71.47, 92, 93]. This approach assumes that cognitive models are validated when the implementation of a particular model on a robot produces behavior similar to that produced by humans in the same situation; if this does not occur, it is a sign that there may be something wrong with the model or the way it was implemented in the robot [71.46]. Cognitive HRI research involves the development of robotic platforms based on findings from cognitive science and using such platforms to extend knowledge about human cognitive processes.

71.2.1 Developmental Models

Robots are particularly appropriate for exploring theories of embodied and social cognition, which emphasize the centrality of the agent's interactions with its environment and other agents in that environment to cognitive functioning. In the process of synthesizing a robotic system, the researcher is drawn to focus on the dependency of cognition on noncognitive processes, including the social and physical environment in which cognition takes place. Robots such as Cog and Kismet [71.31] have been used to simulate and validate different theories of cognition, perception, and behavior. Cog was used to implement and test cognitive models relating to reaching behavior, rhythmic motor skills, visual search and attention, and social skill acquisition (e.g., joint attention and theory of mind). In the process they were able to validate, extend, and show the limitations of cognitive, behavioral, and developmental theories. In later projects, researchers have developed models inspired by human cognition and behavior such as social referencing [71.94], perception and action loops [71.95], anticipatory actions in collaborative teamwork [71.96], and others.

Robotics researchers apply the idea that the development of intelligence is embedded in social and cultural environment to the construction of robotic artifacts. For example, *Breazeal* [71.31] applied theories relating to infant social development, psychology, ethology, and evolution to design the robot Kismet, which used infant-like social cues to engage a human participant in interactions that would scaffold the robot's learning, as in the case of infant–parent interactions. Researchers have also developed a variety of robotic systems that exhibit cognitive traits such as imitation [71.97, 98], joint attention [71.99–101], and rhythmic synchrony [71.50, 102]. The Infanoid project [71.33] also used a synthetic approach in which development was understood through studying how the robot learns. Situated and embodied models have been applied to robot learning, particularly through imitation. For example, *Bakker* and *Kuniyoshi* [71.103] propose *imitation* as an interaction and learning paradigm in contrast to *robot programming* or *robot learning*. Further, they argue that robot programming is too hard and tedious to specify complex behaviors in sufficient detail and specify how they might be adapted to novel situations.

71.2.2 Robot Spatial Cognition

Systems dedicated to modeling spatial language and interaction, including the theories by *Jackendoff* [71.104], *Landau* and *Jackendoff* [71.105], and *Talmy* [71.106], have been produced for many years. Several previous works have been computational instantiations of the ideas presented in these theories, in particular the implementation and testing of spatial semantics models. *Regier* [71.107] built a system that assigns labels, such as *through* to a movie showing a figure moving relative to a landmark object. *Kelleher* and *Costello* [71.108] and *Regier* and *Carlson* [71.109] built models for the meanings of static spatial prepositions, such as *in front of* and *above*.

Many authors have proposed formalisms for enabling systems to reason about the semantics of natural language use in the context of giving directions. For example, *Bugmann* et al. [71.110] identified a set of 15 primitive procedures associated with clauses in a corpus of spoken natural language directions. *Levit* and *Roy* [71.111] designed *navigational informational units* that break down instructions into components. *MacMahon* et al. [71.112] represented a clause in a set of directions as a compound action consisting of a simple action (move, turn, verify, and declare-goal), plus a set of pre- and post-conditions. Many of these previous representations are expressive but difficult to automatically extract from text. Some authors avoid this problem by using human annotations [71.111, 112] or by specifying the robot's behavior in a controlled language [71.113]. *Matuszek* et al. [71.114] created a system that follows directions using a machine translation approach. Similarly, *Vogel* and *Jurafsky* [71.115] used reinforcement learning to automatically learn a model for understanding route instructions.

71.2.3 Symbol Grounding

Mapping language from the human partner to aspects of the external world – locations, objects, or actions the robot should take – described by the language was referred to as an instance of the *symbol grounding* problem [71.116]. There are three different ways people have approached the symbol grounding problem, which is more general than spatial cognition, in robotics. Starting with *Winograd* [71.117], many have created symbol

systems that map between some language and the external world by manually connecting each term onto a prespecified action space and set of environmental features [71.110, 112, 113, 118–121]. This class of systems takes advantage of the structure of linguistic interaction, but the systems usually do not involve learning, have little perceptual feedback, and have a fixed action space. A second approach involves learning the meaning of words in the sensorimotor space (e.g., joint angles and images) of the robot [71.122–124]. By treating human interaction terms as sensory input, these systems must learn directly from complex features extracted by perceptual systems, resulting in a limited set of commands that can be robustly understood. A third approach is to use learning to convert from an interaction onto aspects of the environment. These approaches may only use linguistic features [71.125, 126], spatial features [71.107] or linguistic, spatial, and semantic features [71.114, 115, 127–129]. These approaches learn the meaning of spatial prepositions (e.g., *above* [71.107]), verbs of manipulation (e.g., *push* and *shove* [71.130]), and verbs of motion (e.g., *follow* and *meet* [71.131]) and landmarks (e.g., *the doors* [71.129]).

Recent progress in probabilistic relational models, such as the generalized grounding graph (G^3), has addressed these issues by exploiting the structure of spatial discourse, breaking down a natural language command into component clauses and connecting each word to a physical interpretation [71.131, 132]. The grounding graph takes full advantage of the hierarchical and compositional structure of natural language commands and is able to ground landmarks, such as *the computers*, by exploiting object co-occurrence statistics between unknown noun phrases and known perceptual features, spatial relations, such as *past* in the path of an agent relative to an object, and motion verbs, such as *follow*, *meet*, *avoid*, and *go* in the path of a single agent or multiple agents. Once trained, the G3 model can ground spatial discourse in a semantic map of the environment; the map can be given a priori or created on the fly as the robot explores the environment. The G3 model is dynamically instantiated as a hierarchical probabilistic graphical model that connects each element in a natural language command to an object, place, path, or event in the environment. Its structure is created according to the compositional and hierarchical structure of the command, learning the mapping from language onto a continuous robot plan. The G3 model is trained on a corpus of natural language commands paired with groundings, and learns meanings for words and phrases in the corpus, including complex verbs, such as *put* and *take*.

71.3 Models of Human–Robot Interaction

Robotic technologies that interact with people – whether they afford closed-loop teleoperation or collaborate autonomously as peers – need to interpret, make decisions about, and respond to their environment, particularly the physical world, the task that they are expected to support, and the actions, goals, and intentions of the other agents – including people. To achieve these goals, robots need *models* that accurately represent the physical and cognitive characteristics of their environment. These models might outline such characteristics as narrowly as control–action relationships in the context of teleoperation or as comprehensively as human–robot joint activity in the context of peer-to-peer collaboration. Cognitive HRI considers the robotic system to be a part of a distributed cognitive system and therefore seeks primarily to develop cognitively inspired models [71.2]. These models might draw on knowledge about human cognition to improve the usability of robotic system, mimic human decision making or behavior mechanisms, or represent the complete human–robot cognitive system, offering cognitive representations for different paradigms of HRI (Fig. 71.11).

71.3.1 Dialog-Based Models

Research on human robot interaction across different interaction paradigms from teleoperation [71.133, 134]

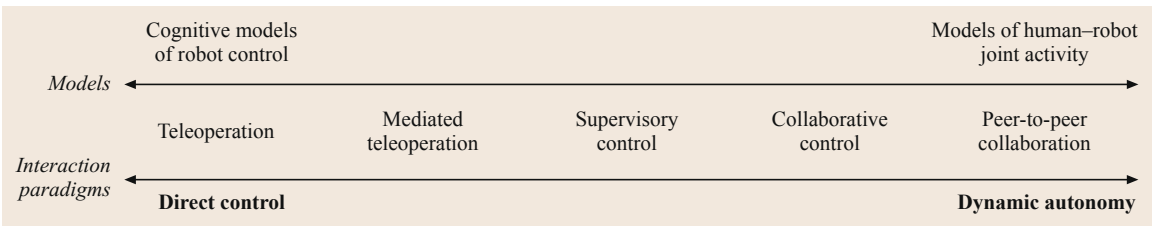


Fig. 71.11 Different paradigms of HRI (after [71.2])

to peer-to-peer interaction [71.135] has highlighted the need for establishing *common ground* [71.136] for effective HRI. In the context of teleoperation, *Burke* et al. [71.133] found that a lack of appropriate shared representations among human team members and the robot resulted in discrepancies in understanding among team members and breakdowns in perceiving and interpreting data provided by the robot. *Stubbs* et al. [71.134] observed such lack of common ground between operators and the robot across varying levels of autonomy. In the context of peer-to-peer interaction, *Kiesler* [71.135] argues that participants in an encounter seek to minimize their collective effort to reach mutual understanding and that the effort needed to establish this understanding between a robot and its users might determine the outcomes and success of HRI. These examples have motivated a large body of research in developing dialog-based models for establishing common ground in human–robot joint activity.

An example of the application of a dialog-based model to a task domain that traditionally involved supervisory control is *Fong*'s et al. [71.136] *collaborative control* system. In this system, the human and the robot collaborated as partners to perform tasks such as navigation, collaborative exploration, and multirobot teleoperation and achieve shared goals within these tasks. The interaction between the robot and its human counterpart involved engaging in dialog to share information and control at key points in the task. For instance, when the robot encountered an obstacle, it asked the user, *Can I drive through <image>?* along with an image of the obstacle. In asking these questions, the robot drew on specific attributes of the user, such as response accuracy, expertise, availability, efficiency, and preferences to determine whether or not it should direct specific questions to its user.

A number of proposed models and systems take the dialog-based interaction paradigm further to involve the robot and its human counterpart jointly addressing the domain task and dialog itself as *joint action* [71.137, 138]. In this peer-to-peer setup, either party selects goals to address and strategies to be used to address them and either party performs any part of the task. The model proposed by *Foster* et al. [71.137] includes a semantic interpretation module and a central decision-making module which draw on resources, such as a history of the ongoing discourse between the robot and its user, a world model, a domain planner, and a representation of the plan that is currently being executed, in order to generate action and communication behaviors.

The model proposed by *Li* et al. [71.138] draws on joint intention theory [71.139], considering the joint activity to involve a common persistent goal of achieving conversational grounding, and explicitly uses elements of grounding in representing conversational contributions. These contributions involve a *presentation* and an *acceptance* phase. For example, when an agent asks a question and the other agent answers, the question becomes the presentation and the answer becomes the acceptance, forming a *grounded exchange*. The model considers exchanges that involve a presentation without an acceptance to be *ungrounded*. Discourse contributions take place at two layers: *intention* and *conversation*. At the intention layer, the system plans communication intentions based on analyses of previous discourse and the robot's control system. These intentions can be *self-* or *other-motivated* for each agent. The conversation layer involves the articulation of communication intentions through verbal and nonverbal behaviors. The two layers form an *interaction unit* (IU) in the model. The model determines whether an IU is presentation or acceptance and whether it is grounded or ungrounded by assessing whether it satisfies joint intentions of the agents. Figure 71.12 illustrates how an other-motivated exchange is assessed by the model to determine whether the exchange is a presentation or an acceptance.

Models of Situated Human–Robot Dialog

The models and systems described above consider task-based and communicative exchanges in HRI as a dialog and extend models of spoken dialog to accommodate requirements that are specific to HRI, such as task-management, mixed-initiative dialog management, and physically situated referencing. Research in cognitive HRI has also explored the development of dialog systems that explicitly integrate these mechanisms into dialog modeling and the development of specific models and mechanisms for these requirements.

An example of dialog systems that are specifically developed for situated human–robot dialog is the pattern-based mixed-initiative (PaMini) HRI framework [71.140]. This framework extends spoken dialog systems with two key components: a task-state protocol and interaction patterns. The task-state protocol component explicitly defines *tasks* that either the robot's perceptual or control subsystems can perform. A task is defined as an execution state and preconditions for execution. The task-state protocol specifies task states and transitions among them to support coordination. The interaction patterns component provides high-level representations of recurring dialog structures such as a *clarification*. A comparison of most commonly used spoken dialog systems and the PaMini framework in the context of a human–robot situated learning scenario is provided by *Peltason* and *Wrede* [71.141].

Another example is the Robot Behavior Toolkit developed by *Huang* and *Mutlu* [71.3], which supports

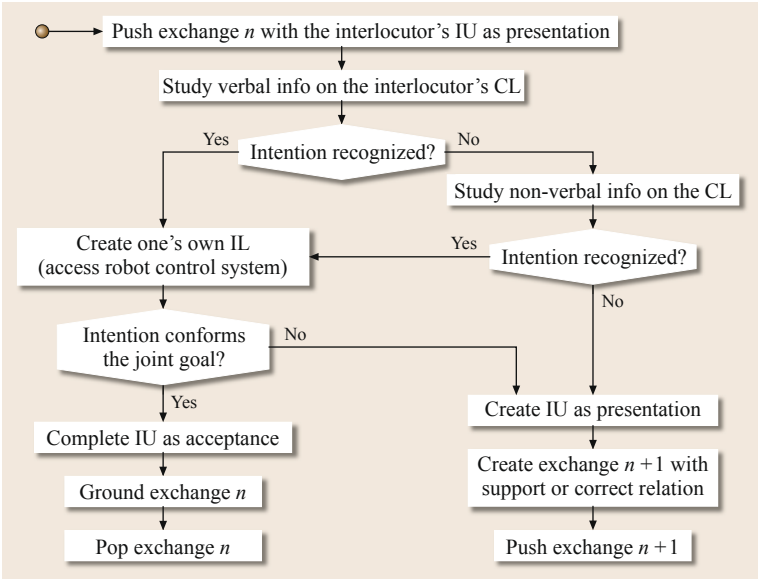


Fig. 71.12 The model evaluates an exchange provided by the interaction partner to determine its *presentation* or *acceptance* status and determine an appropriate action (after [71.138])

situated human–robot dialog by integrating nonverbal cues for task-based referential communication and conversation into the robot's speech. This system uses a *repository* of specifications of situated communication cues based on models of human interactions and an *activity* model (described in more detail below) that specifies the joint human–robot activity including the agents, task context, shared task goals, and expected task outcomes to integrate the situated communication cues that are expected to support these outcomes into the robot's speech. Figure 71.13 displays an example behavior generated by the Toolkit in a collaborative manipulation task. An evaluation of their system showed that interactions in which the robot displayed these situated communication cues as directed by the system more effectively supported desired task outcomes compared with baseline interactions (VIDEO 128).

Research in cognitive HRI has also explored the development of models for specific communication and coordination mechanisms in situated interaction, such as perspective-taking, spatial referencing, reference resolution, and joint attention (VIDEO 129).

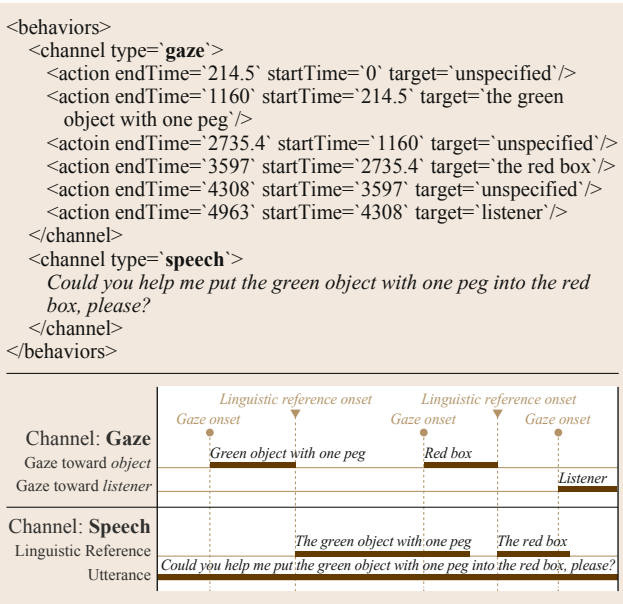


Fig. 71.13 The robot behavior toolkit uses specifications from a repository and a model of the joint activity to integrate effective multimodal task-related dialog behaviors (after [71.3])

Perspective-Taking. A core process in situated interaction toward establishing common ground is *perspective-taking* [71.142]. Research in social cognition has shown that the ability to take another's perspective and share common ground significantly improves collaborative performance in human teams [71.143]. Research in HRI has also explored how robots might employ this core mechanism to establish common ground with their users in situated interactions and has proposed several models that supported perspective taking.

Trafton et al. [71.144] studied interactions among astronauts in a naturalistic collaborative assembly task and found that a quarter of the utterances in the data involved taking the perspective of another and that participants frequently switched among egocentric, exocentric, addressee-centered, and object-centered perspectives. Based on their results, they developed a cog-

nitive model of perspective taking that allowed the robot to maintain multiple perspectives – or *alternative worlds* – at once and explore propositions about these worlds, such as the perspective of an interaction partner. This exploration allowed the robot to make inferences about the perspective of its partner by simulating this alternative world and act on the world from this perspective. The following sequences of actions illustrate the simulations that the robot might carry out based on the command *go to the cone* (adapted from *Trafton* et al. [71.2]). Underlined text describes components of the system implementation:

Simulate current real world (i. e., perceive it)
Perception specialist notices the existence and location of *person*, $cone_1$, $cone_2$, and *obstacle*
Language specialist hears *Coyote, go to the cone* and infers that there is an object, C, that is a cone and that the *person* wants it to go to
Identity hypothesis specialist infers that C can be identical to $cone_1$ or $cone_2$
$C = cone_1, C = cone_2$
Identity constraint specialist notices a contradiction
This contradiction triggers the counterfactual simulation strategy

Simulate the world where $C = cone_1$
Because in this world *person* has referred to $cone_1$, the perspective-simulation strategy is triggered
Simulate the world where $C = cone_1$ **and** *robot* = *person*
The spatial reasoning perspective indicates that $cone_1$ does not exist in this world because *person* cannot see it
Thus, $C \neq cone_1$

Simulate the world where $C = cone_2$
Because in this world *person* has referred to $cone_2$, the perspective-simulation strategy is triggered
Simulate the world where $C = cone_2$ **and** *Robot* = *Person*
Because $cone_2$ is visible in this world, there is no contradiction in this world
Infer that $C = cone_2$ (i. e., *the cone* refers to $cone_2$)

Following a *counterfactual simulation strategy* provides the robot with the ability to make inferences about situated actions across alternative scenarios with alternative physical (e.g., whether or not an object is present) and cognitive (e.g., whether or not the object is visible to the human counterpart) characteristics and determine appropriate next actions, such as carrying out a request or seeking clarification from its human counterpart.

Figure 71.14 illustrates four alternative scenarios with different physical and cognitive properties explored by *Trafton* et al. [71.144]. In each scenario, the robot assesses these properties to determine its next actions, as illustrated below.

Algorithm 71.1

function: Scenario($nCones = 1 \Rightarrow cone_a$)
if $cone_a = visible_{\text{robot}} \wedge cone_a = visible_{\text{human}}$ **then**
  Go to $cone_a$
end if
function: Scenario($nCones = 2 \Rightarrow cone_a, cone_b$)
if $cone_a, cone_b = visible_{\text{robot}} \wedge cone_a = visible_{\text{human}}$
then
  Go to $cone_a$
end if
function: (Scenario($nCones = 1 \Rightarrow cone_a$)
if $cone_a, cone_b \neq visible_{\text{robot}} \wedge cone_a = visible_{\text{human}}$
then
  Check hidden location
end if
function: Scenario($nCones = 2 \Rightarrow cone_a, cone_b$)
if $cone_a, cone_b = visible_{\text{robot}} \wedge cone_a,$
  $cone_b = visible_{\text{human}}$
then
  Request clarification
end if

Berlin et al. [71.145] developed a similar model that enabled the robot to understand its environment from the perspective of an interaction partner by maintaining

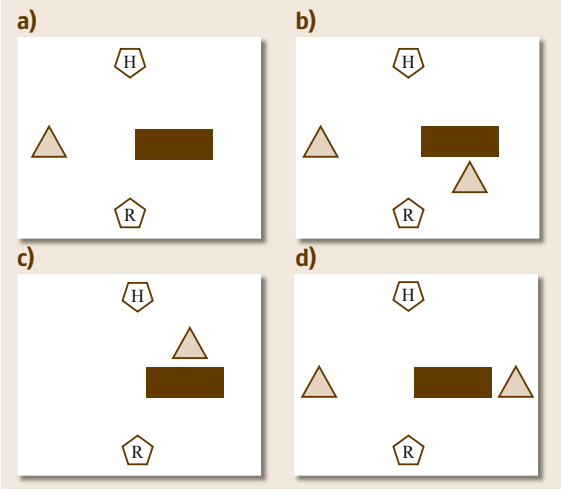


Fig.71.14a–d The alternative scenarios considered by the system in which the robot and its human counterpart are in a room with several objects and possible occlusions from the perspectives of the robot or the human (after [71.144])

separate and potentially different sets of beliefs in its belief system for itself and for its interaction partner. To construct a model of the beliefs of its interaction partner, the robot employed the same mechanisms it used to model its own beliefs but transformed the data it perceived from the world to match the reference frame of its interaction partner. These two sets of beliefs were maintained separately so that the robot can compare differences between its beliefs and its interaction partner's beliefs and plan actions in order to establish common ground or identify discrepancies in its learning in the context of task learning. Figure 71.15 illustrates parallel beliefs maintained by the robot in a button-pressing task.

Spatial Referencing. *Moratz* et al. [71.146] proposed a cognitive model of spatial reference that represented different kinds of spatial reference systems and allowed the robot to interpret instructions from an interaction partner. This model mapped the locations of all objects as projections on a plan view, considering the robot's point of view as *origin* and the location of the object that will be used as *relatum* to determine the reference axis. This axis enabled the robot to interpret directions such as *left of*, *right of*, *in front of*, and *to the back* in relation to the relatum, providing the robot the ability to interpret natural language references to objects in the environment.

Reference Resolution. *Ros* et al. [71.147] extended these approaches to develop a model that enabled the robot to clarify references made by its interaction partner. This model employed several mechanisms including visual perspective taking, spatial perspective taking, symbolic location descriptors, and feature descriptors to determine whether it needed any clarification on its interaction partner's references. The visual perspective taking mechanism allowed the robot to determine whether or not objects in the environment were in its interaction partner's focus of attention (FOA), in its partner's field of view (FOV), or out of its partner's field of view (OOF). The spatial perspective taking mechanism maintained egocentric and addressee-centered perspectives to determine ambiguities in object references. The system also included symbolic location descriptions such as *is in*, *is on*, and *is next to* to determine spatial relationships between objects and the environment. Finally, the robot used feature descriptors such as color and shape to identify ambiguities in the references of its interaction partner. Once the robot determined the need clarification in its partner's references, it used an ontology-based clarification algorithm to ask questions to its partner about the object of reference.

Joint Attention. Another key mechanism in situated interaction is *joint attention* – the ability to use nonverbal cues, such as gaze and pointing, to establish common ground on what referents in the environment are under consideration in the dialogue [71.149]. *Scassellati* [71.99] proposed a task-based decomposition of joint attention skills, including mutual gaze, gaze following, imperative pointing, and declarative pointing, and implemented these skills in a robot as stages for establishing joint attention with a human counterpart. The mutual gaze skill provided the robot with the ability to recognize and maintain eye contact with its interaction partner. At the gaze following stage, the robot followed the eyes of its partner to direct its attention to the object of its partner's attention. Imperative pointing involved pointing at an object that is out of reach in order to request the object. Finally, the declarative pointing stage involved extending an arm and index finger to draw attention to an object that is out of reach without necessarily requesting the object.

Connection Events. *Rich* et al. [71.150] argued that mechanisms such as joint attention serve as *connection events* in situated dialog and establish and maintain engagement among interaction partners. From data on human interactions, they identified a set of key connec-

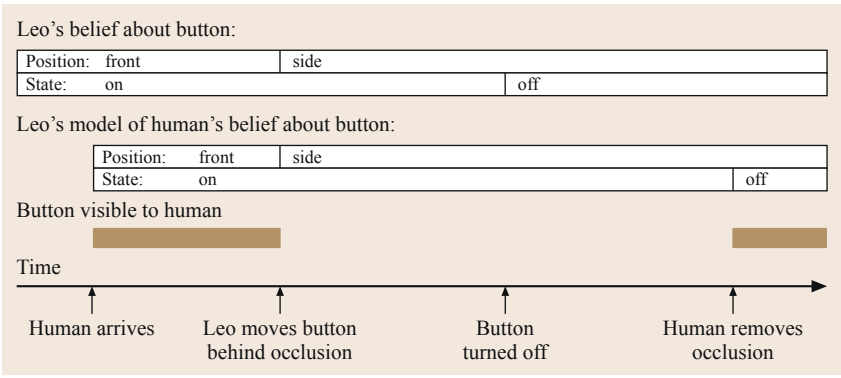


Fig. 71.15 In the system proposed by *Berlin* et al. [71.145], the robot maintains a parallel beliefs for itself and for its human counterpart for the task and updates its beliefs based on sensory input and those of the user based on the user's awareness (after [71.145])

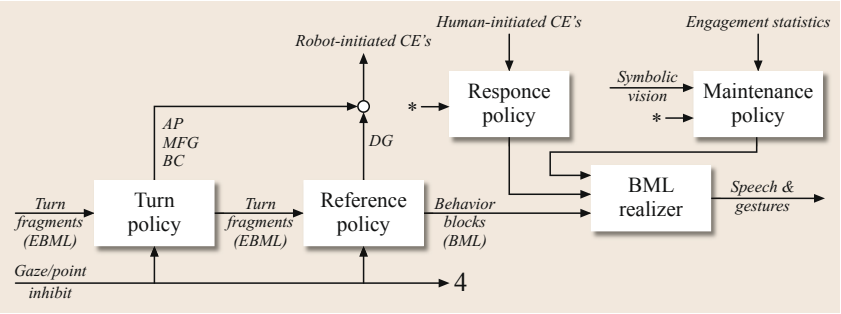


Fig. 71.16 The policy components, the BML realizer, and the generated connection events for the robot in the *engagement generator* module (after [71.148])

tion events, including mutual gaze, directed gaze, adjacency pairs, and backchannels, and developed a system that recognized these events in human counterparts and generated them for a robot (see *Rich* et al. [71.150] for details on the recognizer and *Holroyd* et al. [71.148] for details on generation). The recognizer module included dedicated recognizers for each type of connection event and an estimator for engagement levels for the robot's human counterpart, while the generation module included four policy components and a behavior mark-up language (BML) realizer for generating robot behaviors toward establishing and maintaining engagement. The components of this engagement generator are illustrated in Fig. 71.16.

71.3.2 Simulation-Theoretic Models

Research in cognitive HRI has also been inspired by neurocognitive mechanisms in developing models of human–robot joint activity, building particularly on *simulation theory*, which suggests that people (and primates) represent other people's mental states by adopting their perspective, specifically by tracking or matching their states with resonant states of their own [71.151]. This simulation-theoretic approach led to several models of robot behavior and human–robot joint action that involve the robot imitating or simulating the behaviors of its interaction partner in order to learn from or make inferences about its partner's goals.

As an example of this approach, *Bicho* et al.[71.152] proposed a model for action preparation and decision-making in cooperative human–robot tasks that is inspired by the finding that action observation elicits an automatic activation of motor representations associated with the execution of the observed action. This *motor-resonance mechanism* allows people to internally simulate action consequences using their own motor repertoire and predict the consequences of action of others. In the proposed model, a perception–action linkage enables efficient coordination of actions and decisions between the agents in a human–robot joint action task. The model integrates a mapping between observed actions and complementary actions in memory, while taking into account the inferred goals of the actions of the interaction partner, contextual cues, and shared task knowledge.

Building on simulation theory, *Gray* et al. [71.153] proposed a similar system in which the robot parses user actions and matches the user's movements to movements in its own repertoire toward making inferences about the user's goals and perform a task-level simulation (Fig. 71.17). This simulation allows the robot to determine the preconditions of the schemas that represent the task and track its human partner's progress over the course of the task in order to anticipate its partner's needs and offer relevant help accordingly. The simulation also provided the robot with the ability to make inferences on the beliefs of its partner and simulate its partner's perspective in a fashion similar to the perspective-taking mechanisms proposed by *Trafton* et al. [71.144] and *Berlin* et al. [71.145] (VIDEO 130).

Aspects of the simulation-theoretic approach explicitly taken in these examples can also be seen in other control architectures developed for HRI. *Nicolescu* and *Mataric* [71.154] proposed a control architecture that unifies perception and action to achieve *action-based*

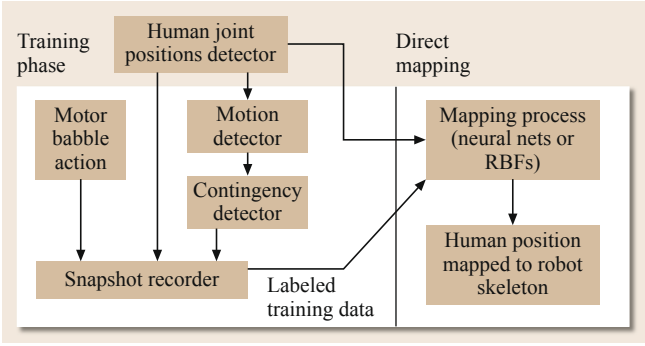


Fig. 71.17 The mapping of perceived human actions onto the robot's body in order to make comparisons and task-level inferences (after [71.153])

interaction. In this architecture, behaviors are built from *perceptual* and *active* components. Perceptual components allow the robot to link its observations and actions and thus to learn to perform a task from the experiences it gains from its interactions with people. Active components enable task-based behaviors that also serve as implicit communication rather than explicit behaviors such as speech and gestures. Behavior representation in the architecture captures two types of behaviors: *abstract* and *primitive*. Abstract behaviors are explicit specifications of the behaviors' activation conditions (preconditions), goals in the form of abstracted environmental states, and effects (postconditions), while primitive behaviors are those that the robot performs to achieve these effects. By linking perceptions and actions, the robot learns what actions of its own might achieve the same observed effects.

71.3.3 Intention- and Activity-Based Models

The models and systems described above are concerned primarily with establishing and maintaining common ground and coordinating actions in task-based interactions using dialog- and simulation-theoretic approaches with limited consideration of the broader context of these interactions as complex activities involving multiple agents with common goals and commitments to these goals. A number of models and systems sought to address this limitation, building on models and theories of human joint activity such as joint intention theory [71.139] and activity theory [71.7].

Building on joint intention theory, *Breazeal* et al. [71.155] proposed a model of human–robot collaboration that involved dynamically meshing subplans into joint activity toward achieving common goals of the human–robot team. In this model, task and goal representations have a goal-centric view, employing an action-tuple data structure that captures preconditions, executables, until-conditions, and goals. Tasks are represented in a hierarchical structure of actions and recursively defined subtasks. Goals are also represented hierarchically as overall intent rather than a chain of low-level goals. The implemented joint intention model dynamically assigns tasks to members of the human–robot team. These intentions are derived based on the robot's actions and abilities, the actions of the human partner, the robot's understanding of the common goal of the team, and its assessment of the current task state. At every stage of the interaction, the robot negotiates who should complete the task. Action at these points might look like turn-taking or simultaneous action (the robot and the human working on different parts of the task).

Alami et al. [71.156, 157] similarly built on joint intention theory to propose a human–robot decision framework in which team members are committed to a *joint persistent goal* and follow cooperation schemes to contribute toward achieving this goal. The framework involves a goal planner called *the agenda* for the robot and human collaborators to pursue, a proxy representation of the human in the robot called *Interaction Agents (IAA)*, *task delegates* that monitor and control the task commitment of the human or the robot for each active, inactive, or suspended goal, and a *robot supervision kernel* that monitors and controls robot activities. For each new active goal, the Robot Supervision Kernel creates a Task Delegate, selects or elaborates a plan, and allocates the roles of each team member.

Fong et al. [71.1] proposed a similar system called the HRI operating system (HRI/OS) to support human–robot teamwork. The system involves a task manager, resource manager, interaction manager, spatial reasoning agent, context manager, human and robot agents, and an open agent architecture (OAA) facilitator. The task manager decomposes the overall goal of the system into high-level tasks and assigns to humans or robots for execution. The manager relies on the agents to complete the low-level steps of the tasks. It communicates with the Resource Manager to find an agent capable of performing the work. Resource manager processes all agent requests, prioritizing the list of agents to be consulted when a task needs to be performed. Interaction manager coordinates dialog-based communication between agents. Context manager keeps track of everything that occurs while the system is running including task status and execution, agent activities, agent dialogue, etc. Spatial reasoning agent (SRA) is used to resolve spatial ambiguities in human–robot dialog through mechanisms such as perspective taking and frames of reference, resolving ambiguities among as ego-, addressee-, object-, and exo-centric references. To do this, SRA transforms the spatial dialog into a geometric reference and perform a *mental simulation* of the interaction to explore how ambiguities might be resolved through multiple references. Finally, the OS includes a software representation of the human – a human proxy agent that represents user capabilities and accepts task assignments in the way that robot agents do. These proxies represent task capabilities, including domains of expertise, and provide health monitoring feedback.

Huang and *Mutlu* [71.3] built on an alternative model of human activity – activity theory [71.7] – to develop a model of human–robot joint activity. Their model builds on five key constructs from activity theory including consciousness, object-orientedness, hierarchical structure, internationalization and external-

ization, and mediation. The *consciousness* construct pertains to attention, intention, memory, reasoning, and speech and includes specific representations for attention and intention. The *object-orientedness* construct describes material artifacts, plans of action, or common ideas to be shared by the members of the joint activity. Following the *hierarchical structure* construct, the model organizes joint activity into three layers: activity, action, and operation. An activity consists of a series of actions that share the same goal, and each action has a defined goal and a chain of operations that are regular routines performed under a set of conditions. *Internalization and externalization* describes cognitive processes; internalization involves transforming external actions or perceptions into mental processes, while externalization is the process of manifesting mental processes in external actions. Finally, the *mediation* construct defines several external and internal tools, such as physical artifacts that might be used in an activity and cultural knowledge or social experience that an individual might have acquired, as mediators of human–robot joint activity. These constructs and their corresponding system elements allow the construction of and planning for joint human–robot *activities*. For each activity, a *motive* governs actions. Each action, by achieving its corresponding goal, helps to fulfill the motive of the activity. Each action may have several *operations* that are constrained by a set of conditions and that can be executed only when all the conditions are met. Actions have predefined *outcomes*, which specify the orientation of an action. Figure 71.18 shows the XML (extensible markup language) representation of a model of a collaborative manipulation task.

71.3.4 Models for Action Planning

The models described above primarily enable communication and coordination between humans and robots toward planning and carrying out joint tasks. In order to successfully contribute to these tasks, robots also need models for planning their actions in a dynamic physical and cognitive environment. Research in cognitive HRI seeks to develop models for *action planning* that help robots estimate the actions that they have to take in order to achieve task goals and learn the parameters of the tasks space. The paragraphs below review research in two common approaches to building such models: *decision-theoretic models* and *model learning*.

Decision-Theoretic Models

One of the simplest approaches to control and decision-making in HRI is to define the interaction as a decision-theoretic planning problem, such as a Markov decision process (MDP). Formally, an MDP consists of the n-tuple $\{S, A, T, R, \gamma\}$. The set S is a set of states, which in the HRI setting typically correspond to the combination of state variables, such as the robot state and the desired outcome of the interaction. For example, if the interaction model allows a human partner to instruct the robot to move to different locations in the environment, one state variable may correspond to different current locations of the robot and another state variable may correspond to the goal states intended by the human partner. The full state space S is given by the combination of possible values for the different state variables.

The action set A represents actions that the robot may take. The actions may include asking a question, performing some physical movement, or even doing nothing. Each action has a cost R depending on the current state, which rewards the robot for performing useful actions, and penalizes the robot for taking actions that either make no immediate progress toward the specified goal (typically a small penalty) or completely unhelpful (a large penalty).

Lastly, the transition function T provides a notion of the dynamics of the environment in terms of how the state changes as robot takes actions, and especially how a human partner's state variables may change as the robot takes actions. The transition function $T(s'|s, a)$ places a probability distribution over the states to which the user in state s may transit if the robot takes action a. The MDP formulation is very appealing, because there exist efficient techniques for solving for interaction policies. Once the policy is computed, the interaction can be managed simply by querying the policy for the

```
<Activity id=`1`>
  <Motive>clear(table)</Motive>
  <Description>Clear objects on table</Description>
  <Participants>Self, User1</Participants>
  <Action id=`1`>
    <Outcome>Task</Outcome>
    <Goal>disappear(object)</Goal>
    <Description>
      Instruct User1 to categorize object
    </Description>
    <Operation type=`utterance`>
    <Condition>present(User1)</Condition>
    <Condition>
      known(the blue object with two pegs)
      </Condition>
      <Condition> known(the blue box)</Condition
      <Info turn=`end`>
        Could you help me put the blue object with
        two pegs into the blue box, please?
      </Info>
    </Operation>
    ...
```

Fig. 71.18 The XML representation of the activity-theory-based model for a collaborative manipulation task (after [71.3])

appropriate action in response to the current state of the robot and the human partner.

A limitation of the MDP approach is that some of the state variables may not be directly observable, in particular the state variables corresponding to human intentional states, such as intended goal locations of the robot. The values of the state variables must be inferred from observations, such as speech acts performed by the human partner, which are inherently noisy. For example, the system may hear the words *coffee machine* when the user asks the robot to go to *copy machine*. While speech recognition errors may be mitigated to some extent by asking the user to use only acoustically distinct keywords when speaking to the system, a system that does not model the likelihood of recognition errors and act accordingly will be brittle; a robust system must be able to infer user intent under uncertainty.

The observations are rarely sufficient to uniquely determine the current state, but more commonly are used to compute a *belief*, or probability distribution over dialog states. If the agent takes some action a and hears observation o from an initial belief b, it can easily update its belief using Bayes rule

$$b^{a,o}(s) = \frac{\Omega(o|s',a)\sum_{s\in S} T(s'|s,a)b(s)}{\sum_{\sigma\in S}\Omega(o|\sigma,a)\sum_{s\in S} T(\sigma|s,a)b(s)} . \quad (71.1)$$

This probability distribution will evolve as the dialog manager asks clarification questions and receives responses. In Fig. 71.19, we show a cartoon of a simple dialog model. Initially, we model the user as being in a *start* state. Then, at some point in time, the user speaks to the robot to indicate that he or she wants it to perform a task. We denote this step by the set of vertical stack of nodes in the center of the model. Each node represents a different task. The dialog manager must now interact with the user to determine what is wanted. Once the task is successfully completed, the user transitions to the right-most *end* node, in which he or she again does not desire anything from the robot. We note that it can be easily augmented to handle more complex scenarios. For example, by including the time of day as part of the state, we can model the fact that the user may usually wish to go to certain locations in the morning and other locations in the afternoon.

Intuitively, we can see how the belief can be used to select an appropriate action. For example, if the dialog manager believes that the user may wish to go to either the coffee machine or the copy machine (but not the printer), then it may ask the user for clarification before commanding the wheelchair to one of the locations. More formally, we call the mapping from beliefs to actions a policy. We represent this mapping using the concept of a value function $V(b)$. The value of a belief is defined to be the expected long-term reward the dialog manager will receive if it starts a user interaction in belief b. The optimal value function is piecewise-linear and convex, so we represent V with the vectors V_i; $V(b) = \max_i V_i \cdot b$. The optimal value function satisfies the Bellman equation [71.158]

$$V(b) = \max_{a\in A} Q(b,a) ,$$
$$Q(b,a) = R(b,a) + \gamma \sum_{o\in O} \Omega(o|b,a)V(b_a^o) , \quad (71.2)$$

where $Q(b,a)$ represents the expected reward for starting in belief b, performing action a, and then acting optimally. The belief b_a^o is b after a Bayesian update of b using (71.1), and $\Omega(o|b,a)$, the probability of seeing o after performing a in belief b ($\sum_{s\in S}\Omega(o|s,a)b(s)$).

There are also non-Bayesian approaches for acting in uncertain environments. Many interaction systems provide the dialog manager with a set of rules to follow given particular outputs from a speech recognition system. The drawback to rule-based systems is that they often have difficulty managing the many uncertainties that stem from noisy speech recognition or linguistic ambiguities. The ability to manage the trade-off between gathering additional information and servicing a user's request have made partially observable Markov decision process (POMDP) planners particularly useful in dialog management; applications include a Nursebot robot, designed to interact with the elderly in nursing homes [71.159], a vision-based system that aids Alzheimer's patients with basic tasks such as hand-washing [71.160], an automated tele-

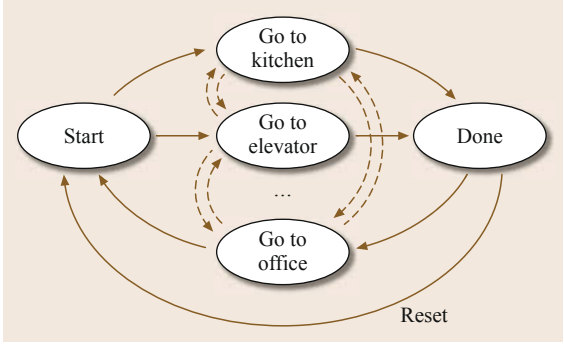


Fig. 71.19 A toy example of a dialog POMDP. The *nodes* in the graph are different states of the dialog (i. e., user requests). *Solid lines* indicate likely transitions; we assume that the user is unlikely to change their request before their original request is fulfilled. The system automatically resets once we reach the end state

phone operator [71.161], and a tourist information kiosk [71.162].

Beyond the initial formulations of cognitive HRI as a decision-theoretic problem, there have been a number of algorithmic improvements that increase the domains of applicability of this approach. For example, the conventional MDP and POMDP algorithms have typically assumed that each observation and action takes approximately the same amount of time, which can lead to an implicit bias toward longer actions. Representing time explicitly leads to computational intractability, but *Broz* et al. [71.163] demonstrated that the similar states that vary only by the time-index can be aggregated, leading to reduced-order models that can be solved very efficiently. Similarly, *Doshi* and *Roy* [71.164] showed that symmetries in human intentional states could be exploited to dramatically reduce the size of the planning problem, also leading to very efficient solutions. Most recently, again in the non-Bayesian line, *Wilcox* et al. have shown that the temporal dynamics of task-based HRI can be formulated as a scheduling problem [71.165].

Model Learning

The behavior of the dialog manager derived from solving (71.2) depends critically on accurate choices of the transition probabilities, observation probabilities, and the reward. For example, the observation parameters affect how the system associates particular keywords with particular requests. Similarly, the reward function affects how aggressive the dialog manager will be in assuming that it understands a user's request, given limited and noisy information. An incorrect specification of the dialog model may lead to behavior that is either overly optimistic or conservative, depending on how accurately the model captures the user's expectations on the interaction.

A common approach in other domains is to collect data using a fixed policy, typically referred to as system identification. In HRI, this is easiest to perform using so-called *Wizard of Oz* studies where a human experimenter executes the policy unseen to generate data or evaluate a policy. *Prommer* et al. [71.166] showed that Wizard-of-Oz studies could be used effectively not only to learn model parameters for an MDP dialog model, but also to learn an effective policy.

At the same time, learning all the parameters required to specify a rich dialog model can require a prohibitively large amount of data. While the model parameters may be difficult to specify exactly, either by hand or from data, we can often provide the dialog manager with an initial estimate of the model parameters that will generate a reasonable policy that can be executed while the model is improved. For example, even though we may not be able to attach an exact numerical value to driving a wheelchair user to the wrong location, we can at least specify that this behavior is undesirable. Similarly, we can specify that the exact numerical value is initially uncertain. As data about model parameters accumulate, the parameter estimates should converge to the correct underlying model with a corresponding reduction in uncertainty.

Figure 71.20a depicts the conventional model, where the arrows in the graph show which parts of the model affect each other from time t to $t+1$. Although the variables below the *hidden* line in Fig. 71.20a are not directly observed by the dialog manager, the parameters defining the model (i. e., the parameters in the function giving the next state) are fixed and known

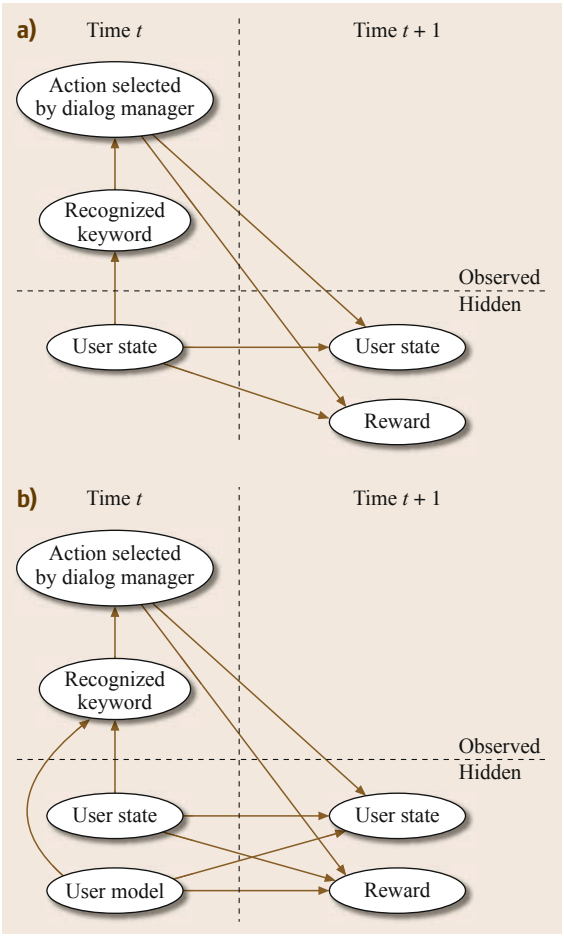


Fig. 71.20 (a) The standard POMDP model. (b) The extended POMDP model. In both cases, the *arrows* show which parts of the model are affected by each other from time t to $t+1$. Not drawn are the dependencies from time $t+1$ onward, such as the user state and user model's effect on the recognized keyword at time $t+1$

a priori. For instance, the reward at time t is a function of the state at the previous time and the action chosen by the dialog manager.

If the model parameters are not known a priori because the model is uncertain – for example, how much reward is received by the agent given the previous state and the action selected – then the concept of the belief can be extended to also include the agent's uncertainty over possible models. In this new representation, which we call the *model-uncertainty POMDP*, both the user's request and the model parameters are hidden. Figure 71.20b shows this extended model, in which the reward at time t is still a function of the state at the previous time and the action chosen by the dialog manager, but the parameters are not known a priori and are therefore hidden model variables that must be estimated along with the user state. The system designer can encode their knowledge of the system in the dialog manager's initial belief over what dialog models it believes are likely – a Bayesian prior over models – and let the agent improve upon this belief with experience.

Poupart et al. treated the unknown MDP parameters as hidden state in a larger POMDP and derived an analytic solution (based on [71.167]) for a policy that will trade optimally between learning the MDP and maximizing reward. Unfortunately, these techniques did not extend tractably to the model-uncertainty POMDP, which is continuous in both the POMDP parameters (like the MDP) and the belief state (unlike the MDP). *Doshi* and *Roy* [71.168, 169] provided an approximate, Bayes risk action selection criterion that allows the dialog manager to function in this complex space of dialog models. This approach was applied to the intelligent wheelchair assistant shown in Fig. 71.21. Their goal was to design an adaptable HRI system, or dialog manager, that allows both the user of the wheelchair and a caregiver to give natural instructions to the wheelchair, as well as ask the wheelchair computer for general information that may be relevant to the user's daily life.

In contrast to the Bayesian approach, *Cakmak* and *Thomaz* [71.170] pursued an active learning approach and identified three types of queries that a robot could generate while learning a new task (VIDEO 237). While this result does not provide a comparison to an approach embedded in an ongoing dialogue, their results do provide guidelines for model designers.

71.3.5 Cognitive Models of Robot Control

A final line of research in cognitive HRI seeks to achieve greater task efficiency in human–robot teams, thus addressing common problems between operators and robots such as those identified by *Burke* et al. [71.133] and *Stubbs* et al. [71.134], by developing models and control interfaces that exploit mechanisms of human cognition such as working memory and mental models [71.171, 172]. This research includes formalisms such as *neglect time*, the amount of time that an operator can neglect a robot before the robot's performance drops below a certain threshold [71.171], and *fan out*, a measure of how many robots an operator can effectively manage in a human–robot team [71.172]. Such formalisms inform the development of guidelines for designing effective control mechanisms such as the following principles proposed by *Goodrich* and *Olsen* [71.171]:

1. *Implicitly switch interfaces and autonomy modes.* Context determines the mode of use. For instance, the user starts using a joystick and the interaction modality automatically switches, rather than the user explicitly selecting a modality.
2. *Let the robot use natural human cues.* The robot uses the cues to provide feedback and present information that the human uses to provide the commands or present information to the robot.
3. *Manipulate the world instead of the robot.* Control interfaces integrate knowledge about the task and the world to minimize low-level control of the robot and maintaining of a mental model of the robot's functioning.
4. *Manipulate the relationship between the robot and world.* Control interfaces provide real-world

Fig. 71.21 Our dialog manager allows for more natural human communication with a robotic wheelchair (after [71.168, 169])

representations for control to minimize low-level control.
5. *Let people manipulate presented information.* Interfaces present information in a way that represents the real world and allows users to provide input directly into the representation rather than translating information readings to a different modality or representation.
6. *Externalize memory.* Different types of information are integrated into a single representation to reduce the working memory load for the user.
7. *Help people manage attention.* The robot provides appropriate indicators to capture the attention of the operator.
8. *Learn.* Control mechanisms adapt system activity to the user's mental models.

71.4 Conclusion and Further Reading

This chapter presented an overview of research in cognitive human–robot interaction, the area of research concerned with modeling human, robot, or joint human–robot cognitive processes in the context of HRI. This research seeks to gain a better understanding of people's interaction with robots and build robotic systems with the necessary cognitive mechanisms to communicate and collaborate with their human counterparts. Three key themes fall within this research area. The first theme seeks to build a better understanding of human cognition in HRI; specifically, people's mental models of robots as ontological entities, social cognition of robot behaviors, and the use of robots as experimental platforms to study cognitive development in humans. The second theme includes research that seeks to build models for simulating human cognition in robots, gaining cognitive capabilities through imitation and interaction with the physical environment, and mapping aspects of interaction, such as commands from or references by human counterparts to objects in the environment. The final theme seeks to build models that support human–robot joint activity, including dialog-, simulation-theoretic-, joint-intention-, activity- and action-planning-based models that enable robots to reason about the physical and cognitive properties of the environment and the actions of their human counterparts and to plan actions toward achieving communicative or collaborative goals. The common thread among these three themes of research is the consideration of humans and robots as part of a cognitive system in which cognitive processes – natural or designed – shape how humans and robots communicate and collaborate.

As an interdisciplinary area of research, cognitive human-robot interaction receives contributions from a diverse set of research fields including robotics, cognitive science, social psychology, communication studies, and science and technology studies. Further reading on the topic is also available in a diverse set of venues such as:

- *The Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI)*
- *The Proceedings of the Annual Meeting of the Cognitive Science Society (CogSci)*
- *International Conference on Epigenetic Robotics (EpiRob)*
- *The Proceedings of the AAAI Conference on Artificial Intelligence*
- *The Proceedings of the IEEE International Symposium on Robots and Human Interactive Communication (RO-MAN)*
- *The Proceedings of the Robotics: Science and Systems (RSS) Conference*
- *Sun*, [71.173]
- *Journal of Human–Robot Interaction*
- *Interaction Studies: Social Behaviour and Communication in Biological and Artificial Systems.* John Benjamins
- *International Journal of Social Robotics*. Sage.

Video-References

VIDEO 128 Gaze and gesture cues for robots
available from http://handbookofrobotics.org/view-chapter/71/videodetails/128

VIDEO 129 Robotic secrets revealed, Episode 1
available from http://handbookofrobotics.org/view-chapter/71/videodetails/129

VIDEO 130 Robotic secrets revealed, Episode 2: The trouble begins
available from http://handbookofrobotics.org/view-chapter/71/videodetails/130

VIDEO 236 Human-robot jazz improvization
available from http://handbookofrobotics.org/view-chapter/71/videodetails/236

VIDEO 237 Designing robot learners that ask good questions
available from http://handbookofrobotics.org/view-chapter/71/videodetails/237
VIDEO 238 Active keyframe-based learning from demonstration
available from http://handbookofrobotics.org/view-chapter/71/videodetails/238

References

71.1 T. Fong, C. Kunz, L.M. Hiatt, M. Bugajska: The human-robot interaction operating system, Proc. 1st ACM SIGCHI/SIGART Conf. HRI, Salt Lake City (2006) pp. 41–48

71.2 J.G. Trafton, A.C. Schultz, N.L. Cassimatis, L.M. Hiatt, D. Perzanowski, D.P. Brock, M.D. Bugajska, W. Adams: Communicating and collaborating with robotic agents. In: *Cognition and Multi-Agent Interaction*, ed. by R. Sun (Cambridge Univ. Press, New York 2006) pp. 252–278

71.3 C.-M. Huang, B. Mutlu: Robot behavior toolkit: Generating effective social behaviors for robots, Proc. 7th ACM/IEEE Intl. Conf. HRI, Boston (2012) pp. 25–32

71.4 A.H. Vera, H.A. Simon: Situated action: A symbolic interpretation, Cogn. Sci. **17**(1), 7–48 (1993)

71.5 T. Winograd, F. Flores: *Understanding Computers and Cognition: A New Foundation for Design* (Ablex Publ., New York 1986)

71.6 L.A. Suchman: *Plans and Situated Actions: The Problem of Human-Machine Communication* (Cambridge Univ. Press, Cambridge 1987)

71.7 A.N. Leont'ev: The problem of activity in psychology, J. Russ. East Eur. Psychol. **13**(2), 4–33 (1974)

71.8 E. Hutchins: The social organization of distributed cognition. In: *Perspectives on Socially Shared Cognition*, ed. by L.B. Resnick, J.M. Levine, S.D. Teasley (American Psychological Association, Wachington, DC 1991)

71.9 B. Gates: A robot in every home, Sci. Am. **296**, 58–65 (2007)

71.10 C. Pantofaru, L. Takayama, T. Foote, B. Soto: Exploring the role of robots in home organization, Proc. 7th Annu. ACM/IEEE Intl. Conf. HRI, Boston (2012) pp. 327–334

71.11 K.M. Tsui, M. Desai, H.A. Yanco, C. Uhlik: Exploring use cases for telepresence robots, ACM/IEEE 6th Int. Conf. HRI, Lausanne (2011) pp. 11–18

71.12 F. Tanaka, A. Cicourel, J.R. Movellan: Socialization between toddlers and robots at an early childhood education center, Proc. Natl. Acad. Sci. USA **104**(46), 17954–17958 (2007)

71.13 T. Kanda, R. Sato, N. Saiwaki, H. Ishiguro: A two-month field trial in an elementary school for long-term human-robot interaction, IEEE Trans. Robotics **23**(5), 962–971 (2007)

71.14 B. Reeves, C. Nass: *The Media Equation: How People Treat Computers, Television and New Media Like Real People and Places* (Cambridge University Press, Cambridge 1996)

71.15 S. Turkle, O. Daste, C. Breazeal, B. Scassellati: Encounters with Kismet and Cog: Children respond to relational artifacts, Proc. IEEE-RAS/RSJ Int. Conf. Humanoid Robots, Los Angeles (2004) pp. 1–20

71.16 S. Turkle: *Alone Together: Why We Expect More from Technology and Less from Each Other* (Basic Books, New York, 2011)

71.17 C. Nass, Y. Moon: Machines and mindlessness: Social responses to computers, J. Soc. Issues **56**(1), 81–103 (2000)

71.18 S. Turkle: *Evocative Objects: Things We Think With* (MIT Press, Cambridge 2011)

71.19 K. Wada, T. Shibata, Y. Kawaguchi: Long-term robot therapy in a health service facility for the aged – A case study for 5 years, Proc. 11th IEEE Int. Conf. Rehabil. Robotics, Kyoto (2009) pp. 930–933

71.20 B.R. Duffy: Anthropomorphism and the social robot, Robotics Auton. Syst. **42**(3-4), 177–190 (2003)

71.21 S. Kiesler, A. Powers, S.R. Fussell, C. Torrey: Anthropomorphic interactions with a software agent and a robot, Soc. Cogn. **26**(2), 168–180 (2008)

71.22 S.R. Fussell, S. Kiesler, L.D. Setlock, V. Yew: How people anthropomorphize robots, Proc. 3rd ACM/IEEE Int. Conf. HRI, Amsterdam (2008) pp. 145–152

71.23 D.S. Syrdal, K. Dautenhahn, S.N. Woods, M.L. Walters, K.L. Koay: Looking good? Appearance preferences and robot personality inferences at zero acquaintance, AAAI Spring Symp.: Multidiscip. Collab. Socially Assist. Robotics, Stanford (2007) pp. 86–92

71.24 K.F. MacDorman, T. Minato, M. Shimada, S. Itakura, S. Cowley, H. Ishiguro: Assessing human likeness by eye contact in an android testbed, Proc. XXVII Annu. Meet. Conf. Cogn. Sci. Soc., Stresa (2005) pp. 1373–1378

71.25 F. Hegel, S. Krach, T. Kircher, B. Wrede, G. Sagerer: Understanding social robots: A user study on anthropomorphism, Proc. 17th IEEE Int. Symp. Robot Hum. Interact. Commun., Munich (2008) pp. 574–579

71.26 M. Mori: The uncanny valley, Energy **7**(4), 33–35 (1970)

71.27 C. Bartneck, T. Kanda, H. Ishiguro, N. Hagita: My robotic Doppelganger – A critical look at the Uncanny Valley Theory, IEEE 18th Intl. Symp. Robot Hum. Interact. Commun., Toyama (2009) pp. 269–276

71.28 M.L. Walters, D.S. Syrdal, K. Dautenhahn, R. te Boekhorst, K.L. Koay: Avoiding the uncanny valley: Robot appearance, personality and consistency of behavior in an attention-seeking home scenario for a robot companion, Auton. Robots **24**(2), 159–178 (2008)

71.29 A.P. Saygin, T. Chaminade, H. Ishiguro, J. Driver, C. Frith: The thing that should not be: Predictive coding and the uncanny valley in perceiving human and humanoid robot actions, Soc. Cogn. Affect. Neurosci. **7**(4), 413–422 (2012)

71.30 W. Mitchell, K.A. Szerszen Sr., A.S. Lu, P.W. Schermerhorn, M. Scheutz, K.F. MacDorman: A mismatch in the human realism of face and voice produces an uncanny valley, i-Perception **2**, 10–12 (2011)

71.31 C. Breazeal: *Designing Sociable Robots* (MIT Press, Cambridge 2002)

71.32 N. Matsumoto, H. Fujii, M. Okada: Minimal design for human-agent communication, Artif. Life Robotics **10**(1), 49–54 (2006)

71.33 H. Kozima, H. Yano: A Robot that Learns to Communicate with Human Caregivers, Proc. 1st Int. Workshop Epigenetic Robotics, Lund (2001) pp. 47–52

71.34 A. Powers, A.D.I. Kramer, S. Lim, J. Kuo, S-I. Lee, S. Kiesler: Eliciting information from people with a gendered humanoid robot, IEEE 14th Int. Workshop Robot Hum. Interact. Commun., Nashville (2005) pp. 158–163

71.35 K.M. Lee, N. Park, H. Song: Can a robot be perceived as a developing creature?: Effects of a robot's long-term cognitive developments on its social presence and people's social responses toward it, Human Commun. Res. **31**(4), 538–563 (2005)

71.36 J. Goetz, S. Kiesler, A. Powers: Matching robot appearance and behavior to tasks to improve human–robot cooperation, Proc. 12th IEEE Int. Workshop Robot Hum. Interact. Commun., Silicon Valley (2003) pp. 55–60

71.37 M.K. Lee, S. Kiesler, J. Forlizzi, S. Srinivasa, P. Rybski: Gracefully mitigating breakdowns in robotic services, Proc. 6th ACM/IEEE Int. Conf. HRI, Lausanne (2010) pp. 203–210

71.38 B. Shore: *Culture in Mind: Cognition, Culture, and the Problem of Meaning* (Oxford Univ. Press, Oxford 1996)

71.39 V. Evers, H. Maldonado, T. Brodecki, P. Hinds: Relational vs. group self-construal: Untangling the role of national culture in HRI, Proc. 3rd ACM/IEEE Int. Conf. HRI, Amsterdam (2008)

71.40 L. Wang, P.-L.P. Rau, V. Evers, B.K. Robinson, P. Hinds: When in Rome: The role of culture and context in adherence to robot recommendations, Proc. 5th ACM/IEEE Int. Conf. HRI, Osaka (2010) pp. 359–366

71.41 S. Sabanovic: Robots in society, society in robots – Mutual shaping of society and technology as a framework for social robot design, Int. J. Soc. Robotics **2**(4), 439–450 (2010)

71.42 G. Shaw-Garlock: Looking forward to sociable robots, Int. J. Soc. Robotics **1**(3), 249–260 (2009)

71.43 P.H. Kahn, A.L. Reichert, H.E. Gary, T. Kanda, H. Ishiguro, S. Shen, J.H. Ruckert, B. Gill: The new ontological category hypothesis in human–robot interaction, Proc. 6th ACM/IEEE Int. Conf. HRI, Lausanne (2011) pp. 159–160

71.44 P.H. Kahn, N.G. Freier, B. Friedman, R.L. Severson, E.N. Feldman: Social and moral relationships with robotic others?, IEEE 13th Int. Workshop Robot Hum. Interact. Commun., Kurashiki (2004) pp. 545–550

71.45 S. Turkle: *A Nascent Robotics Culture: New Complicities for Companionship* (AAAI, Boston 2006)

71.46 B. Scassellati: How developmental psychology and robotics complement each other, NSF/DARPA Workshop Dev. Learn. (MIT Press, CSAIL, Cambridge 2006)

71.47 H. Ishiguro: Android science – toward a new cross-interdisciplinary framework, ICCS/CogSci Workshop Toward Soc. Mech. Android Sci., Stresa (2005) pp. 1–6

71.48 K.F. MacDorman, H. Ishiguro: The uncanny advantage of using androids in cognitive and social science research, Interact. Stud. **7**(3), 297–337 (2006)

71.49 M. Stanley, J. Sabini: On maintaining social norms: A field experiment in the subway. In: *Advances in Environmental Psychology: The Urban Environment*, ed. by A. Baum, J.E. Singer, S. Valins (Erlbaum Associates, Hillsdale 1978) pp. 31–40

71.50 H. Kozima, M.P. Michalowski, C. Nakagawa: Keepon: A playful robot for research, therapy, and entertainment, Int. J. Soc. Robotics **1**(1), 3–18 (2009)

71.51 K.F. MacDorman: Introduction to the special issue on android science, Connect. Sci. **18**(4), 313–317 (2006)

71.52 J.J. Gibson: *The Ecological Approach to Visual Perception* (Houghton Mifflin, Boston 1979)

71.53 E.S. Reed: *Encountering the World: Toward an Ecological Psychology* (Oxford Univ. Press, Oxford 1996)

71.54 Y. Yamaji, T. Miyake, Y. Yoshiike, P.R.S. De Silva, M. Okada: STB: Human-dependent sociable trash box, Proc. 5th ACM/IEEE Int. Conf. HRI, Osaka (2010) pp. 197–198

71.55 H. Ishiguro: Android science: Conscious and subconscious recognition, Connect. Sci. **18**(4), 319–332 (2006)

71.56 S. Nishio, H. Ishiguro, N. Hagita: Geminoid: Teleoperated android of an existing person. In: *Humanoid Robots, New Developments*, ed. by A.C. De Pina Filho (InTech, Vienna 2007) pp. 343–352

71.57 S. Nishio, H. Ishiguro, N. Hagita: Can a teleoperated robot represent personal presence? – A case study with children, Psychologia **50**(4), 330–342 (2007)

71.58 M. Shimada, K. Yamauchi, T. Minato, H. Ishiguro, S. Itakura: Studying the influence of the chameleon effect on humans using an android, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Nice (2008)

71.59 J. Wainer, D.J. Feil-Seifer, D.A. Shell, M.J. Mataric: Embodiment and human-robot interaction: A taskbased perspective, Proc. 2nd ACM/IEEE Int. Conf. HRI, Washington (2007) pp. 872–877

71.60 P. Schermerhorn, M. Scheutz: Disentangling the effects of robot affect, embodiment, and au-

tonomy on human team members in a mixed-initiative task, Proc. 4th Int. Conf. Adv. Comput.–Hum. Interact., Gosier (2011) pp. 235–241
71.61 W.A. Bainbridge, J.W. Hart, E.S. Kim, B. Scassellati: The benefits of interactions with physically present robots over video-displayed agents, Int. J. Soc. Robotics **1**(2), 41–52 (2010)
71.62 B. Mutlu: Designing embodied cues for dialog with robots, AI Magazine **32**(4), 17–30 (2011)
71.63 E.T. Hall: *The Hidden Dimension* (Anchor Books, New York 1966)
71.64 L. Takayama, C. Pantofaru: Influences on proxemic behaviors in human–robot interaction, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), St. Louis (2009) pp. 5495–5502
71.65 D.S. Syrdal, K.L. Koay, M.L. Walters, K. Dautenhahn: A personalized robot companion? The role of individual differences on spatial preferences in HRI scenarios, Proc. 16th IEEE Int. Symp. Robot Hum. Interact. Commun., Jeju Island (2007) pp. 1143–1148
71.66 J. Mumm, B. Mutlu: Human-robot proxemics: Physical and psychological distancing in human-robot interaction, Proc. 6th ACM/IEEE Int. Conf. HRI, Lausanne (2011) pp. 331–338
71.67 M.L. Walters, M.A. Oskoei, D.S. Syrdal, K. Dautenhahn: A long-term human–robot proxemic study, Proc. 20th IEEE Int. Symp. Robot Hum. Interact. Commun., Atlanta (2011) pp. 137–142
71.68 C. Yu, M. Scheutz, P. Schermerhorn: Investigating multimodal real-time patterns of joint attention in an HRI word learning task, Proc. 5th ACM/IEEE Int. Conf. HRI, Osaka (2010) pp. 309–316
71.69 T. Yonezawa, H. Yamazoe, A. Utsumi, S. Abe: Gaze-communicative behavior of stuffed-toy robot with joint attention and eye contact based on ambient gaze-tracking, Proc. 9th Int. Conf. Multimodal Interfaces, Nagoya (2007) pp. 140–145
71.70 Y. Yoshikawa, K. Shinozawa, H. Ishiguro, N. Hagita, T. Miyamoto: Responsive robot gaze to interaction partner, Robotics Sci. Syst., Philadelphia (2006)
71.71 B. Mutlu, T. Shiwa, T. Kanda, H. Ishiguro, N. Hagita: Footing in human-robot conversations: How robots might shape participant roles using gaze cues, Proc. 4th ACM/IEEE Int. Conf. HRI, San Diego, California (2009) pp. 61–68
71.72 M. Staudte, M.W. Crocker: Visual attention in spoken human-robot interaction, Proc. 4th ACM/IEEE Int. Conf. HRI, San Diego (2009) pp. 77–84
71.73 B. Mutlu, J. Forlizzi, J.K. Hodgins: A storytelling robot: Modeling and evaluation of human-like gaze behavior, IEEE-RAS Conf. Humanoid Robots, Genoa (2006) pp. 518–523
71.74 C. Yu, P. Schermerhorn, M. Scheutz: Adaptive eye gaze patterns in interactions with human and artificial agents, ACM Trans. Interact. Intell. Syst. **1**(2), 1–25 (2012)
71.75 H. Admoni, C. Bank, J. Tan, M. Toneva, B. Scassellati: Robot gaze does not reflexively cue human attention, Proc. 33rd Annu. Conf. Cogn. Sci. Soc., Boston (2011) pp. 1983–1988
71.76 W.S. Condon: Cultural microrhythms. In: *Interaction Rhythms: Periodicity in Communicative Behavior*, ed. by M. Davis (Human Sciences Press, New York 1982) pp. 53–76
71.77 E. Goffman: Some context for content analysis: A view of the origins of structural studies of face-to-face interaction. In: *Conducting Interaction: Patterns of Behavior in Focused Encounters*, ed. by A. Kendon (Cambridge Univ. Press, Cambridge 1990) pp. 15–49
71.78 C. Trevarthen: Can a robot hear music? Can a robot dance? Can a robot tell what it knows or intends to do? Can it feel pride or shame in company? – Questions of the nature of human vitality, Proc. 2nd Int. Workshop Epigenet. Robotics, Edinburgh (2002)
71.79 M. Michalowski, S. Sabanovic, H. Kozima: A dancing robot for rhythmic social interaction, Proc. 2nd ACM/IEEE Int. Conf. HRI, Washington DC (2007) pp. 89–96
71.80 M.P. Michalowski, R. Simmons, H. Kozima: Rhythmic attention in child-robot dance play, Proc. 18th IEEE Int. Symp. Robot Hum. Interact. Commun., Toyama (2009) pp. 816–821
71.81 E. Avrunin, J. Hart, A. Douglas, B. Scassellati: Effects related to synchrony and repertoire in perceptions of robot dance, Proc. 6th ACM/IEEE Int. Conf. HRI, Lausanne (2011) pp. 93–100
71.82 G. Hoffman, C. Breazeal: Anticipatory perceptual simulation for human-robot joint practice: Theory and application study, Proc. 23rd AAAI Conf. Artif. Intell., Chicago (2008) pp. 1357–1362
71.83 G. Hoffman, G. Weinberg: Interactive improvisation with a robotic marimba player, Auton. Robots **31**(2–3), 133–153 (2011)
71.84 G. Deàk, I. Fasel, J. Movellan: The emergence of shared attention: Using robots to test developmental theories, Proc. 1st Int. Workshop Epigenet. Robotics, Lund (2001) pp. 95–104
71.85 K. Dautenhahn: Roles and functions of robots in human society: Implications from research in autism therapy, Robotica **21**(4), 443–452 (2003)
71.86 H. Kozima, C. Nakagawa, Y. Yasuda: Wowing together: What facilitates social interactions in children with autistic spectrum disorders, Proc. 6th Int. Workshop Epigenet. Robotics Model. Cogn. Dev. Robotics Syst., Paris (2006) p. 177
71.87 B. Scassellati: How social robots will help us to diagnose, treat, and understand autism, Proc. 12th Int. Symp. Robotics Res., San Francisco, ed. by S. Thrun, R.A. Brooks, H. Durrant-Whyte (Springer, Berlin, Heidelberg 2005) pp. 552–563
71.88 D.J. Feil-Seifer, M.J. Mataric: B3IA: An architecture for autonomous robot-assisted behavior intervention for children with autism spectrum disorders, Proc. 17th IEEE Int. Workshop Robot Hum. Interact. Commun., Munich (2008) pp. 328–333
71.89 H. Kozima, C. Nakagawa, Y. Yasuda: Interactive robots for communication-care: A case-study in autism therapy, Proc. 14th IEEE Int. Workshop Robot Hum. Interact. Commun., Nashville (2005) pp. 341–346

71.90 H. Kozima, Y. Yasuda, C. Nakagawa: Social interaction facilitated by a minimally-designed robot: Findings from longitudinal therapeutic practices for autistic children, Proc. 16th IEEE Int. Symp. Robot Hum. interact. Commun., Jeju Island (2007) pp. 599–604

71.91 H.A. Simon: *The Sciences of the Artificial* (MIT Press, Cambridge 1969)

71.92 B. Adams, C.L. Breazeal, R.A. Brooks, B. Scassellati: Humanoid robots: A new kind of tool, IEEE Intell. Syst. Appl. **15**(4), 25–31 (2000)

71.93 L.W. Barsalou, C. Breazeal, L.B. Smith: Cognition as coordinated non-cognition, Cogn. Process. **8**(2), 79–91 (2007)

71.94 A.L. Thomaz, M. Berlin, C. Breazeal: An embodied computational model of social referencing, Proc. 14th IEEE Int. Workshop Robot Hum. Interact. Commun., Nashville (2005) pp. 591–598

71.95 G. Hoffman, C. Breazeal: Robotic partners? Bodies and minds: An embodied approach to fluid human-robot collaboration, Proc. 5th Int. Workshop Cogn. Robotics, Boston (2006) pp. 95–102

71.96 G. Hoffman: Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team, Proc. 2nd ACM/IEEE Int. Conf. HRI, Washington D.C. (2007) pp. 1–8

71.97 Y. Demiris, A. Meltzoff: The robot in the crib: A developmental analysis of imitation skills in infants and robots, Infant Child Dev. **17**(1), 43–53 (2008)

71.98 C. Nehaniv, K. Dautenhahn (Eds.): *Imitation and Social Learning in Robots, Humans and Animals: Behavioural, Social and Communicative Dimensions* (Cambridge Univ. Press, Cambridge 2009)

71.99 B. Scassellati: Imitation and mechanisms of joint attention: A developmental structure for building social skills on a humanoid robot, Lect. Notes Comput. Sci. **1562**, 176–195 (1999)

71.100 Y. Nagai, K. Hosoda, A. Morita, M. Asada: A constructive model for the development of joint attention, Connect. Sci. **15**(4), 211–229 (2003)

71.101 F. Kaplan, V. Hafner: The challenges of joint attention, Proc. 4th Int. Workshop Epigenet. Robotics, Lund (2004) pp. 67–74

71.102 C. Crick, M. Munz, B. Scassellati: Synchronization in social tasks: Robotic drumming, Proc. 15th IEEE Int. Workshop Robot Hum. Interact. Commun., Hatfield (2006) pp. 97–102

71.103 P. Bakker, Y. Kuniyoshi: Robot see, robot do: An overview of robot imitation, AISB-96 Workshop Learn. Robots Animals, Brighton (1996) pp. 3–11

71.104 R.S. Jackendoff: On beyond zebra: The relation of linguistic and visual information, Cognition **26**, 89–114 (1987)

71.105 B. Landau, R.S. Jackendoff: *What* and *where* in spatial language and spatial cognition, Behav. Brain Sci. **16**, 217–265 (1993)

71.106 L. Talmy: The fundamental system of spatial schemas in language. In: *From Perception to Meaning: Image Schemas in Cognitive Linguistics*, ed. by B. Hamp (Mouton de Gruyter, Berlin 2005)

71.107 T.P. Regier: The Acquisition of Lexical Semantics for Spatial Terms: A Connectionist Model of Perceptual Categorization, Ph.D. Thesis (University of California at Berkeley, Berkeley 1992)

71.108 J.D. Kelleher, F.J. Costello: Applying computational models of spatial prepositions to visually situated dialog, Comput. Linguist. **35**(2), 271–306 (2008)

71.109 T.P. Regier, L.A. Carlson: Grounding spatial language in perception: An empirical and computational investigation, J. Exp. Psychol. **130**(2), 273–298 (2001)

71.110 G. Bugmann, E. Klein, S. Lauria, T. Kyriacou: Corpus-based robotics: A route instruction example, Proc. 8th Conf. Intell. Auton. Syst. (IAS-8), Amsterdam (2004) pp. 96–103

71.111 M. Levit, D. Roy: Interpretation of spatial language in a map navigation task, IEEE Trans. Syst. Man Cybern. B **37**(3), 667–679 (2007)

71.112 M. MacMahon, B. Stankiewicz, B. Kuipers: Walk the talk: Connecting language, knowledge, and action in route instructions, Proc. Natl. Conf. Artif. Intell., Boston (2006) pp. 1475–1482

71.113 H. Kress-Gazit, G.E. Fainekos: Translating structured English to robot controllers, Adv. Robotics **22**, 1343–1359 (2008)

71.114 C. Matuszek, D. Fox, K. Koscher: Following directions using statistical machine translation, Proc. 5th ACM/IEEE Int. Conf. HRI, Nara (2010) pp. 251–258

71.115 A. Vogel, D. Jurafsky: Learning to follow navigational directions, Proc. 48th Annu. Meet. Assoc. Comput. Linguist., Uppsala (2010) pp. 806–814

71.116 S. Harnad: The symbol grounding problem, Physica D **43**, 335–346 (1990)

71.117 T. Winograd: Procedures as a Representation for Data in a Computer Program for Understanding Natural Language, MIT Tech. Rep. TMAC-TR-84 (MIT, Cambridge 1971)

71.118 K.Y. Hsiao, N. Mavridis, D. Roy: Coupling perception and simulation: Steps towards conversational robotics, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Las Vegas (2003) pp. 928–933

71.119 D. Roy, K.Y. Hsiao, N. Mavridis: Conversational Robots: Building blocks for grounding word meanings, Proc. HLT-NAACL 2003 Workshop Learn. Word Mean. Non-Linguist. Data, Stroudsburg (2003) pp. 70–77

71.120 D. Roy: Semiotic schemas: A framework for grounding language in action and perception, Artif. Intell. **167**(1-2), 170–205 (2005)

71.121 J. Dzifcak, M. Scheutz, C. Baral, P. Schermerhorn: What to do and how to do it: Translating natural language directives into temporal and dynamic logic representation for goal management and action execution, IEEE Int. Conf. Robotics Autom. (ICRA), Kobe (2009) pp. 4163–4168

71.122 Y. Sugita, J. Tani: Learning semantic combinatoriality from the interaction between linguistic and behavioral processes, Adapt. Behav. – Animals Animat. Softw. Agents Robots Adapt. Syst. **13**(1), 33–52 (2005)

71.123 J. Modayil, B. Kuipers: Autonomous development of a grounded object ontology by a learning

robot, Proc. 22nd AAAI Conf. Artif. Intell., Vancouver (2007) pp. 1095–1101
71.124 D. Marocco, A. Cangelosi, K. Fischer, T. Belpaeme: Grounding action words in the sensorimotor interaction with the world: Experiments with a simulated iCub humanoid robot, Front. Neurorobotics **4**, 1–15 (2010)
71.125 R. Ge, R.J. Mooney: A statistical semantic parser that integrates syntax and semantics, Proc. 9th Conf. Comput. Nat. Lang. Learn., Ann Arbor (2005) pp. 9–16
71.126 N. Shimizu, A. Haas: Learning to follow navigational route instructions, Proc. 21st Int. Jt. Conf. Artif. Intell., Pasadena (2009) pp. 1488–1493
71.127 S.R.K. Branavan, H. Chen, L.S. Zettlemoyer, R. Barzilay: Reinforcement learning for mapping instructions to actions, Proc. 47th Jt. Conf. Annu. Meet. Assoc. Comput. Linguist. 4th Int. Jt. Conf. Nat. Lang. Process. (AFNLP), Singapore (2009) pp. 82–90
71.128 S.R.K. Branavan, D. Silver, R. Barzilay: Learning to win by reading manuals in a Monte-Carlo framework, Proc. 49th Annu. Meet. Assoc. Comput. Linguist. Hum. Lang. Technol., Portland (2011)
71.129 T. Kollar, S. Tellex, D. Roy, N. Roy: Toward understanding natural language directions, Proc. 5th ACM/IEEE Int. Conf. HRI, Osaka (2010) pp. 259–266
71.130 D. Bailey: When Push Comes to Shove: A Computational Model of the Role of Motor Control in the Acquisition of Action Verbs, Ph.D. Thesis (Univ. of California, Berkeley 1997)
71.131 T. Kollar, S. Tellex, D. Roy, N. Roy: Grounding verbs of motion in natural language commands to robots, Proc. Int. Symp. Exp. Robotics, New Delhi (2010) pp. 31–47
71.132 S. Tellex, T. Kollar, S. Dickerson, M.R. Walter, A.G. Banerjee, S. Teller, N. Roy: Understanding natural language commands for robotic navigation and mobile manipulation, Proc. Natl. Conf. Artif. Intell., San Francisco (2011)
71.133 J.L. Burke, R.R. Murphy, M.D. Coovert, D.L. Riddle: Moonlight in Miami: Field study of human-robot interaction in the context of an urban search and rescue disaster response training exercise, Hum.–Comput. Interact. **19**(1/2), 85–116 (2004)
71.134 K. Stubbs, P.J. Hinds, D. Wettergreen: Autonomy and common ground in human-robot interaction: A field study, IEEE Intell. Syst. **22**(2), 42–50 (2007)
71.135 S. Kiesler: Fostering common ground in human-robot interaction, Proc. 14th IEEE Int. Workshop Robot Hum. Interact. Commun., Nashville (2005) pp. 729–734
71.136 T. Fong, C. Thorpe, C. Baur: Collaboration, dialogue, human-robot interaction, Robotics Res. **6**, 255–266 (2003)
71.137 M.E. Foster, T. By, M. Rickert, A. Knoll: Human-robot dialogue for joint construction tasks, Proc. 8th Int. Conf. Mulltimodal Interfaces, Banff (2006) pp. 68–71
71.138 S. Li, B. Wrede, G. Sagerer: A computational model of multi-modal grounding for human robot interaction, Proc. 7th SIGdial Workshop Discourse Dialogue, Sydney (2009) pp. 153–160
71.139 P.R. Cohen, H.J. Levesque: Teamwork, Nous **25**(4), 487–512 (1991)
71.140 J. Peltason, B. Wrede: Pamini: A framework for assembling mixed-initiative human-robot interaction from generic interaction patterns, Proc. 11th SIGdial Annu. Meet. Special Interest Group Discourse Dialogue, Tokyo (2010) pp. 229–232
71.141 J. Peltason, B. Wrede: The curious robot as a case-study for comparing dialog systems, AI Magazine **32**(4), 85–99 (2011)
71.142 M.F. Schober: Spatial perspective-taking in conversation, Cognition **47**(1), 1–24 (1993)
71.143 J.E. Hanna, M.K. Tanenhaus, J.C. Trueswell: The effects of common ground and perspective on domains of referential interpretation, J. Mem. Lang. **49**(1), 43–61 (2003)
71.144 J.G. Trafton, N.L. Cassimatis, M.D. Bugajska, D.P. Brock, F.E. Mintz, A.C. Schultz: Enabling effective human–robot interaction using perspective-taking in robots, IEEE Trans. Syst. Man Cybern. A **35**(4), 460–470 (2005)
71.145 M. Berlin, J. Gray, A.L. Thomaz, C. Breazeal: Perspective taking: An organizing principle for learning in human–robot interaction, Proc. 21st Natl. Conf. Artif. Intell., Boston (2006) p. 1444
71.146 R. Moratz, K. Fischer, T. Tenbrink: Cognitive modeling of spatial reference for human-robot interaction, Int. J. Artif. Intell. Tools **10**(04), 589–611 (2001)
71.147 R. Ros, S. Lemaignan, E.A. Sisbot, R. Alami, J. Steinwender, K. Hamann, F. Warneken: Which one? Grounding the referent based on efficient human-robot interaction, Proc. 19th IEEE Int. Symp. Robot Hum. Interact. Commun., Viareggio (2010) pp. 570–575
71.148 A. Holroyd, C. Rich, C.L. Sidner, B. Ponsler: Generating connection events for human-robot collaboration, Proc. 20th IEEE Int. Symp. Robot Hum. Interact. Commun., Atlanta (2011) pp. 241–246
71.149 G. Butterworth, L. Grover: Joint visual attention, manual pointing, and preverbal communication in human infancy. In: *Attention and Performance, Vol. 13: Motor Representation and Control*, ed. by M. Jeannerod (Lawrence Erlbaum Assoc., Mahwah 1990) pp. 605–624
71.150 C. Rich, P. Ponsler, A. Holroyd, C.L. Sidner: Recognizing engagement in human-robot interaction, Proc. 5th ACM/IEEE Int. Conf. HRI, Osaka (2010) pp. 375–382
71.151 V. Gallese, A. Goldman: Mirror neurons and the simulation theory of mind-reading, Trends Cogn. Sci. **2**(12), 493–501 (1998)
71.152 E. Bicho, W. Erlhagen, L. Louro, E. Costa e Silva: Neuro-cognitive mechanisms of decision making in joint action: A human–robot interaction study, Hum. Mov. Sci. **30**(5), 846–868 (2011)
71.153 J. Gray, C. Breazeal, M. Berlin, A. Brooks, J. Lieberman: Action parsing and goal inference using self as simulator, Proc. 14th IEEE Int. Workshop

Robot Hum. Interact. Commun., Nashville (2005) pp. 202–209
71.154 M.N. Nicolescu, M.J. Mataric: Linking perception and action in a control architecture for human-robot domains, Proc. 36th Annu. Hawaii Int. Conf. Syst. Sci., Big Island (2003) pp. 10–20
71.155 C. Breazeal, G. Hoffman, A. Lockerd: Teaching and working with robots as a collaboration, Proc. 3rd Int. Jt. Conf. Auton. Agents Multiagent Syst., New York, Vol. 3 (2004) pp. 1030–1037
71.156 R. Alami, A. Clodic, V. Montreuil, E.A. Sisbot, R. Chatila: Task planning for human–robot interaction, Proc. 2005 Jt. Conf. Smart Obj. Ambient Intell. Innov. Context-Aware Serv. Usages Technol., Grenoble (2005) pp. 81–85
71.157 R. Alami, A. Clodic, V. Montreuil, E.A. Sisbot, R. Chatila: Toward human-aware robot task planning, AAAI Spring Symp.: To Boldly Go where No Human-Robot Team Has Gone Before, Palo Alto (2006) pp. 39–46
71.158 R. Bellman: *Dynamic Programming* (Princeton Univ. Press, Princeton 1957)
71.159 N. Roy, J. Pineau, S. Thrun: Spoken dialog management for robots, Proc. Assoc. Comput. Linguist., Hong Kong (2000) pp. 93–100
71.160 J. Hoey, P. Poupart, C. Boutilier, A. Mihailidis: POMDP models for assistive technology, Proc. AAAI Fall Symp. Caring Mach., AI in Eldercare (2005)
71.161 J. Williams, S. Young: Scaling up POMDPs for dialogue management: The *summary POMDP* method, Proc. IEEE Autom. Speech Recognit. Underst. Workshop, Cancun (2005)
71.162 D. Litman, S. Singh, M. Kearns, M. Walker: NJFun: A reinforcement learning spoken dialogue system, Proc. ANLP/NAACL 2000 Workshop Conversat. Syst., Seattle (2000) pp. 17–20
71.163 F. Broz, I. Nourbakhsh, R. Simmons: Planning for human–robot interaction using time-state aggregated POMDPs, Proc. 23rd Conf. Artif. Intell., Chicago (2008) pp. 1339–1344
71.164 F. Doshi, N. Roy: The permutable POMDP: Fast solutions to POMDPs for preference elicitation, Proc. 7th Int. Conf. Auton. Agents Multiagent Syst., Estoril (2008) pp. 493–500
71.165 R. Wilcox, S. Nikolaidis, J. Shah: Optimization of temporal dynamics for adaptive human-robot interaction in assembly manufacturing, Proc. Robotics Sci. Syst., Sydney (2012) p. 441
71.166 T. Prommer, H. Holzapfel, A. Waibel: Rapid simulation-driven reinforcement learning of multimodal dialog strategies in human–robot interaction, 9th Int. Conf. Spoken Lang. Process., Pittsburgh (2006)
71.167 J.M. Porta, N. Vlassis, M. Spaan, P. Poupart: Point-based value iteration for continuous POMDP, J. Mach. Learn. Res. **7**, 2329–2367 (2006)
71.168 F. Doshi, N. Roy: Efficient model learning for dialog management, Proc. 2nd ACM/IEEE Int. Conf. HRI, Arlington (2007) pp. 65–72
71.169 F. Doshi, N. Roy: Spoken language interaction with model uncertainty: An adaptive human-robot interaction system, Connect. Sci. **20**(4), 299–319 (2008)
71.170 M. Cakmak, A.L. Thomaz: Designing robot learners that ask good questions, Proc. 7th Annu. ACM/IEEE Int. Conf. HRI, Boston (2012) pp. 17–24
71.171 M.A. Goodrich, D.R. Olsen: Seven principles of efficient human robot interaction, IEEE Int. Conf. Syst. Man Cybern., Washington D.C. (2003) pp. 3942–3948
71.172 J.W. Crandall, M.A. Goodrich, D.R. Olsen, C.W. Nielsen: Validating human-robot interaction schemes in multitasking environments, IEEE Trans. Syst. Man Cybern. A **35**(4), 438–449 (2005)
71.173 R. Sun (Ed.): *Cognition and Multiagent Interaction: From Cognitive Modeling to Social Simulation* (Cambridge Univ. Press, Cambridge 2005)

Multimedia Contents

72. Social Robotics

Cynthia Breazeal, Kerstin Dautenhahn, Takayuki Kanda

This chapter surveys some of the principal research trends in Social Robotics and its application to human–robot interaction (HRI). Social (or Sociable) robots are designed to interact with people in a natural, interpersonal manner – often to achieve positive outcomes in diverse applications such as education, health, quality of life, entertainment, communication, and tasks requiring collaborative teamwork. The long-term goal of creating social robots that are competent and capable partners for people is quite a challenging task. They will need to be able to communicate naturally with people using both verbal and nonverbal signals. They will need to engage us not only on a cognitive level, but on an emotional level as well in order to provide effective social and task-related support to people. They will need a wide range of social-cognitive skills and a theory of other minds to understand human behavior, and to be intuitively understood by people. A deep understanding of human intelligence and behavior across multiple dimensions (i. e., cognitive, affective, physical, social, etc.) is necessary in order to design robots that can successfully play a beneficial role in the daily lives of people. This requires a multidisciplinary approach where the design of social robot technologies and methodologies are informed by robotics, artificial intelligence, psychology, neuroscience, human factors, design, anthropology, and more.

72.1 **Overview** 1936

72.2 **Social Robot Embodiment** 1936
72.2.1 Anthropomorphic Design 1936
72.2.2 Design Space 1936

72.3 **Social Robots and Social-Emotional Intelligence** 1938
72.3.1 Theories of Emotion 1939
72.3.2 Example: A Synthesis of Emotion Theories in Action .. 1939
72.3.3 Emotional Empathy 1940

72.4 **Socio-Cognitive Skills** 1941
72.4.1 Shared Attention 1941
72.4.2 Mental Perspective Taking 1941
72.4.3 Perspective Taking in Collaboration and Teamwork 1942

72.5 **Human Social Responses to Social Robots** 1944
72.5.1 Social Judgments 1944
72.5.2 Physical Versus Virtual Embodiment 1944
72.5.3 Social Stimulus to Learn About People 1944
72.5.4 Social Rapport 1945
72.5.5 Social Support 1945

72.6 **Social Robots and Communication Skills** 1946
72.6.1 Verbal/Nonverbal Communication 1946
72.6.2 Mechanisms for Human–Robot Communication 1947
72.6.3 Challenges 1950

72.7 **Long-Term Interaction with Robot Companions** 1950
72.7.1 Robot Companions 1950
72.7.2 Engagement and Long-Term Relationships 1950
72.7.3 Robot Home Companions for Supporting Elderly Users 1951
72.7.4 Example: Long-Term Interactions with a Robotic Weight Loss Coach 1952
72.7.5 Challenges 1953

72.8 **Tactile Interaction with Social Robots** .. 1954
72.8.1 Touch for Social Interaction 1954
72.8.2 Touch Sensors and Mechanisms Used for HRI 1954

72.8.3 Example: Teaching Children with Autism About Tactile Interaction 1956
72.8.4 Challenges and Opportunities .. 1957
72.9 **Social Robots and Teamwork** 1958
72.9.1 Human–Robot Teamwork and Collaboration 1958
72.9.2 Robots as Social Mediators 1959
72.9.3 Research Direction 1959
72.10 **Conclusion** 1959
72.11 **Further Reading** 1960
Video-References 1960
References 1961

72.1 Overview

The way a person interacts with a social robot (or sociable robot) is quite different from interacting with the majority of autonomous mobile robots today. Modern autonomous robots are generally viewed as tools that human specialists use to perform hazardous tasks in remote environments (i. e., sweeping minefields, inspecting oil wells, mapping mines, etc.). In dramatic contrast, social (or sociable) robots are designed to engage people in an interpersonal manner, often as partners, in order to achieve positive outcomes in domains such as education, therapy, or health, or task-related goals in areas such as coordinated teamwork for manufacturing, search and rescue, domestic chores, and more.

The development of socially intelligent and socially skillful robots drives research to develop autonomous or semiautonomous robots that are natural and intuitive for the general public to interact with, communicate with, work with as partners, and teach new capabilities. *Dautenhahn*'s work is among the earliest in thinking about robots with interpersonal social intelligence where relationships between specific individuals are important [72.1, 2]. These early works pose the question:

> *What are the common social mechanisms of communication and understanding that can produce efficient, enjoyable, natural and meaningful interactions between humans and robots?*

Promisingly, there have been initial and ongoing strides in all of these areas ([72.3–11], etc.). In addition, this domain motivates new questions for robotics researchers, such as how to design for a successful long-term relationship where the robot remains appealing and provides consistent benefit to people over weeks, months, and even years. The benefit that social robots provide people extends far beyond strict task-performing utility to include educational (Chap. 79), health and therapeutic (Chap. 64), domestic (Chap. 65), social and emotional goals (e.g., entertainment, companionship, communication, etc.), and more.

We begin this chapter with a brief overview of a wide assortment of embodiments of socially interactive robots that have been developed around the world (Sect. 72.2). We follow with selected topics that highlight some of the representative research themes: social-emotional intelligence and emotion-based interaction (Sect. 72.3), and social-cognitive skills (Sect. 72.4). Human social responses to robots(Sect. 72.5), verbal and non-verbal communication (Sect. 72.6), long-term interaction (Sect. 72.7), touch-based interaction (Sect. 72.8), and teamwork with robot partners (Sect. 72.9). We rely on examples from our own research programs to illustrate these trends, while making reference to other excellent works performed in other research labs.

72.2 Social Robot Embodiment

72.2.1 Anthropomorphic Design

Social robots are designed to interact with people in human-centric terms and to operate in human environments alongside people. Many social robots are humanoid or animal-like in form, although this does not have to be the case. A unifying characteristic is that that social robots engage people in an interpersonal manner, communicating and coordinating their behavior with humans through verbal, nonverbal, or affective modalities. People have a strong tendency to anthropomorphize social robots [72.12] and to reason about their behavior in terms of having their own mental states (e.g., thoughts, intents, beliefs, desires, etc.). Hence, anthropomorphic design principles, spanning from the physical appearance of robots, to how they move and behave, and how they interact with people, are often employed to facilitate interaction and acceptance.

72.2.2 Design Space

The design space of social robotics is quite large. It is important to note that a more human-like design

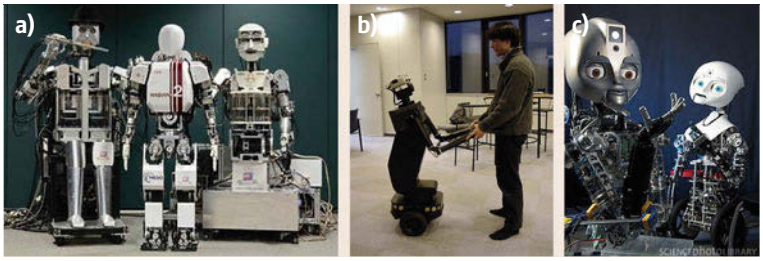


Fig.72.1a–c Examples of socially interactive humanoid robots: **(a)** Humanoid robots developed at Waseda University from left to right: a flutist robot WF-4RII (after [72.13]), WABIAN-2 (after [72.14]), and WE-4RII (after [72.15]); **(b)** Robovie developed at ATR (Advanced Telecommunications Research Institute, Kyoto) is able to gesture with its arms and give a hug; **(c)** Nexi and Maddox, developed at MIT, are mobile and dexterous social robots used to study collaborative human–robot teamwork (after [72.16])

does not necessarily correlate with a *better* design. One needs to balance the robot design with the task, user, and context [72.17]. As can be seen in the following examples, social robots exploit many different modalities to communicate and express social-emotional behavior. These include whole-body motion, proxemics (i. e., interpersonal distance), gestures, facial expressions, gaze behavior, head orientation, linguistic or emotive vocalization, touch-based communication, and an assortment of display technologies.

For social robots to close the communication loop and coordinate their behavior with people, they must also be able to perceive, interpret, and respond appropriately to verbal and nonverbal cues from humans. VIDEO 219 shows dog-inspired robot behaviour to faciliate intention reading by people. Given the richness of human behavior and the complexity of human environments, many social robots are among the most sophisticated, articulate, behaviorally rich, and intelligent robots today.

As shown in Fig. 72.1, a number of socially interactive humanoid robots have been developed (Chap. 65) that can participate in whole body social interaction with people such as dancing [72.22], walking hand-in-hand [72.23, 24], playing a musical duet [72.13], or transferring skills to unskilled persons [72.25], or collaborating as a team with people in search and retrieve tasks [72.16]. Their arms and hands are designed to exhibit human-like gestures such as pointing, shrugging shoulders, shaking hands, or giving a hug [72.26–28]. Some of them are designed with mechanical faces to communicate with humans via facial expressions [72.18, 20, 29].

Whereas many of these humanoids have a mechanical appearance, android robots are designed to have a very human-like appearance with skin, teeth, hair, and clothes (Fig. 72.2). A design challenge of android robots is to avoid the *uncanny valley* where the appearance and movement of the robot resemble more of an animate corpse than a living human. Designs that fall within the *uncanny valley* elicit a strong negative reaction from people [72.37]. In contrast to trying to look as human-like as possible, there are also more doll-like robots that are intentionally designed to have simplified facial cues and predictable movements to be suitable for therapeutic contexts [72.21].

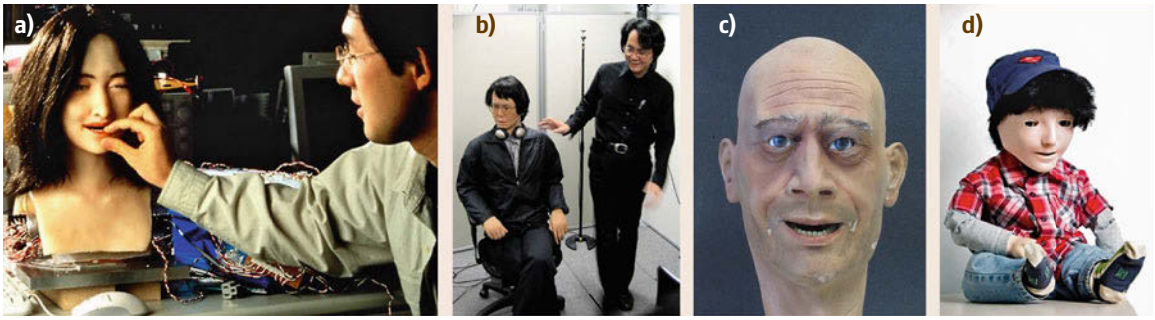


Fig.72.2a–d Some examples of androids: **(a)** One of the earliest face robots developed at the Science University of Tokyo (after [72.18]); **(b)** Geminoid developed at ATR (after [72.19]); **(c)** ROMAN developed at the University of Kaiserslautern (after [72.20]); **(d)** KASPAR developed at the University of Hertfordshire is a child like robot used during therapeutic interventions to help children with autism (after [72.21])

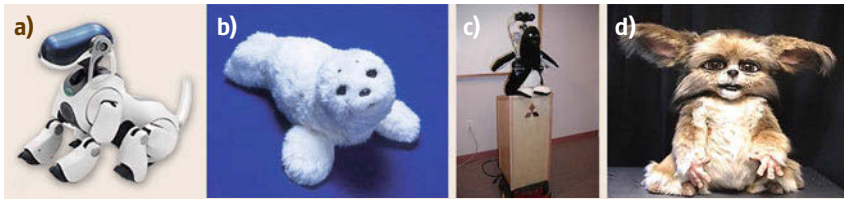


Fig.72.3a–d Examples of social robots inspired by animals with anthropomorphic qualities: **(a)** AIBO, the robotic dog developed by Sony (after [72.30]), **(b)** Paro, the therapeutic seal robot developed at AIST (after [72.31]), **(c)** Mel, the conversational robotic penguin developed at MERL (after [72.32]), and **(d)** Leonardo developed at the MIT Media Lab (after [72.33])

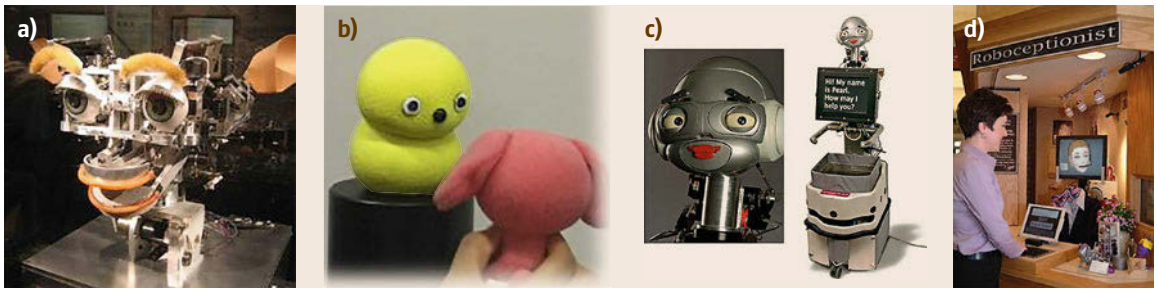


Fig.72.4a–d Examples of social robots that are neither humanoid nor zoomorphic but capture key social attributes: **(a)**: Kismet (after [72.3]); **(b)** Keepon (after [72.34]); **(c)** Pearl (after [72.35]); **(d)** Valerie (after [72.36])

There are a number of more creature-like social robots that take their aesthetic and behavioral inspiration from animals (Fig. 72.3). Given that people pet and stroke companion animals, touch-based communication has been explored in several of these more animal-inspired robots. Sony's entertainment robot dog, AIBO [72.30, 38], is a well-known commercial example. Other robots in this category have a more organic appearance, such as the therapeutic companion robot seal, Paro [72.31]. Researchers have chosen to design robots with a more fanciful appearance, melding anthropomorphic with animal-like qualities such as Leonardo ([72.32, 33, 39], etc.).

Many social robots are not overtly humanoid or zoomorphic, but still capture key social attributes (Fig. 72.4). For instance, one of the best-known and pioneering social robots Kismet [72.3] developed at the MIT Artificial Intelligence Lab. Kismet had a very expressive mechanical face with anthropomorphic features like large blue eyes. Another example is the dancing robot Keepon developed by NiCT (Japan). This small yellow robot has a simplistic face and uses a classic animation technique called *squash and stretch* for expression of the body [72.34].

Many mobile social robots have been fitted with faces to enhance social interaction (Fig. 72.4). Some examples are the eldercare robot, Pearl [72.35], and the robotic receptionist Valerie with a graphical face on a LCD (liquid crystal display) screen [72.36], both developed at Carnegie Mellon University. Other examples are commercial robots like PaPeRo developed by NEC [72.40]. Still, some social robots have no overt social features like faces or eyes, but rely purely on language-based communication. Issues of proxemics on mobile social robots have also been explored such as how a robot should approach a person [72.41], follow a person [72.42], or maintain appropriate interpersonal distance [72.43].

72.3 Social Robots and Social-Emotional Intelligence

Humans are fundamentally emotional beings. Consequently, human communication and social interaction often includes affective or emotive factors. To support the emotional side of human behavior, researchers are exploring affective interaction and communication between people and robots. To participate in emotion-based interaction, robots must be able to recognize and interpret affective signals from humans, they must possess their own internal models of emotions (often inspired by psychological theories), and they must be able

to communicate this affective state to others. In general, emotional displays can inform the interpretations about an individual's internal states (agreement or disagreement about a belief, valuing a particular outcome, an action tendency to fight, etc.) and therefore help to predict future actions. Emotional displays can evoke emotional responses in others (e.g., displays of distress can elicit feelings if empathy and motive another to provide social support). Given that social interactions can serve such a broad range of functions, prominent scientists have argued that emotions evolved because they provide an adaptive advantage to social species where individual relationships matter [72.44].

A growing number of socio-emotional robots have been designed to realize such functions to facilitate human–robot interaction. Some of these robots have been designed with emotional responses or emotion-inspired decision making systems in order to entertain AIBO [72.30], QRIO [72.45, 46], conversationally engage with WE-II [72.47], or bond with people [72.48, 49]. Some have investigated the social-communicative aspects of emotions in coordinating behavior and influencing others, e.g., FEELIX [72.50], Kismet [72.3, 51]. Others have explored the functional role of emotion-inspired processing in order to make robots more intelligent, better able to learn, and better adapted to performing tasks in complex environments [72.52]. More recently, researchers have investigated the role of affect in the context of robots that work with people and perform tasks such as search and rescue/retrieve [72.16, 53, 54]. Finally, others model emotions to make robots better able to handle human emotional states, and to motivate people toward more effective interactions in a range of application domains such as education [72.55], coaching [72.56], or therapeutic systems [72.31].

72.3.1 Theories of Emotion

The robot's computational model of emotion determines the robot's affective responses. This can depend on a myriad of interrelated physical, cognitive, and affective factors that continuously modulate and bias one another. These factors arise from the robot's interactions with the external environment as well as its own internal state (i. e., the current emotional state, the cognitive state, goals, motives, physical states, etc.). The emotional model defines the relationship between these factors and mechanisms that result in the observable behavior of the robot. Many of these computational models of emotion are inspired by theories of human emotions. These theoretical models offer insightful constraints that help researchers to derive coherent computational models.

A number of theoretical perspectives have been particular influential in the development of computational models of emotion. Appraisal theory emphasizes a causal connection between cognition and emotion [72.57]. In appraisal theory, emotion is evoked from patterns of judgments (called appraisal variables) that characterize the personal significance of events (e.g., events and the individual's beliefs, or desires and intentions). An active area of development is in understanding the relationship between appraisal variables and specific emotion labels, or specific behavioral (i. e., facial expressions) and cognitive responses (i. e., coping strategies) [72.58]. This kind of model lends itself to more symbolic AI (artificial intelligence) implementations, with if-then rules [72.59, 60]. In contract, dimensional theories posit that emotion and other affective phenomena are not discrete entities, but rather exist on a continuum of a continuous dimensional space [72.61]. *Smith* and *Scott* [72.62] proposed a three-dimensional (3-D) *PAD* model where P corresponds to pleasure (valence), A corresponds to arousal (intensity), and D corresponds to dominance (coping potential) [72.62]. Reference [72.63] mapped these appraisal dimensions to intensity varying expressions that can be computed as a weighted blend of basis postures corresponding to the main axes [72.63]. The core affect (the emotional state of the individual at any given time) is represented as a point in the 3-D space that is pushed around by eliciting events. Dimensional models are often used for generating the behavior of animated characters, as the dimensional space lends itself nicely to animation blending.

72.3.2 Example: A Synthesis of Emotion Theories in Action

Kismet is the first autonomous robot explicitly designed to explore socio-emotive face-to-face interactions with people [72.3, 51]. Research with Kismet focused on exploring the origins of social interaction and communication in people, namely that which occurs between caregiver and infant, though extensive computational modeling guided by insights from models of emotion [72.64]. Kismet was designed to be dependent on people to help it satisfy its goals and motivations.

Internally, Kismet's models of emotion interacted intimately with its cognitive systems to make affective appraisals about its interactions with the surrounding environment and people. These appraisals characterized the robot's interaction with its environment (e.g., *Is an object too close to the robot so that it might do damage?*, *Is that person speaking to me in a praising tone of voice?* etc.). These appraisals were tagged

with somatic markers [72.65] that characterized how these appraisals mapped to the robot's internal measures of arousal (*A*), valence (*V*), and stance (*S*), an action tendency to approach or avoid). The affective appraisals directly influenced the robot's behavior selection and goal arbitration processes. The somatic markers influenced the robot's core affective state – a continuously adjusted point within a three dimensional [*A*,*V*,*S*] space. The core affect mapped to the robot's emotive expressions conveyed through vocal quality, facial expressions, and body posture [72.66]. Rather than simply triggering a set of discrete emotive expressions, Kismet's facial and postural expressions were continuously computed following a componential approach, as a weighted blend of basis postures of the face and body along the [*A*,*V*,*S*] axes. This subtle variability in expressive behavior was important for enabling the robot to mutually regulate affective states with a person [72.67, 68]. Finally, the core affect and affective appraisals contributed to elicit adaptive behavioral responses (e.g., orient, search, avoid, etc.) as part of the robot's emotive responses (loosely inspired by the idea of emotion circuits [72.69]). Through a process of behavioral homeostasis [72.70], these emotive responses served to influence how people interacted with the robot in order to restore the robot's internal drives, goals, and affective states [72.67, 68]. VIDEO 557 illustrates Kismet's ability to recognize, express and interact with people using emotive cues. Ultimately, the purpose of this *dance* was to keep the robot in a *zone of proximal development* conceptualized by *Vygotzky* to be optimal for learning to propel the robot down a developmental path [72.71].

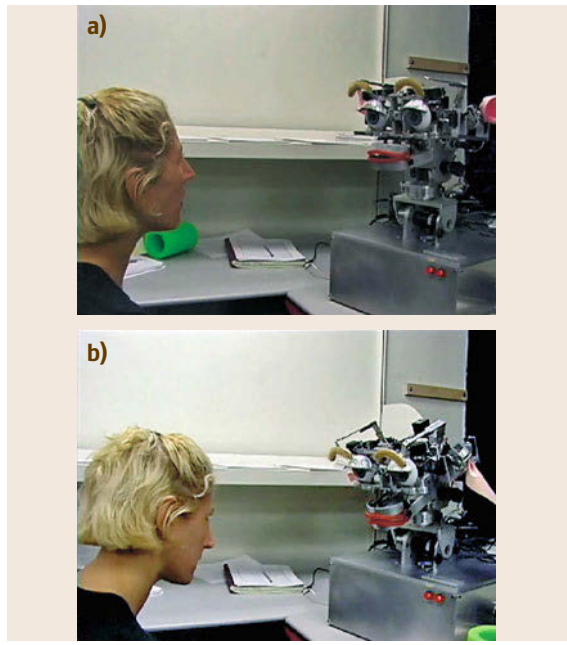


Fig.72.5a,b Kismet and a young woman mirroring affect. Facial expression and affective tone of voice are tightly correlated: **(a)** mirroring interest/arousal; **(b)** mirroring negative affect

72.3.3 Emotional Empathy

For humans, the dynamic coupling of like minds through the actions of similar bodies is critical for acquiring human-like intuitions about the internal states of others. *Dautenhahn* [72.2] is one of the earliest works to explore *empathic* mechanisms of understanding others in social robot–robot interaction.

It is likely that emotional empathy in humans is learned, beginning in infancy. Various experiments with human adults have shown a dual affect–body connection whereby posing one's face into a specific emotive facial expression actually *elicits* the feeling associated with that emotion [72.72, 73]. Hence, imitating the facial expressions of others could cause an infant to feel what the other is feeling, thereby allowing the infant to learn the association of observed emotive expressions of others with the infant's own internal affective states. Other time-locked multimodal cues may facilitate learning this mapping, such as affective speech that accompanies emotive facial expressions during social encounters between caregivers and infants. Using a similar approach, *Breazeal* et al. [72.74] posit that a robot could learn the affective meaning of emotive expressions signaled through another person's facial expressions, body language, and synchronized multimodal cues such as vocal prosody [72.68, 75] (Fig. 72.5). These time-locked multimodal states occur because of the similarity in bodies and body-affect mappings, and they enable the robot to learn to associate its internal affective state with the corresponding observed expression. In later work, they implemented a model of social referencing by which the robot, Leonardo, that combines models of *empathic association* with models of shared attention (Sect. 72.4.1).

72.4 Socio-Cognitive Skills

Socially intelligent robots, however, must understand and interact with animate entities (i. e., people, animals, and other social robots) whose behavior is governed by having a mind and body. In other words, social robots need the ability to recognize, understand, and predict human behavior in terms of the underlying mental states such as beliefs, intents, desires, feelings, etc. Psychology calls this ability *Theory of Mind* (also known as mindreading, mental perception, social commonsense, folk psychology, social understanding, among others).

This section reviews research in implementing models of human socio-cognitive skills and abilities on robots. Social robots shall need a diverse repertoire of such skills to realize their full potential in daily human life – to communicate, cooperate, and learn from people in a human-centric and human compatible manner.

For instance, social robots will need to be aware of people's goals and intentions so that they can appropriately adjust their behavior to help us as our goals and needs change. They will need to be able to flexibly draw their attention to what we currently find of interest so that their behavior can be coordinated and information can be focused about the same thing. They need to realize that perceiving a given situation from different perspectives impacts what we know and believe to be true about it. This will enable them to bring important information to our attention that is not easily accessible to us when we need it. Social robots will need to be deeply aware of our emotions, feelings, and attitudes to be able to prioritize what is the most important thing to do for us according to what pleases us or to what we find to be most urgent, relevant, or significant.

Furthermore, the behavior of social robots will need to adhere to people's expectations. Namely, people will apply their theory of mind to understand the robot in terms of these mental states as well.

72.4.1 Shared Attention

Scassellati [72.76] was one of the earliest works to pose the question of how to endow robots with a theory of other minds. Inspired by the theoretical viewpoints proposed from the study of autism (believed to be a deficit of theory of mind), Scassellati implemented a hybrid model of those models proposed by *Leslie* [72.77] and *Baron-Cohen* [72.78] where shared attention is viewed to be a critical (and missing) precursor to the theory of mind competence. This hybrid model was implemented on the humanoid robot, Cog. The robot was able to exhibit an assortment of social-cognitive skills such as joint attention, distinguishing an entity in the environment as either being animate or inanimate, and imitating only entities deemed to be animate [72.79].

Several researchers have explored models of joint reference, guided by insights provided by developmental psychology and autism research [72.76, 79–81]. Normal human infants first demonstrate the ability to share attention with others at 9 to 12 months of age, such as following the adult's gaze or pointing gestures to the object being referred [72.78, 82]. In these works, joint attention is a learned process. For instance, the robot learns the visual motor mapping from the human's attentional cue (often using head pose as a popular indicator of what the human is currently looking at) to the motor commands necessary to have the robot look at the same thing. This process is often bootstrapped by having the human look to where the robot initiates its gaze. In *Fasel* et al. [72.80], the robot learns a model of joint attention because it discovers that the human's gaze is a reliable indicator of where there is something interesting to look.

Thomaz et al. [72.83] explore attention-monitoring behavior of a robot in a social referencing interaction. In the developmental psychology literature, the ability for babies to actively monitor that others are looking at the same thing is a strong indicator of shared attention [72.84]. Social referencing is considered to be an early demonstration of shared attention because the baby looks back and forth between the novel object and the adult's emotive reaction toward that object to learn the association between the two. To implement shared attention, the robot's attentional state is modeled with two related but distinct foci: the current attentional focus (what is being looked at right now) and the referential focus (the current topic of shared focus, i. e., what communication, activities, etc. are *about*). Furthermore, the robot maintains a model for its own attentional state and a model for the attentional state of the human. The robot uses the heuristic of looking time upon a shared object to infer the referent of the interaction. Once the referent has been identified, the robot monitors the attention of the human in order to associate their emotional reaction about that object to the intended target. Video VIDEO 556 illustrates Leonardo's ability to interact and learn from a person via social referencing.

72.4.2 Mental Perspective Taking

This section explores this empathetic, self-as-simulator approach further to address more general challenges in endowing robots with mental perspective-taking abilities. These approaches are inspired and informed by

theories championed by neuroscience and embodied cognition called *Simulation Theory*.

Simulation Theory

Simulation theory holds that certain parts of the brain have dual use; they are used to not only generate our own behavior and mental states, but also simulate the introceptive states of the other person [72.85]. In other words, we engage in a process of perspective taking and mental simulation.

For instance, *Gallese* and *Goldman* [72.86] proposed that a class of neurons discovered in monkeys (called *mirror neurons*) is a possible neurological mechanism underlying both imitative abilities and Simulation Theory-type prediction of the behavior of others and their mental states. Further, *Meltzoff* and *Decety* [72.87] posit that imitation is the critical link in the story that connects the function of mirror neurons to the development of adult mind-reading skills. From the field of embodied cognition, *Barsalou* et al. [72.88] presents additional evidence from various social embodiment phenomena that when observing an action, people activate some part of their own representation of that action as well as other cognitive states that relate to that action.

Mirror Systems for Recognizing Actions

Inspired by these theories and findings, *Johnson* and *Demiris* [72.89] employ a simulation of visual perception to recreate the visual egocentric sensory space and corresponding egocentric behavioral space of the observed agent to increase the accuracy of action recognition. This approach is based on their HAMMER architecture (hierarchical attentive multiple models for execution and recognition) that takes a mirror neuron inspired approach to action recognition and imitation by directly involving the observer's motor system in the action recognition process. Specifically, during observation of another's actions, all of the observer's inverse models (akin to motor programs) are executed in parallel via simulation using forward models, and then compared to the observed action. The one that matches best is selected as being the recognized action. Perceptual perspective taking is needed to provide meaningful data for comparison. The simulated actions used by the observer during recognition must be generated as though from the point of view of the other person. They demonstrate this approach in an experiment where a robot attributes perceptions and recognizes the actions of a second robot [72.89].

Mental Perspective-Taking for Inferring Beliefs and Goals

Gray et al. [72.90] have implemented computational models of simulation-theoretic mechanisms throughout several systems within Leonardo's cognitive architecture to enable the robot to infer beliefs and goal states of a human collaborator.

The robot reuses its belief-construction systems from the visual perspective of the human to predict the beliefs the human is likely to hold to be true given what he or she can visually observe. This enables the robot to recognize and reason about the beliefs held by a person, even when they diverge from the robot's own beliefs of the same situation.

In psychology, the ability to appreciate the divergent beliefs of another is classically demonstrated by the famous false belief task. In this task, subjects are told a story with pictorial aides that typically proceeds as follows: two children, Sally and Anne, are playing together in a room. Sally places a toy in one of two containers. Sally then leaves the room, and while she is gone, tricky Anne moves the toy into the other container. Sally returns. At this point the human subject is asked *Where will Sally look for the toy*?

The robot, Leonardo, has demonstrated its ability to pass these sorts of false belief tasks where it observes two humans playing the roles of Sally and Anne [72.91]. Within the robot's goal-directed behavior system (where schemas relate preconditions and actions with desired outcomes) motor information is used along with perceptual and other contextual clues (such as hierarchically structured task knowledge) to infer the human's goals and how he or she might be trying to achieve them (i. e., plan recognition).

72.4.3 Perspective Taking in Collaboration and Teamwork

By using a simulation-theoretic methodology, mental inferences made across different cognitive systems can interact in interesting and useful ways to support collaborative behavior where a robot offers its human teammate appropriate assistance.

Using Visual Perspective Taking to Resolve Ambiguous Referents

Trafton et al. [72.92] have developed and implemented visual and spatial perspective taking abilities based on mental simulation to support human–robot interaction and collaboration. Their cognitive architecture, *Polyscheme*, is designed to model how humans integrate multiple representational methods, reasoning, and planning methods to keep track of the world, including rich facilities for representing counterfactual worlds. It thus supports simulations of other people's visual perspective to reason about interactions and the world from this alternate point of view.

They have demonstrated these skills in a number of experiments, such as demonstrating the robot's ability to learn how to play *hide and seek* with a person where the robot learns what makes a good hiding place with respect to being completely occluded from the human seeker's point of view [72.93]. They have also demonstrated the usefulness of this system for a robot that solves a series of perspective-taking problems using the same frames of references and spatial reasoning abilities that astronauts do to facilitate collaborative problem solving – such as repairing a vehicle with another person that has a different vantage point [72.94]. For instance, the robot can handle egocentric requests (i. e., *hand me the cone to my right*), addressee-centric requests (i. e., *hand me the cone to your right*), or object-centered requests (i. e., *hand me the cone in front of the box*).

In *Trafton* et al. [72.92], a human interacts with the robot using a multimodal interface that supports speech and gesture. The robot's perspective taking skills are used to resolve ambiguous referents that can arise when a person asks a robot perform an action in relation to an object (i. e., asking the robot to *hand me the wrench* when there are multiple wrenches to choose from). In particular, a visual occlusion in the workspace might hide another candidate wrench from the person's viewpoint but not from the robot's viewpoint (Fig. 72.6). The robot can infer which is the intended object by taking the visual perspective of the human and applying principles of *joint salience* and *least effort*. If there still remains an ambiguity, the robot can act to resolve it by asking *which one*?

Providing Informational or Instrumental Support

Gray et al. have demonstrated the ability for the Leonardo robot to successfully infer its human partner's beliefs, desires, and intentions from real-time behavior during collaborative tasks. The shared workspace can have either visual occlusions [72.90] or can change dynamically where not all participants know of these changes [72.91]. The robot can integrate these mental state inferences to decide how to best help the person such as offering instrumental support (acting on the environment to help the human complete their goal) or provide informational support (giving relevant information the person needs to successfully achieve his or her goal).

Consider the following scenario: a helpful robot is introduced to two people, Sally and Anne. All three watch as Anne hides chips in a box to the left of the robot and cookies in a box to the right. Sally leaves the room, at which point Anne plays a trick on Sally by swapping the contents of the boxes and then locks both boxes with a combination lock. Anne leaves, and Sally soon returns craving the chips she saw placed in one of the boxes. Sally remembers seeing the chips placed in the left box and attempts to open it by working the combination lock. The robot has matching chips and cookies that it can give out. What should the robot do to assist Sally?

Mindreading skills play an important role in this plan recognition scenario where the robot must observe Sally in real time to infer Sally's misconception of where the chips are (Anne switched the location when Sally was out of the room), to infer what her desire is based on her behavior (Sally never explicitly said she wants the chips), and to recognize that Sally's plan for how to get the chips is actually invalid (she is trying to open the wrong box). The robot has true knowledge of the situation, and must then reason about how to best help Sally get the object of her desire.

Gray et al. [72.91] (VIDEO 563) combines these three kinds of mental inferences to demonstrate intention recognition with divergent beliefs for collaborative robots. Specifically, for the case of informa-

Fig. 72.6 Robonaut using visual perspective taking to disambiguate the intended referent when asked to *hand me the wrench*. The human can only see one wrench, but the robot can see both. The robot correctly hands the wrench that both can see

tional support, Leonardo relates its own beliefs about the state of the shared workspace to those of the human based on the visual perspective of each. If a visual occlusion exists or an event occurs that prevents the human from knowing important information about the workspace, the robot knows to direct the human's attention to bring that information into common ground. For instance, Leonardo points to the box that actually holds the chips. For the case of instrumental support, Leonardo helps the person by directly giving the person a matching bag of chips. *Gray* and *Breazeal* [72.95] explore the robot reasoning and taking explicit action to deceive a human competitor.

72.5 Human Social Responses to Social Robots

72.5.1 Social Judgments

Anthropomorphic design is important for social robots given that the appearance, interface, and function of a technology or product impact how people perceive it, interact with it, and engage with it over time [72.12]. Robots and other technologies with humanlike design cues elicit social responses from people [72.96–99]. For instance, people have been found to respond more positively to artifacts that exhibit humanlike cues such as emotive expressions over those with a purely functional design, although user preferences were task and context dependent [72.100, 101]. Adding humanlike cues to a technological artifact can foster people's social connection to it, aid people in learning how to use it, and enhance liking, engagement, and the desire to collaborate [72.102–104]. Others have found that people tend to hold richer mental models of as anthropomorphic robots than mechanistic ones [72.103, 105]. Others have explored a number of anthropomorphic design features such personality, backstory, use of humor, and even the notions of self (e.g., referring to itself as *I*), deception, politeness, and moral regard [72.106–108]. However, it is important that the appearance and interface of the robot's design match its capabilities and the users' expectations or negative effects can result [72.109].

72.5.2 Physical Versus Virtual Embodiment

When considering the role and advantages of social embodiment, one might ask whether there is a difference between physical and virtual counterparts. Indeed, physical social embodiment offers a number of advantages over purely graphical representations. First, while many social interactions involve exchanging only visual and auditory cues, robots support communication and collaboration through physical contact as well. Robots support the joint manipulation of artifacts and the sharing of physical space with people. Both are important for all sorts of collaborative activities such as assembly and manufacturing [72.110, 111] search and rescue [72.16, 53], domestic assistance [72.103], and more. Further touch is not only an interesting and important communication modality with noted heath and therapeutic benefits, but it can also influence the social judgments people make of robots such as how caring or persuasive it is perceived to be [72.103, 104]. The modality of touch will be discussed further in section social touch [72.31, 39].

In addition, a growing number of studies that directly compare virtual to physical agents report that people show more trust, compliance, and enjoyment with physical robots [72.112–115]. Beyond user preference of physical over virtual agents, a number of studies have also shown improved human performance and outcomes on a wide variety of tasks ranging from games [72.112] to educational contexts [72.115], assistive tasks [72.114], health-related activities [72.116], and Wizard-of-Oz user studies [72.113].

Finally, because virtual agents are a representational form, interpretation and mapping may prove too challenging for individuals with cognitive or social deficits. Hence, robots may prove advantageous as therapeutic interventions for children with autism who can show little or no interest in forms on video monitors or televisions [72.117]. Finally, social robots tend to support face-to-face group dynamics [72.118] – while screens tend to *capture eyeballs* at the risk of diminishing face-to-face interaction [72.119]. This has important considerations for the design of learning technologies for young children, in particular, where supporting the participation of the parent in a social way is of great benefit to how children learn.

72.5.3 Social Stimulus to Learn About People

The impact of physically embodied social robots on human social responses has opened new applications for robots as an interesting tool to help scientists learn about human social behavior and judgments. People make a variety of social judgments through the dynamic exchange of nonverbal cues such as postural

shifts, subtle head gestures, arm gestures, facial cues, tone of voice, etc. Indeed, ample evidence indicates that humans regularly use specific cues, often without conscious awareness, to infer the motivations of others with some level of accuracy [72.120, 121]. If people employ such cues without conscious awareness, then it becomes difficult to use people as experimental confederates where such cues must be carefully controlled. Thus, in one paradigm, social robots can be used as a highly controllable social stimulus in the place of a confederate in human participant studies. This is particularly useful when trying to understand how human nonverbal cues influence people's social judgments, such as how trustworthy another is perceived to be from a brief encounter [72.122].

72.5.4 Social Rapport

One important skill for social robots is the ability to build and maintain social rapport with its user. Social rapport exists in the interaction between individuals. It creates a powerful interpersonal influence and responsiveness based on mutual attentiveness, positivity, and mutual responsiveness [72.123–126]. In joint activities, the ability to establish a good rapport often results in improved outcomes. For instance, students learn better when they have a good rapport with their teachers [72.127], patients are more adherent and have better health outcomes when they have a good rapport with their doctors [72.124], teams make better decisions and work more effectively when they share a good rapport with each other [72.128]. The quality of rapport between people is influenced by the exchange of nonverbal behaviors between individuals [72.129, 130]. Although the specifics of how people build a strong rapport with one another are still a topic of scientific inquiry, the more in-sync and open the participants' nonverbal cues are in relation to one another, the more positive rapport results. For instance, appropriate mirroring or synchrony of body posture, head movements, facial expression, and vocal prosody can all contribute to positive rapport. Open cues signal a receptiveness to interact, e.g., making appropriate eye contact, leaning toward another, and arm gestures that tend to not occlude the body or face.

Hence, when considering how social robots can effectively collaborate with people, their ability to build and maintain good social rapport with people (at least perceived rapport) is important. Thus, it is not just a robot's ability to perform such nonverbal cues that ultimately matters, but the robot's ability to coordinate them with those of people in real-time. For instance, researchers have explored the coordination of an agent's nonverbal cues with people to improve rapport [72.56], engagement [72.32], trustworthiness [72.122], and deictic cues [72.131].

72.5.5 Social Support

Improved rapport also facilitates the ability of social robots to provide people with effective forms of social support. Social support is recognized to play an effective role in helping people to attain personal goals and improved outcomes in broad domains such as education, mental health, physical health, aging, coping, and more. It is conceptualized as the perception and actuality that one is cared for, has assistance available from others, and that one is part of a supportive social network [72.123]. These supportive sources can be emotional, instrumental, informational, or companionship. Emotional support, for instance, conveys to an individual know that they he or she is valued; it includes offering empathy, concern, nurturance, encouragement, and acceptance to name a few [72.124, 125]. Instrumental support concerns the concrete, utilitarian ways to provide assistance such as financial assistance, material goods, or services. Informational support includes offering useful advice, guidance, or information to help others problem-solve. Finally, companionship support gives someone a sense of social belonging and having another to participate in shared social activities. These forms of social support can come from people, professionals, pets, and even social robots.

The ability to provide a user with social support is one of the effective ways that social robots can help people through social means. Robots can provide this assistance through direct interaction with its user, or by helping to mediate the provision of social support from people (e.g., connecting people). A wide variety of social robots today are being developed to interact with people as tutors [72.132], learning companions [72.118, 133], coaches [72.116], domestic helpers for the elderly [72.134], therapeutic aids [72.21, 31], and more. Through dialog, nonverbal cues, expressive displays, and physical actions, these robots assist people by providing information, monitoring performance, offering feedback, incentivizing and sustaining motivation, giving encouragement, offering companionship, performing physical tasks, etc.

As such, social robots have broad applicability in many domains where it is a technology that can extend and augment the social support provided by people. This is particularly relevant for societal challenges, e.g., eldercare, health and chronic disease management, and education, where social support is recognized as being critical for positive outcomes, but where there is a rec-

ognized shortage of trained professionals to meet the demand. Further, whereas frequent meetings with human professionals is cost prohibitive, social robots have the potential to be used to *fill the gaps* in a cost-effective way. Importantly, social robots are not being designed to replace or obviate human professionals, but rather to serve an effective tool that supports human networks in a scalable and cost-effective way.

72.6 Social Robots and Communication Skills

Communication implies an exchange of information through natural language. Thus, one might consider that very good ASR (automatic spoken-language recognition) is the primary function required to fulfill the communication skills for robots. However, *social* robots are expected to engage in casual communication with people in as natural a way as people communicate themselves. In such casual communication, information is often exchanged nonverbally as well as verbally. Thus, robots need to be well equipped to recognize people's nonverbal cues and to express nonverbal cues via their nonverbal behavior. The required computation even takes into account good perceptual and cognitive capability of surrounding environments in addition to the targeted person. This section introduces the history of research in social robots and communication skills, and provides insights for future challenges.

72.6.1 Verbal/Nonverbal Communication

Historically, even first-generation humanoid robots developed in the 1970s had primitive capabilities for communicating using natural language. For instance, WABOT and WABOT-2 had a conversation capability in natural language, which is based on the model as simple combinations of speech input/output mappings [72.135, 136].

Furthermore, early pioneers had noticed the importance of nonverbal information in human conversation. Nonverbal behavior was classified into three roles:

1. Regulators: expressions such as gaze, poses, and vocalizations that are used to regulate/control conversational turn-taking.
2. State displays: indication of internal state including affect, cognitive, or conversational states that improve interface transparency.
3. Illustrators: gestures that supplement information for the utterance. These include deictic gestures (pointing) and iconic gestures.

In this scope, nonverbal information is considered as supplemental. It is natural language that communicates the primary information in turn-based exchanges. We provide examples below.

Regulatory Cues

Even some of the earliest social robots displayed nonverbal information to regulate interactions with people. Hadaly 2 was the first robot to use mutual gaze as a nonverbal cue to regulate conversation [72.137, 138]. The mutual gaze is approximated using face recognition to determine when the human's face was facing the robot; when a mutual gaze occurred, Hadaly 2 expressed readiness to commence conversation by blinking its eyes. People's gaze toward a robot is also considered cues to inform whether he/she is engaging in the conversation with the robot [72.139].

Other examples are Kismet [72.3, 140, 141] and Leonardo [72.33, 142], which had the capability for nonverbal cues called envelope displays to regulate the exchange of speaking turns. Backchannelling cues were found to reduce stress and cognitive load during complex human–robot teaming task (Fig. 72.7). See VIDEO 559 for how Kismet uses envelope displays to regulate speaking turns with people. The regulatory role for a speaking turn is demonstrated in multi-party interaction too. People tend to make eye contact and raise their eyebrows when they are ready to relinquish their speaking turn and tend to break their gaze and blink when starting their speaking turn. Recognition of these cues was implemented for smoothing and syn-

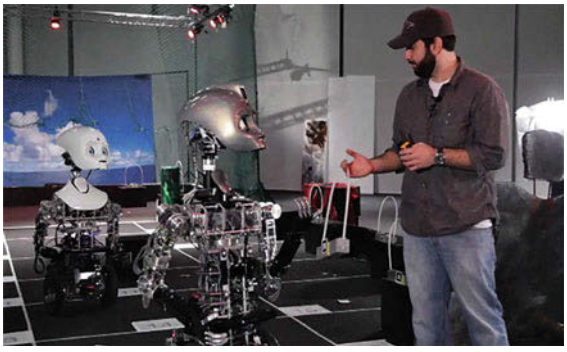


Fig. 72.7 Backchannelling cues were found to reduce stress and cognitive load during complex human–robot teaming task. Teams where the robots engaged in backchannel cues to human requests also tended to find more items in a search and retrieve task (after [72.16]), VIDEO 555

chronizing the exchange of speaking turns. Gaze is also known to convey the speaker's intention who may take the next turn. *Mutlu* et al. have successfully replicated it in human–robot interaction [72.143, 144]. *Kirchner* and *Alempijevic* revealed that gaze is also effective in communicating who should receive an item provided by a robot [72.145]. In opposite way, gaze was also used as a cue to interpret whether one would wish to take turn [72.146].

Paralinguistic information is also processed. People frequently provide short acknowledgment utterances (e.g., *uh-huh*, *um-hmm*, *huh*, etc.) as the robot explains something. These responses are either acknowledgments, or ask-backs. It is very difficult to distinguish these two kinds of utterances from only the linguistic information as represented by the transcription of the utterance. *Fujie* et al. demonstrated a method to distinguish the utterance as either an acknowledgment or an ask-back from the prosody of utterance [72.47].

State–Display Cues

The facial expression and gaze were used to indicate the conversational or cognitive state of a robot. Such state-display cues make the robot's internal state more transparent to a user and thus enables him/her to better understand the robot's state. For instance, ROBITA used the tightness of its facial expression to indicate readiness to engage in conversation; a tight face was used to express conversational readiness, while a loose face communicated a lack of readiness to engage [72.147]. Emotional state and attention target can also be displayed nonverbally (Sects. 72.3 and 72.4).

State of listening and level of understanding are also displayed nonverbally. Human listeners use back-channel responses, such as head nods to convey the fact that he/she is successfully following the conversation. Imitating human behavior, robots indicate their state of listening using head nods [72.148], facial expressions [72.5], and bodily motion ([72.149] and VIDEO 810). Such state-display cues are used in a human-like telepresence robots to indicate operators' state of being engaging to the conversation [72.150, 151].

Another back-channel signal is an expression of confusion by the listener (verbal or nonverbal). This flags the speaker to stop and try to repair the broken communication. Robots such as Leonardo and ROBITA use facial displays of confusion when speech recognition fails in order to intuitively communicate to the human that he or she should repeat their last utterance.

There were robots that process humans' back channel feedback [72.32] A sophisticated head nod recognition system was developed whereby the robot, Mel, could successfully distinguish small feedback nods from other kinds of head nods such as those that communicate agreement. Mel used this information to determine its own nodding behavior in order to be an appropriate response for the human. In a series of human subject studies, *Sidner* et al. found these nonverbal cues to enhance the social engagement of the robot to people [72.32].

Illustrator Cues

Deictic gestures have often been implemented in robots for pointing to an object, such as using index-finger pointing [72.152, 153], gaze [72.143], and the combination of the two [72.74, 154–157]. Other types of gestures, such as iconic gestures [72.156] and region pointing ([72.158] and VIDEO 811), were also successfully used in robots. The effect of gestures has been successfully demonstrated. For instance, in a direction giving scenario, even though turn-by-turn direction is verbally given thus could be comprehend without gesture, supplemental pointing gesture improved listeners' comprehension about the given direction [72.159].

A number of robots are able to recognize deictic gestures of a person conveyed either through pointing gestures or head poses. For example, Leonardo is able to infer the object referent in an interaction by considering a number of factors including pointing gesture, head pose, and speech. *Sugiyama* et al. associated verbal spatial deixis and pointing gestures to better recognize the pointed target ([72.160] and VIDEO 807). *Brooks* and *Breazeal* [72.153] developed a deictic recognition system that enabled a robot to infer the correct object referent from correlated speech and deictic gesture. Interestingly, it was found that the accuracy of the human's pointing gesture is surprisingly poor. As a result, the deictic recognition system relies on coordinated speech and gesture information, with spatial knowledge provided by a three-dimensional (3-D) spatial database constructed by the robot using real-time vision, and a deictic spatial reasoning system. This system was successfully demonstrated on the dexterous humanoid Robonaut developed at National Aeronautics and Space Administration (NASA) Johnson Space Center (JSC) (Fig. 72.8b) where the human points to and labels a set of four bolts on a wheel to be fastened in order by the robot.

72.6.2 Mechanisms for Human–Robot Communication

Mechanisms of turn-based communication are well studied. In linguistics, it is considered that conversation is formed as the repetition of turn-taking [72.161].

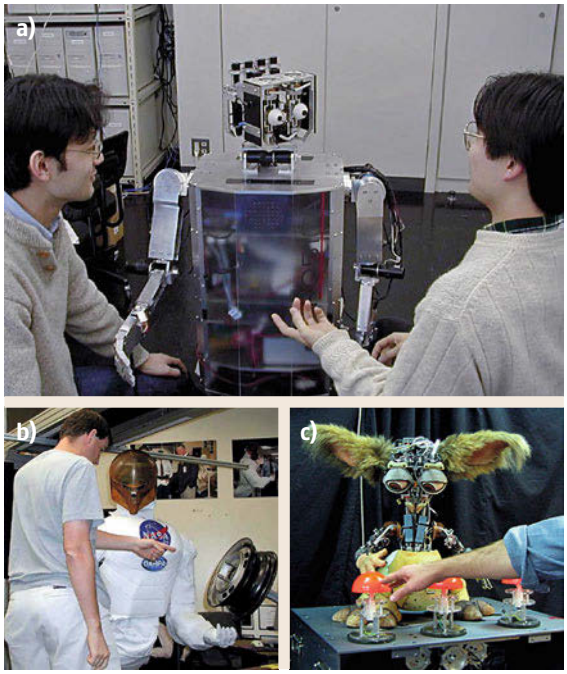


Fig.72.8a–c Examples of conversational robots: **(a)** ROBITA performing group conversation; **(b)** Robonaut interpreting the pointing gestures of a human to determine which nut to fasten on the wheel; **(c)** Leonardo uses gaze and joint attention to ground the human's pointing gesture for the desired referent

There is one speaker who takes floor and speaks while listeners listen to him/her [72.162]. When the speaker finishes speaking, then the floor is taken by one of the listeners. There could be bystanders who also listen to, but do not intend to take turn. This turn-based model is the typical model used in dialog modeling [72.163]. That is, a dialog management system identifies who owns floor, recognizes words in utterances the speaker spoke, and generate utterance when the system takes floor, often implemented with series of rules. It is successfully extended for human–robot communication. For instance, *Nakano* et al. developed rule-based architecture in which the planner deals with robot's task-based actions as well as dialog management [72.164]. *Scheutz* et al. developed software architecture, named DIARC. It uses a rule-based planner that receives inputs from all perception modules, and addresses effect and goal-directed actions in addition to natural language dialog management [72.165]. In such an approach, the robot communicates through utterances accompanied with nonverbal cues [72.166]. Some systems deal with multi-party dialog, in which robot's gaze-cue (Fig. 72.8a) is used to regulate who is the addressee, the active listener who is expected to take the next turn [72.143, 167–171].

Researchers have also been well aware of the importance of the time-sensitive nature of communication. For instance, interruption in the middle of a speaker's turn has also recently been taken into account in dialog management systems [72.164, 172]. *Chao* and *Thomaz*, proposed more elaborated model, in which time-synchrony in verbal and nonverbal cues, e.g., gaze and gesture, are addressed using a Petri-net-based representation [72.173]. Empirical studies have revealed what is good synchrony and timing. For instance, *Yamamoto* and *Watanabe*, have studied synchrony within a robot's utterance and motion, and revealed that people prefer the robot whose utterance is slightly delayed from the start of its motion [72.174]. This would mirror what humans do everyday, as it is reported that humans' gestures are performed slightly ahead than their utterances [72.175]. *Shiwa* et al. revealed that people prefer small delay in the robot's response, and conversational filler such as *etto* would be useful to buy time when the robot's response is delayed ([72.176] and VIDEO 806).

However, while above studies are under the assumption that information is communicated through turn-based dialog, recent studies have revealed more dynamic cases of human–robot communication, in which the way robots communicate information is sometimes out from the turn-based dialog paradigm. For instance, during the moment a robot and a person are going to initiate interaction both of them communicate their intention that they would like to meet and talk. A couple of studies investigated a way for a robot to express its intention to welcome the initiation of interaction [72.177]. When the target user is seated, Dautenhahn et al. revealed that they prefer the robot approach from side ([72.178] and VIDEO 258). Figure 72.9 shows a scene in which the robot approached to pedestrians. Satake et al. revealed that such interaction failed if the robot failed to communicate its intention to talk. As the robot was operated in noisy shopping mall, when it used only verbal utterance, the robot was simply ignored. Instead, they found that the robot needs to communicate its intention nonverbally. It needs to approach from a frontal direction and needs to be responsive in adjusting its body orientation toward the targeted person; such nonverbal behavior made the robot more success in initiating conversation with pedestrians [72.179].

Pedestrians also communicate their intention nonverbally; for instance, when he/she does not wish to talk to the robot, they avoid approaching to the robot. Thus, a robot could use the proximity information to estimate people's willingness in initiating interac-

Fig. 72.9 A robot that approaches pedestrians

tion. *Michalowski* et al. classified spatial zones around a robot, and let the robot talk to the person if he/she approached to the robot and entered to social distance [72.177]. There are some empirical studies about proxemics that revealed that people often prefer to stay at social distance when they talk to a robot [72.180–182].

Joint attention (as introduced in earlier Sect. 72.4) can be silent. Figure 72.10 and VIDEO 257 show a scene where a person share his attention target nonverbally with the robot. The person is curious about the computer and stands in front of it; then, the robot moves to the location convenient to explain the object. When the person moves to the other exhibit, the robot follows him and explains the one in front of him. Here, it is their standing position that communicates the target of attention [72.183, 184].

These examples show the cases where communication is not necessarily turn-based. This is because information can be exchanged nonverbally in such casual communication. Unlike the speech channel that needs to be typically occupied by only one speaker (speaking person), nonverbal signals can be mutually exchanged at the same time. For instance, when a pedestrian and a robot meet, their positions continuously change while they walk which communicates whether they would like to initiate conversation. As social robots aim to operate in people's daily environment, such a casual and nonverbal exchange of information is not a trivial part of human–robot communication. The research for fully unveiling required mechanism for such continuous exchange of social signals is still premature.

On the other hand, some studies started to highlight the needs of a mechanism that connects communication and background knowledge. It is revealed that the model of common ground makes daily communication effective [72.185]. For example, a direction-giving scenario would be a case where environmental knowledge is useful. For instance, when a robot provides directions (Fig. 72.11 and VIDEO 259), it could be more comprehensive if the robot is aware of a visible landmark so that it could say *please turn at the book store* instead of saying *please turn at the third corner* [72.186]. Thus, a good model of environment, e.g., [72.187], would greatly improve a robot's capability in communicating about a route. Moreover, if a robot has a model of users' memory of location, it could provide destination-based direction, such as *a café is nearby the book store you just visited* [72.188], which is much easier to comprehend than complex turn-by-turn destination like *to go to a café, please turn right at first corner, turn left at third corner, and* ...

Fig. 72.10 A person and robot implicitly share the target of attention via spatial formation

Fig. 72.11 A robot that provides directions (after [72.189])

72.6.3 Challenges

To summarize, turn-based conversation has been studied to a certain degree. However, social robots often engage in casual communication in which robots would need to deal with social signals and background knowledge. Here, we denote a couple of key challenges.

Revealing Repertory of Communication Skills

Previous studies started to reveal different forms of communication where various perceptual and cognitive capabilities are required. Early studies have revealed the importance of social signal processing, like recognizing and expressing gaze and facial expressions. Studies about initiation of conversation have revealed the mutual real-time exchange of information via their positions. Studies about giving direction demonstrated further needs of associating language and environmental knowledge, e.g., a *cognitive map*, how people perceive environments, and remembering landmarks.

To what extent do we need to cover repertories? We believe that there could be many elements we will find. Many of them might be interrelated with perception and cognition. Such a case would improve our understanding about what communication skills truly are. Thus, one of the important challenges is here.

Architecture for Communication Skills

Along with finding out the repertory of communication skills, the other important challenge is to deal with the integration, i. e., the architecture of communication skill. Moving from a turn-based structure to a dynamic structure is the real challenge.

72.7 Long-Term Interaction with Robot Companions

A number of social robots have been introduced as companion robots. Companion robots may have the sole purpose of providing companionship, e.g., in studies using toy robots such as the Pleo [72.190], but often they combine two aspects: *being useful*, i. e., being able to carry out certain tasks for the user, and carrying those tasks out in a manner that is socially acceptable [72.191]. The latter notion has been used, e.g., in several European projects [72.192–195]. The notion of a companion often entails repeated and long-term interactions. This poses particular challenges not only for the design of the robot, the interaction design, the choices of tasks/settings/scenarios, but also on the how to ensure satisfactory and successful interaction with the robot.

A number of tele-presence robots have been developed, but this chapter will focus on autonomous or partially tele-operated companion robots. Note, Cody targets the domain of patient hygiene care that is related to issues discussed in the section on tactile human–robot interaction. Regardless of whether human–robot interaction with companions include tactile interaction, or is a hands-off approach (Chapt. 73), developing such system involves a careful study of the roles and functions of such robots in the application domain, and of users' perception of and attitudes toward such a system.

Figure 72.12 and VIDEO 218 show the Care-o-bot robot (Fraunhofer) used in the ACCOMPANY project on home assistance for elderly people. The robot is shown in the University of Hertfordshire's Robot House, a domestic setting for experiments into robot home assistance where participants regularly visit for human–robot interaction sessions. Such *living lab* settings are increasingly being used to design, experiment and evaluate innovative systems (e.g., the European Network of Living Labs, [72.196]). Providing such environments facilitates progress toward complex robotic systems that can be deployed in real-life environments.

72.7.1 Robot Companions

Recently more and more researchers are moving toward studying application areas that develop such companion robots, e.g., the use of robots in assistive and rehabilitation robotics. The use of robots to assist elderly users in their homes, with a view to extend the period they can stay and live independently in their own homes, is currently being studied extensively by different research groups worldwide. Several companies are also marketing their robots as such assistive companions, e.g., the Wakamaru robot (Mitsubishi Heavy Industries Ltd) that was introduced in 2005, or the Human Support Robot (Toyota) revealed in 2012. In addition, numerous research prototypes exist, e.g., Cody (Georgia Tech), Herb (CMU), Care-O-bot 3 (Fraunhofer), Hector (CompanionAble project).

72.7.2 Engagement and Long-Term Relationships

The concept of a robot companion entails repeated, long-term interactions, which may also afford the development of *relationships* between robots and people. Relationships with robots can take many shapes and forms, and the formation of relationships will be influenced by many factors. For example, the specific

role of the robot (Butler? Friend? Tutor? Assistant? Tool? etc., [72.197]) will matter, as well as the robots' specific embodiment and behavioral and expressive repertoire that is often used to present them as *relational artifacts* [72.198, 199] that are designed to build and maintain social-emotional relationships with users.

Many applications with social robots do only involve short-term interactions, e.g., robots meant to function as a museum guide [72.200] and VIDEO 808, a receptionist, etc. Here, the *novelty effect* can often be exploited, i. e., the general interest that many people have in new robots. However, after repeated interactions this effect can wear off and people can lose interest in a robot. Thus, a main research challenge is how to keep people engaged in the interaction and keep them motivated to interact with the robot. Establishing a useful and enjoyable relationship with a robot is not an easy task. And what exactly is *long-term*? *Tanaka* et al. [72.22] mentions at *10-hour barrier* (total interaction time), *Sung* et al. [72.201] pose the goal of long-term interaction over more than 3 months as a main challenge. *Hüttenrauch* and *Severinson-Eklundh* [72.202] performed a user study with the service robot CERO meant to assist one user over three months in an office environment. *Kanda* et al. [72.189] and VIDEO 809 investigated a partially tele-operated robot over a 25-day period. The nature of these interactions depends on whether studies take place in a school, a home, a public place, etc. It will also depend on the frequency and duration of the interaction, e.g., how many interaction episodes per time interval does an individual have with the robot?

There is also the issue of quantity and quality of interaction that may be realized very differently in different application areas of social robots. More and more studies are bringing robots into the *wild* [72.203]. VIDEO 564 shows how preschool age children interacting with a storytelling learning companion robot during repeated encounters over a 2 month period. Personalization of the robot stories to the children led to improved vocabulary learning. Field studies are often said to be preferable, and more ecologically valid than data collected in the laboratory. However, practical, technical, and methodological constraints may limit the length and nature of the field studies. Also, bringing a robot in the *wild* does not necessarily make the interactions with people more natural, and the more *messy* conditions pose hard methodological problems and can often interfere with controlled data collection and statistical data analysis [72.204]. A field study by *Heylen* et al. [72.204] placed a simple dialogue Nabaztag (rabbit) robot in a few people's homes over 10 days, and it illuminates a number of those real-life problems with field studies which impacts on the validity of the results that can be gained and on the user experience. The latter will then ultimately decide on whether people will consider a long-term companion robot *amusing or a nuisance* [72.204]. A taxonomy to characterize studies in the *wild* for child-robot interactions is provided in *Salter* et al. [72.205]. Systematic analyses of different experimental conditions and tasks, settings, etc., whether in the laboratory or in field studies, will help the planning, design and comparison of different long-term studies.

Note, often the expectations of users are not met by the robot's design and abilities, and nor does it fit into daily activities of users [72.190]. The advancement of robot technology and knowledge of HRI will enable more and more field studies, in schools [72.206], nurseries [72.22], private homes [72.190], care homes [72.31], etc.

In many of the above long-term studies, commercially available robots have been used. The reasons for this are not only availability, but also robustness and reliability, as well as safety. The use research prototypes in field studies should always entail securing Institutional Review Board (IRB) approval to meet ethical guidelines when human participants are involved. Even prototypes that are generally safe to use, may still have *cables sticking out*, etc. Hence often it is a necessity to have researchers present at all times during these studies. Research prototypes are also by their very nature more prone to break down, and again this requires the constant attention of researchers. The design of research prototypes is also often not ideally suited for a field study, e.g., wheels when being used on cluttered surfaces, etc. Interesting insights on those *practical* but nevertheless very important points can be found, e.g., in *Hüttenrauch* et al. [72.207] where a modified Peoplebot (Mobile Robotics) robot was brought into eight different homes for approximately 1 hour each in order to study spatial management in HRI situations. Note, even when leaving a robot alone with a person or family in their own home, people may still be acutely aware of the *experimental* nature of the interaction [72.204].

Establishing and maintaining relationships between users and robots in long-term interactions needs careful consideration, and many insights can also be gained from related long-term studies with virtual agents, e.g., as exercise advisors [72.208, 209].

72.7.3 Robot Home Companions for Supporting Elderly Users

Several HRI studies in the home focus on the use of a companion robot for the general population. *Koay* et al. [72.210] report on a long-term, 5-week study

in a domestic environment with a partially remote-controlled robot involving several different scenarios (e.g., a collaborative task, sharing physical space, recording and revealing personal information, interrupting a person to serve them, and seeking assistance from a person using combinations of physical and verbal cues). The results highlight concerns and expectations that people have in such scenarios that are relevant for the domestic use of robots. Note, the robot used in this study was partially remotely controlled, which allowed the investigation of complex scenarios.

Due to demographic changes, a lot of interest exists worldwide into developing robot technology for the care of elderly people that would allow elderly people to remain in their own homes for longer, or to provide assistance in sheltered or otherwise specifically designed environments. A robot may assist an elderly person in the home in different ways, e.g., provide physical assistance (standing up, walking, fetching and carry objects, etc.), social assistance (e.g., engaging the user in interactions with other people) and cognitive assistance (e.g., reminder functions).

In HRI, there is prior work investigating robot's ability to remind users of future or prior activities. Autominder using the Nursebot represents an initial attempt to make reminders more intelligent and dynamic [72.211]. Intelligence and dynamics were also key aspects of the system proposed by the Robocare project [72.212]. More recently, the KSERA project [72.213] focused on the ability of the robot to draw on information not readily available to the user as well as its ability to persuade in order to safeguard the health of the user. Other examples for assistance and reminders for elderly people with dementia through multimodal interactions and smart-house integration include the EU FP7 CompanionAble project and the MOBISERV project [72.214]. The project, Florence [72.215], introduces a commercially available robot as an autonomous lifestyle device for ambient assisted living; it provided multiple services to users including an agenda reminder application that allowed the elderly to share information with caregivers, etc. The European projects SRS [72.216] and ACCOMPANY [72.217] both use the care-o-bot 3 (Fraunhofer) (Fig. 72.12) as their target robotic platform. While SRS involves the use of the robot in a remote-control scenario, the ACCOMPANY project develops fully autonomous behaviors for an empathic and assistive home companion.

Several projects combine a robotic companion with an intelligent smart/ambient environment as part a variety of ambient assisted living (AAL) solutions to assist with older adults, or people with disabilities, to live independently by providing support in activities of daily living (ADLs). Such topics are currently being studied world-wide. See, e.g., the quality of life technology (QOLT) [72.218] Centre of Carnegie Mellon University or the Centre for Affective Solutions for Ambient Living Awareness (CASALA) [72.219].

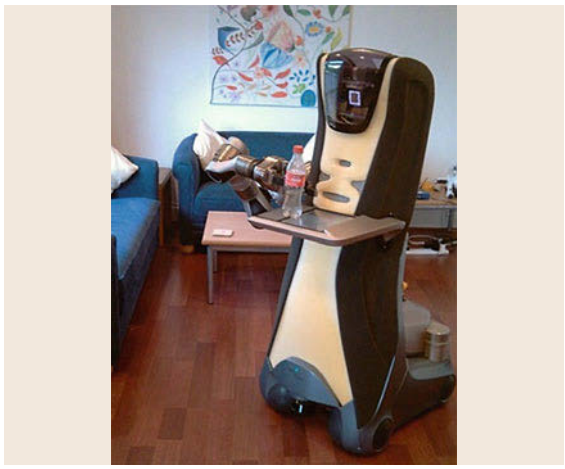


Fig. 72.12 Care-O-bot 3 robot (courtesy of Fraunhofer)

72.7.4 Example: Long-Term Interactions with a Robotic Weight Loss Coach

A number of recent studies investigate the use of robotic companion as health *coaches* or *advisors*. Kidd and Breazeal developed Autom [72.116, 220] (VIDEO 558), a robot specifically designed for long-term interaction with people in the role of a robotic weight loss coach. Autom has a clear function and role, but also needs to interact with users in a socially acceptable and comfortable manner, so that users intuitively understand the robot, are willing to engage with the robot, and *listen* to it. This robot is also an example how academic research in long-term HRI can lead to new innovations and commercialization. The main purpose of the robot is to help the user to lose weight, trying to persuade the user to change his or her behavior [72.221].

A key ingredient supporting this functionality is the creation of a relationship between the user and the robot. Autom is thus an example of a robot that presents itself as a relational artifact and encourages people to develop social-emotional relationships with them [72.198, 199]. Autom uses a psychologically inspired relationship model, with the robot's role as a caregiver. It engages the use in a dialog modeled after patient-care professional dialog to provide social support (Sects. 72.5.4 and 72.5.5) in order to build a working alliance as well as to be helpful, persuasive, positive, and supportive. For a health care coaching robot, where behavior change is a central objective,

such a relationships is very important. People who need to lose weight typically do not lack the intelligence or knowledge necessary to understand that weight loss would improve their health and general well-being. The motivational factor is often a crucial point, so a weight loss robot's role on providing social support is different from that of a kiosk robot only needs to provide information.

A long-term study with 45 participants an Autom prototype robot over 6 weeks was the first study designed to create behavior change in people with a weight management goal (Fig. 72.13). The results were very encouraging: participants developed a close relationship with the robot, and they tracked their calorie consumption and exercise much longer when using the robot, compared to other methods (a computer running the same dialog or a paper log for manual entry). Since these factors are indicators of longer term weight loss success, the study provided evidence for the effectiveness of sociable robots for long-term HRI [72.116].

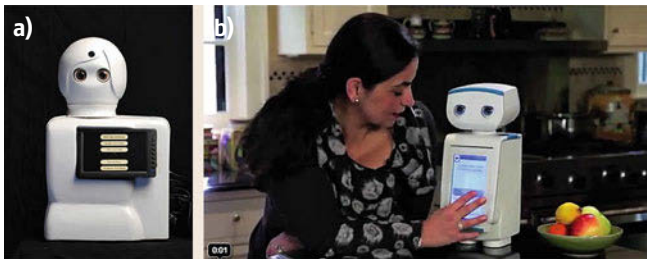


Fig.72.13a,b Autom, a weight management and exercise coach. **(a)** The Autom prototype robot developed at MIT; **(b)** the commercial version of the Autom robot

72.7.5 Challenges

Many of the projects studying the use of robot companions (e.g., supporting elderly users, providing assistance in therapy, or helping in office environments) are still at an initial stage. Future results from extensive, long-term, multi-site and even cross-cultural evaluation studies are still needed in order to illuminate the usability, usefulness, and acceptability of such systems. It is particularly advantageous if the same research platform is used in different projects so that direct comparisons and the sharing of research developments are possible.

The challenges of supporting engagement in long-term studies are beginning to emerge. New sensors and interfaces, such as brain–computer interfaces, could provide richer information on people's emotional, attentional engagement states when interacting with a robot. Robots are providing richer social cues to explore the whole spectrum of human–human interaction modalities, e.g., using gaze, gesture, proxemics, dialogue, contingency, etc. [72.177, 222–226]. Measuring engagement in real time and allowing a robot to respond to it is a key challenge. In a user study with 37 participants and a robot that could perceive user engagement *Sidner* et al. [72.32] found that engagement gestures were perceived more positively than a robot without such gestures. *Rich* et al. [72.139] proposed an initial computational model for a robot to model engagement, based on the recognition of different events involving gesture and speech.

A clear framework of what engagement means in HRI in general, and for long-term HRI interactions in particular, is needed and will needs to be validated in extensive testing. Several other proof of concept studies of robots that adapt to user engagement and interaction show encouraging results. *François* et al. [72.227] evaluated a robot that adapts to user interaction in a therapeutic context based on tactile information (how children touched the robot), while *Szafir* and *Mutlu* [72.228] demonstrated how a robot can adapt to the engagement of the user, utilizing techniques from brain–computer interfaces in order to assess engagement. Future applications of this research into adaptive robots target educational settings where a teacher robot can automatically adapt to the pupils' engagement, or therapeutic/rehabilitation applications where the robot can automatically adapt to the patient's needs and preferences in light of therapeutic objectives.

Future research needs to illuminate how best to design interaction and embodiment of social robots that are successful in long-term interactions with people. Since adding social interaction skills to a robot is costly, the identification of a set of robot characteristics (appearance, behavior, cognition) that are necessary and sufficient to create meaningful, acceptable, and efficient long-term interaction with a robot would be beneficial. Here, it is important to provide frameworks that connect to empirical methodologies, cf. the discussion of a conceptual and methodology framework for robot believability in *Rose* et al. [72.229]. Note, while predominantly humanoid or zoomorphic robots are currently being used in long-term HRI experiments, even mechanically looking robots such as the Roomba robot may invite people to develop a social-emotional relationship with its users [72.201, 230].

Using robots in long-term repeated interactions with people also involves a number of ethical issues, as it has already been highlighted also for embodied conversational agents and virtual characters [72.208]. While in some applications the robot may have a useful function if portrayed as a care-receiver [72.133], in most applications the companion robot should provide

care to its users [72.191]. As pointed out by *Turkle* et al. [72.198, 199], robots that present themselves as relational artifacts may influence people's understanding and expectations of the nature of social relationships and friendship. Ethical issues are particularly important when users are vulnerable people such as adults with special needs, elderly people, or children. Many researchers have commented on these issues [72.231, 232].

Practical issues can also become crucial to the acceptance of a robot for long-term interaction, as it has been highlighted in a long-term study with Pleo – a commercially available robot, but which requires maintenance that does not fit easily into people's routines [72.190]. Fernaeus et al. also bring up the important issue of interaction design that needs to support long-term interaction, an issue that was also brought up in user feedback from the participants in their long-term study. Novel approaches to how to design human–robot interaction may be required to facilitate long-term use of companion robots [72.190], and in real-world settings attention must be paid to how the robot technology is being introduced to people and how people and robots must adapt to each other [72.230].

Recently, the blending of virtual and robotic characters area a growing area of research whereby seamless transitions from a virtual to and from a robotic character [72.233], or migration between different robot and other digital embodiments [72.234] pose interesting challenges for future long-term companions. Indeed, it may change the nature of what we usually perceive as a *companion robot* [72.148, 233, 235–242].

72.8 Tactile Interaction with Social Robots

In robotics and artificial intelligence research, tactile interaction has long been exploited primarily as a necessity to enable, e.g., collision detection, grasping and object manipulation, particularly in combination with vision and other sensor modalities, and led to adding touch sensors to a robot's gripper or hand [72.243].

72.8.1 Touch for Social Interaction

However, the importance of *human–robot* tactile interaction has been highlighted recently in a number of research projects, inspired by evidence from human–human interaction and child development. The recent interest in the *sense of touch* goes beyond the necessity of touch for interactions with the physical environment, but focuses on the important role of tactile interaction in interactions with the social environment. For instance, *Siegel* et al. [72.244] found that social touch, such as shaking hands, can impact how persuasive a robot is perceived to be [72.244]. In other work, the same kind of touch can have a different impact on people's response to the robot depending on the context [72.245].

Indeed, humans are born as *tactile creatures*. Physical touch is one of the most basic forms of human communication. In human development, touch plays a crucial role in developing cognitive, social, and emotional skills, as well as establishing and maintaining attachment and social relationships. Deprivation of touch in early child development can have devastating effects on a child's development [72.246]. A comprehensive survey on communicative functions of touch in humans and other animals can be found in *Hertenstein* [72.247].

72.8.2 Touch Sensors and Mechanisms Used for HRI

Recently, more and more social robots are being equipped with tactile *skin*, thus allowing the robot to react according to the person touching the robot. Recent trends, e.g., in the European project Roboskin [72.248] tend toward covering the whole, or most of the robot's body, see e.g., *Schmitz* et al. [72.249] for an example using modular capacitive sensors to cover the humanoid robot iCub. Related work by *Dahiya* et al. [72.250] surveys a variety of different technological approaches toward tactile sensors and mechanisms of tactile sensor for robots. They show how one may take inspiration from biological tactile sensors in humans and derive design hints for robotic tactile sensing. *Stiehl* et al. [72.251] designed a tactile system inspired by the human skin as well as somatic processing [72.251], and was later developed to recognize social and affective communicative intent of how a human touches the robot [72.252, 253]. The new compliant skin technology developed in the Roboskin project has two primary functions: (a) to allow a robot to operate safely and efficiently, and (b) to use tactile sensors for communication, interaction, and cooperation with people.

The field of tactile human–robot interaction is indeed a growing area of research, and a recent survey discusses tactile HRI from the perspectives of the types of interactions that may occur between a robot and a human, and the types of sensors that allow to detect these interactions in various robotic systems such as the Robovie series of robots, RI-MAN, the Huggable robot [72.253], or Paro [72.254]. Note, equipping

a robot with tactile sensors may add its functionality, but in some cases the *sense of touch* is crucial for the key functionality of the robot (e.g., in the case of Paro [72.251] with a therapeutic/care function, or in the case of the Huggable to improve social relationships [72.255]). Many toy robots built primarily for human–robot interaction have touch sensors for petting and stroking, e.g., the AIBO or more recent Pleo robots. Indeed, the use of tactile HRI to support human–human communication over the distance is a promising area of research [72.255–258]. Tactile feedback can also improve teaching a robot by demonstration [72.259].

In order to realize robots' capability of a *sense of touch*, there are difficult sensor processing and perception problems to be addressed. One problem is that we need to identify the geometrical relationship between a number of sensors embedded in a whole body of a robot and body parts being touched when sensors are activated. For better perception of various touching, one can embed sensors into or under a soft skin. For instance, Robovie II-F has 274 sensor elements (Piezofilms) embedded in soft silicone rubber (Fig. 72.14). Tactile action activates multiple sensors at the same time, thus nearby sensors are all activated when touch action occurs to one place of body. For this problem, *Noda* et al. [72.260] took a bottom-up approach using observed signal patterns. Their method works in a self-organizing manner to identify mapping between signal patterns and touched location.

The second problem is the semantic relationship between signal pattern and people's communicative intention of touch action. For instance, *Tajika* et al. [72.261], applied clustering method for signal pattern to retrieve hierarchical structure of haptic actions. For instance, tickling-chest (Fig. 72.15) is found to be similar to stroking-chest, grouped together as lightly touching-chest action. *Knight* et al. [72.262] developed an algorithm based on identification of typical touch types, such as tickle, pet, and poke. *Yohanan* and *MacLean* [72.263] categorized humans' intent to communicate affect state and mapped touch action to each category, for example, comforting intent is mapped with actions like stroke and pat. *Stiehl* and *Breazeal* [72.264] trained a neural net to classify touch according to affect and communicative intent by recognizing types of socio-affective touch (e.g., pleasant or unpleasant ways of touching or teasing based on the way a person tickles, touches, pats, slaps, rubs, squeezes, etc.)

François et al. [72.227] describe an algorithm for pattern recognition in HRI, the cascaded information bottleneck method and apply it to real-time autonomous recognition of human–robot tactile interaction styles. This method uses an information theoretic approach

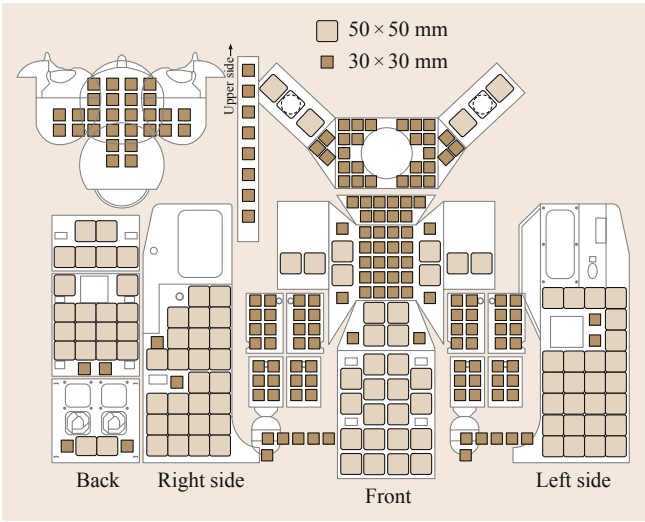


Fig. 72.14 An example of layouts of piezofilm sensors embedded in a soft skin (*left*)

Fig. 72.15 A child tickling the chest of a robot

and enables to progressively extract relevant information from time series. An evaluation with real interaction data obtained with a Sony AIBO robot shows that the algorithm is capable of classifying interaction styles (frequency and gentleness of the interaction), with a good accuracy and a very acceptable delay. The cascaded information bottleneck method was later successfully applied to create a socially adaptive robot that can recognize and adapt to children's play styles in real time [72.265]. The robot rewards well-balanced interaction styles and encourages children to engage in the interaction. The potential impact of such an adaptive robot in robot-assisted play for children with autism is evaluated through a study conducted with seven children with autism in a school. A statistical analysis of the results shows the positive impact of such an adaptive

robot on the children's play styles and on their engagement in the interaction with the robot.

72.8.3 Example: Teaching Children with Autism About Tactile Interaction

Using robots for the therapy and education for children with autism was first proposed by *Kerstin Dautenhahn* [72.266, 267] and it has recently attracted a lot of attention in the research community. A number of such approaches focus on *robot-assisted play*, since play has a crucial role in a child's development. During play, children can learn about themselves and their environments as well as develop cognitive, social, and perceptual skills. Autism, or better autistic spectrum disorders (ASD), is a life-long development disorder with key impairments in communication and social interaction and imagination (DSM IV, 1995) [72.268, 269]. Robots allow for a simplified, predictable, and reliable environment, where the complexity of interaction can be controlled and gradually increased [72.270]. Children can play dyadic games with robots, or triadic games involving other children or adults. In the latter case, scenarios emphasize the role of the robot as a *social mediat* or [72.271].

Robot mediated playing and learning activities, if successful, have the potential to enable children with cognitive disabilities to learn and acquire basic social skills thus getting support to develop/enhance their individual potential especially in the areas of communication and interaction [72.270, 272]. A number of different social robots have been studied in this domain, including zoomporphic robots, humanoid robots, or mechanically looking robots, e.g., Labo-1 [72.273], NAO [72.274], Probo [72.275], Robota [72.276, 277], Keepon [72.278], Aibo [72.279], Tito [72.280], KASPAR [72.21], and others. While we find encouraging results from case-study evaluations, further long-term and clinical studies are required [72.281].

Diehl et al. [72.281] in their review on the clinical use of robots for individuals with autism suggested that the use of interactive robots is a promising development in light of the research showing that individuals with autism exhibit strengths in understanding the physical world and relative weaknesses in understanding the social world. Children with autism often find it very difficult to appropriately interact with their social environment. In interactions between children and their peers, teachers, and family, usually touch has an important communicative and emotional role.

However, children with ASD may be hyposensitive or hypersensitive to touch so that the children may crave or avoid touch. As part of the above-mentioned European Roboskin, a number of case study evaluations have been performed investigating how a humanoid robot equipped with touch sensors can teach children with autism about appropriate tactile interaction and the associated emotional responses. The studies used the child-sized, minimally expressive robot KASPAR [72.21]; [72.282]), a low-cost robot specifically designed for interaction – many of its features lend themselves to children with autism, e.g., the human-like but minimally expressive shape and form of the robot. The robot has 8 DOFs (degrees of freedom) in the head and face and 6 DOF in each arm, as well as one DOF in the torso. A comprehensive set of play scenarios for robot-assisted play for children with various special needs, and tactile play scenarios for children with autism have been developed [72.283–285].

KASPAR can play a variety of games with children, either operating completely autonomously [72.285, 286] and VIDEO 220, or being partially remotely controlled by the children or an adult as part of the play scenario. Case studies on tactile child–robot interaction have shown encouraging results (*Robins* et al. [72.283, 287]). In the research on tactile interactions KASPAR was equipped with patches of tactile sensors, and it produced some responses autonomously, e.g., a ticking of the chest resulted in laughing, or hitting in the face results in the robot turning away from the child, covering its face with its hands and saying *ouch, that hurts*. If the child touches areas that are not covered by skin patches, or if the recognition of different types of touch is not reliable enough [72.288], then the experimenter can trigger the robot's reaction.

Figure 72.16 shows a child with autism first hitting the robot's leg and then exploring the robot's reaction. Since the legs cannot detect forceful touch, the experimenter triggers the robot's reactions. Note, this hybrid approach of autonomous behaviors combined with a remote control allows the robot to be perceptually more advanced than current state-of-the-art robotics sensing technology allows. Robots used with children with autism need to be predictable, and so a mistake in the robot's sensing abilities can be compensated for by the adult present (experimenter, teacher or caregiver). A person who knows the child very well will also be able to trigger certain useful robot behaviors at certain movements in time which are not observable directly from the child or the context and can only be inferred from detailed knowledge about the child, his/her needs and preferences, and therapeutic goals and objectives for this particular child.

Note, for a robot to detect and respond to tactile interactions of children with autism is not only beneficial to teach about appropriate tactile interaction, but it can

also provide a basis for adapting to a child's individual interaction style [72.265].

Tactile interaction is often not the sole focus of interaction, it can be embedded in multimodal play scenarios with the therapeutic objective to increase children's social skills through play. *François* et al. [72.279] present a long-term study where six children diagnosed with autism interact with an autonomous zoomorphic robot (Sony Aibo) over 10 sessions. The study is inspired by nondirective play therapy to encourage children's proactivity and initiative-taking. The behavior of each child is analyzed in detail according to three dimensions (play, reasoning, affect). Unique trajectories for children's progressions along these three dimensions were observed, resulting in unique profiles. The work highlights methodological issues in the domain of robot-assisted therapy, and also points out and formalizes different potential roles of the experimenter in the sessions, who may be a passive or an active participant [72.289]. A regulation process is introduced whereby the experimenter can regulate the interaction under specific conditions in order to:

1. Prevent or discourage repetitive behaviors
2. Help the child engage in play
3. Give a better pace to the game if it has already been experienced by the child
4. Bootstrap a higher level of play, and
5. Ask questions related to reasoning or affect.

72.8.4 Challenges and Opportunities

While many studies assume that tactile HRI will result in a more enjoyable, meaningful, and efficient interaction with a robot, many issues are still unclear and need to be investigated further. For example, a video-based study, where participants watched videos of interactions rather than interacting with the robot themselves, showed the impact of a robot's level of autonomy, and suggests that touch behaviors are considered more appropriate for proactive as compared to reactive robots [72.290]. Much further research is needed to find out when and how tactile interaction can enhance a person's experience of interaction and benefit the overall performance of the human–robot triads. One can also expect that individual differences will play a role in which types of tactile interaction with a robot are more appropriate, depending also on the tasks involved and the overall context and setting. The perception of people in terms of the robot's roles and its

Fig. 72.16 **(a)** Child hits the robot's leg and then explores the robot's reaction. **(b)** The child sees the robot looking *sad*, so he tickles its tummy (*left*) to make it *happy* (*right*). **(c)** The minimally expressive humanoid robot KASPAR

relationship with its users is also likely to impact these issues.

In recent years a number of projects worldwide have tried to use robots for therapy of children with autism. Different modes of robot control and autonomy need to be investigated, from fully autonomous systems [72.285], to Wizard-of-Oz controlled robots [72.291], to using a hybrid approach where remote control is an integral part of the interaction and triggers autonomous behaviors [72.287, 289]. Realistically, for therapeutic tools to be used widely outside the laboratory and the experimental setting, the technology needs to be highly robust, reliable, easy to operate by nonresearchers as well as cost-effective.

A number of technological, methodological, and design challenges still need to be tackled, but tactile interactions with social robots open up a number of new research avenues as well as exciting applications.

72.9 Social Robots and Teamwork

Verbal and nonverbal communication play a very important role in coordinating joint action during collaborative tasks. Sharing information through communication acts is critical given that each teammate often has only partial knowledge relevant to solving the problem, different capabilities, and possibly diverging beliefs about the state of the task. For instance, all teammates need to establish and maintain a set of mutual beliefs regarding the current state of the task, the respective roles and capabilities of each member, and the responsibilities of each teammate [72.33, 153]. This is called *common ground* [72.162].

72.9.1 Human–Robot Teamwork and Collaboration

Dialog certainly plays an important role in establishing common ground. Each conversant is committed to the shared goal of establishing and maintaining a state of mutual belief with the other. To succeed, the speaker composes a description that is adequate for the purpose of being understood by the listener, and the listener shares the goal of understanding the speaker. This communication act serves to achieve robust team behavior despite adverse conditions, including breaks in communication and other difficulties in achieving the team goals.

Humans also use nonverbal skills such as visual perspective taking and shared attention to establish common ground with others. They orient their own gaze and direct the gaze of their teammate through deictic cues such as pointing gestures in order to establish common ground. Given the visual perspective taking, shared attention, and the use of deictic cues to direct attention are core psychological processes that people use to coordinate joint action about objects and events in the world, robot teammates must be able to display and interpret these behaviors and cues when working with humans in a manner that adheres to human expectations.

Breazeal et al. [72.142] investigated the impact grounding using nonverbal social cues and behavior on task performance by a human–robot team. In a human subject experiment, participants guided Leonardo to perform a physical task using speech and gesture. The robot communicates either implicitly through behavior (such as gaze and facial expressions) or explicitly through nonverbal social cues (i. e., explicit pointing gestures). The robot's explicit grounding acts include visually attending to the human's actions to acknowledge their contributions, issuing a short nod to acknowledge the success and completion of the task or subtask, visually attending to the person's attention directing cues such as to where the human looks or points, looking back to the human once the robot operates on an artifact to make sure its contribution is acknowledged, and pointing to artifacts in the workspace to direct the human's attention toward them. Both self-reporting via questionnaire and behavioral analysis of video support the hypothesis that implicit nonverbal communication positively impacts human–robot task performance with respect to understandability of the robot, efficiency of task performance, and robustness to errors that arise from miscommunication [72.142].

Common ground is grown along with partners that work together over time. In a simple example, if two persons repeatedly work in a sequence of collaborative manufacturing tasks, one will be able to easily predict what the other person will do next and thus proactively help each other. For instance, if one always needs a spanner to be passed at certain moment in the task, another person will probably take anticipatory action passing the spanner before being asked. *Hoffman* and *Breazeal* developed adaptive system that learn such task structure which enabled a robot's anticipatory action [72.292]. They further revealed the importance of perceptual simulation [72.293].

The importance of such common ground and cognitive similarity is demonstrated in partnership in casual social interaction too. For instance, *Morales* et al. re-

vealed that a robot efficiently performed side-by-side walking with a human partner when it anticipated where a human partner would walk [72.186]. Such anticipatory computing was enabled by a capability of computing a preferred walking course in a similar way as humans would do. There are robots that explicitly learn common knowledge, like one that learns names of places [72.294].

72.9.2 Robots as Social Mediators

Researchers have started to explore the use of robots as a *social mediator*. One approach is to use a human-like presence of robots as social stimuli. It is known that presence of people facilitate others. For instance, people perform simple math-calculation faster when being watched by someone else. This is known to be *social facilitation effect* [72.295]. *Riether* et al. [72.296] reported that people's performance on easy math-calculation is improved because of presence of robots [72.296]. *Takano* et al. [72.297], put an android robot as a bystander where patients meet a medical doctor in a hospital, and found that it moderated clients' anxiety and let them believe that the doctor pays more attention to them [72.297].

Another approach is to use a robot as an active coordinator in humans' social settings. When there are many people, a coordinator could make their activity more efficient. Such a role can also successfully replicated by a robot. Consider the situation where an interactive robot is placed in the middle of a group of kids, and they start to push each other away when they want to play with the robot differently from each other. *Shomi* et al. [72.298] , developed a technique to identify when crowd of people around a robot is disordered, and let the robot perform attention-controlling behavior so that children play together with the robot in coordinated way [72.299] (Fig. 72.17). In other work, robots have been used to facilitate elderly people's group conversation. *Matsuyama* et al. [72.300], developed a robot to participate in a quiz game, conducted as a recreation activity in elderly-care facility, and provide inspiring answer to facilitate other elderly people to continue the game [72.300]. In these works, robots actively model the social situation and intervene in people's activities based on its understanding of the situation.

Fig. 72.17 The robot interacted with a crowd of people with coordinating their attention

72.9.3 Research Direction

Human activities are often social, involving multiple people who often have different skills, desires, and goals. Although the research is only in an early stage we have started to model such social situations while revealing potential roles of robots in such social settings. Along with further advance of relevant technologies, e.g., manipulation capability, navigation capability, and language capabilities, there should be many potential uses to be unveiled. However, it is most likely that the underlying theoretical work still needs a lot more work as well.

72.10 Conclusion

In this chapter, we have presented some of the principal research trends in social robotics and human–robot interaction. We have relied heavily on examples from our own research to illustrate these trends, and have used excellent examples drawn from other research groups around the world.

From this overview, we have shown that one of the most important goals of social robotics as applied HRI is the creation of robots that are human-compatible and human-centered in their design. Their differences from human abilities should complement and enhance our strengths and support how people help one another. Their similarities to human abilities, such as computationally implementing human cognitive, affective, or multimodal communication models make them more intuitive for people to understand and interact with. Further, such robots are also being used as a scientific tool to help us to understand ourselves better. With this broadening understanding, social robots are being designed to offer increasingly sophisticated levels of social, affective, cognitive, and task-based support for people, opening new applications for robots

in education, health, therapy, communication, domestic tasks, physical tasks requiring coordination and teamwork, and more. As the field advances, social robots are being applied to increasingly sophisticated tasks, in increasingly complex human environments, for longer deployment periods. We expect that in the coming decades, many other researchers, especially young researchers, will actively contribute to the transition from today's robots into capable robot partners of tomorrow.

72.11 Further Reading

For further reading, we recommend the following conference proceedings, journals, books, articles:

- Annual conference proceedings:
  - Proceedings of the ACM/IEEE International Conference on Human–Robot Interaction (HRI)
  - Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (ROMAN)
  - AAAI Symposium Series
  - AISB Symposium Series
- Journals:
  - Journal of Human-Robot Interaction (http://humanrobotinteraction.org/journal/)
  - Interaction Studies-Social Behaviour and Communication in Biological and Artificial Systems published by John Benjamins Publishing Company
  - International Journal of Social Robotics by Springer
  - IEEE Transactions on Autonomous Mental Development (TAMD)
  - IEEE Transactions on Human–Machine Systems
  - IEEE Transactions on Affective Computing
  - Paladyn, Journal of Behavioral Robotics, de Gruyter
  - PLoS ONE

Reviews and overviews can be found in several books and articles:

- Books:
  - C. Breazeal: *Designing Sociable Robots* (MIT Press, Cambridge 2002)
  - R. W. Picard: *Affective Computing* (MIT Press, Cambridge 1997)
  - J.-M. Fellous, M. Arbib (Eds.): *Who Needs Emotions: The Brain Meets the Robot* (Oxford, Oxford Univ. Press 2005)
  - K. Dautenhahn, J. Saunders (Eds.): *New Frontiers in Human–Robot Interaction*, Advances in Interaction Studies, (John Benjamins Publishing, Amsterdam 2011)
  - T. Kanda, H. Ishiguro (Eds.): *Human-Robot Interaction in Social Robotics* (CRC Press, Boca Raton 2012)
- Review Articles:
  - T. Fong, I. Nourbakshsh, K. Dautenhahn: A survey of social robots, Robotics and Autonomous Systems **42**, 143–166 (2003)
  - M.A. Goodrich, A.C. Schultz: Human–robot interaction: A survey, Foundations and Trends in Human-Computer Interaction **1**(3), 203–275 (2007)

Video-References

VIDEO 218 Home assistance companion robot in the Robot House
available from http://handbookofrobotics.org/view-chapter/72/videodetails/218

VIDEO 219 Visual communicative non-verbal behaviours of the Sunflower Robot
available from http://handbookofrobotics.org/view-chapter/72/videodetails/219

VIDEO 220 Playing triadic games with KASPAR
available from http://handbookofrobotics.org/view-chapter/72/videodetails/220

VIDEO 221 Explaining a typical session with Sunflower as a home companion in the Robot House
available from http://handbookofrobotics.org/view-chapter/72/videodetails/221

VIDEO 257 A robot that forms a good spatial formation
available from http://handbookofrobotics.org/view-chapter/72/videodetails/257

VIDEO 258 A robot that approaches pedestrians
available from http://handbookofrobotics.org/view-chapter/72/videodetails/258

VIDEO 259 A robot that provides a direction based on the model of the environment
available from http://handbookofrobotics.org/view-chapter/72/videodetails/259

VIDEO 555 Human-robot teaming in a search and retrieve task
available from http://handbookofrobotics.org/view-chapter/72/videodetails/555
VIDEO 556 Social referencing behavior
available from http://handbookofrobotics.org/view-chapter/72/videodetails/556
VIDEO 557 Overview of Kismet's expressive behavior
available from http://handbookofrobotics.org/view-chapter/72/videodetails/557
VIDEO 558 Overview of Autom: A robotic health coach for weight management
available from http://handbookofrobotics.org/view-chapter/72/videodetails/558
VIDEO 559 Non-verbal envelope displays to support turn-taking behavior
available from http://handbookofrobotics.org/view-chapter/72/videodetails/559
VIDEO 560 Learning how to be a learning companion for children
available from http://handbookofrobotics.org/view-chapter/72/videodetails/560
VIDEO 562 Social learning applied to task execution
available from http://handbookofrobotics.org/view-chapter/72/videodetails/562
VIDEO 563 Mental state inference to support human-robot collaboration
available from http://handbookofrobotics.org/view-chapter/72/videodetails/563
VIDEO 564 A learning companion robot to foster pre-K vocabulary learning
available from http://handbookofrobotics.org/view-chapter/72/videodetails/564
VIDEO 806 Influence of response time
available from http://handbookofrobotics.org/view-chapter/72/videodetails/806
VIDEO 807 A scene of deictic interaction
available from http://handbookofrobotics.org/view-chapter/72/videodetails/807
VIDEO 808 An example of a social robot in a museum
available from http://handbookofrobotics.org/view-chapter/72/videodetails/808
VIDEO 809 An example of repeated long-term interaction
available from http://handbookofrobotics.org/view-chapter/72/videodetails/809
VIDEO 810 A robot that exhibits its listening attitude with its motion
available from http://handbookofrobotics.org/view-chapter/72/videodetails/810
VIDEO 811 Region pointing gesture
available from http://handbookofrobotics.org/view-chapter/72/videodetails/811

References

72.1 K. Dautenhahn: Getting to know each other – Artificial social intelligence for autonomous robots, Robotics Auton. Syst. **16**, 333–356 (1995)
72.2 K. Dautenhahn: I could be you: The phenomenological dimension of social understanding, Cybern. Syst. **28**, 417–453 (1997)
72.3 C. Breazeal: *Designing Sociable Robots* (MIT Press, Cambridge 2002)
72.4 H. Miwa, A. Takanishi, H. Takanobu: Experimental study on robot personality for humanoid head robot, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2001) pp. 1183–1188
72.5 T. Tojo, Y. Matsusaka, T. Ishii, T. Kobayashi: A conversational robot utilizing facial and body expressions, IEEE Int. Conf. Syst. Man Cybern., Vol. 2 (2000) pp. 858–863
72.6 J. Cassell, J. Sullivan, S. Prevost, E. Churchill (Eds.): *Embodied Conversational Agents* (MIT Press, Cambridge 2000)
72.7 G. Hoffman, C. Breazeal: Robots that work in collaboration with people, AAAI Symp. Intersect. Cogn. Sci. Robotics, Washington DC (2004)
72.8 K. Dautenhahn, A.H. Bond, L. Canamero, B. Edmonds (Eds.): *Socially Intelligent Agents: Creating Relationships with Computers and Robots* (Kluwer, Boston 2002)
72.9 R.W. Picard: *Affective Computing* (MIT Press, Cambridge 1997)
72.10 T. Fong, I. Nourbakshsh, K. Dautenhahn: A survey of socially interactive robots, Robotics Auton. Syst. **42**, 143–166 (2003)
72.11 S. Schaal: Is imitation learning the route to humanoid robots?, Trends Cogn. Sci. **3**(6), 233–242 (1999)
72.12 J. Fink: Anthropomorphism and human likeness in the design of robots and human-robot interaction, Lect. Notes Comput. Sci. **7621**, 199–208 (2012)
72.13 J. Solis, K. Chida, K. Suefuji, A. Takanishi: The development of the anthropomorphic flutist robot at Waseda University, Int. J. Humanoid Robotics **3**(2), 1–25 (2006)
72.14 Y. Ogura, H. Aikawa, K. Shimomura, H. Kondo, A. Morishima, H. Lim, A. Takanishi: Development of a new humanoid robot WABIAN-2, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 76–81
72.15 H. Miwa, K. Itoh, H. Takanobu, A. Takanishi: Mechanical design and motion control of emotion expression Humanoid robot WE-4R, 15th CISM-Symp. Robot Des. Dyn. Control (2004) pp. 255–262
72.16 M.F. Jung, J.J. Lee, N. DePalma, S.O. Adalgeirsson, P.J. Hinds, C. Breazeal: Engaging robots: Easing complex human-robot teamwork using backchanneling, Proc. ACM Conf. Comput. Suppor. Coop. Work (CSCW), San Antonio (2013) pp. 1555–1566

72.17 J. Goetz, S. Kiesler, A. Powers: Matching robot appearance and behavior to tasks to improve human-robot cooperation, Proc. 12th IEEE Int. Workshop Robot Hum. Interact. Commun. (ROMAN) (2003) pp. 55–60

72.18 F. Iida, M. Tabata, F. Hara: Generating personality character in a face robot through interaction with human, Proc. 7th IEEE Int. Workshop Robot Hum. Commun. (ROMA) (1998) pp. 481–486

72.19 M. Shimada, T. Minato, S. Itakura, H. Ishiguro: Evaluation of android using unconscious recognition, Proc. 6th IEEE-RAS Int. Conf. Humanoid Robots (2006) pp. 157–162

72.20 K. Berns, J.J. Hirth: Control of facial expressions of the humanoid robot head ROMAN, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Bejing (2006) pp. 3119–3124

72.21 K. Dautenhahn, C.L. Nehaniv, M.L. Walters, B. Robins, H. Kose-Bagci, N.A. Mirza, M. Blow: KASPAR – A minimally expressive humanoid robot for human-robot interaction research, Appl. Bionics Biomech. **6**(3), 369–397 (2009)

72.22 F. Tanaka, J.R. Movellan, B. Fortenberry, K. Aisaka: Daily HRI evaluation at a classroom environment: Reports from dance interaction experiments, Proc. 1st ACM Conf. Hum.-Robot Interact. (HRI), Salt Lake City (2006) pp. 3–9

72.23 H. Lim, S. Hyon, S.A. Setiawan, A. Takanishi: Quasi-human biped walking, Int. J. Inform. Educ. Res. Robotics Artif. Intell. **24**(2), 257–268 (2006)

72.24 H. Lim, A. Ishii, A. Takanishi: Emotion-based biped walking, Int. J. Inform. Educ. Res. Robotics Artif. Intell. **22**(5), 577–586 (2004)

72.25 J. Solis, S. Isoda, K. Chida, A. Takanishi, K. Wakamatsu: Anthropomorphic flutist robot for teaching flute playing to beginner students, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 146–151

72.26 H. Miwa, K. Itoh, M. Matsumoto, M. Zecca, H. Takanobu, S. Roccella, M.C. Carrozza, P. Dario, A. Takanishi: Effective emotional expressions with emotion expression humanoid robot WE-4RII, Proc. 2004 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 2203–2208

72.27 S. Roccella, M.C. Carrozza, G. Cappiello, P. Dario, J. Cabibihan, M. Zecca, H. Miwa, K. Itoh, M. Matsumoto, A. Takanishi: Design, fabrication and preliminary results of a novel anthropomorphic hand for humanoid robotics: RCH-1, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Sendai (2004) pp. 266–271

72.28 H. Miwa, K. Itoh, H. Takanobu, A. Takanishi: Design and control of 9-DOFs emotion expression humanoid arm, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 128–133

72.29 K. Hayashi, Y. Onishi, K. Itoh, H. Miwa, A. Takanishi: Development and evaluation of face robot to express various face shape, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 481–486

72.30 M. Fujita: On activating human communications with pet-type robot AIBO, Proceedings IEEE **92**(11), 1804–1813 (2004)

72.31 K. Wada, T. Shibata, K. Sakamoto, K. Tanie: Long-term interaction between seal robots and elderly people – Robot assisted activity at a health service facility for the aged, Proc. 3rd Int. Symp. Auton. Minirobots Res. Edutainment (2005) pp. 325–330

72.32 C.L. Sidner, C. Lee, C.D. Kidd, N. Lesh, C. Rich: Explorations in engagement for humans and robots, Artif. Intell. **166**(1/2), 140–164 (2005)

72.33 C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd, D. Chilongo: Tutelage and collaboration for humanoid robots, Int. J. Humanoid Robotics **1**(2), 315–348 (2004)

72.34 H. Kozima: An anthropologist in the children's world: A field study of children's everyday interaction with an interactive robot, Proc. Int. Conf. Dev. Learn. (ICDL), Bloomington (2006)

72.35 M.E. Pollack, S. Engberg, J.T. Matthews, S. Thrun, L. Brown, D. Colbry, C. Orosz, B. Peintner, S. Ramakrishnan, J. Dunbar-Jacob, C. McCarthy, M. Montemerlo, J. Pineau, N. Roy: Pearl: A mobile robotic assistant for the elderly, Proc. AAAI Workshop Autom. Eldercare (2002)

72.36 R. Gockley, R. Simmons, J. Forlizzi: Modeling affect in socially interactive robots, Proc. 15th IEEE Int. Symp. Robot Hum. Interact. Commun. (ROMAN) (2006) pp. 558–563

72.37 M. Mori: Bukimi no tani the uncanny valley, Energy **7**(4), 33–35 (1970)

72.38 K. Wada, T. Shibata: Living with seal robots in a care house – Evaluations of social and physiological influences, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2007) pp. 4940–4945

72.39 W. Stiehl, J. Lieberman, C. Breazeal, L. Basel, L. Lalla, M. Wolf: Design of a therapeutic robotic companion for relational, affective touch, Proc. 14th IEEE Workshop Robot Hum. Interact. Commun. (ROMAN) (2005) pp. 408–415

72.40 NEC Corporation Japan: http://jpn.nec.com/robot/en/

72.41 M.L. Walters, K. Dautenhahn, S.N. Woods, K.L. Koay: Robotic etiquette: Results from user studies involving a fetch and carry task, Proc. 2nd ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI), Washington (2007)

72.42 R. Gockley, J. Forlizzi, R. Simmons: Natural person-following behavior for social robots, Proc. 2nd ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2007) pp. 17–24

72.43 A.G. Brooks, R.C. Arkin: Behavioral overlays for non-verbal communication expression on a humanoid robot, Auton. Robots **22**(1), 55–74 (2007)

72.44 C. Darwin, P. Ekman: *The Expression of the Emotions in Man and Animals*, 3rd edn. (Oxford Univ. Press, Oxford 1998)

72.45 C. Breazeal: Emotion and sociable humanoid robots, Int. J. Hum. Comput. Interact. **58**, 119–155 (2003)

72.46 F. Tanaka, K. Noda, T. Sawada, M. Fujita: Associated emotion and its expression in an entertainment robot QRIO, Proc. 3rd Int. Conf. Entertain. Comput., Eindhoven (2004) pp. 499–504

72.47 S. Fujie, Y. Ejiri, K. Nakajima, Y. Matsusaka, T. Kobayashi: A conversation robot using head gesture recognition as paralinguistic information, Proc. IEEE Int. Symp. Robot Hum. Interact. Commun. (ROMAN) (2004) pp. 158–164

72.48 L. Hall, S. Woods, R. Aylett, L. Newall, A. Paiva: Achieving empathic engagement through affective interaction with synthetic characters, Lect. Notes Comput. Sci. **3784**, 731–738 (2005)

72.49 R.C. Arkin: Moving up the food chain: Motivation and emotion in behavior-based robots. In: *Who Needs Emotions: The Brain Meets the Robot*, ed. by J. Fellous, M. Arbib (Oxford Univ. Press, Oxford 2005)

72.50 Feelix growing, a European project coordinated by University of Hertfordshire: https://en.wikipedia.org/wiki/Feelix_Growing

72.51 C. Breazeal: Function meets style: Insights from emotion theory applied to HRI, IEEE Trans. Syst. Man Cybern. C **34**(2), 187–194 (2003)

72.52 A. Sloman: Beyond shallow models of emotion, Cogn. Process. Int. Q. Cogn. Sci. **2**(1), 177–198 (2001)

72.53 S. Chernova, N. DePalma, C. Breazeal: Crowdsourcing real world human-robot dialog and teamwork through online multiplayer games, AAAI Magazine **32**(4), 100–111 (2011)

72.54 M. Scheutz, P. Schermerhorn, J. Kramer: The utility of affect expression in natural language interactions in joint human-robot tasks, Proc. 1st ACM Conf. Hum.-Robot Interact. (HRI) (ACM, New York, USA 2006) pp. 226–233

72.55 J.C. Lester, S.G. Towns, C.B. Callaway, J.L. Voerman, P.J. Fitzgerald: Deictic and emotive communication in animated pedagogical agents. In: *Embodied Conversational Agents*, ed. by J. Casell, S. Prevost, J. Sullivan, E. Churchill (MIT Press, Cambridge 2000)

72.56 M.E. Hoque: My automated conversation helper (MACH): Helping people improve social skills, Proc. 14th ACM Int. Conf. Multimodal Interact. (ICMI), Santa Monica (2012)

72.57 R. Lazarus: *Emotion and Adaptation* (Oxford Univ. Press, New York 1991)

72.58 A. Ortony, G. Clore, A. Collins: *The Cognitive Structure of Emotions* (Cambridge Univ. Press, New York 1988)

72.59 A. Marsella, J. Gratch: EMA: A process model of appraisal dynamics, J. Cogn. Syst. Res. **10**, 70–90 (2009)

72.60 C. Elliott: The Affective Reasoner: A Process Model of Emotions in a Multi-Agent System, Ph.D. Thesis (Northwestern Univ., Northwestern 1992)

72.61 J.A. Russell: Core affect and the psychological construation of emotion, Psychol. Rev. **110**, 145–172 (2003)

72.62 A. Mehrabian, J.A. Russell: *An Approach to Environmental Psychology* (MIT Press, Cambridge 1974)

72.63 C. Smith, H. Scott: A componential approach to the meaning of facial expressions. In: *The Psychology of Facial Expression*, ed. by J. Russell, J. Fernandez-Dols (Cambridge Univ. Press, Cambride 1997) pp. 229–254

72.64 C. Breazeal, B. Scassellati: Infant-like social interactions between a robot and a human caregiver, Adapt. Behav. **8**(1), 47–72 (2000)

72.65 A. Damasio: *Decartes' Error: Emotion, Reason and the Human Brain* (Putnam, New York 1994)

72.66 C. Breazeal: Emotive qualities in lip synchronized robot speech, Adv. Robotics **17**(2), 97–113 (2003)

72.67 C. Breazeal: Early experiments using motivations to regulate human-robot interaction, AAAI Fall Symp. Emot. Intell., Orlando (1998) pp. 31–36

72.68 C. Breazeal, L. Aryananda: Recognizing affective intent in robot directed speech, Auton. Robots **12**(1), 85–104 (2002)

72.69 J. Panskepp: *Affective Neuroscience: The Foundations of Human and Animal Emotions* (Oxford Univ. Press, New York 1998)

72.70 R. Plutchik: Emotions: A general psychoevolutionary theory. In: *Approaches to Emotion*, ed. by K. Sherer, P. Elkman (Lawrence Erlbaum Associates, Hillsdale 1984) pp. 197–219

72.71 L. Vygotsky: *Mind in Society: The Development of Higher Psychological Processes* (Harvard Univ. Press, Cambridge 1978)

72.72 F. Strack, L. Martin, S. Stepper: Inhibiting and facilitating conditions of the human smile: A nonobtrusive test of the facial feedback hypothesis, J. Person. Soc. Psychol. **54**, 768–777 (1988)

72.73 P.M. Niedenthal, L.W. Barsalou, P. Winkielman, S. Krauth-Gruber, F. Ric: Embodiment in attitudes, social perception, and emotion, Personal. Soc. Psychol. Rev. **9**(3), 184–211 (2005)

72.74 C. Breazeal, D. Buchsbaum, J. Gray, D. Gatenby, B. Blumberg: Learning from and about others: Towards using imitation to bootstrap the social understanding of others by robots, Artif. Life **11**(1–2), 31–62 (2005)

72.75 A. Fernald: Intonation and communicative intent in mother's speech to infants: Is the melody the message?, Child Dev. **60**, 1497–1510 (1989)

72.76 B. Scassellati: Theory of mind for a humanoid robot, Auton. Robots **12**(1), 13–24 (2002)

72.77 A. Leslie: How to acquire a representational theory of mind. In: *Metarepresentation: A Multidisciplinary Perspective*, ed. by D. Sperber (Oxford Univ. Press, Oxford 1994) pp. 197–223

72.78 S. Baron-Cohen: Precursors to a theory of mind: Understanding attention in others. In: *Natural Theories of Mind*, ed. by A. Whiten (Blackwell, Oxford 1991) pp. 233–250

72.79 B. Scassellati: Mechanisms of shared attention for a humanoid robot, AAAI Fall Symp. Embodied Cogn. Action (1996)

72.80 G.O. Fasel, J. Deak, J. Triesch, J. Movellan: Combining embodied models and empirical research for understanding the development of shared attention, Proc. 2nd IEEE Int. Conf. Dev. Learn. (ICDL) (2002) pp. 21–27

72.81 Y. Nagai, M. Asada, K. Hosoda: Learning for joint attention helped by functional development, Adv. Robotics **20**(10), 1165–1181 (2006)

72.82 G. Butterworth: The ontogeny and phylogeny of joint visual attention. In: *Natural Theories of*

Mind, ed. by A. Whiten (Blackwell, Oxford 1991) pp. 223–232
72.83 A.L. Thomaz, M. Berlin, C. Breazeal: An embodied computational model of social referencing, Proc. 14th IEEE Workshop Robot Hum. Interact. Commun. (ROMAN), Nashville (2005)
72.84 S. Feinman: Social referencing in infancy, Merrill-Palmer Q. **28**, 445–470 (1982)
72.85 M. Davies, T. Stone: *Mental Simulation* (Blackwell, Oxford 1995)
72.86 V. Gallese, A. Goldman: Mirror neurons and the simulation theory of mind-reading, Trends Cogn. Sci. **2**(12), 493–501 (1998)
72.87 A. Meltzoff, J. Decety: What imitation tells us about social cognition: A rapprochement between developmental psychology and cognitive neuroscience, Philos. Trans. R. Soc. Lond. B Biol, Sci. **358**, 491–500 (2003)
72.88 L.W. Barsalou, P.M. Niedenthal, A. Barbey, J. Ruppert: Social embodiment. In: *The Psychology of Learning and Motivation*, Vol. 43, ed. by B. Ross (Academic, Amsterdam 2003) pp. 43–92
72.89 M. Johnson, Y. Demiris: Perceptual perspective taking and action recognition, Int. J. Adv. Robotics Syst. **2**(4), 301–308 (2005)
72.90 J. Gray, C. Breazeal, M. Berlin, A. Brooks, J. Lieberman: Action parsing and goal inference using self as simulator, Proc. 14th IEEE Workshop Robot Hum. Interact. Commun. (ROMAN), Nashville (2005)
72.91 J. Gray, M. Berlin, C. Breazeal: Intention recognition with divergent beliefs for collaborative robots, Proc. AISB Symp. Mindful Environ., Newcastle Upon Tyne (2007)
72.92 G. Trafton, A. Schultz, M. Bugajska, F. Mintz: Perspective-taking with Robots: Experiments and models, Proc. 14th IEEE Workshop Robot Hum. Interact. Commun. (ROMAN), Nashville (2005)
72.93 G. Trafton, A.C. Shultz, D. Perzanowsli, W. Adams, M. Bugajska, N. Cassimatis, D. Brock: Children and robots learning to play hide and seek, Proc. 1st Ann. Conf. Hum.-Robot Interact. (HRI), Salt Lake City (2006)
72.94 J.G. Trafton, N. Cassimatis, M. Bugajska, D. Brock, F. Mintz, A. Schultz: Enabling effective human-robot interaction using perspective-taking in robots, IEEE Trans. Syst. Man Cybern. A Syst. Hum. **35**(4), 460–470 (2005)
72.95 J. Gray, C. Breazeal: Manipulating mental states through physical action, Int. Conf. Soc. Robotics, Chendu (2012)
72.96 C. Nass, S. Brave: *Wired for Speech: How Voice Activates and Advances the Human-Computer Relationship* (MIT Press, Cambridge, MA 2005)
72.97 C. Nass, Y. Moon: Machines and mindlessness: Social responses to computers, J. Soc. Issues **56**, 81–103 (2000)
72.98 C. Nass, J.S. Steuer, E. Tauber, H. Reeder: Anthropomorphism, agency, and ethopoeia: Computers as social actors, ACM Conf. Companion Hum. Factors Comput. Syst., Amsterdam (1993) pp. 111–112
72.99 N. Epley, A. Waytz, J.T. Cacioppo: On seeing human: A three-factor theory of anthropomorphism, Psychol. Rev. **114**, 864–886 (2007)
72.100 J. Goetz, S. Kiesler: Cooperation with a robotic assistant, ACM Ext. Abstr. Hum. Factors Comput. Syst. (2002) pp. 578–579
72.101 L. Axelrod, K. Hone: E-motional advantage: Performance and satisfaction gains with affective computing, ACM Ext. Abstr. Hum. Factors Comput. Sci. (2005) pp. 1192–1195
72.102 S.O. Adalgeirsson, C. Breazeal: MeBot: A robotic platform for socially embodied presence, Proc. 5th ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2010) pp. 15–22
72.103 S. Lee, I.Y. Lau, S. Kiesler, C.-Y. Chiu: Human mental models of humanoid robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 2767–2772
72.104 L.D. Riek, T.-C. Rabinowitch, B. Chakrabarti, P. Robinson: How anthropomorphism affects empathy toward robots, Proc. 4th ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2009) pp. 245–246
72.105 S. Krach, F. Hegel, B. Wrede, G. Sagerer, F. Binkofski, T. Kircher: Can machines think? Interaction and perspective taking with robots investigated via fMRI, PLoS ONE **3**(7), e2597 (2008)
72.106 E. Short, J. Hart, M. Vu, B. Scassellati: No fair!! An interaction with a cheating robot, Proc. 5th ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2010) pp. 219–226
72.107 S.R. Fussell, S. Kiesler, L.D. Setlock, V. Yew: How people anthropomorphize robots, Proc. 3rd ACM/IEEE Int. Conf. Human Robot Interact. (HRI) (2008) pp. 145–152
72.108 P.H. Kahn Jr., N.G. Freier, B. Friedman, R.L. Severson, E. Feldman: Social and moral relationships with robotic others?, Proc. 13th Int. Workshop Robot Hum. Interact. Commun. (ROMAN) (2004) pp. 545–550
72.109 C. Bartneck, J. Forlizzi: A design-centred framework for social human-robot interaction, 13th IEEE Int. Workshop Robot Hum. Interact. Commun. (ROMAN) (2004) pp. 591–594
72.110 J. Shah, C. Breazeal: Improved human-robot team performance using chaski, A human-inspired plan execution system, Proc. ACM/IEEE Int. Conf. Hum. Robot Interact. (HRI) (2011)
72.111 Rethink Robotics: http://www.rethinkrobots.com
72.112 C. Kidd, C. Breazeal: Effect of a robot on user perceptions, Proc. 2004 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Sendai, Vol. 4 (2004) pp. 3559–3564
72.113 W. Bainbridge, J. Hart, E. Kim, B. Scassellati: The effect of presence on human-robot interaction, IEEE Int. Symp. Robot Hum. Interact. Commun., Munich (2008)
72.114 J. Wainer, D. Feil-Seifer, D. Shell, M. Matarić: The role of physical embodiment in human-robot interaction, Proc. Int. Workshop Robot Hum. Interact. Commun. (ROMAN) (2006) pp. 6–8
72.115 J. Wainer, D. Feil-Seifer, D. Shell, M. Matarić: Embodiment and human-robot interaction: A task-based perspective, Proc. 16th IEEE Int. Workshop

Robot Hum. Interact. Commun. (ROMAN), Jeju Island (2007)
72.116 C. Kidd, C. Breazeal: Robots at home: Understanding long-term human-robot interaction, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Nice (2008)
72.117 F.R. Volkmar, C. Lord, A. Bailey, R.T. Schultz, A. Klin: Autism and pervasive developmental disorders, J. Child Psychol. Psych. **45**(1), 1–36 (2004)
72.118 N. Freed: This is the Fluffy Robot That Only Speaks French: Language Use Between Preschoolers, Their Families, and a Social Robot While Sharing Virtual Toys. Masters Sci. Thesis (MIT, Cambridge 2012)
72.119 S. Turkle: *Along Together* (Basic Books, New York 2012)
72.120 N. Ambady, M. Weisbuch: Nonverbal behavior. In: *Handbook of Social Psychology*, 5th edn., ed. by D.T. Gilbert, S.T. Fiske, G. Lindzey (Wiley, Hoboken 2010)
72.121 J.A. Hall, E.J. Coats, L. Smith-Lebeau: Nonverbal behavior and the vertical dimension of social relations: A meta-analysis, Psychol. Bull. **131**, 898–924 (2005)
72.122 D. DeSteno, C. Breazeal, R. Frank, D. Pizarro, J. Baumann, L. Dickens, J.J. Lee: Detecting the trustworthiness of novel partners in economic exchange, Psychol. Sci. **23**(12), 1549–1556 (2012)
72.123 C.P.H. Langford, J. Bowsher, J.P. Maloney, P.P. Lillis: Social support: A conceptual analysis, J. Adv. Nurs. **25**, 95–100 (1997)
72.124 T.A. Wills: Supportive functions of interpersonal relationships. In: *Social Support and Health*, ed. by S. Cohen, L. Syme (Academic, Orlando 1985) pp. 61–82
72.125 T.A. Wills: Social support and interpersonal relationships. In: *Prosocial Behavior*, ed. by M.S. Clark (Sage, Newbury Park 1991) pp. 265–289
72.126 F.J. Bernieri, J.M. Davis, R. Rosenthal, C.R. Knee: Inter-actional synchrony and rapport: Measuring synchrony in displays devoid of sound and facial affect, Personal. Soc. Psychol. Bull. **20**, 303–311 (1994)
72.127 F.J. Bernieri: Coordinated movement and rapport in teacher-student interactions, J. Nonverbal Behav. **12**, 120–138 (1988)
72.128 N. Sonalkar, M. Jung, A. Mabogunje: Emotion in engineering design teams. In: *Emotional Engineering: Service Development*, ed. by S. Fukuda (Springer, London 2010)
72.129 J.E. Grahe, F.J. Bernieri: The importance of nonverbal cues in judging rapport, J. Nonverbal Behav. **23**, 253–269 (1999)
72.130 L.K. Miles, L.K. Nind, C.N. Macrae: The rhythm of rapport: Interpersonal synchrony and social perception, J. Exp. Soc. Psychol. **45**(3), 585–589 (2009)
72.131 Y. Nagai: Learning to comprehend deictic gestures in robots and human infants, IEEE Int. Workshop Robot Hum. Interact. Commun. (ROMAN) (2005) pp. 217–222
72.132 Z. Kasap, N. Magnenat-Thalmann: Building long-term relationships with virtual and robotic characters: The role of remembering, Vis. Comput. **28**, 87–97 (2012)
72.133 F. Tanaka, S. Matsuzoe: Children teach a care-receiving robot to promote their learning: Field experiments in a classroom for vocabulary learning, J. Hum.-Robot Interact. **1**(1), 78–95 (2012)
72.134 C. Pastor, G. Gaminde, A. Renteria: COMPANION-ABLE: Integrated cognitive assistive and domotic companion robotic systems for ability and security, Int. Symp. Robotics, Barcelona (2009)
72.135 H. Fujisawa, K. Shirai: An algorithm for spoken sentence recognition and its application to the speech input-output system, Proc. IEEE Trans. Syst. Man Cybern. **4**(5), 475–479 (1974)
72.136 T. Kobayashi, Y. Komori, N. Hashimoto, K. Iwata, Y. Fukazawa, J. Yazawa, K. Shirai: Speech conversation system of the musician robot, Proc. Int. Conf. Adv. Robotics (ICAR) (1985) pp. 483–488
72.137 H. Kikuchi, M. Yokoyama, K. Hoashi, Y. Hidaki, T. Kobayashi, K. Shirai: Controlling gaze of humanoid in communication with human, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1998) pp. 255–260
72.138 S. Hashimoto: Humanoid robots in Waseda University: Hadaly2 and WABIAN, Auton. Robots **12**(1), 25–38 (2002)
72.139 C. Rich, B. Ponsler, A. Holroyd, C.L. Sidner: Recognizing engagement in human-robot interaction, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2010) pp. 375–382
72.140 C. Breazeal, A. Edsinger, P. Fitzpatrick, B. Scassellati: Active vision systems for sociable robots, IEEE Trans. Syst. Man Cybern. **31**(5), 443–453 (2001)
72.141 C. Breazeal, C.D. Kidd, A.L. Thomaz, G. Hoffman, M. Berlin: Effects of nonverbal communication on efficiency and robustness in human-robot teamwork, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005) pp. 383–388
72.142 C. Breazeal, C. Kidd, A.L. Thomaz, G. Hoffman, M. Berlin: Effects of nonverbal communication on efficiency and robustness in human-robot teamwork, Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS) (2005)
72.143 B. Mutlu, T. Shiwa, T. Kanda, H. Ishiguro, N. Hagita: Footing in human-robot conversations: How robots might shape participant roles using gaze cues, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2009) pp. 61–68
72.144 B. Mutlu, F. Yamaoka, T. Kanda, H. Ishiguro, N. Hagita: Nonverbal leakage in robots: Communication of intentions through seemingly unintentional behavior, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2009) pp. 69–76
72.145 N. Kirchner, A. Alempijevic: A robot centric perspective on HRI, J. Hum.-Robot Interact. **1**(2), 135–157 (2012)
72.146 A. Yamazaki, K. Yamazaki, T. Ohyama, Y. Kobayashi, Y. Kuno: A techno-sociological solution for designing a Museum guide robot: Regarding

choosing an appropriate visitor, Proc. 17th Annu. ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2012) pp. 309–316
72.147 S. Fujie, K. Fukushima, T. Kobayashi: Back-channel feedback generation using linguistic and nonlinguistic information and its application to spoken dialogue system, Proc. Interspeech (2005) pp. 889–892
72.148 M. Imai, T. Ono, H. Ishiguro: Physical relation and expression: Joint attention for human-robot interaction, Proc. ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2001) pp. 512–517
72.149 T. Kanda, M. Kamasima, M. Imai, T. Ono, D. Sakamoto, H. Ishiguro, Y. Anzai: A humanoid robot that pretends to listen to route guidance from a human, Auton. Robots **22**, 87–100 (2007)
72.150 H. Ogawa, T. Watanabe: Interrobot: A speech driven embodied interaction robot, IEEE Int. Workshop Robot Hum. Interact. Commun. (ROMAN) (2000) pp. 322–327
72.151 D. Sakamoto, T. Kanda, T. Ono, H. Ishiguro, N. Hagita: Android as a telecommunication medium with a human-like presence, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2007) pp. 193–200
72.152 H. Kuzuoka, S. Oyama, K. Yamazaki, K. Suzuk, M. Mitsuishi: GestureMan: A mobile robot that embodies a remote instructor's actions, Proc. ACM Conf. Comput.-Suppor. Coop. Work (CSCW) (2000)
72.153 A. Brooks, C. Breazeal: Working with robots and objects: Revisiting deictic reference for achieving spatial common ground, Proc. ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2006)
72.154 B. Scassellati: *Investigating Models of Social Development Using a Humanoid Robot. Biorobotics* (MIT Press, Cambridge 2000)
72.155 H. Kozima, E. Vatikiotis-Bateson: Communicative criteria for processing time/space-varying information, IEEE Int. Workshop Robot Hum. Commun. (ROMAN) (2001)
72.156 O. Sugiyama, T. Kanda, M. Imai, H. Ishiguro, N. Hagita: Humanlike conversation with gestures and verbal cues based on a three-layer attention-drawing model, Connect. Sci. **18**(4), 379–402 (2006)
72.157 V. Ng-Thow-Hing, P. Luo, S. Okita: Synchronized gesture and speech production for humanoid robots, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 4617–4624
72.158 Y. Hato, S. Satake, T. Kanda, M. Imai, N. Hagita: Pointing to space: Modeling of deictic interaction referring to regions, Proc. ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2010) pp. 301–308
72.159 Y. Okuno, T. Kanda, M. Imai, H. Ishiguro, N. Hagita: Providing route directions: Design of robot's utterance, gesture, and timing, Proc. ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2009) pp. 53–60
72.160 O. Sugiyama, T. Kanda, M. Imai, H. Ishiguro, N. Hagita: Three-layer model for generation and recognition of attention-drawing behavior, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 5843–5850
72.161 H. Sacks, E.A. Schegloff, G. Jefferson: A simplest systematics for the organization of turn-taking for conversation, Language **50**, 696–735 (1974)
72.162 H.H. Clark: *Using Language* (Cambridge Univ. Press, Cambridge 1996)
72.163 M.F. McTear: Spoken dialogue technology: Enabling the conversational user interface, ACM Comput. Surv. **34**, 90–169 (2002)
72.164 M. Nakano, Y. Hasegawa, K. Nakadai, T. Nakamura, J. Takeuchi, T. Torii, H. Tsujino, N. Kanda, H.G. Okuno: A two-layer model for behavior and dialogue planning in conversational service robots, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2005) pp. 3329–3335
72.165 M. Scheutz, P. Schermerhorn, J. Kramer, D. Anderson: First steps toward natural human-like Hri, Auton. Robots **22**, 411–423 (2006)
72.166 C. Shi, T. Kanda, M. Shimada, F. Yamaoka, H. Ishiguro, N. Hagita: Easy development of communicative behaviors in social robots, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010) pp. 5302–5309
72.167 K. Sakita, K. Ogawara, S. Murakami, K. Kawamura, K. Ikeuchi: Flexible cooperation between human and robot by interpreting human intention from gaze information, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 846–851
72.168 Y. Matsusaka, T. Tojo, T. Kobayashi: Conversation robot participating in group conversation, IEICI Trans. Inform. Syst. **86**(1), 26–36 (2003)
72.169 R. Nisimura, T. Uchida, A. Lee, H. Saruwatari, K. Shikano, Y. Matsumoto: ASKA: Receptionist robot with speech dialogue system, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2002) pp. 1314–1319
72.170 Y. Matsusaka, S. Fujie, T. Kobayashi: Modeling of conversational strategy for the robot participating in the group conversation, Eur. Conf. Speech Commun. Technol. (EUROSPEECH) (2001)
72.171 M. Hoque, D. Das, T. Onuki, Y. Kobayashi, Y. Kuno: An integrated approach of attention control of target human by nonverbal behaviors of robots in different viewing situations, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 1399–1406
72.172 A. Raux, M. Eskenazi: A finite-state turn-taking model for spoken dialog systems, Proc. Hum. Lang. Technol. Annu. Conf. North Am. Chapt. Assoc. Comput. Linguist. (2009) pp. 629–637
72.173 C. Chao, A.L. Thomaz: Timing in multimodal turn-taking interactions: Control and analysis using timed petri nets, J. Hum.-Robot Interact. **1**, 4–25 (2012)
72.174 M. Yamamoto, T. Watanabe: Time lag effects of utterance to communicative actions on Cg character-human greeting interaction, IEEE Int. Symp. Robot Hum. Interact. Commun. (ROMAN) (2006) pp. 629–634
72.175 D. McNeill: *Psycholinguistics: A New Approach* (HarperRow, New York 1987)
72.176 T. Shiwa, T. Kanda, M. Imai, H. Ishiguro, N. Hagita: How quickly should a communication robot re-

spond? Delaying strategies and habituation effects, Int. J. Soc. Robotics **1**, 141–155 (2009)
72.177 M.P. Michalowski, S. Sabanovic, R. Simmons: A spatial model of engagement for a social robot, IEEE Int. Workshop Adv. Motion Control (2006) pp. 762–767
72.178 K. Dautenhahn, M.L. Walters, S. Woods, K.L. Koay, C.L. Nehaniv, E.A. Sisbot, R. Alami, T. Siméon: How may i serve you? A robot companion approaching a seated person in a helping context, ACM/IEEE Int. Conf. Hum.-Robots Interact. (HRI) (2006) pp. 172–179
72.179 S. Satake, T. Kanda, D.F. Glas, M. Imai, H. Ishiguro, N. Hagita: How to approach humans? Strategies for social robots to initiate interaction, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2009) pp. 109–116
72.180 M.L. Walters, K. Dautenhahn, R.T. Boekhorst, K.L. Koay, C. Kaouri, S. Woods, C. Nehaniv, D. Lee, I. Werry: The influence of subjects' personality traits on personal spatial zones in a human-robot interaction experiment, IEEE Int. Workshop Robot Hum. Interact. Commun. (ROMAN) (2005) pp. 347–352
72.181 H. Hüttenrauch, K.S. Eklundh, A. Green, E.A. Topp: Investigating spatial relationships in human-robot interactions, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 5052–5059
72.182 D. Feil-Seifer: Distance-based computational models for facilitating robot interaction with children, J. Hum.-Robot Interact. **1**, 55–77 (2012)
72.183 F. Yamaoka, T. Kanda, H. Ishiguro, N. Hagita: Developing a model of robot behavior to identify and appropriately respond to implicit attention-shifting, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2009) pp. 133–140
72.184 C. Shi, M. Shimada, T. Kanda, H. Ishiguro, N. Hagita: Spatial formation model for initiating conversation, Robotics Sci. Syst. Conf. (RSS) (2011)
72.185 C. Torrey, A. Powers, M. Marge, S.R. Fussell, S. Kiesler: Effects of adaptive robot dialogue on information exchange and social relations, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2006) pp. 126–133
72.186 Y. Morales, S. Satake, T. Kanda, N. Hagita: Modeling environments from a route perspective, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2011) pp. 441–448
72.187 T. Kollar, S. Tellex, D. Roy, N. Roy: Toward understanding natural language directions, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2010) pp. 259–266
72.188 T. Matsumoto, S. Satake, T. Kanda, M. Imai, N. Hagita: Do you remember that shop? – Computational model of spatial memory for shopping companion robots, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2012) pp. 447–454
72.189 T. Kanda, M. Shiomi, Z. Miyashita, H. Ishiguro, N. Hagita: A communication robot in a shopping mall, IEEE Trans. Robotics **26**(5), 897–913 (2010)
72.190 Y. Fernaeus, M. Håkansson, M. Jacobsson, S. Ljungblad: How do you play with a robotic toy animal?: A long-term study of Pleo, Proc. 9th Int. Conf. Interact. Des. Child. (IDC) (ACM, New York, USA 2010) pp. 39–48
72.191 K. Dautenhahn: Socially intelligent robots: Dimensions of human–robot interaction, Philos. Trans. R. Soc. B Biol, Sci. **362**(1480), 679–704 (2007)
72.192 COGNIRON, FP6 project, coordinated by LAAS: http://www.cogniron.org/final/Home.php
72.193 LIREC, FP7 project, coordinator Queen Mary University of London: http://lirec.eu/project
72.194 Companions, FP6 project, coordinated by University of Teesside: http://www.companions-project.org/
72.195 CompaniAble, FP7 project, coordinated by University of Reading: http://www.companionable.net/
72.196 http://www.openlivinglabs.eu/aboutus, see also the MIT Living labs, http://livinglabs.mit.edu/ or the Placelab http://architecture.mit.edu/house_n/placelab.html
72.197 K. Dautenhahn, S. Woods, C. Kaouri, M. Walters, K.L. Koay, I. Werry: What is a robot companion – Friend, assistant or butler?, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Edmonton (2005) pp. 1488–1493
72.198 S. Turkle: Authenticity in the age of digital companions, Interact. Stud. **8**(3), 501–517 (2007)
72.199 S. Turkle, W. Taggart, C.D. Kidd, O. Daste: Relational artifacts with children and elders: The complexities of cybercompanionship, Connect. Sci. **18**(4), 347–361 (2006)
72.200 M. Shiomi, T. Kanda, H. Ishiguro, N. Hagita: Interactive humanoid robots for a science museum, IEEE Intell. Syst. **22**, 25–32 (2007)
72.201 J.-Y. Sung, L. Guo, R.E. Grinter, H.I. Christensen: My Roomba Is Rambo: Intimate home appliances, Lect. Notes Comput. Sci. **4717**, 145–162 (2007)
72.202 H. Hüttenrauch, K.S. Eklundh: Fetch-and-carry with Cero: Observations from a long-term user study with a service robot, IEEE Int. Workshop Robot Hum. Interact. Commun. (ROMAN) (2002) pp. 158–163
72.203 S. Šabanović, M.P. Michalowski, R. Simmons: Robots in the wild: Observing human-robot social interaction outside the lab, Proceedings AMC 2006 (2006) pp. 576–581
72.204 D. Heylen, B. van Dijk, A. Nijholt: Robotic rabbit companions: Amusing or a nuisance?, J. Multimodal User Interfaces **5**, 53–59 (2012)
72.205 T. Salter, F. Michaud, H. Larouche: How wild is wild? A taxonomy to categorize the wildness of child-robot interaction, Int. J. Soc. Robotics **2**(4), 405–415 (2010)
72.206 T. Kanda, R. Sato, N. Saiwaki, H. Ishiguro: A two-month field trial in an elementary school for long-term human-robot interaction, IEEE Trans. Robotics **23**(5), 962–971 (2007)
72.207 H. Hűttenrauch, E.A. Topp, E.K. Severinson: The art of gate-crashing – Bringing HRI into users' homes, Interact. Stud. **10**(3), 274–297 (2009)
72.208 T. Bickmore, R. Picard: Establishing and maintaining long-term human-computer relation-

ships, ACM Trans. Comput. Hum. Interact. **59**(1), 21–30 (2005)
72.209 T. Bickmore, L. Caruso, K. Clough-Gorr, T. Heeren: It's just like you talk to a friend – Relational agents for older adults, Interact. Comput. **17**(6), 711–735 (2005)
72.210 K.L. Koay, D.S. Syrdal, M.L. Walters, K. Dautenhahn: Five weeks in the robot house – exploratory human-robot interaction trials in a domestic setting, IEEE 2nd Int. Conf. Adv. Comput.-Hum. Interact. (ACHI) (2008) pp. 219–226
72.211 M.E. Pollack, L. Brown, D. Colbry, C.E. McCarthy, C. Orosz, B. Peintner, I. Tsamardinos: Autominder: An intelligent cognitive orthotic system for people with memory impairment, Robotics Auton. Syst. **44**(3), 273–282 (2003)
72.212 A. Cesta, F. Pecora: The robocare project: Intelligent systems for elder care, AAAI Fall Symp. Caring Mach. AI Elder Care, USA (2005)
72.213 R. Cuijpers, M. Bruna, J. Ham, E. Torta: Attitude towards robots depends on interaction but not on anticipatory behaviour, Lect. Notes Comput. Sci. **7072**, 163–172 (2011)
72.214 C. Huijnen, A. Badii, H. van den Heuvel, P. Caleb-Solly, D. Thiemert: Maybe it becomes a buddy, but do not call it a robot – Seamless cooperation between companion robotics and smart homes, Lect. Notes Comput. Sci. **7040**, 324–329 (2011)
72.215 Florence, FP7 European project, coordinated by Philips Electronics Nederland B.V.: http://www.florence-project.eu/
72.216 SRS, FP7 European project, coordinated by Cardiff University: http://srs-project.eu/
72.217 ACCOMPANY, FP7 European project, coordinated by University of Hertfordshire: http://accompanyproject.eu/
72.218 Quality of Life Technology Center: http://www.cmu.edu/qolt/index.html
72.219 CASALA, Dundalk Institute of Technology: http://www.casala.ie/
72.220 Intuitive Automata: http://www.intuitiveautomata.com/
72.221 B.J. Fogg: Persuasive computers: Perspectives and research directions, Proc. ACM/SIGCHI Conf. Hum. Factors Comput. Syst., ed. by C.-M. Karat, A. Lund, J. Coutaz, J. Karat (1998) pp. 225–232
72.222 B. Mutlu, T. Kanda, J. Forlizzi, J. Hodgins, H. Ishiguro: Conversational gaze mechanisms for human-like robots, ACM Trans, Interact. Intell. Syst. **1(2)**, 33 (2012)
72.223 M. Salem, S. Kopp, I. Wachsmuth, K. Rohlfing, F. Joublin: Generation and evaluation of communicative robot gesture, Int. J. Soc. Robotics **4**(2), 201–217 (2012)
72.224 J. Mumm, B. Mutlu: Human-robot proxemics: Physical and psychological distancing in human-robot interaction, ACM/IEEE 6th Int. Conf. Hum.-robot Interact. (HRI) (2011) pp. 331–338
72.225 F. Yamaoka, T. Kanda, H. Ishiguro, N. Hagita: How contingent should a lifelike robot be? The Relationship between contingency and complexity, Connect. Sci. **19**(2), 143–162 (2007)
72.226 M. Shimada, T. Kanda: What is the appropriate speech rate for a communication robot?, Interact. Stud. **13**(3), 408–435 (2012)
72.227 D. François, D. Polani, K. Dautenhahn: Towards socially adaptive robots: A novel method for real time recognition of human-robot interaction styles, Proc. Humanoids 2008, Daejeon (2008) pp. 353–359
72.228 D. Szafir, B. Mutlu: Pay attention! Designing adaptive agents that monitor and improve user engagement, Proc. 30th ACM/SIGCHI Conf. Hum. Factors Comput. (2012)
72.229 R. Rose, M. Scheutz, P. Schermerhorn: Towards a conceptual and methodological framework for determining robot believability, Interact. Stud. **11**(2), 314–335 (2010)
72.230 J. Forlizzi, C. DiSalvo: Service robots in the domestic environment: A study of the roomba vacuum in the home, Proc. 1st ACM SIGCHI/SIGART Conf. Hum.-Robot Interact. (2006) pp. 258–265
72.231 N.E. Sharkey, A.J.C. Sharkey: The crying shame of robot nannies: An ethical appraisal, J. Interact. Stud. **11**, 161–190 (2010)
72.232 A. Sharkey, N. Sharkey: Granny and the robots: Ethical issues in robot care for the elderly, Ethics Inform. Technol. **14**(1), 27–40 (2012)
72.233 D. Robert, C. Breazeal: Blended reality characters, Proc. Seventh Annu. ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2012) pp. 359–366
72.234 M. Imai, T. Ono, T. Etani: Agent migration: Communications between a human and robot, IEEE Int. Conf. Syst. Man Cybern., Vol. 4 (1999) pp. 1044–1048
72.235 D.S. Syrdal, K.L. Koay, M.L. Walters, K. Dautenhahn: The boy-robot should bark! – Children's impressions of agent migration into diverse embodiments, Proc. New Front. Hum.-Robot Interact. Symp. AISB Convention (2009) pp. 116–121
72.236 K.L. Koay, D.S. Syrdal, M.L. Walters, K. Dautenhahn: A user study on visualization of agent migration between two companion robots, 13th Int. Conf. Hum.-Comput. Interact. (HCII) (2009)
72.237 E.M. Segura, H. Cramer, P.F. Gomes, S. Nylander, A. Paiva: Revive!: Reactions to migration between different embodiments when playing with robotic pets, Proc. 11th Int. Conf. Interact. Des. Child. (IDC) (2012) pp. 88–97
72.238 M. Kriegel, R. Aylett, P. Cuba, V.M.A. Paiva: Robots meet IVAs: A mind-body interface for migrating artificial intelligent agents, Proc. Intell. Virtual Agents, Reykjavik (2011)
72.239 K.L. Koay, D.S. Syrdal, K. Dautenhahn, K. Arent, L. Malek, B. Kreczmer: Companion migration – Initial participants' feedback from a video-based prototyping study. In: *Mixed Reality and Human-Robot Interact*, ed. by X. Wang (Springer, Berlin, Heidelberg 2011) pp. 133–151
72.240 W.C. Ho, M. Lim, P.A. Vargas, S. Enz, K. Dautenhahn, R. Aylett: An initial memory model for virtual and robot companions supporting migration and long-term interaction, 18th IEEE Int.

Symp. Robot Hum. Interact. Commun. (ROMAN) (2009)
72.241 G.M.P. O'Hare, B.R. Duffy, J.F. Bradley, A.N. Martin: Agent chameleons: Moving minds from robots to digital information spaces, Proc. Auton. Minirobots Res. Edutainment (2003) pp. 18–21
72.242 Y. Sumi, K. Mase: AgentSalon: Facilitating face-to-face knowledge exchange through conversations among personal agents, Proc. 5th Int. Conf. Auton. Agents (AGENTS) (2001) pp. 393–400
72.243 P.K. Allen, A. Miller, P.Y. Oh, B.B. Leibowitz: Integration of vision, force and tactile sensing for grasping, Int. J. Intell. Mach. **4**(1), 129–149 (1999)
72.244 M. Siegel, M.C. Breazeal, M. Norton: Persuasive robotics: The influence of robot gender on human behavior, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 2563–2568
72.245 T. Chen, C.-H. King, A. Thomaz, C. Kemp: Touched by a robot: An investigation of subjective responses to robot-initiated touch, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2011)
72.246 P.K. Davis: *The Power of Touch – The Basis for Survival, Health, Intimacy, and Emotional Well-Being* (Hay House, Carlsbad 1999)
72.247 M.J. Hertenstein, J.M. Verkamp, A.M. Kerestes, R.M. Holmes: The communicative functions of touch in humans, non-human primates, and rats: A review and synthesis of the empirical research, Genet. Soc. Gen. Psychol. Monogr. **132**(1), 5–94 (2006)
72.248 Roboskinproject
72.249 A. Schmitz, P. Maiolino, M. Maggiali, L. Natale, G. Cannata, G. Metta: Methods and technologies for the implementation of large-scale robot tactile sensors, IEEE Trans. Robotics **27**(3), 389–400 (2011)
72.250 R.S. Dahiya, M. Getta, M. Valle, G. Sandini: Tactile sensing – From humans to humanoids, IEEE Trans. Robotics **26**(1), 1–20 (2010)
72.251 W. Stiehl, L. Lalla, C. Breazeal: A somatic alphabet approach to sensitive skin for robots, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), New Orleans (2004) pp. 2865–2870
72.252 W. Stiehl, C. Breazeal: Design of a therapeutic robotic companion for relational, affective touch, Proc. 14th IEEE Workshop Robot Hum. Interact. Commun. (ROMAN), Nashville (2005) pp. 408–415
72.253 W.D. Stiehl, C. Breazeal: A sensitive skin for robotic companions featuring temperature, force and electric field sensors, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 1952–1959
72.254 B.D. Argall, A. Billard: A survey of tactile human-robot interactions, Robotics Auton. Syst. **58**(10), 1159–1176 (2010)
72.255 J.K. Lee, R.L. Toscano, W.D. Stiehl, C. Breazeal: The design of a semi-autonomous robot avatar for family communication and education, Proc. 17th IEEE Int. Symp. Robot Hum. Interact. Commun. (ROMAN) (2008) pp. 166–173
72.256 F. Mueller, F. Vetere, M. Gibbs, J. Kjeldskov, S. Pedell, S. Howard: Hug over a distance, Proc. Conf. Hum. Factors Comput. Syst. (2005) pp. 1673–1676
72.257 J.K.S. The, A.D. Cheok, R.L. Peiris, Y. Choi, V. Thuong, S. Lai: Huggy Pajama: A mobile parent and child hugging communication system, Proc. 7th Int. Conf. Interact. Des. Child. (IDC) (2008) pp. 250–257
72.258 F. Papadopoulos, K. Dautenhahn, W.C. Ho: Exploring the use of robots as social mediators in a remote human-human collaborative communication experiment, Paladyn **3**(1), 1–10 (2012)
72.259 B.D. Argall, E. Sauser, A. Billard: Tactile guidance for policy adaptation, Found. Trends Robotics **1**(2), 79–133 (2011)
72.260 T. Noda, T. Miyashita, H. Ishiguro, N. Hagita: Super-flexible skin sensors embedded on the whole body, self-organizing based on haptic interactions, Robotics Sci. Syst. Conf. (2008)
72.261 T. Tajika, T. Miyashita, H. Ishiguro, N. Hagita: Automatic categorization of haptic interactions-what are the typical haptic interactions between a human and a robot?, Proc. 6th IEEE-RAS Int. Conf. Humanoid Robots (Humanoids), Genova (2006)
72.262 H. Knight, R. Toscano, W.D. Stiehl, A. Chang, Y. Wang, C. Breazeal: Real-time social touch gesture recognition for sensate robots, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 3715–3720
72.263 S. Yohanan, K.E. MacLean: The role of affective touch in human-robot interaction: Human intent and expectations in touching the haptic creature, Int. J. Soc. Robotics **4**, 163–180 (2011)
72.264 W.D. Stiehl, C. Breazeal: Affective touch for robotic companions, Proc. Affect. Comput. Intell. Interact., Bejing (2005)
72.265 D. François, K. Dautenhahn, D. Polani: Using real-time recognition of human-robot interaction styles for creating adaptive robot behaviour in robot-assisted play, Proc. 2nd IEEE Symp. Artif. Life, Nashville (2009) pp. 45–52
72.266 Aurora project, University of Hertfordshire: http://www.aurora-project.com/
72.267 K. Dautenhahn: Robots as social actors: AURORA and the case of autism, Proc. 3rd Int. Cogn. Technol. Conf., San Francisco (1999)
72.268 American Psychiatric Association : *Diagnostic and Statistical Manual of Mental Disorders DSM-IV* (APA, Washington 1995)
72.269 R. Jordan: *Autistic Spectrum Disorders – An Introductory Handbook for Practitioners* (David Fulton, London 1999)
72.270 K. Dautenhahn, I. Werry: Towards interactive robots in autism therapy: Background, motivation and challenges, Pragmat. Cogn. **12**(1), 1–35 (2004)
72.271 I. Werry, K. Dautenhahn, B. Ogden, W. Harwin: Can social interaction skills be taught by a social agent? The role of a robotic mediator in autism therapy, Proc. 4th Int. Conf. Cogn. Technol. Instrum. Mind (CT), ed. by M. Beynon, C.L. Nehaniv, K. Dautenhahn (Springer, London 2001) pp. 57–74

72.272 B. Scassellati, H. Admoni, M. Mataric: Robots for use in autism research, Annu. Rev. Biomed. Eng. **14**, 275–294 (2012)

72.273 I. Werry, K. Dautenhahn, W. Harwin: Evaluating the response of children with autism to a robot, Proc. RESNA Annu. Conf. Rehabil. Eng. Assist. Technol. Soc. N. Am., Nevada (2001)

72.274 A. Tapus, A. Peca, A. Aly, C. Pop, L. Jisa, S. Pintea, A.S. Rusi, D.O. David: Children with autism social engagement in interaction with Nao, an imitative robot – A series of single case experiments, Interact. Stud. **13**(3), 315–347 (2012)

72.275 B. Vanderborght, R. Simut, J. Saldien, C. Pop, A.S. Rusu, S. Pintea, D. Lefeber, D.O. David: Using the social robot probo as a social story telling agent for children with ASD, Interact. Stud. **13**(3), 348–372 (2012)

72.276 A. Billard, B. Robins, K. Dautenhahn, J. Nadel: Building robota, a mini-humanoid robot for the rehabilitation of children with autism, RESNA Assist. Technol. J. **19**(1), 37–49 (2006)

72.277 B. Robins, K. Dautenhahn, R. te Boekhorst, A. Billard: Robotic assistants in therapy and education of children with autism: Can a small humanoid robot help encourage social interaction skills?, Univ. Access Inform. Soc. **4**(2), 105–120 (2005)

72.278 H. Kozima, M.P. Michalowski, C. Nakagawa: Keepon: A playful robot for research, therapy, and entertainment, Int. J. Soc. Robotics **1**(1), 3–18 (2009)

72.279 D. François, S. Powell, K. Dautenhahn: A long-term study of children with autism playing with a robotic pet: Taking inspirations from non-directive play therapy to encourage children's proactivity and initiative-taking, Interact. Stud. **10**(3), 324–373 (2009)

72.280 A. Duquette, F. Michaud, H. Mercier: Exploring the use of a mobile robot as an imitation agent with children with low-functioning autism, Auton. Robots **24**, 147–157 (2008)

72.281 J.J. Diehl, L.M. Schmitt, M. Villano, C.R. Crowell: The clinical use of robots for individuals with autism spectrum disorders: A critical review, Res. Autism Spectr. Disord. **6**, 249–262 (2012)

72.282 KASPAR, University of Hertfordshire: http://www.kaspar.herts.ac.uk/

72.283 B. Robins, K. Dautenhahn: Developing play scenarios for tactile interaction with a humanoid robot: A case study exploration with children with autism, Lect. Notes Comput. Sci. **6414**, 243–252 (2010)

72.284 B. Robins, E. Ferrari, K. Dautenhahn, G. Kronrief, B. Prazak, G.J. Gerderblom, F. Caprino, E. Laudanna, P. Marti: Human-centred design methods: Developing scenarios for robot assisted play informed by user panels and field trials, Int. J. Hum.-Comput. Stud. **68**, 873–898 (2010)

72.285 J. Wainer, K. Dautenhahn, B. Robins, F. Amirabdollahian: Collaborating with Kaspar: Using an autonomous humanoid robot to foster cooperative dyadic play among children with autism, Proc. IEEE-RAS Int. Conf. Humanoid Robots (2010) pp. 631–638

72.286 B. Robbins, K. Dautenhahn, E. Ferrari, G. Kronreif, B. Prazak-Aram, P. Marti, I. Iacono, G.J. Gelderblom, T. Bernd, F. Caprino, E. Laudanna: Scenarios of robot-assisted play for children with cognitive and physical disabilities, Interact. Stud. **13**(2), 189–234 (2012)

72.287 B. Robins, K. Dautenhahn, P. Dickerson: Embodiment and cognitive learning – Can a humanoid robot help children with autism to learn about tactile social behaviour?, Lect. Notes Comput. Sci. **7621**, 6675 (2012)

72.288 Z. Ji, F. Amirabdollahian, D. Polani, K. Dautenhahn: Histogram based classification of tactile patterns on periodically distributed skin sensors for a humanoid robot, Proc. 20th IEEE Int. Symp. Robot Hum. Interact. Commun. (ROMAN) (2011) pp. 433–440

72.289 B. Robins, F. Amirabdollahian, Z. Ji, K. Dautenhahn: Tactile interaction with a humanoid robot for children with autism: A case study analysis involving user requirements and results of an initial implementation, Proc. 19th IEEE Int. Symp. Robot Hum. Interact. Commun. (ROMAN) (2010) pp. 704–711

72.290 H. Cramer, N. Kemper, A. Amin, B. Wielinga, V. Evers: Give me a hug: The effects of touch and autonomy on people's responses to embodied social agents, Comput. Animat. Virtual Worlds **20**, 437–445 (2009)

72.291 E.S. Kim, L.D. Berkovits, E.P. Bernier, D. Leyzberg, F. Shic, R. Paul, B. Scassellati: Social robots as embedded reinforcers of social behavior in children with autism, J. Autism Dev. Disord. **43**(5), 1038–1049 (2012)

72.292 G. Hoffman, C. Breazeal: Cost-based anticipatory action selection for human–robot fluency, IEEE Trans. Robotics **23**, 952–961 (2007)

72.293 G. Hoffman, C. Breazeal: Effects of anticipatory perceptual simulation on practiced human-robot tasks, Auton. Robots **28**, 403–423 (2010)

72.294 T. Spexard, S. Li, B. Wrede, J. Fritsch, G. Sagerer, O. Booij, Z. Zivkovic, B. Terwijn, B. Kröse: Biron, Where are you? Enabling a robot to learn new places in a real home environment by integrating spoken dialog and visual localization, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 934–940

72.295 S. Woods, K. Dautenhahn, C. Kaouri: Is someone watching me? – Consideration of social facilitation effects in human-robot interaction experiments, IEEE Int. Symp. Comput. Intell. Robotics Autom. (CIRA) (2005) pp. 53–60

72.296 N. Riether, F. Hegel, B. Wrede, G. Horstmann: Social facilitation with social robots?, 7th ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2012) pp. 41–47

72.297 E. Takano, T. Chikaraishi, Y. Matsumoto, Y. Nakamura, H. Ishiguro, K. Sugamoto: Psychological effects on interpersonal communication by by-

stander android using motions based on human-like needs, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2009) pp. 3721–3726

72.298 M. Shiomi, T. Kanda, S. Koizumi, H. Ishiguro, N. Hagita: Group attention control for communication robots with Wizard of Oz approach, ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI) (2007) pp. 121–128

72.299 M. Shiomi, K. Nohara, T. Kanda, H. Ishiguro, N. Hagita: Estimating group states for interactive humanoid robots, IEEE-RAS Int. Conf. Humanoid Robots (Humanoids) (2009) pp. 318–323

72.300 Y. Matsuyama, H. Taniyama, S. Fujie, T. Kobayashi: Framework of communication activation robot participating in multiparty conversation, AAAI Fall Symp. Ser. (2010) pp. 68–73

Multimedia Contents

73. Socially Assistive Robotics

Maja J. Matarić, Brian Scassellati

This chapter reviews the critical societal issues that have motivated research into socially assistive robotics (SAR) (Sect. 73.2) and describes the reason why *physical robots* rather than virtual agents are essential to this effort (Sect. 73.3). It highlights the major research issues within this area (Sects. 73.4–73.7), describes the primary application domains and populations where SAR research has shown an impact (Sects. 73.8–73.11), and closes with some of the ethical and safety issues that SAR research necessitates (Sect. 73.12).

73.1 **Overview** 1973
73.2 **The Need for Socially Assistive Robotics** 1974
73.3 **Advantages of Embodied Robots over Virtual Agents** 1975
73.3.1 Embodiment and Social Influence 1975
73.3.2 Embodiment and Learning 1976
73.3.3 Embodiment and Special Populations 1976
73.3.4 Evaluation of SAR Systems 1977
73.4 **Motivation, Autonomy, and Companionship** 1977
73.5 **Influence and the Dynamics of Assistive Interaction** 1978
73.6 **Personalization and Adaptation to Specific Needs and Abilities** 1978
73.7 **Creating Long-Term Engagement and Behaviour Change** 1979
73.8 **SAR for Autism Spectrum Disorder (ASD) Therapy** 1980
73.8.1 Summary of ASD-Related Challenges 1980
73.8.2 User Study Insights 1980
73.8.3 Research Directions 1981
73.9 **SAR Supporting Rehabilitation** 1982
73.9.1 Summary of Rehabilitation Challenges 1982
73.9.2 Contact Versus SAR (Noncontact) Robotics for Rehabilitation 1982
73.9.3 User Study Insights 1983
73.9.4 Research Directions 1984
73.10 **SAR and Eldercare** 1985
73.10.1 Summary of Eldercare Challenges 1985
73.10.2 User Study Insights 1985
73.11 **SAR for Alzheimer's Dementia and Cognitive Rehabilitation** 1986
73.11.1 User Study Insights 1986
73.11.2 Research Directions 1987
73.12 **Ethical and Safety Considerations** 1987
References 1988

73.1 Overview

As robots become increasingly integrated into human environments, different forms of interaction are spawning new research topics and subfields. Human–robot interaction (HRI) can be broadly classified into two categories: physical and social/emotional. Physical HRI (Chap. 69) involves the research areas of manipulation and haptics, among others, and is used in medical and rehabilitation robotics (Chaps. 63 and 64). In contrast, social/emotional interaction involves verbal and nonverbal expression and communication, and thus the research areas of assistive robotics, social robotics, and SAR.

Robots have long been used to provide assistance to individual users through physical interaction, typically by supporting direct physical rehabilitation (Chap. 64) or by providing a service such as retrieving items or cleaning floors (Chap. 55). Socially assistive robotics (SAR) is a comparatively new field of robotics that focuses on developing robots capable of assisting users through *social* rather than *physical* interaction. Just as a good coach or teacher can provide motivation, guidance, and support without making physical contact with a student, socially assistive robots attempt to provide the appropriate emotional, cognitive, and social cues to encourage development, learning, or therapy for an individual. SAR focuses on the challenges of providing motivation, coaching, training, and rehabilitation through nonphysical interaction; such systems have been validated in hands-off stroke rehabilitation, social skill training of children with autism, and eldercare, among others. In contrast, most of rehabilitation robotics focuses on physical interaction with the patient; such systems have been validated in hands-on stroke rehabilitation. Social robotics (Chap. 72) focuses on endowing robots with the ability to behave in socially aware and engaging ways; such systems have been validated in museums, movies, classrooms, and informal settings. Service robotics can be seen as the overarching field that encompasses much of the work above.

A socially assistive robot is a system that employs hands-off interaction strategies, including the use of speech, facial expressions, and communicative gestures, to provide assistance in accordance with a particular assistive context. SAR systems equipped with motivational, social, pedagogical, and therapeutic capabilities have the potential to improve access to personalized care, training and rehabilitation to large populations, including individuals post-stroke, elderly, and children with social and developmental disabilities, in order to enhance their quality of life. An effective socially assistive robot must understand and interact with its environment, exhibit social behaviour, focus its attention and communication on the user, sustain engagement with the user, and achieve specific assistive goals. The robot must do all of this in a way that is safe, ethical and effective for the potentially vulnerable user. Socially assistive robots have been shown to have promise as therapeutic tool for children, the elderly, stroke patients, and other special-needs populations requiring personalized care.

73.2 The Need for Socially Assistive Robotics

The need for socially assistive robots is driven not only by rapidly growing populations that need additional assistance (including the elderly, individuals with cognitive disabilities, and individuals who require nonphysical support during rehabilitation) but also by the recognition that social robots can have a substantial positive impact on personal challenges such as maintaining a healthy diet, supporting an active lifestyle, maintaining therapeutic programs, and supporting individualized education [73.1].

One of the most rapidly expanding populations requiring daily assistance is the elderly. The United Nations estimates that the population of those aged 60 and older will double in the next thirty years, with the elderly outnumbering children for the first time in history by 2045 [73.2]. Space and staff shortages at nursing homes and other care facilities are already an issue today. As the elderly population continues to grow, a great deal of attention and research will be dedicated to assistive systems aimed at promoting ageing-in-place, facilitating living independently in one's own home as long as possible. While many of these users will require some degree of physical assistance (such as standing up or carrying heavy groceries), many more are expected to require social and cognitive support to reduce isolation, maintain active and healthy lifestyles, and to assist in daily therapy routines.

Individuals with cognitive disabilities and developmental and social disorders constitute another key growing population that can benefit from SAR in the contexts of special education, therapy, and training. Autism spectrum disorders (ASDs) have been the focus of much of the research in this area due to their prevalence, rapid growth of diagnoses, inherently social nature, and the critical need for long term, intensive treatment. Current research suggests that 1 in every 60 children in the United States will be diagnosed with ASD; the rate of diagnosis has increased sixfold between 1994 and 2003 [73.3]. The cause of the increase is not yet known. However, early intervention is critical to a positive long-term outcome, and many individuals need high levels of support and care throughout their lives [73.4–8]. Research into applying robots as therapy tools for ASD has focused on how to support repetitive, but necessary, social and behavioral therapies [73.9].

Motivation is recognized as the most significant challenge in physical rehabilitation and training [73.10]. SAR technology has the potential to provide novel means for monitoring, motivating, and

coaching. Post-stroke rehabilitation is one of the largest potential application domains, since stroke is a dominant cause of severe disability in the growing ageing population. In the United States alone, over 700 000 people suffer a new stroke each year, with the majority sustaining some permanent loss of movement [73.11]. Stroke patients frequently have difficulty with everyday functional movements and activities; the loss of function can be decreased through rehabilitation therapy during the critical post-stroke period. Such rehabilitation therapy involves carefully designed repetitive exercises, which can be passive and active. In passive exercises, the therapist (or a robot) actively helps the patient to repeatedly move the stroke-affected limb as prescribed. In active exercises, the patient does the work him/herself, with no physical assistance.

Finally, there is growing interest in using socially assistive robots to support an active and healthy lifestyle. These applications are again motivated by rapidly growing prevalence of obesity and a lack of effective interventions that are not labor-intensive. Childhood and adolescent obesity has been related to a host of adverse proximal and distal health outcomes, including high cholesterol and triglycerides, hypertension [73.12], insulin resistance [73.13], type 2 diabetes [73.14], nonalcoholic fatty liver disease [73.15], as well as breast, colorectal, and other cancers [73.16]. Furthermore, childhood obesity sustains adverse consequences over the entire life span. In one review, all of the 25 longitudinal studies surveyed showed that overweight and obese youth were at higher risk of becoming overweight adults [73.17]. Per capita medical spending for an obese person is roughly 42% higher than for a person of normal weight [73.18] and outranks the health costs of both smoking and drinking alcohol [73.19]. Understanding, preventing, and treating childhood obesity is therefore a top public health priority [73.20]. Education about health impacts of obesity in addition to coaching and motivation for a healthy lifestyle are known to be effective means of lowering obesity rates [73.21]. SAR has the potential to play an important role in this context as an affordable and accessible means of providing personalized coaching, motivation, and training for exercise and healthy eating habits [73.22].

73.3 Advantages of Embodied Robots over Virtual Agents

Compared to virtual agents, robotic agents are typically more expensive, more difficult to replicate and distribute, more difficult to maintain and support, require a longer development time, and more fragile and susceptible to damage and system failure. In physically assistive tasks, virtual agent alternatives are not possible; manipulation of physical objects requires a physical presence. For socially assistive tasks, the added costs of a physical robot must be justified in comparison with a virtual agent solution.

73.3.1 Embodiment and Social Influence

One primary reason for using a physical robot in a socially assistive task is that robots can often exert more social influence over the user compared to a virtual agent. Socially assistive tasks often require the user to engage in tasks (such as therapy or homework) that are challenging and that the user might need encouragement or guidance to maintain. Multiple studies have demonstrated that physically embodied and collocated robots are more capable of generating the social presence to convince users to take these challenging steps. A physically present robot yields significantly more compliance to its requests, even when those requests are unexpected or unconventional, than a video representation of the same robot [73.23]. *Kidd* and *Breazeal* [73.24] developed a table-top robot to serve as a daily weight-loss advisor; it engaged users through a touch–screen interface, tracked user progress and the state of the user–robot relationship over time, and was tested in a six-week field study with participants at home. The study compared the robot interactions against alternative weight loss methods, including using a standalone computer and maintaining paper logs, and found that participants exercised and tracked their caloric intake for nearly twice as long when interacting with the robot than with the other methods.

Interactions with physical robots are also more engaging and credible than virtual agents in cooperative task scenarios. For example, following a cooperative block-stacking task with a talking agent, participants found an agent more engaging, enjoyable, informative, and credible if it were a physically embodied robot, than if it were a virtually embodied animated character [73.24]. Other work has shown that after playing a cooperative game, people most enjoyed, and attributed the most watchfulness and helpfulness to a physically embodied robot than to a virtual embodiment (a graphical simulation) of the same robot [73.26]. *Bartneck* [73.27] conducted a study comparing the effectiveness of an emotionally expressive robot, eMuu, with its screen character version in engaging users in a simple negotiation task, and found that participants

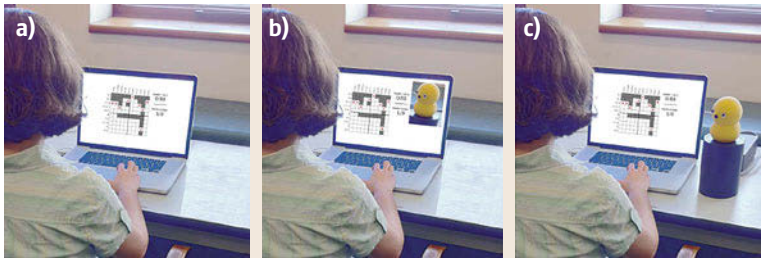


Fig.73.1a–c The physical presence of a robot tutor **(c)** has been shown to increase learning gains during a puzzle solving task substantially more than when the same tutoring lessons are delivered by a video of the same robot **(b)** or from a disembodied voice **(a)** (after [73.25])

exerted more effort and received higher task scores when interacting with the physical eMuu than with the simulated eMuu. *Jung* and *Lee* [73.28] also demonstrated the positive effects of physical embodiment in relation to interactions with both a Sony Aibo robot and an anthropomorphic dancing robot, April. Another study compared a physical co-present robot, to a remotely located one shown on the video screen, to a simulation of the same robot, all in the same interaction context; the study participants rated the co-present robot as more helpful, watchful, and appealing than remote and simulated counterparts [73.29]. These findings suggest that physical embodiment affords greater social attribution or enjoyment of an agent, than does virtual embodiment.

73.3.2 Embodiment and Learning

Physical robots have also been shown to enhance learning and impact later behavioral choice more substantially than virtual agents. *Leyzberg* et al. [73.25] showed that even when delivering the same lessons on solving cognitive puzzles, users improved their performance during a 20 min session almost twice as much when those lessons were delivered by a physical robot rather than by an on-screen video of the same robot. *Kiesler* et al. [73.30] used task performance measures to find that participants who received health advice from a physically present robot were more likely to choose a healthy snack than participants who received the same information in robot-video or on-screen agent conditions. *Kidd* and *Breazeal* [73.22] demonstrated that users were more likely to continue sessions with a robot weight-loss coach than with a virtual coach over a period of six weeks (Fig. 73.1).

73.3.3 Embodiment and Special Populations

While most of the investigations of the impact of physical embodiment on socially assistive tasks have been conducted with typical adults, embodiment effects have also been demonstrated on many of the particular populations that SAR projects typically target. *Tapus* and *Matarić* [73.31] found that individuals suffering from cognitive impairment and/or Alzheimer's disease reported being more engaged with a robot treatment than a similar on-screen agent treatment. Numerous studies have demonstrated that children and adolescents with ASD engage readily with robots even when those children are unlikely to engage with animated characters on a screen [73.9, 32].

This impact of embodiment on preschool-age children has been of particular interest to educators and developmental psychologists. Although preschoolers learn from watching television and on-screen agents, many studies have focused on how learning and behavior shaping are more robust and generalize more readily when embedded in real time interactions with a physical agent:

1. Preschool children are multimodal learners, learning and retaining information better if there are overlapping modalities of experience [73.33–35].
2. Infants and preschool children off-load cognitive functions to the 3-D (three-dimensional) spatial structure of the world, for example, remembering the direction of attention rather than the content and then looking back to a location to re-perceive the content [73.36–38].
3. Children learn by physically manipulating and grouping objects [73.39, 40].
4. Preschool children learn from gross (whole body) movements; they do not have the fine motor skills to work in a limited spatial arena (such as a computer screen) and do not sustain attention when their gross motor activity is limited [73.41].

These facts suggest that learning and development will be best achieved in physical environments with physically realized teachers. Robotics may offer a mechanism to augment and support teacher–child and parent–child interactions.

Embodiment studies that have targeted the elderly population include the work of *Heerink* et al. [73.42], who investigated the acceptance of assistive social agents by older adults. The robot used in their evaluation was a table-top robot (the iCat), and was either controlled via a human operator during interaction with

elderly users (Wizard of Oz study), or interacted with users through a touch–screen interface. The interactions consisted primarily of short informational or utility interactions (e.g., medication/agenda reminders, weather forecast, companionship). *Fasola* and *Matarić* [73.43] also investigated the role of embodiment in a SAR exercise coach system, with a between-subjects study with participants split into two groups: physical robot coach versus virtual robot coach. Participants engaged in four exercise sessions over a two-week period, with each session lasting 20 min. The results of the between-subjects data analysis showed participants rated the physical robot significantly more positively than the virtual robot across all of the subjective measures, including enjoyableness and usefulness of the interaction, and also in terms of helpfulness, intelligence, social presence, and companionship of the robot coach. Direct comparison among the conditions also favored the physical robot significantly.

73.3.4 Evaluation of SAR Systems

One of the unique features of SAR is its evaluation process. All robotics requires the implementation of a physical system and its evaluation in the physical world, ideally outside of the laboratory setting, as close to the intended final use as possible. In reality, however, the majority of research robots are evaluated in laboratory settings, because the alternative is prohibitive due to cost or access or both.

HRI puts the human in the loop and therefore requires evaluation with human users. Since human behavior is complex, it is not readily simulated. Consequently, SAR, which involves HRI, also requires evaluation with human users. However, to provide truly relevant and realistic evaluation for SAR systems, such systems must be tested with the intended populations, since testing with healthy adults is not appropriate for systems aimed at stroke patients, testing with adults is not appropriate for systems aimed at children, and so on.

Consequently, for a SAR approach to be truly evaluated, it needs to be tested with members of the intended end-user population, the more rigorously the better, according to the principles and best practices of human subject evaluations, as well as consistent with all the ethical rules associated with such procedures (Sect. 73.11). As a result, SAR development and evaluation can be time-consuming and requires multidisciplinary expertise that goes beyond robotics into social science and, as relevant, health science or education, among other disciplines.

73.4 Motivation, Autonomy, and Companionship

SAR systems equipped with motivational, social, and therapeutic capabilities have the potential to enhance user skills and quality of life. Motivation is a fundamental tool in establishing adherence to a training, rehabilitation, or therapy regimen and in promoting behavior change; therefore methods aimed at improving motivation are likely to be useful in the SAR contexts. There are two forms of motivation: intrinsic motivation, which comes from within a person, and extrinsic motivation, which comes from sources external to a person. Intrinsically motivated activities are those characterized by enjoyment [73.44]. Extrinsic motivation, though effective for short-term task compliance, has been shown to be less effective than intrinsic motivation for long-term task compliance and behavior change [73.45]. Intrinsic motivation, however, can be and often is affected by external factors, thereby offering a promising avenue for future SAR research. In a task scenario, for example, the instructor (or a socially assistive robot) can impact the user's intrinsic motivation through verbal feedback. Praise is considered a form of positive feedback and has the potential to increase the user's intrinsic motivation for performing the task, while criticism, a form of negative feedback, tends to negatively impact the user's intrinsic motivation [73.46, 47]. The effect of positive feedback is closely tied to the user's own perceived competence at the task; once the user believes he is competent at the task, then additional praise no longer affects his intrinsic motivation.

Verbal feedback provided to the user by the instructor plays an important role in task-based motivation, but the task itself and how it is presented perhaps plays an even more significant role. *Csikszentmihalyi*'s research suggests that [73.44]:

> *when one engages in an optimally challenging activity with respect to one's capacities there is a maximal probability for task-involved enjoyment or flow.*

Therefore, an instructor that oversees user performance in a task scenario should also be aware of the user's current affective state or try to infer that state from task performance. The task must be continually adjusted to meet the appropriate needs of the user in order to increase or maintain intrinsic motivation to perform the task.

Another task characteristic with the potential to influence user enjoyment is the incorporation of direct

user input. Studies have shown that tasks that support user autonomy and self-determination lead to increased intrinsic motivation, self-esteem, creativity, and other related variables among the participants [73.48], all of which are essential for achieving task adherence and long term behavior change. Self-determination represented in the task such as choice of activity [73.49], choice of difficulty level [73.50], and choice of rewards [73.51] have been shown to increase or be less detrimental to intrinsic motivation when compared to similar task conditions that do not involve choice. The positive impact of self-determination in task scenarios has also been supported by studies investigating behavior change [73.52, 53].

73.5 Influence and the Dynamics of Assistive Interaction

While social robotics has approached the issues of modeling the dynamic interplay of social interactions (Chap. 69), assistive interactions are unique among these social engagements as they must simultaneously support the needs of the engagement itself (by maintaining interest, novelty, and the conventions of social behavior) as well as guide the interaction toward the long-term behavioral or educational goals of the system. These two goals can often be in conflict; at times a good teacher (or coach, or socially assistive robot) must sacrifice some of the enjoyment of the interaction, or bend a social rule, in order to promote an educational goal. *Huang* and *Mutlu* [73.54] recently found some evidence that a robot could make some of these tradeoffs by changing its nonverbal behavior, focusing either on improving task performance (recall in a storytelling task) or on improving social engagement by varying the type of gaze behavior used by the robot.

Because the socially assistive robot must take an active part in shaping this interaction toward particular goals, the nature of the representations and the way in which this influence can be applied to shape the dynamics of the assistive interaction represents a core research question for SAR. Researchers address questions such as:

- What abstractions allow a robot to model the unique dynamics of a social interaction with a user in order to shape the user's behavior?
- Can we define a set of computational primitives that describe social roles, concepts, or properties in such interactions?
- Can this primitive set provide a basis of representations that allows robots to capture in real time the nuances of interaction, to shape and have their behavior shaped by that interaction, while allowing generalization and comparisons across users, time scales, and contexts?

Much of the research in this area is based upon foundational psychological properties of human–human social interaction including agency, intentionality, causality, social roles, and goal attribution. These foundational elements discriminate between agents and objects, delineate which elements of the world can move with goal-directed purpose, and provide the primitive structure for describing cause and effect. If we are to build socially assistive robots, these systems must see the world in the same way that we humans do, in the same categories of animate and inanimate, and attribute cause and blame in the same way that we do.

Current computational models represent these features explicitly, with users overtly detailing their goals and intentions in an operationalized way [73.55]. In contrast, people are able to detect and analyze each other's agency and intentions *on line*, in a way that is often implicit, automatic, effortless, and even irresistible. To be successful, a socially assistive robot must understand, model, and engage in the dynamics of social interaction, so that computational systems can detect, analyze, and manipulate agency, intention, and other social primitives in dynamic environments in a way that is akin to how people do the same for social perception and behavior. These models must operate over a wide range of time scales and across input modalities (visual, auditory, tactile, etc.) and must be capable of interpreting and responding to social overtures in real time and with only minimal latency. Multiple studies have examined particular foundational aspects of the dynamics of interaction [73.56–58] as well as how persuasion might occur within those dynamics [73.59], but a complete model of the dynamics of social interaction remains one of the key research areas addressed within SAR.

73.6 Personalization and Adaptation to Specific Needs and Abilities

A second core research area for SAR concerns the personalization of the robot to the ever changing needs and abilities of the user. While existing studies in social robotics often focus on a single matching process

by which the behavior of the robot is changed to optimize the properties of the interaction at a single moment in time, adaptation and personalization in assistive tasks should ideally treat the goal as an ever-changing target. As the assistive interaction is often sustained over substantially longer periods of time than other social robot interactions (lasting perhaps months or years), SAR systems should assume that the preferences, needs, and capabilities of the user change over the course of the interaction. Research in this area focuses on questions such as: How can we design a robot that adapts to each user's individual social, physical, and cognitive differences? How can a robot utilize patterns of interactions observed with other users (and possibly other robots) to bootstrap its own attempts at interaction? How do novelty and continuing personalization in interaction create and support trust, learning, and adaptation for improved outcomes for its user?

Socially assistive robots could greatly benefit from adapting to each individual user's unique social, cognitive, and physical abilities. Computationally, the challenge in constructing such a system stems from the individual differences among users and adaptation must occur at multiple levels of abstraction and time scales. At a given moment, robot responses that one user might find frustrating might be optimally engaging to another, or even to the same user under different conditions or at different times. Their responses should also change over time as the user becomes tired, frustrated, or bored. Responses must also vary from session to session or day to day so that interactions remain engaging and interesting.

Very little research to date addresses learning through personalized social interaction [73.60], especially with populations like children, the elderly, or individuals with social and cognitive deficits. Various learning approaches for HRI have been proposed [73.61], but to date, none have included the user's profile, preferences, and/or personality. The robot's capability to adapt to an individual user's needs and abilities must be bounded by the social and cultural context and while filling the role of a supportive partner or peer. The majority of current machine learning methods are focused on learning specific policies or parameters for tasks or behaviors. While there is a plethora of available learning methods, and some are particularly relevant to SAR, such as user state and activity modeling [73.62, 63], learning by imitation and from demonstration [73.64–68], and learning interaction models from crowd–sourced interaction [73.69], no work to date has focused on adapting to individual differences.

73.7 Creating Long-Term Engagement and Behaviour Change

The ultimate objective for any socially assistive robot is to alter the long-term behavior of the user in line with the behavioral, therapeutic, or educational goals that the robot was designed to support. For many of the application domains covered by SAR systems, this change might require months or years of interaction. A critical research issue for SAR is the development of systems that can maintain these long-term social interactions, monitor the behavioral changes of the user over these long-term interactions, and promote activities that support selective changes in behavior.

Research questions related to this area include: How can a robot continue to be interesting and engaging to a user over an extended time period? What social behaviors and attributes are needed to establish and sustain trust, rapport, and comfort with the user over time to build a successful relationship that continues to provide value? How can a robot continue to adapt to and autonomously guide its user through successive milestones in order to achieve desired learning or behavior change outcomes over longitudinal time scales? What strategy should a robot employ to motivate and maintain lessons or activities, and how should this strategy evolve over time? How can the robot recognize when its goals related to the behavior change of the user have been achieved, or when intermediary goals have been achieved?

To better fill the social role of supportive partners, socially assistive robots should adapt their behavior and actively planning the trajectory of interactions to guide users toward learning or behavioral goals. The ability to adapt and plan should be demonstrated at multiple time-scales and abstraction levels so as to enable long-term engagement over days, weeks, months, and eventually years. Such capability is particularly important for special populations who may have different responses to techniques and methods that are often employed with typical adults. Almost no work to date has focused on the computational challenges involved in creating social interactions that persist over multiple weeks, months, or years. Early efforts in the area of life-long learning still operate on short time-scales (e.g., two or three cases/tasks) compared to the weeks and months that will be necessary for SAR systems. This particular goal presents a myriad of interesting and challenging applications of complex computational methods such as machine learning. Socially assistive robots capable of long-term adaptation must constantly

sample new interaction and teaching strategies while exploiting previously learned preferences of each individual user and between users, all in an online setting. Extending engagement, social support, and guidance to long-term interaction requires more than scaling up existing methods and measures. A major scientific goal for SAR research is to develop new insights that enable long-term adaptive support and guidance to the user in order to sustain a productive assistive relationship.

73.8 SAR for Autism Spectrum Disorder (ASD) Therapy

Some of the earliest and most extensive investigations of SAR have focused on the diagnosis and treatment of autism spectrum disorders. Driven by pressing needs from a rapidly expanding population and remarkable early results, a diverse set of robotics research groups have examined the responses of individuals with autism spectrum disorders to robots [73.9, 32].

73.8.1 Summary of ASD-Related Challenges

Autism is a pervasive developmental disorder marked by severe deficits in social functioning such as difficulties recognizing body language, making eye contact, and understanding the emotions of other people [73.4]. It is typically associated with a range of cognitive and attentional deficits [73.70, 71] as well as tendency to engage in preservative and self-stimulatory behaviors. Expression of particular behaviors is highly heterogeneous and associated with notable intra-individual variation in the level of functioning in specific domains. That is, while some children have exceptional cognitive skills, others struggle with major challenges in these areas.

ASD has received a heightened interest from researchers in recent years due to, in part, an unprecedented increase in the incidence rate. ASD is estimated to affect one in every 60 children in the United States, a rate that is roughly eight times higher than the best estimates from 20 years ago [73.72]. ASD is a behaviorally specified disorder [73.5]; there is no blood test, no genetic screening, and no functional imaging test that can diagnose ASD. Diagnosis relies on the clinician's intuitive feel for the child's social skills including eye-to-eye gaze, facial expression, body postures, and gestures. These observational judgments are then quantified according to standardized protocols [73.6, 7, 73, 74] that are both imprecise and subjective. The broad disagreement of clinicians on individual diagnoses creates difficulties both for selecting appropriate treatment for individuals and for reporting the results of population-based studies [73.8, 75].

ASD is a lifelong disorder; there is no known cure. However, early, sustained, and intensive intervention using a variety of behavioral techniques have been demonstrated to produce substantive gains in many individuals with ASD [73.5]. These therapeutic approaches often focus on traditional behavioral therapies in which a skilled therapist guides an individual with ASD through a series of repetitive exercises aimed at improving daily living and/or functional social skills. These techniques require a tremendous amount of labor, and attempts to use technological innovations to alleviate some of this burden have been a popular research topic. For example, pet-assisted therapy [73.76, 77], computer-assisted therapy [73.78–80] and virtual reality-based approaches [73.81, 82] have all been studied. While these have shown some success, there has been limited investigation of the parameters of the conditions necessary to generalize the benefits to interactions with human partners. Among nonhuman partners, robots that provide instruction, support, and assistance through social interaction have been seen as a potential mechanism for supporting therapy and daily living for individuals with ASD [73.1, 83].

Robots promise unique practical advantages over other nonhuman interactive partners. First, they can potentially provide identical delivery of stimuli, establishing a uniquely high level of control in diagnosis and assessment [73.83]. Second, they do not require the months or years of training that animal assistants may need, as robots can be designed to allow flexible customization. Third, they offer a potential for tactile interface and the immediacy of embodied agency, which computer programs and virtual realities cannot provide. In cases in which human or animal therapeutic aids may be unavailable or are prohibitively expensive, and where software or virtual reality therapeutic tools cannot provide sufficient embodiment, robots may provide an especially useful addition to therapy.

73.8.2 User Study Insights

Research in this area has shown exciting, but often preliminary, benefits to individuals with ASD, including increased engagement in tasks, increased levels of attention, and novel social behaviors such as joint attention and spontaneous imitation, when robots are part of

the interaction [73.83–89]. Much of the early work in this domain was exploratory, performed on small sample sizes (typically fewer than 5 individuals), lacked quantitative measurements of success, failed to include a control group, and had minimal diagnostic and inclusionary criteria. Given the heterogeneity of ASD, these exploratory studies were frequently dismissed by the clinical research community. These exploratory studies however were instrumental in laying the groundwork for the more extensive and detailed studies that are being performed today [73.9].

More recent studies have focused on how robots can enhance human-to-human social behavior in children with ASD and have included statistically valid sample sizes, carefully designed control conditions, and extensive diagnostics to accurately describe the population being studied. For example, *Kim* et al. [73.90] recently demonstrated a robot that engaged children in a vocal prosody training task. Children with ASD were more likely to use appropriate inflection in their voices when engaging the robot than when interacting with a familiar clinician. Furthermore, some children in this study demonstrated increased frequency of social gaze behaviors to an adult (including social referencing) immediately following interaction with a social robot. *Feil-Seifer* and *Matarić* [73.91] recently demonstrated that children with ASD demonstrated an increased social response to a robot that was socially responsive than to one that was not. The study included a comparable toy as a control and its results included novel insights about child–robot interactions for verbal and nonverbal children with ASD (Fig. 73.2,73.3).

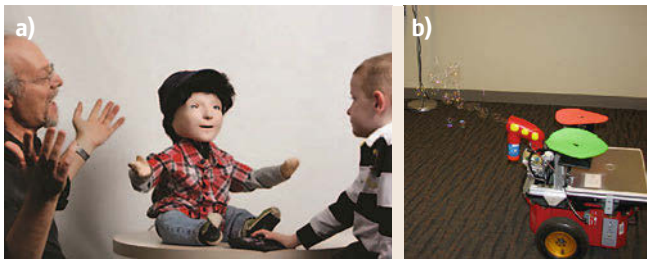


Fig.73.2a,b Both highly anthropomorphic robots ((**a**) Kaspar from the University of Hertfordshire) and highly mechanistic robots ((**b**) Bubblebot from the University of Southern California) have shown increased social responses of children with ASD during interactions with robots

73.8.3 Research Directions

While these HRI studies offer the hope of a novel therapeutic approach, they have to date neither explained why robots provoke these responses in children nor demonstrated that a long-term behavioral change in children can be generated that generalizes to human–human interactions. A number of possible hypotheses could explain the increase in social behavior observed during and after interactions between an individual with ASD and a robot. Perhaps robots offer a more predictable interaction, or a more selective and simplified set of social cues to be interpreted, or are more overt in their behavioral responses and easier to engage in reciprocal interactions. There is anecdotal evidence that some of these may not be a satisfactory explanation [73.83, 90], but a robust explanation of why these effects are so notable in interactions with individuals with ASD has yet to be described.

Constructing autonomous systems is particularly challenging for this population. The diversity of abilities and deficits within the ASD population results in vast differences in the rate of interaction, the mode of interaction (e.g., verbal versus nonverbal), the social skills to be included in the training, and even the criteria for what constitutes a successful application of a social skill [73.32]. While a few studies [73.92] have begun to incorporate completely autonomous systems, most are still utilizing wizard-of-Oz techniques to deal with the variations across users.

Perhaps the most clinically important research direction is to establish that HRIs provide a long-term, measurable change in human–human interactions. Nearly all of the studies carried out to date (with the exception of [73.87]), focus on a small number of short interactions (typically spanning a combined time of less

Fig.73.3a–c *Kim* et al. [73.89, 90] demonstrated increases in child-adult interactions following a short (5 min) interaction with a robot. These images come from video captured before (**a**), during (**b**), and after (**c**) an interaction with a Pleo robot, which focused on prosodic production

than an hour). As behavioral change even with an expert clinician can require hundreds of hours of training, our research studies must extend to more lasting interactions that span weeks and months. Furthermore, most studies to date focus on the development of social behaviors directed at the robot. These robot-directed behaviors have minimal clinical value unless they can be shown to generalize to human–human interactions. (One exception is a the studies by *Kim* et al. [73.89, 90] which demonstrated that children with ASD show a significant improvement in face-to-face interactions with an adult immediately following an interaction with a robot tutor). To provide therapeutic value to an individual with ASD, interactive robots must provide either a measurable benefit to their ability to engage with other people.

73.9 SAR Supporting Rehabilitation

SAR is especially well suited for use in the process of rehabilitation, which involves regaining function lost due to illness or trauma. Where rehabilitation robotics focuses on hands-on HRI, in contrast, SAR focuses on providing motivation and coaching in order to facilitate the rehabilitation process. Since motivation has long been recognized as a major factor in rehabilitation, SAR has the potential to play a key role in this health context. SAR systems for supporting rehabilitation have been developed and tested with stroke patients [73.93, 94] and Alzheimer's patients [73.95], and are being considered for use in traumatic brain injury rehabilitation.

73.9.1 Summary of Rehabilitation Challenges

Stroke is the leading cause of serious, long-term disability in the US and the third leading cause of death [73.96]. In the US alone, over 800 000 people suffer a stroke each year, and nearly 400 000 survive with some form of neurologic disability [73.97]. The number of stroke survivors with disability is expected to double by the year 2025 [73.98]. Over 80% of first-time strokes involve loss of function of the upper limb that significantly impacts the independence and health of the stroke survivor [73.99–101]. The residual disability after stroke is substantial, with over 65% of patients unable to incorporate the affected hand effectively into daily activities. Functional recovery of the arm and hand has generally been resistant to the traditional approaches of stroke rehabilitation compared with that for the lower extremities program [73.99, 102].

Stroke often leaves individuals unable to perform movements with the affected limb even though the limb is not completely paralyzed. This loss of function, termed *learned disuse*, can improve with rehabilitation therapy [73.103, 104]. Clinical studies using motor training have found improvement in functional upper limb performance in patients more than one year post stroke and cortical reorganization and recruitment of adjacent brain areas associated with intensive use of the affected upper limb have been documented several years after the initial stroke injury [73.105–107]. Repetitive, intense, task-specific practice is effective in rehabilitation [73.108–111]. Indeed, practice is fundamental to motor learning – it is the single most important variable for skill acquisition. However, such practice requires trained coaching and motivation.

73.9.2 Contact Versus SAR (Noncontact) Robotics for Rehabilitation

SAR creates a bridge between rehabilitation robotics, which has historically been contact-based [73.112, 113], with noncontact functionalities, such as companion robotics [73.114, 115]. The noncontact approach defines a key niche for SAR that involves an autonomous physically embodied robot providing contact-free monitoring, coaching, and encouragement, while also providing detailed assessments of client progress to key stakeholders including and importantly the user. Unlike rehabilitation robotics, where the application and perception of forces are the main focus, and which are consequently currently limited to in-clinic settings, work with noncontact robots is instead directed at the interaction modalities and motivation strategies for achieving specific goals in the home and community settings.

One of the most important elements of any rehabilitation program is the practice of carefully directed, well-focused and repetitive exercises, which can be passive or active. In passive exercises (also known as hands-on rehabilitation), the patient is helped by a human (or robotic) therapist to appropriately exercise the affected limb(s). In contrast, in active exercises, the patient performs the exercises with no physical assistance. The majority of existing work in rehabilitation robotics focuses on hands-on robotic systems for passive exercise, focusing on attempting to recover upper-limb function primarily through robotic manipulation of the affected-limb. *Burgar* et al. [73.116] developed a robot-assisted arm therapy workstation, in which patients

can exercise their upper limbs and evaluate their performance. A similar device, which also depends on hands-on robotic technology, was developed by *Krebs* et al. [73.117]. Other related systems have been investigated [73.118] and [73.119]. Because human–robot contact involves complex issues of safety, such hands-on robotics methods remain areas of active ongoing research, with many outstanding challenges. Currently, contact rehabilitation robots are not portable; the logistics of contact robot therapy are further limited by cost, availability of adequate amount of practice, expertise necessary to program and execute trials, and liability. Risk of injury to the patient is of concern when movement of a limb with sensorimotor loss is imposed by a robot [73.120, 121]. Finally, intense functional exercise is more likely to sustain the behavior (upper limb functional use) in the long term in contrast to the decline in long-term performance with contact robot systems after therapy has ceased [73.120–122].

The SAR system acts as a *coach*, providing encouragement, and direction, and ensuring proper adherence with the therapy regimen. Given the size of the stroke-affected population and the time-intensive personalized requirements, such coaching is not possible with human therapists, creating a niche for technological means of filling the gap. By providing the opportunity for time-extended monitoring and encouragement of rehabilitation exercises in any setting (at the clinic or at home), SAR systems complement human care [73.123–126].

73.9.3 User Study Insights

This section reviews some of the insights from SAR studies performed with stroke patients. One study [73.93, 127] focused on how different robot behaviors may affect the patient's willingness to adhere to the rehabilitation program. The study evaluated different voices, movements, and levels of patience on the part of the robot, and correlated those with patient adherence. The robot was equipped with navigation capabilities, so it could find and follow the patient, maneuver itself to an appropriate position for monitoring the patient, and leave when it was not wanted. The robot acquired real-time information about the patient's use of the stroke-affected limb from inertial measurement unit (IMU)-based sensors worn by the patient. Wearable sensors present an important alternative to machine vision, which may not be acceptable to all users due to privacy concerns. The robot used information provided by the wearable sensors to encourage the patient to continue using the limb, or to use the limb more or in a different way, as appropriate. The robot was well received by the patients. Some continued to perform the activity beyond the end of the experiment, providing further evidence of improved adherence well beyond any novelty effects. As expected, there were significant personality differences among the patients; some were highly adherent but appeared un-engaged by the robot, while others were highly engaged and even entertained, but preferred to interact with the robot rather than perform the coached exercises. This leads toward interesting research questions of how to define adaptive robot-assisted rehabilitation protocols that will serve the diverse user population as well as the time-extended and evolving needs of every individual user.

Modeling personality in order to customize HRI has a particular relevance to stroke rehabilitation because pre-stroke personality has a great influence on post-stroke recovery [73.96]; individuals classified as extroverted before the stroke mobilize their strength more easily to recover than do introverted subjects [73.128]. Further, work in human–computer interaction (HCI) has demonstrated the similarity-attraction principle, which posits that individuals are more attracted to others manifesting the same personality as theirs [73.129–131]. *Tapus* and *Matarić* [73.132] performed a SAR study to test related effects. In a series of experiments with a simple mobile robot equipped with a camera and a microphone, participants were asked to perform four tasks (designed as functional activities) similar to those used during standard stroke rehabilitation. The participants completed a personality assessment questionnaire before the experiment; the resulting personality based specifically on the extroversion–introversion dimension was used to determine the robot's personality. The behavior control architecture was based on the model of reciprocal influences on behavior [73.133]. The robot expressed its personality through several means: (1) proxemics (social use of space; extroverted personalities used smaller personal distances) [73.134]; (2) speed and amount of movement (extroverted personalities moved more and faster); and (3) verbal content (extroverted personalities talked more assertively (*You have done only x movements, I'm sure you can do more!*), using a challenge-based style compared to a nurture-based style (*I know it's hard, but remember it's for your own good.*). The experiment compared personality-matched versus personality-mismatched (random) conditions. The results showed that the robot's personality was fundamental to the quality of the interaction. Two statistically significant results were found: (1) participants consistently performed better on the task when interacting with the personality-matched robot; (2) participants reported preferring the personality-matched robot. *Tapus* et al. [73.94] then expanded on this work by enabling the SAR system to adapt its personality based on the past interactions with the user.

Fig.73.4a,b Seated exercise coaching with a socially assistive humanoid robot on wheels, **(a)** Bandit, from the University of Southern California, and **(b)** upper extremity rehabilitation manipulandum

Formulated as policy gradient reinforcement learning (PGRL), the adaptation allowed for optimizing the interaction parameters: interaction distance/proxemics, speed, and verbal content (what the robot says and how it says it). The learning algorithm was initialized with parameter values that were in the vicinity of what was thought to be acceptable for both extroverted and introverted individuals, based on the user–robot personality matching study described earlier. Gradual adaptation around a solution rather than random probing of the solution space is important; if the robot's initial personality is far off the mark, or if it changes drastically, users are dissatisfied with the interaction. The results provided evidence for the effectiveness of robot behavior adaptation to user personality and performance: users tended to perform more or longer trials under the personality matched and therapy style-matched conditions (nurturing styles were correlated with the introversion, and challenging styles with extroversion).

Another study with stroke patients compared matched humanoid robot and simulated agent implementation of the SAR coach. The study participants were seated at a desk, and were asked by the robot to move magazines from the lower to the higher rack and to de-shelf as many magazines as possible in 6 min. The robot monitored the participant's motions using a revised version of the motion sensors described above [73.135, 136]. The participants could use the button on a WiiMote to stop the interaction at any time. The IMU data and shelf state (using a scale) were used to classify the movement into states of progress (e.g., resting, picking up magazine, lifting arm, shelving book, lowering arm). In addition to the verbalizations, the robot used *beat gestures* whenever it spoke. This consisted of moving the head, arms, and hands in a beat with amplitude and frequency governed by the characteristics of the phrase that was being uttered. Baseline movement data were assessed using standard stroke movement tests and task improvement was measured after repeated sessions. The experimental design involved all participants experimenting with both the robot and the computer. Two-thirds of the participants preferred the physical robot, with one third having no preference.

73.9.4 Research Directions

SAR for rehabilitation presents multi-faceted research challenges.

Personalizing the interaction to adapt to the user's changing state over multiple timescales remains a major challenge, since rehabilitation aims for long-term improvement, but must contend with mood, fatigue, and performance variations within each session, as well as short term trends that may plateau or even reverse. Personalizing the interaction thus requires new ways of integrating robot learning, user modelling, and motivation strategies, among many other SAR research challenges. Creating robotic systems capable of adapting their behavior in order to provide an engaging and motivating customized protocol is a challenging target, especially when working with vulnerable user populations. In the SAR context, behavior adaptation must address both short-term changes that represent individual differences and long-term changes that allow the interaction to continue to be engaging over a period of months and even years. Various learning approaches for HRI have been proposed in the literature [73.61, 65] but this remains a major challenge in robotics in general and in SAR in particular.

Modeling the user and predicting user behavior is a challenge common to HRI and HCI, as well as research in signal processing and related fields. However, SAR also brings up the issue of modelling the identity of the robot character. Specifically, studies in SAR are beginning to show that having not only a personality but also a back story for the robot facilitates user engagement.

Research into relational agents [73.137] has been applied to short-term interactions in medical settings, such as hospital checkout procedures. Some of the insights from that field, including politeness theory, empathy, etc., are being explored in SAR for use in long-term interactions involved in rehabilitation.

Finally, one of the much needed new directions of research is the exploration of bringing together contact and noncontact rehabilitation robotics in relevant domains, in order to enhance the impact of both robotics subfields.

73.10 SAR and Eldercare

73.10.1 Summary of Eldercare Challenges

The growing population of aging adults is increasing the demand for healthcare services worldwide. By the year 2050, the number of people over the age of 85 will increase threefold [73.138], while the shortfall of nurses and caregivers is already an issue [73.139–141]. Regular physical exercise has been shown to be effective at maintaining and improving overall health [73.142–145]. Physical fitness is associated with higher functioning in the executive control processes [73.146] and is correlated with less atrophy of frontal cortex regions [73.147] and with improved reaction times [73.148]. Social interaction, and specifically high perceived interpersonal social support, has also been shown to have a positive impact on general mental and physical wellbeing [73.149], in addition to reducing the likelihood of depression [73.150–153]. Thus, the availability of physical exercise therapy, social interaction, and companionship will be critical for the growing elderly population.

The probability for acquired motor, cognitive, and perceptual disabilities dramatically increases with age [73.154]. Over 50 million Americans have a disability that affects one or more of their major life activities [73.155, 156] and only 30% are employed [73.157]. The economic cost of the associated loss of functioning is high; e.g., the cumulative total of stroke-affected Americans is over 4 million, and the estimated annual burden from stroke-related disability is $53.6 billion, of which $20.6 billion is in indirect costs due to lost productivity and income [73.96, 158]. Given projections that the number of people older than 65 years is anticipated to double between 1997 and 2025 [73.156] and that 30% of the population will be over 65 [73.159, 160], we will soon face new major technical and social challenges to attain, prolong, and preserve quality of life.

A especially vulnerable and sizeable population consists of individuals who are aging with disabilities that were acquired at any stage of life, from birth or early in life to stroke or other disorders later in life [73.161]. The number of middle aged and older adults living with disabilities will grow significantly as the U.S. population ages rapidly [73.162]. People who have managed their lives very successfully for years may find that they are experiencing the effects of aging earlier than others or are developing secondary health conditions as they reach their 30s or 40s [73.163, 164]. As persons with disabilities age, progressive declines in health and medical status can challenge the adaptive resources required to maintain functional independence and quality of life [73.165, 166].

Motor and cognitive dysfunctions that accompany chronic illness and disability in late life are major precursors for people who age into disability. Cognitive impairments are associated with increased disability in aging populations [73.167]; 19% of women and 13% of men aged 65–69 live with a disability in mobility [73.168]. The respective percentages increase to 83% and 63% for women and men aged 90–95 [73.168]. These dysfunctions greatly compound the disruptions in overall lifestyle and occupational status experienced by individuals aging with disabilities primarily associated with cognitive or motor deficits. Thus, there is a critical need to develop effective and affordable assistive technologies to improve the motor and cognitive functioning, especially in individuals aging with disabilities. This creates an important niche for SAR.

73.10.2 User Study Insights

The literature that addresses assistive robotics intended for and evaluated by the elderly is limited but growing.

Researchers have investigated the use of robots to help address the social and emotional needs of the elderly, including reducing depression and increasing social interaction with peers. *Wada* et al. [73.114] studied the psychological effects of a seal robot, Paro, used to engage seniors at a day service center. The study found that Paro, always accompanied by a human handler, was able to consistently improve the moods of elderly participants who spent time petting and engaging with it over the course of a 6-week period. *Kidd* et al. [73.115] used Paro in another study that found it to be useful as a catalyst for social interaction. They observed that seniors who participated with the robot in a group were more likely to interact socially with each other when the robot was present and powered on, rather than when it was powered off or absent.

Social agents that aim to assist individuals in health-related tasks such as physical exercise have also been developed in the HCI community. *Bickmore* and *Picard* [73.137] developed a computer-based virtual relational agent that served as a daily exercise advisor by engaging the user in conversation and providing educational information about walking for exercise, asking about the user's daily activity levels, tracking user progress over time while giving feedback, and engaging the user in relational dialogue.

Fasola and *Matarić* [73.169] conducted a study to investigate the role of praise and relational discourse (politeness, humor, empathy, etc.) in a SAR exercise

Fig.73.5a–c Paro robot (a) from the University of Tokyo, Autom robot (b) used in an MIT weight-loss study, and Healthbot robot (c) from the University of Auckland used in a large-scale elderly care study

system designed for older adults. The study compared a relational robot versus a nonrelational robot. The relational robot employed praise and relational discourse, while the nonrelational robot guided the exercise interaction without such praise or discourse. Results showed participants rated the relational robot significantly more positively than the nonrelational robot in terms of enjoyableness and usefulness of the interaction, and as a companion and exercise coach. *Fasola* and *Matarić* [73.169] also conducted a user study to investigate the role of choice and user autonomy within the same SAR exercise system. The results of the study showed no clear preference for one condition over the other, as the user enjoyment level of the interaction was reported to be equally high for choice and no-choice conditions. This suggests that a hybrid approach that includes both user and robot decision making, personalized and tuned automatically for each user, might ultimately be the best solution for achieving a fluid and enjoyable task interaction for all users. This once again underscores the need for personalized methods of HRI and user modeling and adaptation.

73.11 SAR for Alzheimer's Dementia and Cognitive Rehabilitation

Dementia is a progressive disabling neurological condition that can be a symptom of several diseases, including Alzheimer's, which makes up about half of all cases of dementia. The latest estimate is that 26.6 million people were suffering from Alzheimer's disease worldwide in 2006, and it will rise to 100 million by 2050 – 1 in 85 of the total population [73.170]. More than 40% of those cases will be in late-stage Alzheimer's, requiring a high level of attention equivalent to nursing home care. Individuals suffering from Alzheimer's experience a decrease in verbal and cognitive skills along with the deterioration of strength and flexibility that comes with aging. Rehabilitation therapies for individuals with dementia often focus on both cognitive and motor exercises to in order to preserve patients' autonomy for as long as possible.

Therapy for individuals with dementia must be pervasive and ongoing in their everyday lives. Mental and physical exercise involving art, music, and playing with/tending to animals have been used to preserve cognitive function in older adults who have begun experiences symptoms of dementia. However, these therapies can only be offered by well-trained personnel. In particular, animal therapy can involve safety risks for both the animal and the patient. The goal of SAR in this therapy context is to provide a means of providing therapy that is safe, easy to use, and does not require special training.

73.11.1 User Study Insights

In a study by *Tamura* et al. [73.171], an entertainment robot was used as occupational therapy instead of animal-assisted therapy to avoid any danger or injury to the patient and maintain cleanliness. The study compared the effectiveness of a robot animal, AIBO, with a toy. AIBO responded to spoken commands and was demonstrated to severely demented elderly people living in a geriatric home. The most frequent reactions to AIBO consisted of looking at, communicating with, and caring for the robot. The patients recognized that AIBO was a robot but, once the robot was addressed by the experimenters, the patients perceived it as either a dog or a baby. The presentation resulted in positive outcomes for the severe dementia patients, including increased communication between the patients and the robot.

Tapus et al. [73.172] performed a long-term (6-month) study of socially assistive robots in the context of cognitive training for Alzheimer's patients to test if

elderly individuals with Alzheimer's impairments can maintain attention to music with the help of a SAR system. A small group of four participants over 70 years old, suffering from mild to severe cognitive impairment and/or Alzheimer's disease, took part in the study and received per- and post-study cognitive assessments. Each interacted in one-on-one twice-weekly sessions with the robot, in which the robot presented a music recognition game with three different levels of challenge: difficult (no hints), medium (hints given), and easy (name of song given). The challenge level was adjusted to each participant based on performance. The results obtained over 6 months of robot interaction (excluding the 2 months of learning) suggest that elderly people suffering of dementia and/or Alzheimer's can sustain attention to music across a long period of time and even improve on the music recognition task when coached by a socially assistive robot; the SAR system was able to adapt the challenge level of the game it was presenting to the user in order to encourage task improvement and attention training. Importantly, the participants demonstrated fondness for the robot and included it into their daily narratives (e.g., *My buddy is coming to visit, I don't want to miss it.*).

The multiyear Healthbots project at the University of Auckland is examining the role of telepresence and humanoid robots in assisted living communities [73.173]. This large study included three different types of robots, ranging from large telepresence to small humanoid, which were deployed as companions and nursing assistants in a retirement village of around 650 residents. Several insights were gained from the long-term deployment, including the preferences of both the elderly users and staff members. The most promising avenue for robotic assistance was a robot that was sent into residents' rooms to remind them to take their medications or fetch them for appointments.

73.11.2 Research Directions

As with all areas of SAR, the user's willingness to engage with a socially assistive robot to accept advice, interact, and ultimately alter behavior depends on the robot's ability to obtain the user's trust and sustain the user's interest. User interfaces and input devices that are easy and intuitive for a range of users, including those with special needs, need to be developed. Social interaction is inherently bidirectional and thus involves both multimodal perception and communication, including verbal and nonverbal means. Toward that end, automated behavior detection and classification as well as activity recognition, including user intent, task-specific attention, and failure recognition, are critical enabling components being developed.

73.12 Ethical and Safety Considerations

SAR, with its focus on developing human–robot relationships and influencing human behavior, naturally brings about several distinct types of ethical considerations.

Safety of the user is the first and most obvious consideration in any area of HRI. Because SAR systems do not involve physical contact initiated by either people or robots, they present a minimum of physical safety issues. However, while physical safety is one of the largest concerns in other areas of robotics research (Chap. 30), in SAR, emotional and other types of nonphysical safety must be considered.

As SAR research strives to achieve user engagement and attachment, some critics have argued that such relationships should be reserved for humans only [73.174]. Evaluations of specific SAR systems have triggered ethical dilemmas [73.174, 175]. *Turkle* [73.174] demonstrated that some participants interacting with robots can correctly identify the robot's intended emotional abilities and operational capabilities. These participants could also correctly distinguish equivalent capabilities in a person, pet, or other relational artifact. However, it was also demonstrated that some users formed attachments and emotional bonds with robots they were interacting with. These attachments led to misconceptions about the robots' emotional capabilities. For example, one user felt that the robot would miss him when he was gone, something that the robot was not capable of doing. *Sharkey* and *Sharkey* [73.175] argue strongly that such attachments in children could lead to malformed development and emotional problems. Some such arguments are based on the notion that human care is replaced by robot care, which is not consistent with the premise of most of SAR research. Most SAR research clearly state that their aim is to *augment rather than replace human care*, as discussed in Sect. 73.7 of this chapter, in the context of SAR for autism spectrum disorders.

Some more pragmatic ethical considerations take into account the inevitable progress of technology, which will render any particular system obsolete well within a user's lifetime, therefore undermining attachment and likely making long-term system operation impossible. Since long-term robot system operation is

a goal of the majority of robotics research, this is not yet a pressing ethical concern, but is one that is well recognized as important.

As one example of ethics applied to SAR in particular, *Feil-Seifer* and *Mataric* [73.176] outlines the ethical issues of SAR around the core principles of ethics applied to human subjects, namely beneficence, nonmaleficence, autonomy and justice. The first two principles encompass the SAR issues of relationships, authority and attachment, perception and personifications of the robot, and replacement of human care/changes to human–human interaction. The third principle, autonomy, spans the issues of privacy, choice, and intentional user deception. Finally, justice spans the complex issues of cost/benefit analysis, and locus of responsibility in the case of failure or harm.

As robots get closer to people, ethical issues become more pressing. It is a good sign that so early in the development of SAR those issues are being actively discussed and considered, toward the development of most ethically informed methods and systems.

References

73.1 A. Tapus, M.J. Matarić, B. Scassellati: The grand challenges in socially assistive robotics, IEEE Robotics Autom. Mag. Spec. Issue Grand Chall. Robotics (2007)

73.2 United Nations Department of Economic and Social Affairs, Population Division (2011): *World Population Aging 1950–2050*, http://www.un.org/esa/population/publications/worldageing19502050/ (2013)

73.3 Department of Health and Human Services, Centers for Disease Control and Prevention (2009)

73.4 American Psychiatric Association: *Diagnostic and Statistical Manual of Mental Disorders: DSM-IV-TR* (American Psychiatric Association, Arlington 2000)

73.5 F.R. Volkmar, C. Lord, A. Bailey, R.T. Schultz, A. Klin: Autism and pervasive developmental disorders, J. Child Psychol. Psychiatry **45**(1), 135–170 (2004)

73.6 S.S. Sparrow, D. Balla, D. Cicchetti: *Vineland Adaptive Behavior Scales, Expanded Edition* (Pearson, New York 1984)

73.7 E. Mullen: *Mullen Scales of Early Learning: AGS Edition* (Pearson, New York 1995)

73.8 A. Klin, J. Lang, D.V. Cicchetti, F.R. Volkmar: Brief report: Interrater reliability of clinical diagnosis and DSM-IV criteria for autistic disorder: results of the DSM-IV autism field trial, J. Autism Dev. Disord. **30**(2), 163–167 (2000)

73.9 B. Scassellati, H. Admoni, M. Matarić: Robots for use in autism research, Annu. Rev. Biomed. Eng. **14**, 275–294 (2012)

73.10 J. Kiratli: Telehealth technologies for monitoring adherence and performance of home exercise programs for persons with spinal cord injury: Tele-exercise, Exerc. Recreat. Technol. People With Disabil.: State Sci. (2006)

73.11 National Institute of Neurological Disorders and Stroke (NINDS): Post-Stroke Rehabilitation Fact Sheet, http://www.ninds.nih.gov/disorders/stroke/poststrokerehab.htm (2013)

73.12 D.S. Freedman, W.H. Dietz, S.R. Srinivasan, G.S. Berenson: The relation of overweight to cardiovascular risk factors among children and adolescents: The Bogalusa Heart Study, Pediatrics **103**(6), 1175–1182 (1999)

73.13 G.Q. Shaibi, M.I. Goran: Examining metabolic syndrome definitions in overweight Hispanic youth: A focus on insulin resistance, J. Pediatr. **152**(2), 171–176 (2008)

73.14 O. Pinhas-Hamiel, P. Zeitler: Insulin resistance, obesity, and related disorders among black adolescents, J. Pediatr. **129**(3), 319–320 (1996)

73.15 M.L. Cruz, G.Q. Shaibi, M.J. Weigensberg, D. Spruijt-Metz, G.D. Ball, M.I. Goran: Pediatric obesity and insulin resistance: Chronic disease risk and implications for treatment and prevention beyond body weight modification, Annu. Rev. Nutr. **25**, 435–468 (2005)

73.16 E.E. Calle, R. Kaaks: Overweight, obesity and cancer: Epidemiological evidence and proposed mechanisms, Nat. Rev. Cancer **4**(8), 579–591 (2004)

73.17 A.S. Singh, C. Mulder, J.W. Twisk, W. van Mechelen, M.J. Chinapaw: Tracking of childhood overweight into adulthood: A systematic review of the literature, Obes. Rev. **9**(5), 474–488 (2008)

73.18 E.A. Finkelstein, J.G. Trogdon, J.W. Cohen, W. Dietz: Annual medical spending attributable to obesity: Payer- and service-specific estimates, Health Aff. **28**(5), 822–831 (2009)

73.19 R. Sturm: The effects of obesity, smoking, and drinking on medical problems and costs, Health Aff. **21**(2), 245–253 (2002)

73.20 J.P. Koplan, C.T. Liverman, V.I. Kraak: Preventing childhood obesity: Health in the balance: Executive summary, J. Am. Diet. Assoc. **105**(1), 131–138 (2005)

73.21 D. Spruijt-Metz: Etiology, treatment and prevention of obesity in childhood and adolescence: A decade in review, J. Res. Adolesc. **21**(1), 129–152 (2011)

73.22 C.D. Kidd, C. Breazeal: A Robotic Weight Loss Coach, Twenty-Second Conf. Artif. Intell. (2007)

73.23 W. Bainbridge, J. Hart, E. Kim, B. Scassellati: The effect of presence on human-robot interaction, IEEE Int. Symp. Robot Hum. Interact. Commun. (2008)

73.24 C.D. Kidd, C. Breazeal: Effect of a Robot on Engagement and User Perceptions, Proc. Intell. Robot. Syst. (IROS) (2004)

73.25 D. Leyzberg, S. Spaulding, M. Toneva, B. Scassellati: The physical presence of a robot tutor increases cognitive learning gains, Proc. 34th Annu. Conf. Cogn. Sci. Soc. (2012)

73.26 J. Wainer, D. Feil-Seifer, D. Shell, M. Matarić: Embodiment and human-robot interaction: A taskbased perspective, IEEE Int. Workshop Robot Hum. Interact. Commun. (2007)

73.27 C. Bartneck: Interacting with an embodied emotional character, Proc. 2003 Int. Conf. Des. Pleas. Prod. Interfaces (2003) pp. 55–60

73.28 Y. Jung, K.M. Lee: Effects of physical embodiment on social presence of social robots, Proc. Presence (2004) pp. 80–87

73.29 J. Wainer, D.J. Feil-Seifer, D.A. Shell, M. Matarić: The role of physical embodiment in human-robot interaction, Proc. IEEE Int. Workshop Robot Hum. Interact. Commun. (RO-MAN'06) (2006)

73.30 S. Kiesler, A. Powers, S.R. Fussell, C. Torrey: Anthropomorphic interactions with a robot and robot-like agent, Soc. Cognit. **26**(2), 169–181 (2008)

73.31 A. Tapus, C. Tapus, M.J. Matarić: The use of socially assistive robots in the design of intelligent cognitive therapies for people with dementia, Proc. Int. Conf. Rehabilitation Robotics (ICORR-09) (2009) pp. 924–929

73.32 J.J. Diehl, L.M. Schmitt, M. Villano, C.R. Crowell: The clinical use of robots for individuals with autism spectrum disorders: A critical review, Res. Autism Spectr. Disord. **6**(1), 249–262 (2012)

73.33 E. Thelen, L.B. Smith: *A Dynamic Systems Approach to the Development of Cognition and Action* (MIT Press, Cambridge 1994)

73.34 L.W. Barsalou: Situated simulation in the human conceptual system, Lang. Cognit. Process. **18**(5/6), 513–562 (2003)

73.35 C. Sann, A. Streri: Perception of object shape and texture in human newborns: Evidence from cross-modal transfer tasks, Dev. Sci. **10**(3), 399–410 (2007)

73.36 D.C. Richardson, R. Dale, N.Z. Kirkham: The art of conversation is coordination: Common ground and the coupling of eye movements during dialogue, Psychol. Sci. **18**(5), 407–413 (2007)

73.37 D.C. Richardson, N.Z. Kirkham: Multimodal events and moving locations: Eye movements of adults and 6-month-olds reveal dynamic spatial indexing, J. Exp. Psychol. Gen. **133**(1), 46–62 (2004)

73.38 L.B. Smith: Cognition as a dynamic system: Principles from embodiment, Dev. Rev. **25**(3/4), 278–298 (2005)

73.39 G. Forman: *Where's the Action in Knowing?* (American Psychological Association, Worcester 1983)

73.40 G. Forman, F. Hill: Constructive play in developmentally delayed preschool children, Top. Learn. Learn. Disabil. **1**(1), 31–41 (1981)

73.41 C.K. Whalen, B. Henker, B.E. Collins, D. Finck, S. Dotemoto: A social ecology of hyperactive boys: Medication effects in structured classroom environments, J. Appl. Behav. Anal. **12**(1), 65–81 (1979)

73.42 M. Heerink, B. Kröse, V. Evers, B. Wielinga: Assessing acceptance of assistive social agent technology by older adults: The Almere Model, Int. J. Soc.Robotics **2**(4), 361–375 (2010)

73.43 J. Fasola, M.J. Matarić: A socially assistive robot exercise coach for the elderly, J. Human-Robot Interact. **2**(2), 3–32 (2013)

73.44 M. Csikszentmihalyi: Intrinsic rewards and emergent motivation. In: *The Hidden Costs of Reward*, ed. by M.R. Lepper, D. Greene (Erlbaum, Hillsdale 1978) pp. 205–216

73.45 R.A. Dienstbier, G.K. Leak: Effects of monetary reward on maintenance of weight loss: An extension of the overjustification effect, Am. Psychol. Assoc. Conv. (1976)

73.46 R.J. Vallerand, G. Reid: On the causal effects of perceived competence on intrinsic motivation: A test of cognitive evaluation theory, J. Sport Psychol. **6**, 94–102 (1984)

73.47 R.J. Vallerand: Effect of differential amounts of positive verbal feedback on the intrinsic motivation of male hockey players, J. Sport Psychol. **5**, 100–107 (1983)

73.48 E.L. Deci, A.J. Schwartz, L. Sheinman, R.M. Ryan: An instrument to assess adults' orientations toward control versus autonomy with children: Reflections on intrinsic motivation and perceived competence, J. Educ. Psychol. **73**, 642–650 (1981)

73.49 M. Zuckerman, J. Porac, D. Lathin, R. Smith, E.L. Deci: On the importance of self-determination for intrinsically motivated behavior, Personality Soc. Psychol. Bull. **4**, 443–446 (1978)

73.50 C.D. Fisher: The effects of personal control, competence, and extrinsic reward systems on intrinsic motivation, Organ. Behav. Hum. Perform. **21**(3), 273–288 (1978)

73.51 R.B. Margolis, C.R. Mynatt: *The Effects of Self and Externally Administered Reward on High Base Rate Behavior, Unpublished manuscript* (Bowling Green State University, Bowling Green 1979)

73.52 F.H. Kanfer, L.G. Grimm: Freedom of choice and behavioral change, J. Consult. Clin. Psychol. **46**, 873–878 (1978)

73.53 G.R. Liem: Performance and satisfaction as affected by personal control over salient decisions, J. Personal. Soc. Psychol. **31**, 232–240 (1975)

73.54 C.-M. Huang, B. Mutlu: The repertoire of robot behavior: Enabling robots to achieve interaction goals through social behavior, J. Hum.-Robot Interact. **2**(2), 80–102 (2013)

73.55 T.L. Griffiths, C. Kemp, J.B. Tenenbaum: Bayesian models of cognition. In: *Cambridge Handbook of Computational Cognitive Modeling*, ed. by R. Sun (Cambridge Univ. Press, Cambridge 2008) pp. 59–100

73.56 C. Crick, B. Scassellati: Inferring narrative and intention from playground games, 7th IEEE Int. Conf. Dev. Learn. (ICDL) (2008)

73.57 C. Breazeal, J. Gray, M. Berlin: Mindreading as a foundational skill for socially intelligent roots, Int. Symp. Robotics Res. (ISRR-07) (2007)

73.58 E.K. Mower, D. Feil-Seifer, M. Matarić, S. Narayanan: Investigating implicit cues for user state estimation in HRI using physiological measurements, 16th IEEE Int. Symp. Robot Human Interact. Commun. (RO-MAN) (2007)

73.59 V. Chidambaram, Y.-H. Chiang, B. Mutlu: Designing persuasive robots, 17th Annu. ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI '12) (2012) pp. 293–300

73.60 T. Fong, I. Nourbakhsh, K. Dautenhahn: A survey of socially interactive robots, Robotics Auton. Syst. **42**(3/4), 143–166 (2003)

73.61 M. Berlin, J. Gray, A. Thomaz, C. Breazeal: Perspective taking: An organizing principle for learning in human-robot interaction, 21st Natl Conf. Artif. Intell. (AAAI) (2006)

73.62 Y.-L. Lin, G. Wei: Speech emotion recognition based on HMM and SVM, 4th Int. Conf. Mach. Learn. Cybern. (2005)

73.63 A. Metallinou, S. Lee, S. Narayanan: Audio-visual emotion recognition using Gaussian mixture models for face and voice, 2008 IEEE Int. Symp. Multimed. (ISM) (2008) pp. 250–257

73.64 M. Matarić: Sensory-motor primitives as a basis for learning by imitation: Linking perception toaction and biology to robotics. In: *Imitation in Animals and Artifacts*, ed. by K. Dautenhahn, C. Nehaniv (MIT Press, Cambridge 2002)

73.65 C. Breazeal, B. Scassellati: Robots that imitate humans, Trends Cognit. Sci. **6**(11), 481–487 (2002)

73.66 M. Nicolescu, M. Matarić: Task learning through imitation and human-robot interaction. In: *Models and Mechanisms of Imitation and Social Learning in Robots, Humans and Animals: Behavioural, Social and Communicative Dimensions*, ed. by K. Dautenhahn, C. Nehaniv (Cambridge Univ. Press, Cambridge 2005) pp. 407–424

73.67 O.C. Jenkins, M.J. Matarić, S. Weber: Primitive-Based Movement Classification for Humanoid Imitation, 1st IEEE-RAS Int. Conf. Humanoid Robotics (Humanoids-2000) (2000)

73.68 P. Bakker, Y. Kuniyoshi: Robot see, robot do: An overview of robot imitation, AISB96 Workshop Learn. Robots Animals (1996)

73.69 S. Chernova, C. Breazeal: Learning Temporal Plans from Observation of Human Collaborative Behavior, AAAI Spring Symp., It's all Timing: Represent. Reason. about Time Interact. Behav. (2010)

73.70 K. Chawarska, F.R. Volkmar: Autism in infancy and early childhood. In: *Handbook of Autism and Pervasive Developmental Disorders*, Vol. 1, (Wiley, New York 2005) pp. 223–246

73.71 K.D. Tsatsanis: Neuropsychological characteristics in autism and related conditions. In: *Handbook of Autism and Pervasive Developmental Disorders*, Vol. 1, (Wiley, New York 2005) pp. 365–381

73.72 CDC: Facts about Autism Spectrum Disorders. Facts About ASDs, http://www.cdc.gov/ncbddd/autism/data.html (2015)

73.73 C. Lord, S. Risi, L. Lambrecht, E.H. Cook Jr., B.L. Leventhal, P.C. DiLavore, M. Rutter: The autism diagnostic observation schedule—Generic: A standard measure of social and communication deficits associated with the spectrum of autism, J. Aut. Devel. Disorder. **30**(3), 205–223 (2000)

73.74 J. Edwards, M. Lanyado: Autism: Clinical and theoretical issues. In: *The Handbook of Child & Adolescent Psychotherapy, Psychoanalytic Approaches*, ed. by M. Lanyado, A. Horne (Routledge, London 1999) pp. 429–443

73.75 F.R. Volkmar, A. Klin: Issues in the classification of autism and related conditions. In: *Handbook of Autism and Pervasive Developmental Disorders*, Vol. 1, (Wiley, New York 2005) pp. 5–41

73.76 F. Martin, J. Farnum: Animal-assisted therapy for children with pervasive developmental disorders, West J. Nurs. Res. **24**(6), 657–670 (2002)

73.77 L.A. Redefer, J.F. Goodman: Brief report: Pet-facilitated therapy with autistic children, J. Autism Dev. Disord. **19**(3), 461–467 (1989)

73.78 A. Bosseler, D.W. Massaro: Development and evaluation of a computer-animated tutor for vocabulary and language learning in children with autism, J. Autism Dev. Disord. **33**(6), 653–672 (2003)

73.79 O.E. Hetzroni, J. Tannous: Effects of a computer-based intervention program on the communicative functions of children with autism, J. Autism Dev. Disord. **34**(2), 95–113 (2004)

73.80 M. Silver, P. Oakes: Evaluation of a new computer intervention to teach people with autism or Asperger syndrome to recognize and predict emotions in others, Autism **5**(3), 299–316 (2001)

73.81 S. Parsons, P. Mitchell: The potential of virtual reality in social skills training for people with autistic spectrum disorders, J. Intell. Disabil. Res. **46**(5), 430–443 (2002)

73.82 D. Strickland: Virtual reality for the treatment of autism, Stud. Health Technol. Inform. **44**, 81–86 (1997)

73.83 B. Scassellati: How social robots will help us to diagnose, treat, and understand autism, 12th Int. Symp. Robotics Res. (ISRR) (2005)

73.84 I.P. Werry, K. Dautenhahn: Applying mobile robot technology to the rehabilitation of autistic children, 7th Int. Symp. Intell. Robotics Syst. (SIRS'99) (1999)

73.85 F. Michaud, A. Clavet: Robotoy contest – designing mobile robotics toys for autistic children, Am. Soc. Eng. Educ. (ASEE'01) (2001)

73.86 B. Robins, K. Dautenhahn, R. Te Boekhorst, A. Billard: Robotic assistants in therapy and education of children with autism: Can a small humanoid robot help encourage social interaction skills?, Proc. Univers. Access Inf. Soc. (UAIS) (2005)

73.87 H. Kozima, C. Nakagawa, Y. Yasuda: Interactive robots for communication-care: A case-study in autism therapy, IEEE Int. Workshop Robot Hum. Interact. Commun. (RO-MAN) (2005) pp. 341–346

73.88 E. Kim, E. Newland, R. Paul, B. Scassellati: Robotic tools for prosodic training for children with ASD: A case study, Int. Meet. Autism Res. (IMFAR) (2008)

73.89 E.S. Kim, L.D. Berkovits, E.P. Bernier, D. Leyzberg, F. Shic, R. Paul, B. Scassellati: Social robots as em-

bedded reinforcers of social behavior in children with autism, J. Autism Dev. Disord. **43**(5), 1038–1049 (2013)
73.90 E. Kim, R. Paul, F. Shic, B. Scassellati: Bridging the research gap: Making HRI useful to individuals with autism, J. Hum.-Robot Interact. **1**(1), 26–54 (2012)
73.91 D. Feil-Seifer, M.J. Matarić: Toward socially assistive robotics for augmenting interventions for children with autism spectrum disorders, Exp. Robotics – 11th Int. Symp. (2009)
73.92 D. Feil-Seifer, M.J. Matarić: Towards the integration of socially assistive robots into the lives of children with ASD. Working, Human-Robot Interact. Workshop Soc. Impact: How Socially Accepted Robots Can be Integr. our Soc. (2009)
73.93 M. Matarić, J. Eriksson, D. Feil-Seifer, C. Winstein: Socially assistive robotics for post-stroke rehabilitation, Int. J. Neuroeng. Rehabil. **4**, 5 (2007)
73.94 A. Tapus, C. Tapus, M.J. Matarić: User-robot personality matching and assistive robot behavior adaptation for post-stroke rehabilitation therapy, Intell. Serv. Robotics Spec. Issue Multidiscip. Collab. Soc. Assist, Robotics **1**(2), 169–183 (2008)
73.95 A. Tapus, C. Tapus, M.J. Matarić: Music therapist robot: A solution for helping people with cognitive impairments, Int. Jt. Conf. Artif. Intell. (IJCAI) (2009)
73.96 American Heart Association: *Heart Disease and Stroke Statistics – 2003 Update* (American Heart Association and American Stroke Association, Dallas 2003)
73.97 M. Kelly-Hayes, J.T. Robertson: The American Heart Association stroke outcome classification, Stroke **29**(6), 1274–1280 (1998)
73.98 J.P. Broderick: William M. Feinberg Lecture: Stroke therapy in the year 2025: Burden, breakthroughs, and barriers to progress, Stroke **35**(1), 205–211 (2004)
73.99 T.S. Olsen: Arm and leg paresis as outcome predictors in stroke rehabilitation, Stroke **21**(2), 247–251 (1990)
73.100 N.E. Mayo, S. Wood-Dauphinee, R. Côté, L. Durcan, J. Carlton: Activity, participation, and quality of life 6 months poststroke, Arch. Physic. Med. Rehabil. **83**(8), 1035–1042 (2002)
73.101 B.H. Dobkin: Clinical practice. Rehabilitation after stroke, N. Eng. J. Med. **352**(16), 1677–1684 (2005)
73.102 J. Desrosiers, F. Malouin, D. Bourbonnais, C.L. Richards, A. Rochette, G. Bravo: Arm and leg impairments and disabilities after stroke rehabilitation: relation to handicap, Clin. Rehabil. **17**(6), 666–673 (2003)
73.103 R.J. Nudo, E.J. Plautz, S.B. Frost: Role of adaptive plasticity in recoery of function after damage to motor cortex, Muscle Nerve **24**(8), 1000–1019 (2001)
73.104 E.G. Taub, G. Uswatte, D.M. Morris: Improved motor recovery after stroke and massive cortical reorganization following constraint-induced movement therapy, Phys. Med. Rebail. Clin. N. Am **14**(1), 77–91 (2003)
73.105 J. Van der Lee, R.C. Wagenaar, G. Lankhorst, T.W. Vogelaar, W.L. Deville, L.M. Bouter: Forced use of the upper extremity in chronic stroke patients; results from a single-blind randomized clinical trial, Stroke **30**, 2369–2375 (1999)
73.106 D. Reinkensmeyer, M. Averbuch, A. McKenna-Cole, D.B. Schmit, W.Z. Rymer: Understanding and treating arm movement impairment after chronic brain injury: Progress with the arm guide, J. Rehabil Res. Dev. **37(6)**, 653–662 (2000)
73.107 J. Schaechter, E. Kraft, T.S. Hilliard, R.M. Dijkhuizen, T. Benner, S.P. Finklestein, B.R. Rosen, S.C. Cramer: Motor recovery and cortical reorganization after constraint-induced movement therapy in stroke patients: A preliminary study, Neurorehabil. Neural Repair **16**(4), 326–338 (2002)
73.108 C. Butefisch, H. Hummelsheim, P. Denzler, K.H. Mauritz: Repetitive training of isolated movements improves the outcome of motor rehabilitation of the centrally paretic hand, J. Neurol. Sci. **130**(1), 59–68 (1995)
73.109 G. Kwakkel, R.C. Wagenaar, J.W. Twisk, G.J. Lankhorst, J.C. Koetsier: Intensity of leg and arm training after primary middle-cerebral artery stroke: A randomized trial, Lancet **354**, 191–196 (1999)
73.110 S. Wolf, S. Blanton, H. Baer, J. Breshears, A.J. Butler: Repetitive task practice: A critical review of constraint induced therapy in stroke, Neurologist **8**(6), 325–338 (2002)
73.111 S.L. Wolf, C.J. Winstein, J.P. Miller, E. Taub, G. Uswatte, D. Morris, C. Giuliani, K.E. Light, D. Nichols-Larsen: Effect of constraint-induced movement therapy on upper extremity function 3 to 9 months after stroke, J. Am. Med. Assoc. (JAMA) **296**(17), 2095–2104 (2006)
73.112 T. Nef, R. Riener: ARMin: Design of a novel arm rehabilitation robot, Proc. Int. Conf. Rehabil. Robotics (2005)
73.113 N. Schweighofer, Y. Choi, C. Winstein, J. Gordon: Task-oriented rehabilitation robotics, Am. J. Phys. Med. Rehabil. **91**(11), 270–279 (2012)
73.114 K. Wada, T. Shibata, T. Saito, K. Tanie: Analysis of factors that bring mental effects to elderly people in robot assisted activity, Proc. Int. Conf. Intell. Robots Syst. (2002) pp. 1152–1157
73.115 C.D. Kidd, W. Taggart, S. Turkle: A sociable robot to encourage social interaction among the elderly, Int. Conf. Robotics Autom. (2006) pp. 3972–3976
73.116 C.G. Burgar, P.S. Lum, P.C. Shor, M. Van der Loos: Development of robots for rehabilitation therapy: The Palo Alto vs/Stanford experience, J. Rehabil. Res. Dev. **37**(6), 663–673 (2000)
73.117 H.I. Krebs, B.T. Volpe, M. Ferraro, S. Fasoli, J. Palazzolo, B. Rohrer, L. Edelstein, N. Hogan: Robot-aided neurorehabilitation: From evidence-based to science-based rehabilitation, Top Stroke Rehabil. **8**(4), 54–70 (2002)
73.118 B.R. Brewer, R. Klatzky, Y. Matsuoka: Feedback distortion to overcome learned nonuse: A system overview IEEE Eng. Med, Biol. **3**, 1613–1616 (2003)

73.119 D.J. Reinkensmeyer, C.T. Pang, J.A. Nesseler, C.C. Painter: Web-based tele-rehabilitation of the upper extremity after stroke, IEEE Tran. Neural Syst. Rehabil. Eng. **10**(2), 102–108 (2002)

73.120 S. Hesse, H. Schmidt, C. Werner, A. Bardeleben: Upper and lower extremity robotic devices for rehabilitation and for studying motor control, Curr. Opin. Neurol. **16**(6), 705–710 (2003)

73.121 S. Hesse, G. Schulte-Tigges, M. Konrad, A. Bardeleben, C. Werner: Robot-assisted arm trainer for the passive and active practice of bilateral forearm and wrist movements in hemiparetic subjects, Arch. Phys. Med. Rehabil. **84**(6), 915–920 (2003)

73.122 S. Hesse, C. Werner: Poststroke motor dysfunction and spasticity: Novel pharmacological and physical treatment strategies, CNS Drugs **17**(15), 1093–1107 (2003)

73.123 C. Winstein, R.A. Schmidt: Reduced frequency of knowledge of results enhances motor skill learning, J. Exp. Psychol. Learn. Mem. Cognit. **16**(4), 677–691 (1990)

73.124 C. Winstein: Knowledge of results and motor learning–implications for physical therapy, Phys. Ther. **71**(2), 140–149 (1991)

73.125 C. Winstein, P.S. Pohl, R. Lewthwaite: Effects of physical guidance and knowledge of results on motor learning: Support for the guidance hypothesis, Res. Q. Exerc. Sport. **65**(4), 316–323 (1994)

73.126 C. Winstein, A. Merians, K. Sullivan: Motor learning after unilateral brain damage, Neuropsychologia **37**(8), 975–987 (1999)

73.127 J. Eriksson, M.J. Matarić, C. Winstein: Hands-off assistive robotics for post stroke arm rehabilitation, Proc. IEEE Int. Conf. Rehabil. Robotics (ICORR'05) (2005) pp. 21–24

73.128 M. Ghahramanlou, J. Arnoff, M.A. Wozniak, S.J. Kittner, T.R. Price: Personality influences psychological adjustment and recovery from stroke, Proc. Am. Stroke Assoc. 26th Int. Stroke Conf. (2001)

73.129 H. Nakajima, Y. Morishima, R. Yamada, S. Brave, H. Maldonado, C. Nass, S. Kawaji: Social intelligence in a human-machine collaboration system: Social responses to agents with mind model and personality, J. Jpn. Soc. Artif. Intell. **19**(3), 184–196 (2004)

73.130 H. Nakajima, S. Braveand, C. Nass, R. Yamada, Y. Morishima, S. Kawaji: The functionality of human-machine collaboration systems mind model and social behavior, Proc. IEEE Conf. Syst. Man Cybern. (2003) pp. 2381–2387

73.131 C. Nass, M.K. Lee: Does computer-synthesized speech manifest personality? Experimental tests of recognition, similarity-attraction, and consistency attraction, J. Exp. Psychol. Appl. **7**(3), 171–181 (2001)

73.132 A. Tapus, M. Matarić: Towards socially assistive robotics, Int. J. Robotics Soc. Jpn. (JRSJ) **24**(5), 14–16 (2006)

73.133 A. Bandura: *Principles of Behavior Modification* (Holt, Rinehart and Winston, New York 1969)

73.134 E.T. Hall: *Hidden Dimension* (Doubleday, Gorden City 1966)

73.135 E. Wade, A. Parnandi, M. Matarić: Automated administration of the wolf motor function test for post-stroke assessment, 4th Int. ICST Conf. Pervasive Comput. Technol. Heal. (Pervasive Health 2010) (2010) pp. 1–7

73.136 A. Parnandi, E. Wade, M. Matarić: Motor function assessment using wearable inertial sensors, 32nd Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC'10) (2010)

73.137 T.W. Bickmore, R.W. Picard: Establishing and maintaining long-term human-computer relationships, ACM Tran. Comput.-Hum. Interact. **12**(2), 293–327 (2005)

73.138 H.F. Davis, J.B. Croft, A.M. Malarcher, C. Ayala, T.L. Antoine, A. Hyduk: Public health and aging: Hospitalizations for stroke among adults aged > 65 years, CDC Morbid. Mortal. Wkl Rep. (MMWR) **52**(25), 581–604 (2003)

73.139 American Association of Colleges of Nursing: *Nursing Shortage Fact Sheet* (American Association of Colleges of Nursing, Washington 2010)

73.140 American Health Care Association: *Summary of 2007 AHCA Survey Nursing Staff Vacancy and Turnover in Nursing Facilities Report* (American Health Care Association, Washington 2008)

73.141 P. Buerhaus: Current and future state of the US nursing workforce J. Am. Med, Assoc. **300**(20), 2422–2424 (2008)

73.142 E. Baum, D. Jarjoura, A. Polen, D. Faur, G. Rutecki: Effectiveness of a group exercise program in a long-term care facility: A randomized pilot trial J. Am. Med. Dir, Assoc. **4**(2), 74–80 (2003)

73.143 D. Dawe, R. Moore-Orr: Low-intensity, range-of-motion exercise: Invaluable nursing care for elderly patients, J. Adv. Nurs. **21**, 675–681 (1995)

73.144 M.D. McMurdo, L.M. Rennie: A controlled trial of exercise by residents of old people's homes, Age Ageing **22**, 11–15 (1993)

73.145 V. Thomas, P. Hageman: Can neuromuscular strength and function in people with dementia be rehabilitated using resistance-exercise training? Results from a preliminary intervention study, J. Gerontol. A Biol. Sci. Med, Sci. **58**, 746–751 (2003)

73.146 S. Colcombe, A. Kramer: Fitness effects the cognitive function of older adults, Psychol. Sci. **14**(2), 125–130 (2003)

73.147 S. Colcombe, A. Kramer, K. Erickson, P. Scalf, E. McAuley, N. Cohen, S. Elavsky: Cardiovascular fitness, cortical plasticity, and aging, Proc. Natl Acad. Sci, USA **101**(9), 3316–3321 (2004)

73.148 W. Spirduso, P. Clifford: Replication of age and physical activity effects on reaction and movement time, J. Gerontol. **33**(1), 26–30 (1978)

73.149 Z.B. Moak, A. Agrawal: The association between perceived interpersonal social support and physical and mental health: Results from the national epidemiological survey on alcohol and related conditions, J. Public Health **32**, 191–201 (2010)

73.150 L. George, D. Blazer, D. Hughes, N. Fowler: Social support and the outcome of major depression, Br. J. Psychiatry **154**, 478–485 (1989)

73.151 E. Paykel: Life events, social support and depression, Acta Psychiatr. Scand. **89**, 50–58 (1994)

73.152 S. Stansfeld, G. Rael, J. Head, M. Shipley, M. Marmot: Social support and psychiatric sickness absence: A prospective study of British civil servants, Psychol. Med. **27**(1), 35–48 (1997)

73.153 E. Stice, J. Ragan, P. Randall: Prospective relations between social support and depression: Differential direction of effects for parent and peer support?, J. Abnorm. Psychol. **113**, 155–159 (2004)

73.154 E. Steinmetz: *American with Disabilities: 2002 Current Population Reports* (Census Bureau, Washington 2006)

73.155 M.J. Field, A.M. Jette: *The Future of Disability in America* (National Acadamies Press, Washington 2007)

73.156 S.L. McGinnis, J. Moore: The impact of the aging population on the health workforce in the United States–summary of key findings, Cah. Sociol. Demogr. Med. **46**(2), 193–220 (2006)

73.157 R.A. Cooper, B.E. Dicianno, B. Brewer, E. LoPresti, D. Ding, R. Simpson, G. Grindle, H. Wang: A perspective on intelligent devices and environments in medical rehabilitation, Med. Eng. Phys. **30**(10), 1387–1398 (2008)

73.158 American Heart Association: *Heart and Stroke Statistics, 2006 Update* (American Heart Association, Dallas 2006)

73.159 M.F. Diagram: *White Paper of the Elderly* (Japan National Council of Social Welfare, Tokyo 2001)

73.160 Eurostatistics: Population Projections 2008-2060: From 2015, deaths projected to outnumber births in the EU27, http://europa.eu/rapid/press-release_STAT-08-119_en.htm (2008)

73.161 J. Rimmer: Exercise and physical activity in persons aging wityh a physical disability, Phys. Med. Rehabil. Clin. N. Am. **16**, 41–56 (2005)

73.162 Experience Corps: *Fact Sheet on Aging in America* (AARP, Washington 2007)

73.163 B.J. Kemp: What the rehabilitation professional and the consumer need to know, Phys. Med. Rehabil. Clin. N. Am. **16**(1), 1–18 (2005)

73.164 B.J. Kemp, L. Mosqueda: *Aging with a Disability: What the Clinician Needs to Know* (Johns Hopkins Univ. Press, Baltimore 2004)

73.165 A.W. Heinemann: State-of-the-science on postacute rehabilitation: Setting a research agenda and developing an evidence base for practice and public policy. An introduction, J. Spinal. Cord. Med. **30**(5), 452–457 (2007)

73.166 A.W. Heinemann: State of the science of postacute rehabilitation: Setting a research agenda and developing an evidence base for practice and public policy. An introduction, Rehabil. Nurs. **33**(2), 82–87 (2008)

73.167 U. Tas, A.P. Verhagen, S.M.A. Bierma-Zeinstra, E. Odding, B.W. Koes: Prognostic factors of disability in older people: A systematic review, Br. J. Gen. Pract. Syst. Rev. **57**(537), 319–323 (2007)

73.168 S.G. Leveille, B.W. Penninx, D. Melzer, G. Izmirlian, J.M. Guralnik: Sex differences in the prevalence of mobility disability in old age: the dynamics of incidence, recovery, and mortality, J. Gerontol. B Psychol. Sci. Soc, Sci. **55**(1), 41–50 (2000)

73.169 J. Fasola, M.J. Matarić: Using socially assistive human-robot interaction to motivate physical exercise for older adults, Proc. IEEE **100**(8), 2512–2526 (2012)

73.170 American Alzheimer Association: About Alzheimer's Disease Statistics (American Alzheimer Association, Chicago 2007)

73.171 T. Tamura, S. Yonemitsu, A. Itoh, D. Oikawa, A. Kawakami, Y. Higashi, T. Fujimooto, K. Nakajima: Is an entertainment robot useful in the care of elderly people with severe dementia?, J. Gerontol. A Biol. Sci. Med, Sci. **59**(1), 83–85 (2004)

73.172 A. Tapus, C. Tapus, M.J. Matarić: The use of socially assistive robots in the design of intelligent cognitive therapies for people with dementia, Int. Conf. Rehabil. Robotics (2009)

73.173 C. Jayawardena, I. Kuo, C. Datta, R.Q. Stafford, E. Broadbent, B.A. MacDonald: Design, implementation and field tests of a socially assistive robot for the elderly: HealthBot version 2, 4th IEEE RAS/EMBS Int. Conf. Biomed. Robotics Biomechatron. (BioRob) (2012) pp. 1837–1842

73.174 W. Taggart, S. Turkle, C.D. Kidd: An interactive robot in a nursing home: Preliminary remarks. In: *Towards Social Mechanisms of Android Science*, (COGSCI Workshop), Stresa, 2005)

73.175 N. Sharkey, A. Sharkey: Living with robots: Ethical tradeoffs in eldercare, Nat. Language Process. **8**, 245–256 (2010)

73.176 D. Feil-Seifer, M.J. Matarić: Ethical principles for socially assistive robotics, IEEE Robotics Autom Mag. **18**(1), 24–31 (2011)

Multimedia Contents

74. Learning from Humans

Aude G. Billard, Sylvain Calinon, Rüdiger Dillmann

This chapter surveys the main approaches developed to date to endow robots with the ability to learn from human guidance. The field is best known as robot programming by demonstration, robot learning from/by demonstration, apprenticeship learning and imitation learning. We start with a brief historical overview of the field. We then summarize the various approaches taken to solve four main questions: when, what, who and when to imitate. We emphasize the importance of choosing well the interface and the channels used to convey the demonstrations, with an eye on interfaces providing force control and force feedback. We then review algorithmic approaches to model skills individually and as a compound and algorithms that combine learning from human guidance with reinforcement learning. We close with a look on the use of language to guide teaching and a list of open issues.

74.1 **Learning of Robots** 1995
74.1.1 Principle 1996
74.1.2 Brief History 1996
74.2 **Key Issues When Learning from Human Demonstrations** 1998
74.2.1 When and Whom to Imitate 1998
74.2.2 How to Imitate and How to Solve the Correspondence Problem 1999
74.3 **Interfaces for Demonstration** 2000
74.4 **Algorithms to Learn from Humans** 2002
74.4.1 Learning Individual Motions 2002
74.4.2 Learning Compound Actions 2003
74.4.3 Incremental Teaching Methods ... 2004
74.4.4 Combining Learning from Humans with Other Learning Techniques 2005
74.4.5 Learning from Humans, a Form of Human–Robot Interaction 2006
74.5 **Conclusions and Open Issues in Robot LfD** 2008
Video-References 2009
References 2009

74.1 Learning of Robots

Robot learning from humans relates to situations in which the robot learns from interacting with a human. This must be contrasted to the vast body of work on robot learning where the robot learns *on its own*, that is, through trial and error and without external guidance. In this chapter, we cover works that combine reinforcement learning (RL) with techniques that use human guidance, e.g., to bootstrap the search in RL. However, we exclude from this survey all works that use purely reinforcement learning, even though one could argue that providing a reward is one form of human guidance. We consider that providing a reward function is akin to providing an objective function and hence refer the reader to the companion chapter on *Machine Learning* for robotics. We also exclude works where the robot learns implicitly from being in presence of a human, while the human is not actively coaching the robot, as these works are covered in the companion chapter on *Social Robotics*. We hence focus our survey to all works where the human is actively teaching the robot, by providing *demonstrations* of how to perform the task.

Various terminologies have been used to refer to this body of work. These include programming by demonstration (PbD), learning from human demonstration (LfD), *imitation learning*, and *apprenticeship learning*. All of these refer to a general paradigm for enabling robots to autonomously perform new tasks from ob-

serving and learning, therefore, from the observation of humans performing these tasks.

74.1.1 Principle

Rather than requiring users to analytically decompose and manually program a desired behavior, work in LfD-PbD takes the view that an appropriate robot controller can be derived from observations of a human's own performance thereof. The aim is for robot capabilities to be more easily extended and adapted to novel situations, even by users without programming ability:

> *The main principle of robot learning from demonstration is that end-users can teach robots new tasks without programming.*

Consider a household robot capable of performing manipulation tasks. One task that an end-user may desire the robot to perform is to prepare a meal, such as preparing an orange juice for breakfast (Fig. 74.1 and VIDEO 29). Doing so may involve multiple subtasks, such as juicing the orange, throwing the rest of the orange in the trash, and pouring the liquid into a cup. Further, every time this meal is prepared, the robot will need to adapt its motion to the fact that the location and type object (cup, juicer) may change.

In a traditional programming scenario, a human programmer would have to code a robot controller that is capable of responding to any situation the robot may face. The overall task may need to be broken down into tens or hundreds of smaller steps, and each one of these steps should be tested for robustness prior to the robot leaving the factory. If and when failures occur in the field, highly-skilled technicians would need to be dispatched to update the system for the new circumstances. Instead, LfD allows the end-user to *program* the robot simply by showing it how to perform the task – no coding is required. Then, when failures occur, the end-user only needs to provide more demonstrations, rather than calling for professional help. LfD hence seeks to endow robots with the ability to learn what it means to perform a task by generalizing from several observations (Fig. 74.1 and VIDEO 29).

LfD is not a record and play technique. LfD implies learning, henceforth, generalization.

Next, we give a brief historical overview of the way the field evolved over the years. This is followed, in Sect. 74.2, by an introduction to the issues at the core of LfD. In Sect. 74.3, we discuss the crucial role that the interface used for LfD plays in the success of the teaching, emphasizing how the choice of interface determines the type of information that can be conveyed to the robot. Finally, in Sect. 74.4, we give a generic view of the main approaches to solving LfD and conclude with an outlook on open issues.

74.1.2 Brief History

Robot learning from demonstration started in the 1980s. Then, and still to a large extent now, robots had to be explicitly and tediously hand programmed for each task they had to perform. PbD sought to minimize, or even eliminate, this difficult step.

The rationale for moving from purely preprogrammed robots to very flexible user-based interfaces for training the robot to perform a task is threefold. First and foremost, PbD is a powerful mechanism for reducing the complexity of search spaces for learning. When observing either good or bad examples, one can reduce the search for a possible solution, by either starting the search from the observed good solution (local optima), or conversely, by eliminating from the search space what is known as a bad solution. Imitation learning is, thus, a powerful tool for enhancing and accelerating learning in both animals and artifacts.

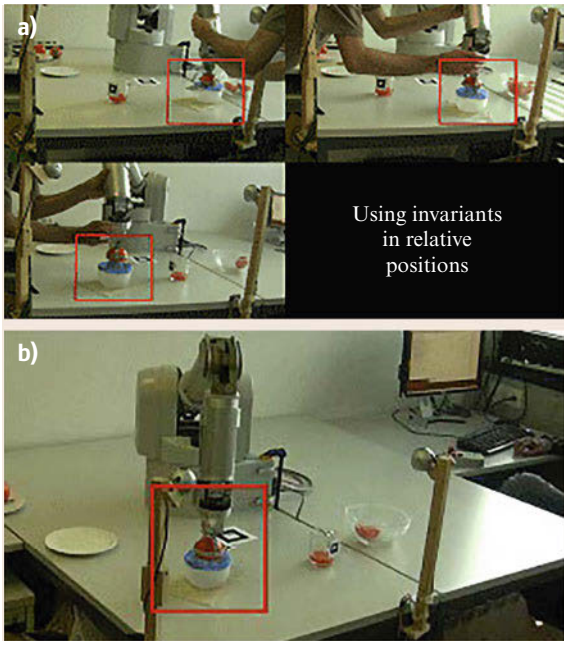


Fig. 74.1 **(a)** The teacher does several demonstrations of the task of juicing an orange, by changing the location of each item to allow the robot to generalize correctly. That is, the robot should be able to infer, by comparing the demonstrations, that only the relative locations matter, as opposed to the exact locations as recorded from a global coordinate system. **(b)** The robot can then reproduce the task even when the objects are located in positions not seen in the demonstrations VIDEO 29

Second, imitation learning offers an implicit means of training a machine, such that explicit and tedious programming of a task by a human user can be minimized or eliminated. Imitation learning is thus a *natural* means of interacting with a machine that would be accessible to lay people.

Third, studying and modeling the coupling of perception and action, which is at the core of imitation learning, helps us to understand the mechanisms by which the self-organization of perception and action could arise during development. The reciprocal interaction of perception and action could explain how competence in motor control can be grounded in the rich structure of perceptual variables, and vice versa, how the processes of perception can develop as means to create successful actions.

PbD promises were thus multiple. On the one hand, one hoped that it would make the learning faster, in contrast to trial-and-error methods trying to learn the skill *tabula rasa*. On the other hand, one expected that being user-friendly, the methods would enhance the application of robots in human daily environments.

At the beginning of the 1980s, LfD, known then as programming by demonstration (PbD), started attracting attention in manufacturing robotics. PbD appeared as a promising route to automate the tedious manual programming of robots, reducing the costs involved in the development and maintenance of robots in the factory.

As a first approach in PbD, symbolic reasoning was commonly adopted in robotics [74.1–5], with processes referred to as *teach-in*, *guiding*, or *playback* methods. In these works, PbD was performed through manual (teleoperated) control. The position of the end-effector and the forces applied on the object manipulated were stored throughout the demonstrations together with the positions and orientations of the obstacles and of the target. This sensorimotor information was then segmented into discrete subgoals (key points along the trajectory) and into appropriate pre-defined actions to attain these subgoals. Actions were commonly chosen to be simple point-to-point movements that industrial robots employed at this time. Examples of subgoals would be, e.g., the robot's gripper orientation and position in relation to the goal [74.3]. Consequently, the demonstrated task was segmented into a sequence of state-action-state transitions.

To take into account the variability of human motion and the noise inherent to the sensors capturing the movements, it appeared necessary to develop a method that would consolidate all demonstrated movements. For this purpose, the state-action-state sequence was converted into symbolic *if-then* rules, describing the states and the actions according to symbolic relationships, such as *in contact*, *close-to*, *move-to*, *grasp-object*, *move-above*, etc. Appropriate numerical definitions of these symbols (i. e., when would an object be considered as *close-to* or *far-from*) were given as prior knowledge to the system. A complete demonstration was thus encoded in a graph-based representation, where each state constituted a graph node and each action a directed link between two nodes. Symbolic reasoning could then unify different graphical representations for the same task by merging and deleting nodes [74.2].

Munch et al. [74.6] suggested the use of machine learning (ML) techniques to recognize elementary operators (EOs), thus defining a discrete set of basic motor skills, with industrial robotics applications in mind. In this early work, the authors already established several key issues of PbD in robotics. These include questions such as how to generalize a task, how to reproduce a skill in a completely novel situation, how to evaluate a reproduction attempt, and how to better define the role of the user during learning. *Munch* et al. [74.6] admitted that generalizing over a sequence of discrete actions was only one part of the problem since the controller of the robot also required the learning of continuous trajectories to control the actuators. They proposed to overcome the missing parts of the learning process by leveraging them to the user, who took an active role in the teaching process.

These early works highlighted the importance of providing a set of examples that are usable by the robot: (1) by constraining the demonstrations to modalities that the robot can understand; and (2) by providing a sufficient number of examples to achieve a desired generality. They noted the importance of providing an adaptive controller to reproduce the task in new situations, that is, how to adjust an already acquired program. The evaluation of a reproduction attempt was also leveraged to the user by letting him/her provide additional examples of the skill in the regions of the learning space that had not been covered yet. In this way, the teacher/expert could control the generalization capabilities of the robot.

With the increasing development of mobile and humanoid robots, the field went on adopting an interdisciplinary approach, taking into account evidence of specific neural mechanisms for visuomotor imitation in primates [74.7–9] and of developmental stages of imitation capacities in children [74.10, 11]. The latter promotes the introduction of socially driven behavior in the robot to sustain interaction and improve teaching [74.12, 13] and of an interactive teaching process, in which the robot takes a more active role and may ask the user for additional sources of information, when

needed [74.14, 15]. Eventually, the notion of *robot programming by demonstration* was replaced by the more biological labeling of *imitation learning*. In essence, a large part of current works in PbD follow a conceptual approach very similar to that followed by these prior works.

Recent progress affected mostly the interfaces at the basis of the teaching. Traditional ways of guiding/teleoperating the robot have been progressively replaced by more user-friendly interfaces, such as vision [74.16, 17], speech command [74.18], data gloves [74.19], the laser range finder [74.20] or kinesthetic teaching (i. e., by manually guiding the robot's arms through the motion) [74.21–23].

The field progressively moved from simply copying the demonstrated movements to generalizing across sets of demonstrations. As machine learning progressed, PbD started incorporating more of those tools to tackle both the perception issue, i. e., how to generalize across demonstrations, and the production issue, i. e., how to generalize the movement to new situations. Initially, tools such as artificial neural networks (ANNs) [74.24, 25], radial-basis function networks (RBFs) [74.26], and *fuzzy logic* [74.27] were quite popular. These have lately been replaced by hidden Markov models (HMMs) [74.28–33] and various non-linear regression techniques [74.21, 34, 35], as we will discuss in more detail in Sect. 74.4.

New learning challenges were, thus, set forth. Robots were expected to show a high degree of flexibility and versatility both in their learning system and in their control system in order to be able to interact naturally with human users and demonstrate similar skills (e.g., by moving in the same rooms and manipulating the same tools as humans). Robots were more and more expected to act *human-like* to enhance the interaction and so that their behavior would be more predictable and, hence, more acceptable.

74.2 Key Issues When Learning from Human Demonstrations

As mentioned in the beginning, learning from demonstration (LfD) has at core to develop algorithms that are generic in their representation of the skills and in the way they generate the skills.

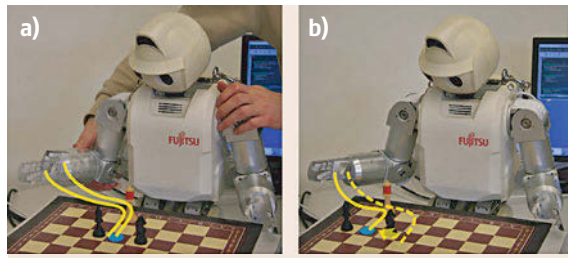
The field has identified a number of key problems that need to be solved for ensuring such a generic approach to transferring skills across various agents and situations [74.36, 37]. These have been formulated as a set of generic questions, namely *what to imitate*, *how to imitate*, *when to imitate*, and *who to imitate*. These questions were formulated in response to the large body of diverse work in robotics LfD [74.18, 26, 38–41] that could not easily be unified under a small number of coherent operating principles. The above four questions and their solutions aim at being generic in the sense of making no assumptions on the type of skills that may be transmitted.

74.2.1 When and Whom to Imitate

Whom and when to imitate has been largely unexplored so far, and hence to date, only the first two questions have really been addressed. Figure 74.2 and VIDEO 97 illustrate how these two problems can be solved in a principled manner through statistical observation of the demonstrations.

How to Determine the Evaluation Metric

What to imitate relates to the problem of determining which aspects of the demonstration should be imitated. For a given task, certain observable or affectable properties may be irrelevant and safely ignored. For instance, if the demonstrator always approaches a location from the north, is it necessary for the robot to do the same? The answer to this question strongly in-



Observation of multiple demonstrations | Reproduction of a generalized motion in a different situation

Fig. 74.2 **(a)** A robot learns how to make a chess move (namely moving the queen forward) by generalizing across different demonstrations of the task performed in slightly different situations (different starting positions of the hand). The robot records the trajectories of its joints and learns to extract invariant features (*what-to-imitate*), i. e., that the task constraints are reduced to a subpart of the motion located in a plane defined by the three chess pieces. **(b)** The robot reproduces the skill in a new context (for a different initial position of the chess piece) by finding an appropriate controller that satisfies both the task constraints and constraints relative to its body limitation (*how-to-imitate* problem) (after [74.21])

fluences whether or not a derived robot controller is a successful imitation – a robot that approaches from the south is appropriately trained if direction is not important, but needs further education if it is. This issue is related to questions of signal versus noise and is answered by determining the metric by which the resulting behavior is evaluated. Different ways can be taken to address this issue. The simplest approach is to take a statistical perspective and deem as relevant the parts (dimension, region of input space) of the data that are consistently measured across all demonstration instances [74.21]. If the dimension of the data is too high, such an approach may require too many demonstrations to gather enough statistics. An alternative is then to have the teacher help the robot determine what is relevant by pointing out the parts of the task that are most important.

In summary, what to imitate removes consideration of details that, while perceptible/performable, do not matter for the task. It participates in determining the metric by which the reproduction of the robot can be measured. In continuous control tasks, what to imitate relates to the problem of defining automatically the feature space for learning, as well the constraints and the cost function. In discrete control tasks, such as those treated by reinforcement learning and symbolic reasoning, what to imitate relates to the problem of how to define the state and action space and of how to automatically learn the pre/post conditions in an autonomous decision system.

74.2.2 How to Imitate and How to Solve the Correspondence Problem

How to imitate consists in determining how the robot will actually perform the learned behaviors to maximize the metric found when solving the what to imitate problem. Often, a robot cannot act exactly the same way as a human does, due to differences in physical embodiment. For example, if the demonstrator uses a foot to move an object, is it acceptable for a wheeled robot to bump it, or should it use a gripper instead? If the metric does not have appendage-specific terms, it may not matter.

This issue is closely related to that of the correspondence problem [74.36]. Robots and humans, while inhabiting the same space and interacting with the same objects, and perhaps even superficially similar, still perceive and interact with the world in fundamentally different ways. To evaluate the similarity between human behavior and that of robots, we must first deal with the fact that humans and robots may occupy different state spaces, of perhaps different dimensions. We identify two different ways in which states of demonstrator and imitator can be said to correspond, and give brief examples:

- *Perceptual equivalence*: Due to differences between human and robot sensory capabilities, the same scene may appear to be very different. For instance, while a human may identify humans and

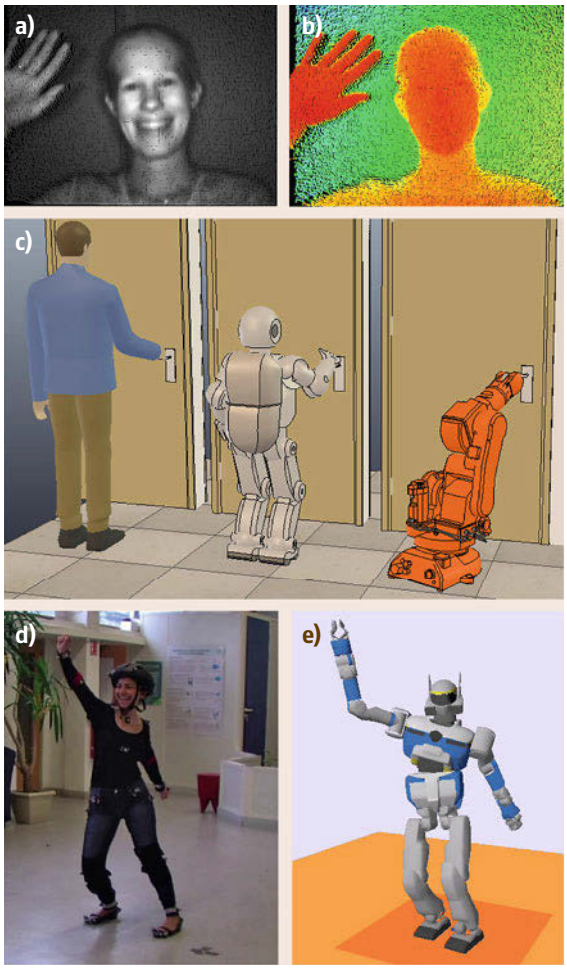


Fig. 74.3 **(a,b)** Perceptual equivalence (adapted from [74.42]). **(c)** Physical equivalence. The humanoid robot has the same arrangement of principal articulations as the human demonstrator, but different limb lengths and joint angle limits. The industrial robot has a different number and arrangement of articulations, which makes the mapping problem more challenging (illustration created with the V-REP simulator [74.43]). **(d,e)** Offline full-body motion transfer by taking into account the kinematic and dynamic disparity between the human and the humanoid [74.44]. See also VIDEO 98 and VIDEO 99 for example of mapping of full body motion from human to humanoids

gestures from color and intensity, a robot may use depth measurements to observe the same scene (Fig. 74.3a). Another point of comparison is tactile sensing. Most tactile sensors allow robots to perceive contact, but do not offer information about temperature, in contrast to the human skin. Moreover, the low resolution of the robots' tactile sensors does not allow robots to discriminate across the variety of existing textures, while human skin does. As the same data may, therefore, not be available to both humans and robots, successfully teaching a robot may require a good understanding of the robot's sensors and their limitations. LfD explores the limits of these perceptual equivalences, by building interfaces that either automatically correct or make explicit these differences.

- *Physical equivalence*: Due to differences between human and robot embodiments, humans and robots may perform different actions to accomplish the same physical effect. For instance, even when performing the same task (soccer), humans and robots may interact with the environment in different ways (Fig. 74.3b). Humans run and kick, while robots roll and bump. Solving this discrepancy in motor capabilities is akin to solving the *how to imitate* problem to achieve the same effect. LfD develops way to solve this problem. Typically, the robot may compute a path (in Cartesian space) for its end-effector that is close to the path followed by the human hand, while relying on inverse kinematics to find the appropriate joint displacements. In the football example above, this would require the robot to determine a path for its center of mass which corresponds to the path followed by the human's right foot when projected on the ground. Clearly, this equivalence is very task dependent. Recent solutions to this problem for hand motion and body motion can be found in [74.45, 46].

We can think of perceptual equivalence as dealing with the manner in which the agents perceive the world. Perceptual equivalence requires to make sure that the information necessary to perform the task is available to both humans and robots. Physical equivalence deals with the manner in which agents affect and interact with the world, so that the task is performable by both agents.

74.3 Interfaces for Demonstration

The interface used to provide demonstration plays a key role in the way the information is gathered and transmitted. We distinguish three major trends:

1. One may directly record human motions. If one is interested solely in the kinematic of the motion, one may use any of the various existing motion tracking systems, whether these are based on vision, exoskeleton, or other types of wearable motion sensors. The left-hand side of Fig. 74.4b and VIDEO 98 show an example of full body motion tracking during walking using vision. The motion of the human body is first extracted from the background using a model of human body. This model is subsequently mapped to an avatar and then to the humanoid robot DB at ATR, Kyoto, Japan.
   These external means of tracking human motion return precise measurement of the angular displacement of the limbs and joints. They have been used in various works for LfD of full body motion [74.33, 47–49]. These methods are advantageous in that they allow the human to move freely. However, they require solutions to the correspondence problem, i. e., the problem of how to transfer motion from human to robot when both differ in the kinematic and dynamics of their body or, in other words, if the configuration space is of different dimension and size. This is typically done when mapping the motion of the joints that are tracked visually to a model of the human body that matches closely that of the robot. Such mapping would be particularly difficult to perform when the walking machine (e.g., a hexapod) differs importantly from the human body. The problem of mapping actions across two dissimilar bodies was already evoked earlier on and refers to the correspondence problem.
2. Second, there are techniques such as kinesthetic teaching, where the robot is physically guided through the task by the human. This approach simplifies the correspondence problem by letting the user demonstrate the skill in the robot's environment with the robot's own capabilities. It also provides a natural teaching interface to correct a skill reproduced by the robot. Recent advances in skin technology offer the possibility to teach robots how to exploit tactile contact on an object (Fig. 74.4 middle and VIDEO 104). By exploiting the compliance of the iCub robot's fingers, the teacher can teach the robot how to adapt the posture of the fingers in response to a change in tactile sensing as measured at the robot's finger tips [74.50]).

One main drawback of kinesthetic teaching is that the human must often use more degrees of freedom to move the robot than the number of degrees of freedom moved on the robot. This is visible in Fig. 74.4. To move the fingers of one hand of the robot, the teacher must use both hands. This limits the type of tasks that can be taught through kinesthetic teaching. Typically tasks that would require moving both hands simultaneously could not be taught this way. One could either proceed incrementally, teaching first the task for the right hand and then, while the robot replays the motion with its right hand, teach the motion of the left hand. However, this may prove to be cumbersome. The use of external trackers as reviewed above are more amenable to teaching coordinated motion between several limbs.

3. Third, there are immersive teleoperation scenarios, where a human operator is limited to using the robot's own sensors and effectors to perform the task. Teleoperation may be done using simple joysticks or other remote control devices, including haptic devices (Fig. 74.4 bottom and VIDEO 101). The later have the advantage that they can allow the teacher to teach tasks that require precise control of forces, while joysticks would only provide kinematic information (position, speed).
   Teleoperation is advantageous compared to external motion tracking systems, as this solves the correspondence problem entirely, since the system directly records the perception and action from the robot's configuration space. It is also advantageous compared to kinesthetic training, as it allows training the robots from a distance and is, hence, particularly suited for teaching navigation and locomotion patterns. The teacher no longer needs to share the same space with the robot. Teleoperation is, usually, used to transmit the kinematics of motion. For instance, in [74.51], the acrobatic trajectories of a helicopter are learned by recording the motion of the helicopter when teleoperated by an expert pilot. In [74.52], a robot dog is taught to play soccer by a human guiding it via a joystick. However, in recent work, teleoperation has been used successfully to teach a humanoid robot balancing techniques [74.53]. Learning to react to perturbations is done through a haptic interface attached to the torso of the demonstrator, which measures the interaction forces when the human is pushed around. The kinematics of motion of the demonstrator are directly transmitted to the robot through teleoperation and are combined with haptic information to train a model of motion conditioned on perceived forces.
   The disadvantage of teleoperation techniques is that the teacher often needs training to learn to use the remote control device. Teleoperation using a simple joystick allows guiding only a subset of degrees of freedom. To control for all degrees of freedom, very complex, exoskeleton type of devices must be used, which can be cumbersome. Moreover, teleoperation prevents the teacher from observing all sensorial information required to perform the task. For instance, teleoperation, even when using haptic device, poorly renders the contacts perceived at the robot's end-effector. To palliate to this, one may provide the teacher with visualization interfaces to simulate the interaction forces.
4. Lastly, one can use explicit information, such as that conveyed by speech, to provide additional advice and comments to the demonstration [74.18, 57, 58] and VIDEO 103. Speech is a very natural means of communication among humans and, hence, is viewed as an easy way to allow the end-user to communicate with robots. However, it necessitates that vocabulary that is understandable to the robot and

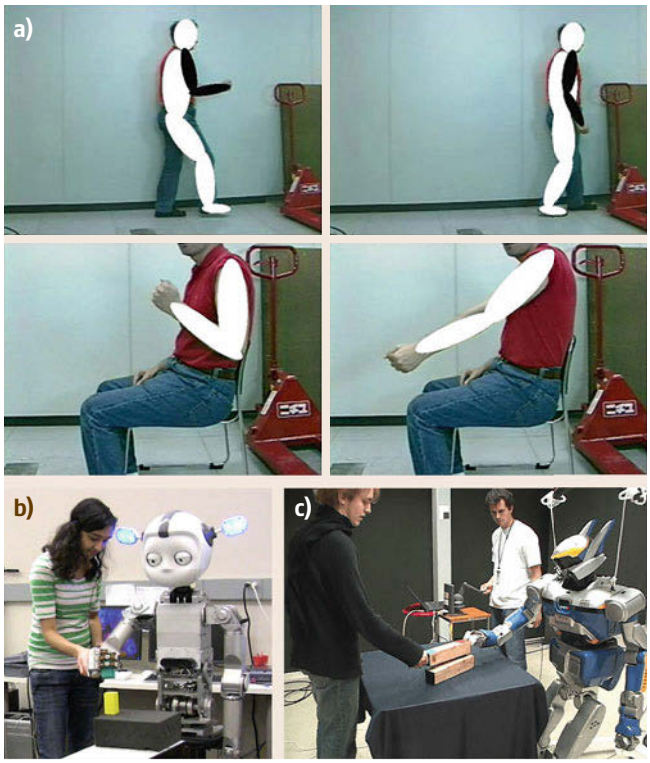


Fig. 74.4 **(a)** Demonstration by visual tracking of gestures (after [74.54], VIDEO 98 and VIDEO 99). **(b)** Demonstration by kinesthetic teaching (after [74.55] and VIDEO 104). **(c)** Demonstration by teleoperation (after [74.56] and VIDEO 101)

grounded in the actions and perceptions of the robot be defined beforehand. While this restricts teaching to discrete state–action pairs, it is particularly useful for symbolic reasoning.

Each teaching interface has its pros and cons. It is thus interesting to investigate how these interfaces could be used in conjunction to exploit complementary information provided by each modality [74.50].

74.4 Algorithms to Learn from Humans

Current approaches to encoding skills through LfD can be broadly divided into two trends: a low-level representation of the skill, taking the form of a non-linear mapping between sensory and motor information, and, a high-level representation of the skill that decomposes the skill into a sequence of action-perception units.

While the majority of work in LfD uses solely the demonstrations for learning, a growing number of works develops methods by which LfD can be combined with other learning techniques. One group of work investigates how to combine imitation learning with *reinforcement learning*, a method by which the robot learns through trial and error to maximize a given reward. Other works take inspiration in the way humans teach each other and introduce interactive and bidirectional teaching scenarios whereby the robot becomes an active partner during the teaching phase. We briefly review the main principles underlying each of these areas below:

74.4.1 Learning Individual Motions

Individual motions/actions (e.g., juicing an orange, trashing it, and pouring liquid into the cup in the example shown in 74.1) could be taught separately instead of simultaneously, as shown in this previous example. The human teacher would then provide one or more examples of each submotion. If learning proceeds from the observation of a single instance of the motion/action, one calls this *one-shot* learning [74.60]. Examples of learning locomotion patterns can be found in [74.61]. To make sure that this is not akin to simple record and play, the controller is provided with prior knowledge in the form of primitive motion patterns. Learning then consists of instantiating the parameters modulating these motion patterns.

Teaching can also proceed in batch mode after recording several demonstrations, or incrementally by adding recursively more information trial by trial [74.12, 50, 62]. When learning in batch mode, learning considers all examples and draws inference by comparing the individual demonstrations. Inference is usually based on a statistical analysis, where the demonstration signals are modeled via a probability density function, exploiting various non-linear regression techniques stemming from machine learning. Popular methods these days include Gaussian processes, Gaussian mixture Models, and support vector machines.

Choosing properly the variables to encode a particular movement is crucial, as it already implies part of the solution to the problem of defining what is important to imitate. Work in LfD encodes human movements in either joint space, task space, or torque space [74.63–65]. The encoding may be specific to cyclic motion [74.22], discrete motion [74.21], or to a combination of both [74.61].

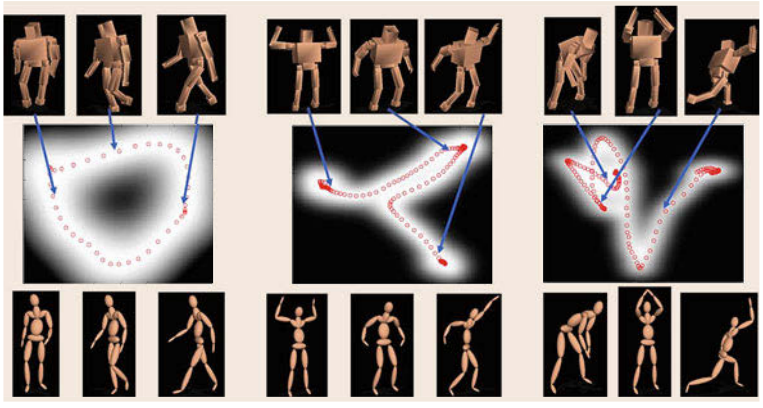


Fig. 74.5 Probabilistic encoding of motion in a subspace of reduced dimensionality (after [74.59] and VIDEO 102)

Encoding often encompasses the use of dimensionality reduction techniques that project the recorded signals into a latent space of motion of reduced dimensionality. These techniques may either perform a local linear transformations [74.66–68] or exploit global nonlinear methods [74.59, 69, 70] (Fig. 74.5). Additionally, task-specific rating functions [74.71] and simulation-based optimization [74.72] are investigated to identify relevant learning features.

Teaching Force-Control Tasks

While most LfD to date work focused on learning the kinematics of motions by recording the position of the end-effector and/or the position of the robot's joints, more recently, some works have investigated transmission of force-based signals through human demonstration [74.56, 73–76]. See VIDEO 478 and VIDEO 479 for examples of kinesthetic teaching of compliant motion. Transmitting information about force is difficult for humans and for robots alike. Force can be sensed only when performing the task ourselves. Current efforts, hence, seek to develop methods by which one may *embody* the robot. This allows human and robot to simultaneously perceive the forces applied when performing the task. A new exciting line of research, hence, leverages on recent advances in the design of haptic devices and tactile sensing, and on the development of torque and variable impedance actuated systems to teach force-control tasks through human demonstration.

74.4.2 Learning Compound Actions

Learning complex tasks, composed of a combination and juxtaposition of individual motions, is the ultimate goal of LfD. There are two major ways to proceed to learning of such complex tasks:

1. One may first learn models of all individual motions, using demonstrations of each of these actions individually. In a second stage, one may learn the right sequence and combination of these actions by observing a human performing the whole task. This approach, however, assumes that one can list all necessary individual actions, so-called *primitive* actions. To date, there does not exist a database of such primitive actions and one may wonder whether the variability of human motion may really be reduced to a finite list of possible motions. A common approach is to first learn models of all of the individual motions, using demonstrations of each of these actions individually [74.77, 78], and then learn the right sequencing/combination in a second stage either by observing a human performing the whole task [74.79, 80] or through reinforcement learning [74.81]. However, this approach assumes that there is a known set of all necessary primitive actions. For specific tasks this may be true, but to date there does not exist a database of general purpose primitive actions, and it is unclear whether the variability of human motion may really be reduced to a finite list.
2. The alternative is to observe the human performing the complete task and to automatically segment the task to extract the primitive actions, which may then become task-dependent, see e.g., [74.82, 83]. This has the advantage of learning, in one swipe, both the primitive actions and the way they should be combined. One issue that arises is that the number of primitive tasks is often unknown, and there could be multiple possible segmentations that must be considered [74.52].

Other examples include learning how to sequence known behaviors to enable complex navigation tasks through the imitation of a more knowledgeable robots or humans [74.9, 84, 85] and learning how to sequence primitive motions for full body motion in humanoid robots [74.25, 33, 86].

A large body of these works uses a symbolic representation of both the learning and the encoding of the task [74.6, 30, 85, 87–91]. This symbolic way of encoding skills may take several forms. One common way is to segment and encode the task according to sequences of *predefined* actions, described symbolically. Encoding and regenerating the sequences of these actions can, however, be done using classical machine learning techniques, such as HMM, [74.30].

Often, these actions are encoded in a hierarchical manner. In [74.85], a graph-based approach is used to generalize an object moving skill, using a wheeled mobile robot. In this model, each node in the graph represents a complete behavior and generalization takes place at the level of the topological representation of the graph. The latter is updated incrementally.

References [74.88, 89] follow a similar hierarchical and incremental approach to encode various household tasks (such as setting the table and putting dishes in a dishwasher) (Fig. 74.6 and VIDEO 103). There, learning consists in identifying a sequence of predefined, elementary actions, which is further combined into a hierarchical task network. By analyzing multiple demonstrations, the ordering of elementary actions is learned, resulting in a precedence graph. The precedence graph defines a partial ordering on the set of learned elementary actions, which can be exploited to execute elementary actions in parallel, extracting symbolic rules that manage the way each object must be

handled. In [74.92], the approach was extended to learning subsymbolic goal and constraint descriptions for each elementary action. In the execution phase, the robot applies motion planning to generate a motion to reach the goals while obeying the constraints. The resulting task description mimics the strategy that humans follow when performing the task. Based on the subsymbolic goal and constraint descriptions, the robot can reason to adapt the strategy to changes in object location, obstacle occurrence, and varying start configurations.

The approaches reviewed above assume a deterministic world, where actions unfold uniquely from perception of the current state of the world. However, robots operating in real environments will observe the world using imperfect sensors and the effects of their actions may be stochastic. To account for the stochasticity of the robot's perceptions and actions, *Schmidt-Rohr* et al. [74.93] use a model of the task with partially observable Markov decision processes (POMDP). At run time, an optimal (in a maximum likelihood sense) decision is then taken.

Reference [74.90] exploits also a hierarchical approach to encoding a skill in terms of pre-defined behaviors. The skill consists in moving through a maze where a wheeled robot must avoid several kinds of obstacles and reach a set of specific subgoals. The particularity of this approach lies in the use of symbolic representations of the skill, which are applied to explore the role of the teacher in guiding incremental learning of the robot.

Finally, [74.91] took a symbolic approach to encoding human motions as sets of pre-defined postures, positions, or configurations, considering different levels of granularity for the symbolic representation of the motion. This a priori knowledge is then used to explore the correspondence problem through several simulated setups, including motion in joint space of arm links and displacements of objects on a two-dimensional (2-D) plane.

The main advantage of these symbolic approaches is that high-level skills (consisting of sequences of symbolic cues) can be learned efficiently through an interactive process. However, because of the symbolic nature of their encoding, the methods rely on a large amount of prior knowledge to predefine the important cues and to segment those efficiently.

74.4.3 Incremental Teaching Methods

The statistical approach described previously is an interesting way to extract autonomously the important features of the task, and, thus to avoid putting too much prior knowledge in the system. However, it requires a large number of demonstrations to draw statistically valid inference. It is not reasonable to assume that a layuser will perform many demonstrations of the same task. Hence, for LfD to be amenable to lay users, learning should require as few demonstrations as possible. Ideally, one would like the robot to be bootstrapped with some initial knowledge, so that the robot can start

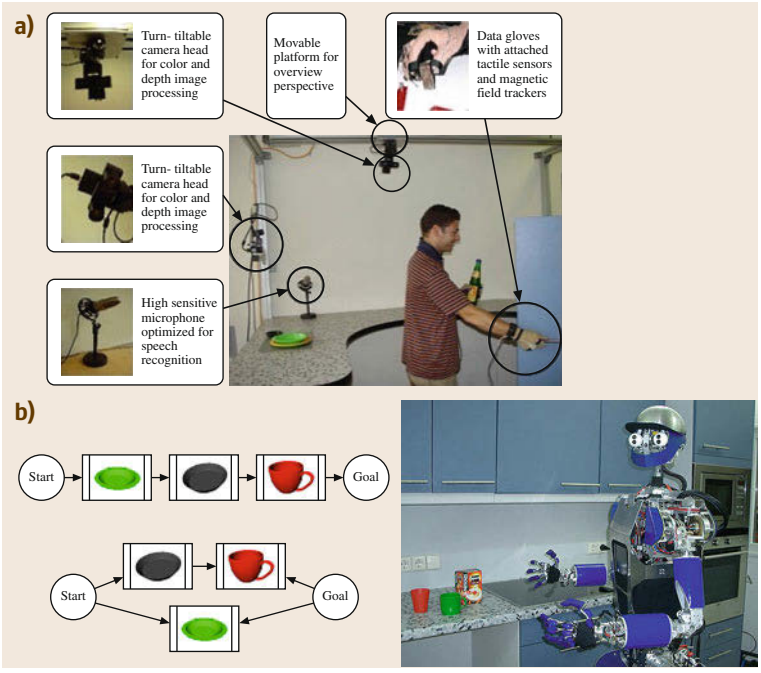


Fig. 74.6 **(a)** Training center with dedicated sensors. **(b)** Precedence graphs learned by the system for the *setting the table* task. **(c)** Initial task precedence graph for the first three demonstrations. **(d)** Final task precedence graph after observing additional examples (after [74.88]) (VIDEO 103)

right away to perform the task, and human training would be used solely to help the robot gradually improve its performance.

Incremental learning approaches that gradually refine task knowledge as more examples become available pave the way towards LfD systems suitable for such continuous and long-life robot learning. Figure 74.7 and VIDEO 104 shows an example of such incremental teaching of a simple skill.

These incremental learning methods use various forms of deixis, as well as verbal and non-verbal interactions, to guide the robot's attention to the important parts of the demonstration or to particular mistakes produced by the robot during the reproduction of the task. Such incremental and guided learning is often referred to as *scaffolding* or *molding* of the robot's knowledge, and is key to teaching robots tasks of increasing complexity [74.90, 94].

Research on the use of incremental learning techniques for robot LfD has contributed to the development of methods for learning complex tasks within the household domain from as few demonstrations as possible. Moreover, it has contributed to the development and application of machine learning that allow a continuous and incremental refinement of the task model. Such systems have sometimes been referred to as *background knowledge-based* or *EM deductive LfD-systems*, as presented in [74.95, 96]. They usually require very few or even only a single user demonstration to generate executable task descriptions. The main objective of this line of research is to build a meta-representation of the knowledge that the robot has acquired on the task and to apply reasoning methods on this knowledge database (Fig. 74.6). In this scenario, reasoning involves recognizing, learning, and representing repetitive tasks.

Pardowitz et al. [74.97] discuss how different forms of knowledge can be balanced in an incremental learning system. The system relies on building *task precedence graphs*. Task precedence graphs encode hypotheses that the system makes on the sequential structure of a task. Learning of the task precedence graphs allows the system to schedule its operations most flexibly, while still meeting the goals of the task ([74.98] for details). Task precedence graphs are directed, acyclic graphs that contain a temporal precedence relation that can be learned incrementally. Incremental learning of task precedence graphs leads to a more general and flexible representation of the task knowledge (Fig. 74.6 and VIDEO 105).

74.4.4 Combining Learning from Humans with Other Learning Techniques

To recall, a main argument for the development of LfD methods was that they would speed up learning by providing examples of *good solutions*. This assumption, however, is realistic only if the context for the reproduction is sufficiently similar to that of the demonstration. We saw previously that the use of dynamical systems-based representation at the trajectory level allows the robot to depart to some extent from a learned trajectory to reach the target, even when both the object and the hand of the robot have moved from the location shown during the demonstration. There are, however, situations in which such an approach would fail, such as, for instance, when placing a large obstacle in the robot's pathway (Fig. 74.8). Besides, robots and humans may differ significantly in their kinematics and dynamics of motion and, although there are varieties of ways to bypass the so-called correspondence problem, relearning a new model may still be required in special cases.

To allow the robot to relearn to perform a task in any new situation, it appeared important to combine LfD methods with other motor learning techniques. Reinforcement learning (RL) appeared particularly suitable for type of problem. Indeed, imitation learning is limiting in that it requires the robot to learn only from what has been demonstrated. Reinforcement learning, in contrast, allows the robot to discover new control policies through free exploration of the state-action space. Approaches that combine imitation learning and reinforcement learning aim at exploiting the strength of both algorithms to overcome their respective drawbacks. Demonstrations are used to guide the exploration

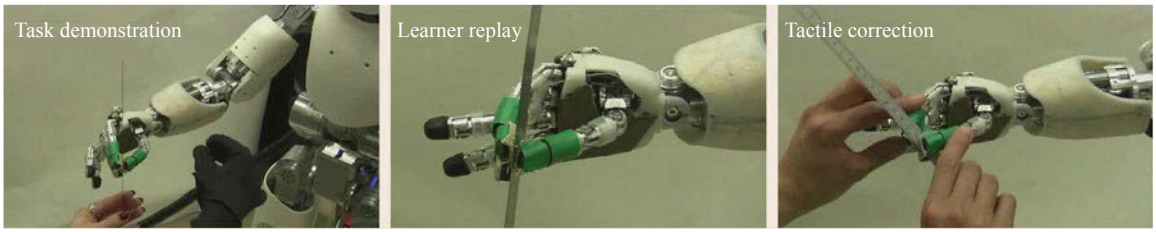


Fig. 74.7 An incremental learning strategy where a manipulation skill is first demonstrated through the use of a data glove. After a first reproduction trial, the skill is refined through kinesthetic teaching, by exploiting the tactile capabilities of the iCub humanoid robot (after [74.50]) (VIDEO 104)

in reinforcement learning (RL). This, hence, reduces the time it takes for RL algorithms to find an adequate control policy, while allowing the robot to depart from the demonstrated behavior. Figures 74.8 show two examples of techniques that use reinforcement learning in conjunction with LfD to improve the robot's performance beyond that of a demonstrator.

Early work on LfD using RL started in the 1990s with learning to swing up and control an inverse pendulum [74.100] and learning industrial tasks like peg-in-hole with a robot arm [74.26]. More recent efforts include [74.101–103], who tackled robust control of the upper body of humanoid robots in various manipulation tasks, learning an archery skill [74.104], and learning how to hit a snooker ball [74.105].

Demonstrations can be used in different ways to bootstrap RL. They may be used as initial roll-outs from which an initial estimate of the policy is computed [74.106–108], or to generate an initial set of primitives [74.81, 103, 107]. In the latter case, RL is then used to learn how to select across these primitives. Demonstrations can also be used to limit the search space covered by RL [74.101, 109], or to estimate the reward function [74.110, 111]. Finally, RL and imitation learning can be used in conjunction at run time, by letting the demonstrator take over part of the control during one trial [74.112].

Another way to enable the robot to learn a control strategy through a combination of self-experimentation and learning from watching others is to evolve population of agents that mimic each other. Such an evolutionary approach using genetic algorithms has been investigated by a number of authors, e.g., for learning of manipulation skills [74.113], navigation strategies [74.114], or sharing a common vocabulary to name sensoriperception and actions [74.115].

Variants on Reinforcement Learning

While most of the works that combine imitation learning with reinforcement learning assume the reward to be known, inverse reinforcement learning (IRL) offers a framework to determine automatically the reward and the optimal control policy [74.116]. When using human demonstrations to guide learning, IRL solves jointly the *what to imitate* and *how to imitate* problems. Other approaches to estimating the reward or cost function automatically have been proposed, see, for instance, the maximum margin planning technique [74.117] and the automatic extraction of constraints [74.118].

Underlying all IRL works is the assumption of a consistent reward function. When demonstrations are provided by multiple experts, this assumes that all experts optimize the same objectives. This is constraining and does not exploit the variability of ways in which humans may solve the same task. Recent IRL works consider multiple experts and identify multiple different reward functions [74.119, 120]. This allows the robots to learn multiple (albeit suboptimal) ways to perform the same task. The hope is that this multiplicity of policies will make the controller more robust, offering alternative ways to complete the task, when the context no longer allows the robot to perform the task in the optimal way.

The vast majority of work on LfD relies on successful demonstrations of the desired task by the human. It hence assumes that all the demonstrations are good demonstrations and discards those that are poor proxy of what would be deemed as a good demonstration. Recent work has also investigated the possibility that demonstrations may instead be failed attempts at performing the task [74.121, 122]. Learning then proceeds from observing solely incorrect demonstrations (VIDEO 476 and VIDEO 477). Note that demonstrations are never completely incorrect. Learning from failed demonstration then attempts to discover which parts of the demonstrations were correct and which were incorrect, so as to improve solely the incorrect parts. In this context, LfD addresses the questions of *what to* and *what not to* imitate. It offers an interesting alternative to approaches that combine imitation learning and reinforcement learning, in that no reward needs to be explicitly determined.

74.4.5 Learning from Humans, a Form of Human–Robot Interaction

Another perspective adopted by LfD to make the transfer of skill more efficient is to focus on the interaction

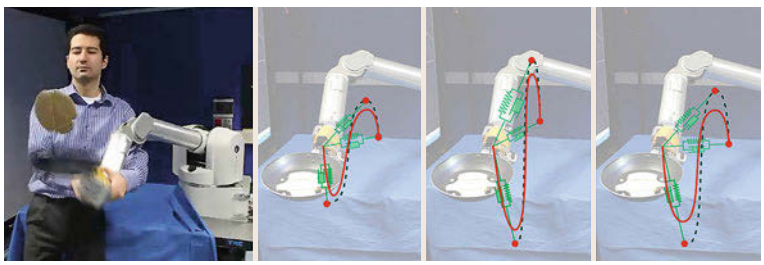


Fig. 74.8 Illustration of the use of reinforcement learning in policy parameter space to refine a skill initially learned from demonstration (after [74.99] and VIDEO 105)

aspect of the transfer process. As this transfer problem is complex and involves a combination of social mechanisms, several insights from human–robot interaction (HRI) were explored to make efficient use of the teaching capabilities of the human user, [74.123–125] for surveys. Next, we briefly survey some of these works.

The development of algorithms for detecting *social cues* given implicitly or explicitly by the teacher during training and the integration of those as part of other generic mechanisms for LfD has become the focus of a large body of work in LfD. Such social cues can be viewed as a way to introduce priors in a statistical learning system, and, by so doing, to speed up learning. Indeed, several hints can be used to transfer a skill not only by demonstrating the task multiple times but also by highlighting the important components of the skill. This can be achieved by various means, using different modalities.

A large body of work explored the use of pointing and gazing (Fig. 74.9 left and VIDEO 106) as a way of conveying the intention of the user [74.79, 126–132]. Vocal deixis, using a standard speech recognition engine, has also been explored widely [74.79, 133]. In [74.88], the user makes vocal comments to highlight the steps of the teaching that are deemed as being the most important. In [74.134, 135], only the prosody of the speech pattern is looked at, rather than the exact content of the speech, as a way to infer some information on the user's communicative intent.

In [74.136], these social cues are learned through an imitative game, whereby the user imitates the robot. This allows the robot to build a user-specific model of these social pointers, and, hence be more robust to detecting those.

Recent lines of research in interactive LfD seeks to give a more active role to the teacher in a bidirectional teaching process [74.15, 137, 138]. Robots become more active partners and can indicate which portion of the demonstration was unclear. Teachers may in turn refine the robot's knowledge by providing complementary information where the robot is performing poorly. This supplementary information may consist of additional rounds of demonstrations of the complete task [74.139], or may be limited to subparts of the task [74.140, 141]. The information can be conveyed through specific task's features, such as a list of waypoints [74.142]. The robot is then left free to interpolate a trajectory using these key points.

The design of such incremental teaching methods calls for machine learning techniques that enable the incorporation of new data in a robust manner. It also opens the door to the design of other human–robot interfacing systems, including the use of speech, which leads to meaningful dialogs between humans and robots. An example of such bidirectional teaching is given on the right-hand side of Fig. 74.10. The robot asks for help during or after teaching, verifying that its understanding of the task is correct [74.14]. This teaching interaction is tailored to let the user become an active participant in the learning process (and not only a model of expert behavior).

By taking inspiration from the human tutelage paradigm, [74.15] shows that a socially guided approach can improve both the human–robot interaction and the machine learning process by taking into account *human benevolence*. That work highlights the role of the teacher in organizing the skill into manageable steps and maintaining an accurate mental model

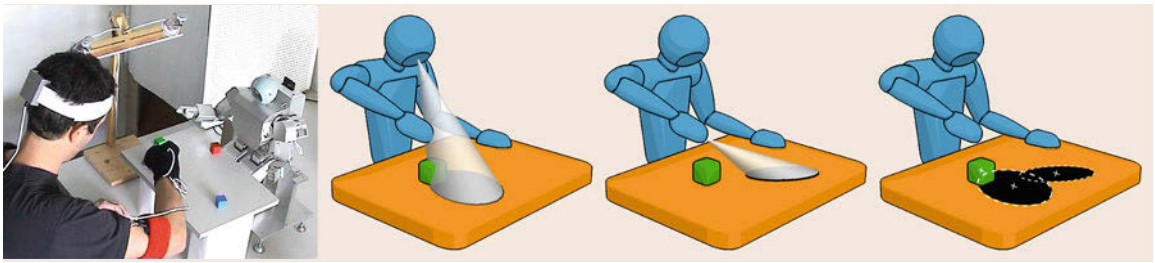


Fig. 74.9 Illustration of the use of social cues to speed up the imitation learning process. Here, gazing and pointing information are used to select probabilistically the objects relevant for the manipulation skill (VIDEO 106)

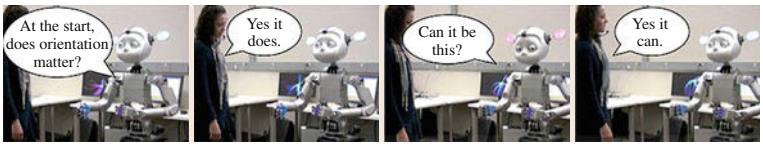


Fig. 74.10 Example of an active teaching scenario. The robot asks for help during or after teaching, verifying that its understanding of the task is correct (after [74.14]) (VIDEO 107)

of the learner's understanding. Reference [74.138] use a similar teaching paradigm and extend the concept to the learning of continuous motion trajectories and of actions on objects, and propose experiments where a humanoid robot learns new manipulation skills by first observing a human demonstrator (through motion sensors) and then gradually refining its skill through teacher support. In this application, the user provides scaffolds to the robot for the reproduction of the skill by moving kinesthetically a subset of the motors. Through the supervision of the user who progressively dismantles the scaffolds after each reproduction attempt, the robot can finally reproduce the skill on its own. Reference [74.143] highlights the importance of an active participation of the teacher not only to demonstrate a model of expert behavior but also to refine the acquired motion through spoken feedback.

Reference [74.90] provides experiments where a wheeled robot is teleoperated through a screen interface to simulate a *molding* process, that is, by letting the robot experience sensory information when exploring its environment through the teacher's support. Their model uses a memory-based approach in which the user provides labels for the different components of the task to teach hierarchically high-level behaviors.

Finally, a core idea of the HRI approach to LfD is that imitation is goal directed, that is, actions are meant to fulfill a specific purpose and to convey the intention of the actor [74.144]. While a longstanding trend in LfD approached the problem from the standpoint of trajectory following [74.84, 145, 146] and joint motion replication, [74.147–150], recent works, inspired by the above rationale, start from the assumption that imitation is not just about observing and replicating the motion, but rather about *understanding* the goals of a given action (see the above survey of approaches to determining automatically the reward or what to imitate).

Determining the way humans learn to both extract the goals of a set of observed actions and give these goals a hierarchy of preference is fundamental to our understanding of the underlying decisional process to imitation. While we have surveyed recent work in that area, it is important to recall other approaches to tackling these issues that have previously followed a probabilistic approach to explain the derivation and sequential application of goals and apply this to enable learning of manipulatory tasks requiring sequencing of subsets of goals [74.97, 145, 151, 152].

Understanding the goal of the task is still only half of the picture, as there may be several ways of achieving the goal of the task. Moreover, what is feasible (or optimal) for the demonstrator may not necessarily be appropriate for the imitator [74.36]. Thus, different models, modes and communication channels, should be used in conjunction to find a solution that is optimal both from the point of view of the imitator and that achieves what the demonstrator seeks to teach the robot.

This concludes our survey. As the reader can see, the issues of what and how to imitate are tightly connected and to a large extent remain only partly solved.

74.5 Conclusions and Open Issues in Robot LfD

Research in LfD or programming by demonstration (PbD) is progressing rapidly, pushing back limits and posing new questions all the time. As such, any list of limitations and open questions is bound to be incomplete and out of date. However, there are a few long-standing limitations and open questions that bear further attention.

Generally, work in LfD assumes a fixed, given form for the robot's control policy, and learns appropriate parameters. To date, there are several different forms of policies in common usage, and there is no clear correct (or dominant) technique. Furthermore, it is possible that a system could be provided with multiple possible representations of controllers and select which is most appropriate.

The combination of reinforcement learning and imitation learning has been shown to be effective in addressing the acquisition of skills that require fine tuning of the robot's dynamics. Likewise, more interactive learning techniques have proven successful in allowing for collaborative improvement of the learnt policy by switching between human-guided and robot-initiated learning. However, there do not yet exist protocols to determine when it is best to switch between the various learning modes available. The answer may, in fact, be task dependent.

In work to date, teaching is usually done by a single teacher, or teachers with an explicit concept of the task to teach. More work needs to be done to address issues related to conflicting demonstrations across teachers with different styles. Similarly, teachers are usually human beings, but could instead be an arbitrary expert agent. This agent could be a more knowledgeable robot or a computer simulation. Finally, another relatively little explored question relates to the problem of how to transfer skills across multiple agents, including multiple robots (i. e., teaching is done from a teacher robot to various learner robots). Early work in this direction was

done in the 1990s [74.115, 153, 154]. This work, however, has so far been reduced to transfer of navigation or communication skills across swarms of simple mobile robots.

Experiments in LfD have mostly focused on a single task (or set of closely related tasks), and each experiment starts with a tabula rasa. As learning of complex tasks progresses, means to store and reuse prior knowledge at a large scale will have to be devised. Learning stages, akin perhaps to those found in child development, may be required. There will need to be a formalism to allow the robot to select information, to reduce redundant information, select features, and store new data efficiently.

Video-References

VIDEO 29 Demonstrations and reproduction of the task of juicing an orange
available from http://handbookofrobotics.org/view-chapter/74/videodetails/29

VIDEO 97 Demonstrations and reproduction of moving a chessman
available from http://handbookofrobotics.org/view-chapter/74/videodetails/97

VIDEO 98 Full-body motion transfer under kinematic/dynamic disparity
available from http://handbookofrobotics.org/view-chapter/74/videodetails/98

VIDEO 99 Demonstration by visual tracking of gestures
available from http://handbookofrobotics.org/view-chapter/74/videodetails/99

VIDEO 100 Demonstration by kinesthetic teaching
available from http://handbookofrobotics.org/view-chapter/74/videodetails/100

VIDEO 101 Demonstration by teleoperation of humanoid HRP-2
available from http://handbookofrobotics.org/view-chapter/74/videodetails/101

VIDEO 102 Probabilistic encoding of motion in a subspace of reduced dimensionality
available from http://handbookofrobotics.org/view-chapter/74/videodetails/102

VIDEO 103 Reproduction of dishwasher unloading task based on task precedence graph
available from http://handbookofrobotics.org/view-chapter/74/videodetails/103

VIDEO 104 Incremental learning of finger manipulation with tactile capability
available from http://handbookofrobotics.org/view-chapter/74/videodetails/104

VIDEO 105 Policy refinement after demonstration
available from http://handbookofrobotics.org/view-chapter/74/videodetails/105

VIDEO 106 Exploitation of social cues to speed up learning
available from http://handbookofrobotics.org/view-chapter/74/videodetails/106

VIDEO 107 Active teaching
available from http://handbookofrobotics.org/view-chapter/74/videodetails/107

VIDEO 476 Learning from failure I
available from http://handbookofrobotics.org/view-chapter/74/videodetails/476

VIDEO 477 Learning from failure II
available from http://handbookofrobotics.org/view-chapter/74/videodetails/477

VIDEO 478 Learning compliant motion from human demonstration
available from http://handbookofrobotics.org/view-chapter/74/videodetails/478

VIDEO 479 Learning compliant motion from human demonstration II
available from http://handbookofrobotics.org/view-chapter/74/videodetails/479

References

74.1 T. Lozano-Perez: Robot programming, Proceedings IEEE **71**(7), 821–841 (1983)

74.2 B. Dufay, J.-C. Latombe: An approach to automatic robot programming based on inductive learning, Int. J. Robotics Res. **3**(4), 3–20 (1984)

74.3 A. Levas, M. Selfridge: A user-friendly high-level robot teaching system, IEEE Int. Conf. Robotics, Altanta (1984) pp. 413–416

74.4 A.B. Segre, G. DeJong: Explanation-based manipulator learning: Acquisition of planning ability through observation, IEEE Conf. Robotics Autom. St. Louis (1985) pp. 555–560

74.5 A.M. Segre: *Machine Learning of Robot Assembly Plans* (Kluwer, Boston 1988)

74.6 S. Muench, J. Kreuziger, M. Kaiser, R. Dillmann: Robot programming by demonstration (RPD) - Using machine learning and user interaction methods for the development of easy and comfortable robot programming systems, Proc. Int. Symp. Indus. Robots (ISIR) (1994) pp. 685–693

74.7 A. Billard: Imitation: A review. In: *The Handbook of Brain Theory and Neural Network*, 2nd edn., ed. by M.A. Arbib (MIT Press, Cambridge 2002) pp. 566–569

74.8 E. Oztop, M. Kawato, M.A. Arbib: Mirror neurons and imitation: A computationally guided review, Neural Netw. **19**(3), 254–271 (2006)

74.9 J. Demiris, G. Hayes: Imitation as a dual-route process featuring predictive and learning components: A biologically-plausible computational model. In: *Imitation in Animals and Artifacs*, ed. by C. Nehaniv, K. Dautenhahn (MIT Press, Cambridge 2002)

74.10 J. Nadel, A. Revel, P. Andry, P. Gaussier: Toward communication: First imitations in infants, low-functioning children with autism and robots, Interact. Stud. **5**(1), 45–74 (2004)

74.11 F. Kaplan, P.-Y. Oudeyer: The progress-drive hypothesis: An interpretation of early imitation. In: *Models and Mechanisms of Imitation and Social Learning: Behavioural, Social and Communication Dimensions*, ed. by K. Dautenhahn, C. Nehaniv (Cambridge Univ. Press, Cambridge 2007) pp. 361–377

74.12 B.D. Argall, M. Veloso, B. Browning: Teacher feedback to scaffold and refine demonstrated motion primitives on a mobile robot, Robotics Auton. Syst. **59**(3/4), 243–255 (2011)

74.13 B. Robins, K. Dautenhahn, C.L. Nehaniv, N.A. Mirza, D. Francois, L. Olsson: Sustaining interaction dynamics and engagement in dyadic child-robot interaction kinesics: Lessons learnt from an exploratory study, IEEE Int. Workshop Robot Human Int. Commun. (ROMAN) (2005) pp. 716–722

74.14 M. Cakmak, A.L. Thomaz: Designing robot learners that ask good questions, IEEE-ACML Int. Conf. Human-Robot Int. (HRI) (2012)

74.15 C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd, D. Chilongo: Tutelage and collaboration for humanoid robots, Human. Robots **1**(2), 315–348 (2004)

74.16 Y. Kuniyoshi, M. Inaba, H. Inoue: Teaching by showing: Generating robot programs by visual observation of human performance, Proc. Int. Symp. Ind. Robots, Tokyo (1989) pp. 119–126

74.17 Y. Kuniyoshi, M. Inaba, H. Inoue: Learning by watching: Extracting reusable task knowledge from visual observation of human performance, IEEE Trans. Robotics Autom. **10**(6), 799–822 (1994)

74.18 M. Ehrenmann, O. Rogalla, R. Zöllner, R. Dillmann: Teaching service robots complex tasks: Programming by demonstation for workshop and household environments, Proc. IEEE Int. Conf. Field Serv. Robotics (FRS) (2001)

74.19 C.P. Tung, A.C. Kak: Automatic learning of assembly task using a dataglove system, IEEE/RSJ Int. Conf. Intell. Robots Syst., Pittsburgh (1995) pp. 1–8

74.20 K. Ikeuchi, T. Suchiro: Towards an assembly plan from observation, Part I: Assembly task recognition using face-contact relations (polyhedral objects), Proc. IEEE Int. Conf. Robot. Autom. (ICRA), Vol. 3 (1992) pp. 2171–2177

74.21 S. Calinon, F. Guenter, A. Billard: On learning, representing and generalizing a task in a humanoid robot, IEEE Trans. Syst. Man Cybern. B **37**(2), 286–298 (2007)

74.22 M. Ito, K. Noda, Y. Hoshino, J. Tani: Dynamic and interactive generation of object handling behaviors by a small humanoid robot using a dynamic neural network model, Neural Netw. **19**(3), 323–337 (2006)

74.23 T. Inamura, N. Kojo, M. Inaba: Situation recognition and behavior induction based on geometric symbol representation of multimodal sensorimotor patterns, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2006) pp. 5147–5152

74.24 S. Liu, H. Asada: Teaching and learning of deburring robots using neural networks, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 339–345

74.25 A. Billard: Learning motor skills by imitation: A biologically inspired robotic model, J. Cybern. Syst. **32**(1/2), 155–193 (2001)

74.26 M. Kaiser, R. Dillmann: Building elementary robot skills from human demonstration, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 3 (1996) pp. 2700–2705

74.27 R. Dillmann, M. Kaiser, A. Ude: Acquisition of elementary robot skills from human demonstration, Int. Symp. Intell. Robotics Syst. (SIRS) (1995) pp. 1–38

74.28 W. Yang: Hidden Markov model approach to skill learning and its application in telerobotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1993) pp. 396–402

74.29 P.K. Pook, D.H. Ballard: Recognizing teleoperated manipulations, Proc. IEEE Int. Conf. Robotics Autom., Atlanta (1993) pp. 578–585

74.30 G.E. Hovland, P. Sikka, B.J. McCarragher: Skill acquisition from human demonstration using a hidden Markov model, Proc. IEEE Int. Conf. Robotics Autom., Minneapolis (1996) pp. 2706–2711

74.31 S.K. Tso, K.P. Liu: Hidden Markov model for intelligent extraction of robot trajectory command from demonstrated trajectories, Proc. IEEE Int. Conf. Ind. Technol. (ICIT) (1996) pp. 294–298

74.32 C. Lee, Y. Xu: Online, interactive learning of gestures for human/robot interfaces, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Vol. 4 (1996) pp. 2982–2987

74.33 D. Kulic, W. Takano, Y. Nakamura: Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden Markov chains, Int. J. Robotics Res. **27**(7), 761–784 (2008)

74.34 D. Nguyen-Tuong, M. Seeger, J. Peters: Local Gaussian process regression for real time online model learning and control, Adv. Neural Inf. Process. Syst. **21**, 1193–1200 (2009)

74.35 S.M. Khansari Zadeh, A. Billard: Learning stable non-linear dynamical systems with Gaussian mixture models, IEEE Trans. Robotics **27**(5), 943–957 (2011)

74.36 C. Nehaniv, K. Dautenhahn: Of hummingbirds and helicopters: An algebraic framework for interdisciplinary studies of imitation and its applications.

In: *Interdisciplinary Approaches to Robot Learning*, Vol. 24, ed. by J. Demiris, A. Birk (World Scientific, Singapore 2000) pp. 136–161

74.37 C.L. Nehaniv: Nine billion correspondence problems and some methods for solving them, Proc. Int. Symp. Imit. Anim. Artifacts (2003) pp. 93–95

74.38 P. Bakker, Y. Kuniyoshi: Robot see, robot do: An overview of robot imitation, AISB Workshop Learn. Robot. Anim., Brighton (1996)

74.39 M. Skubic, R.A. Volz: Acquiring robust, force-based assembly skills from human demonstration, IEEE Trans. Robotics Autom. **16**(6), 772–781 (2000)

74.40 M. Yeasin, S. Chaudhuri: Toward automatic robot programming: Learning human skill from visual data, IEEE Trans. Syst. Man Cybern. B **30**(1), 180–185 (2000)

74.41 J. Zhang, B. Rössler: Self-valuing learning and generalization with application in visually guided grasping of complex objects, Robotics Auton. Syst. **47**(2/3), 117–127 (2004)

74.42 M. Frank, M. Plaue, H. Rapp, U. Koethe, B. Jaehne, F.A. Hamprecht: Theoretical and experimental error analysis of continuous-wave time-of-flight range cameras, Opt. Eng. **48**(1), 013602 (2009)

74.43 M. Freese, S. Singh, F. Ozaki, N. Matsuhira: Virtual robot experimentation platform v-rep: A versatile 3d robot simulator, Proc. Int. Conf. Simul. Model. Progr. Auton. Robots (SIMPAR) (2010) pp. 51–62

74.44 S. Hak, N. Mansard, O. Ramos, L. Saab, O. Stasse: Capture, recognition and imitation of anthropomorphic motion, IEEE-RAS Int. Conf. Robotics Autom. (2012) pp. 3539–3540

74.45 G. Gioioso, G. Salvietti, M. Malvezzi, D. Prattichizzo: An object-based approach to map human hand synergies onto robotic hands with dissimilar kinematics. In: *Robotics - Science and Systems VIII*, ed. by N. Roy, P. Newman, S. Srinivasa (MIT Press, Cambridge 2012) pp. 97–105

74.46 A. Shon, K. Grochow, A. Hertzmann, R. Rao: Learning shared latent structure for image synthesis and robotic imitation, Adv. Neural Inf. Process. Syst. (NIPS) **18**, 1233–1240 (2006)

74.47 A. Ude, C.G. Atkeson, M. Riley: Programming full-body movements for humanoid robots by observation, Robotics Auton. Syst. **47**, 93–108 (2004)

74.48 S. Kim, C. Kim, B. You, S. Oh: Stable whole-body motion generation for humanoid robots to imitate human motions, Proc. IEEE/RSJ Int. Conf. Intell. Robotics Syst. (IROS) (2009)

74.49 S. Nakaoka, A. Nakazawa, F. Kanehiro, K. Kaneko, M. Morisawa, H. Hirukawa, K. Ikeuchi: Learning from observation paradigm: Leg task models for enabling a biped humanoid robot to imitate human dances, Int. J. Robotics Res. **26**(8), 829–844 (2007)

74.50 E.L. Sauser, B.D. Argall, G. Metta, A.G. Billard: Iterative learning of grasp adaptation through human corrections, Robotics Auton. Syst. **60**(1), 55–71 (2012)

74.51 A. Coates, P. Abbeel, A.Y. Ng: Learning for control from multiple demonstrations, Proc. 25th Int. Conf. Mach. Learn. (2008)

74.52 D. Grollman, O.C. Jenkins: Incremental learning of subtasks from unsegmented demonstration, Int. Conf. Intell. Robots Syst. (2010)

74.53 L. Peternel, J. Babic: Humanoid robot posture-control learning in real-time based on human sensorimotor learning ability, IEEE Int. Conf. Robotics Autom. (ICRA) Karlsruhe (2013)

74.54 A. Ude: Robust estimation of human body kinematics from video, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (1999) pp. 1489–1494

74.55 B. Akgun, M. Cakmak, K. Jiang, A.L. Thomaz: Keyframe-based learning from demonstration, Int. J. Soc. Robotics **4**, 343–355 (2012)

74.56 P. Evrard, E. Gribovskaya, S. Calinon, A. Billard, A. Kheddar: Teaching physical collaborative tasks: Object-lifting case study with a humanoid, Proc. IEEE-RAS Int. Conf. Humanoid Robots (Humanoids), Paris (2009) pp. 399–404

74.57 C. Chao, M. Cakmak, A.L. Thomaz: Designing interactions for robot active learners, IEEE Trans. Auton. Mental Dev. **2**(2), 108–118 (2010)

74.58 S. Calinon, A. Billard: PDA interface for humanoid robots, Proc. IEEE Int. Conf. Humanoid Robots (Humanoids) (2003)

74.59 A. Shon, K. Grochow, R. Rao: Robotic imitation from human motion capture using Gaussian processes, Proc. IEEE/RAS Int. Conf. Humanoid Robots (Humanoids) (2005)

74.60 Y. Wu, Y. Demiris: Towards one shot learning by imitation for humanoid robots, IEEE-RAS Int. Conf. Robotics Autom. (ICRA) (2010)

74.61 J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, M. Kawato: Learning from demonstration and adaptation of biped locomotion, Robotics Auton. Syst. **47**(2/3), 79–91 (2004)

74.62 D. Lee, C. Ott: Incremental kinesthetic teaching of motion primitives using the motion refinement tube, Auton. Robot. **31**(2), 115–131 (2011)

74.63 A. Ude: Trajectory generation from noisy positions of object features for teaching robot paths, Robotics Auton. Syst. **11**(2), 113–127 (1993)

74.64 J. Yang, Y. Xu, C.S. Chen: Human action learning via hidden Markov model, IEEE Trans. Syst. Man Cybern. A **27**(1), 34–44 (1997)

74.65 K. Yamane, Y. Nakamura: Dynamics filter – concept and implementation of online motion generator for human figures, IEEE Trans. Robotics Autom. **19**(3), 421–432 (2003)

74.66 S. Vijayakumar, S. Schaal: Locally weighted projection regression: An O(n) algorithm for incremental real time learning in high dimensional spaces, Proc. Int. Conf. Mach. Learn. (ICML) (2000) pp. 288–293

74.67 S. Vijayakumar, A. D'souza, S. Schaal: Incremental online learning in high dimensions, Neural Comput. **17**(12), 2602–2634 (2005)

74.68 N. Kambhatla: Local Models and Gaussian Mixture Models for Statistical Data Processing, PhD Thesis

(Oregon Graduate Institute of Science and Technology, Portland 1996)
74.69 K. Grochow, S.L. Martin, A. Hertzmann, Z. Popovic: Style-based inverse kinematics, Proc. ACM Int. Conf. Comput. Gr. Interact. Tech. (SIGGRAPH) (2004) pp. 522–531
74.70 K.F. MacDorman, R. Chalodhorn, M. Asada: Periodic nonlinear principal component neural networks for humanoid motion segmentation, generalization, and generation, Proc. Int. Conf. Pattern Recogn. (ICPR) (2004) pp. 537–540
74.71 M. Mühlig, M. Gienger, J.J. Steil, C. Goerick: Automatic selection of task spaces for imitation learning, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (2009) pp. 4996–5002
74.72 R. Jäkel, P. Meißner, S. Schmidt-Rohr, R. Dillmann: Distributed generalization of learned planning models in robot programming by demonstration, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2011)
74.73 A. Gams, M. Do, A. Ude, T. Asfour, R. Dillmann: Online periodic movement and force-profile learning for adaptation to new surfaces, Proc. IEEE-RAS Int. Conf. Human. Robot. (2010) pp. 560–565
74.74 P. Kormushev, S. Calinon, D. Caldwell: Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input, Adv. Robotics **25**(5), 581–603 (2011)
74.75 L. Rozo, S. Calinon, D.G. Caldwell, P. Jimenez, C. Torras: Learning collaborative impedance-based robot behaviors, Proc. AAAI Conf. Artif. Intell., Bellevue (2013) pp. 1422–1428
74.76 L. Peternel, T. Petric, E. Oztop, J. Babic: Teaching robots to cooperate with humans in dynamic manipulation tasks based on multi-modal human-in-the-loop approach, Auton. Robots **36**(1/2), 123–136 (2014)
74.77 C. Daniel, G. Neumann, J. Peters: Learning concurrent motor skills in versatile solution spaces, Proc. IEEE Int. Conf. Robotics Intell. Syst. (IROS'2012) (2012) pp. 3591–3597
74.78 O. Mangin, P.-Y. Oudeyer: Unsupervised learning of simultaneous motor primitives through imitation, IEEE Int. Conf. Dev. Learn. (2011)
74.79 R. Dillmann: Teaching and learning of robot tasks via observation of human performance, Robotics Auton. Syst. **47**(2/3), 109–116 (2004)
74.80 A. Skoglund, B. Iliev, B. Kadmiry, R. Palm: Programming by demonstration of pick-and-place tasks for industrial manipulators using task primitives, Int. Symp. Comput. Intell. Robotics Autom. (2007)
74.81 K. Muelling, J. Kober, O. Kroemer, J. Peters: Learning to select and generalize striking movements in robot table tennis, Int. J. Robotics Res. **32**(3), 280–298 (2013)
74.82 D. Kulic, C. Ott, C. Lee, J. Ishikawa, Y. Nakamura: Incremental learning of full body motion primitives and their sequencing through human motion observation, Int. J. Robotics Res. **31**(3), 330–345 (2012)
74.83 S. Niekum, G. Osentoski, A.G. Konidaris, A. Barto: Learning and generalization of complex tasks from unstructured demonstrations, IEEE Int. Conf. Intell. Robotics Syst. (2012) pp. 5239–5246
74.84 P. Gaussier, S. Moga, J.P. Banquet, M. Quoy: From perception-action loop to imitation processes: A bottom-up approach of learning by imitation, Appl. Artif. Intell. **7**(1), 701–729 (1998)
74.85 M.N. Nicolescu, M.J. Mataric: Natural methods for robot task learning: Instructive demonstrations, generalization and practice, Proc. Int. Jt. Conf. Auton. Agents Multiagent Syst. (AAMAS) (2003) pp. 241–248
74.86 J. Tani, M. Ito: Self-organization of behavioral primitives as multiple attractor dynamics: A robot experiment, IEEE Trans. Syst. Man Cybern. A **33**(4), 481–488 (2003)
74.87 H. Friedrich, S. Muench, R. Dillmann, S. Bocionek, M. Sassin: Robot programming by demonstration (RPD): Supporting the induction by human interaction, Mach. Learn. **23**(2), 163–189 (1996)
74.88 M. Pardowitz, R. Zoellner, S. Knoop, R. Dillmann: Incremental learning of tasks from user demonstrations, past experiences and vocal comments, IEEE Trans. Syst. Man Cybern. B **37**(2), 322–332 (2007)
74.89 S. Ekvall, D. Kragic: Learning task models from multiple human demonstrations, Proc. IEEE Int. Symp. Robot Human Int. Commun. (RO-MAN) (2006) pp. 358–363
74.90 J. Saunders, C.L. Nehaniv, K. Dautenhahn: Teaching robots by moulding behavior and scaffolding the environment, Proc. ACM SIGCHI/SIGART Conf. Human-Robot Interaction (HRI) (2006) pp. 118–125
74.91 A. Alissandrakis, C.L. Nehaniv, K. Dautenhahn: Correspondence mapping induced state and action metrics for robotic imitation, IEEE Trans. Syst. Man Cybern. B **37**(2), 299–307 (2007)
74.92 J. Rainer, R. Sven, S. Schmidt-Rohr, W. Rühl, K. Alexander, X. Zhixing, R. Dillmann: Learning of planning models for dexterous manipulation based on human demonstrations, Int. J. Soc. Robotics **4**(4), 437–448 (2012)
74.93 S.R. Schmidt-Rohr, M. Lösch, R. Jäkel, R. Dillmann: Programming by demonstration of probabilistic decision making on a multi-modal service robot, Proc. 2010 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2010)
74.94 C. Breazeal, M. Berlin, A. Brooks, J. Gray, A.L. Thomaz: Using perspective taking to learn from ambiguous demonstrations, Robotics Auton. Syst. **54**, 385–393 (2006)
74.95 Y. Sato, K. Bernardin, H. Kimura, K. Ikeuchi: Task analysis based on observing hands and objects by vision, IEEE/RSJ Int. Conf. Intell. Robots Syst. Lausanne (2002) pp. 1208–1213
74.96 R. Zoellner, M. Pardowitz, S. Knoop, R. Dillmann: Towards cognitive robots: Building hierarchical task representations of manipulations from human demonstration, Int. Conf. Robotics Autom. (ICRA) Barcelona (2005)

74.97 M. Pardowitz, R. Zöllner, R. Dillmann: Incremental learning of task sequences with information-theoretic metrics, Proc. Eur. Robotics Symp. (EUROS06) (2005)

74.98 M. Pardowitz, R. Zöllner, R. Dillmann: Learning sequential constraints of tasks from user demonstrations, Proc. IEEE-RAS Int. Conf. Humanoid Robots (HUMANOIDS05) (2005) pp. 424–429

74.99 S. Calinon, P. Kormushev, D.G. Caldwell: Compliant skills acquisition and multi-optima policy search with EM-based reinforcement learning, Robotics Auton. Syst. **61**(4), 369–379 (2013)

74.100 C.G. Atkeson, A.W. Moore, S. Schaal: Locally weighted learning for control, Artif. Intell. Rev. **11**(1–5), 75–113 (1997)

74.101 J. Peters, S. Vijayakumar, S. Schaal: Reinforcement learning for humanoid robotics, Proc. IEEE Int. Conf. Humanoid Robots (Humanoids) (2003)

74.102 T. Yoshikai, N. Otake, I. Mizuuchi, M. Inaba, H. Inoue: Development of an imitation behavior in humanoid kenta with reinforcement learning algorithm based on the attention during imitation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2004) pp. 1192–1197

74.103 D.C. Bentivegna, C.G. Atkeson, G. Cheng: Learning tasks from observation and practice, Robotics Auton. Syst. **47**(2/3), 163–169 (2004)

74.104 P. Kormushev, S. Calinon, R. Saegusa, G. Metta: Learning the skill of archery by a humanoid robot iCub, Proc. IEEE Int. Conf. Human. Robots Nashville (2010)

74.105 P. Pastor, M. Kalakrishnan, S. Chitta, E. Theodorou, S. Schaal: Skill learning and task outcome prediction for manipulation, IEEE Int. Conf. Robotics Autom. (2011)

74.106 J. Kober, J. Peters: Policy search for motor primitives in robotics, Mach. Learn. **84**(1/2), 171–203 (2011)

74.107 P. Kormushev, S. Calinon, D.G. Caldwell: Robot motor skill coordination with EM-based reinforcement learning, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) Taipei (2010) pp. 3232–3237

74.108 N. Jetchev, M. Toussaint: Fast motion planning from experience: Trajectory prediction for speeding up movement generation, Auton. Robots **34**(1/2), 111–127 (2013)

74.109 F. Guenter, M. Hersch, S. Calinon, A. Billard: Reinforcement learning for imitating constrained reaching movements, RSJ Adv. Robotics **21**(13), 1521–1544 (2007)

74.110 B.D. Ziebart, A. Mass, A. Bagnell, A.K. Dey: Maximum entropy inverse reinforcement learning, Proc. AAAI Conf. Artif. Intell. (2008)

74.111 P. Abbeel, A. Coates, A. Ng: Autonomous helicopter aerobatics through apprenticeship learning, Int. J. Robotics Res. **29**(13), 1608–1639 (2010)

74.112 S. Ross, G. Gordon, J.A. Bagnell: A reduction of imitation learning and structured prediction to no-regret online learning, Proc. 14th Int. Conf. Artif. Intell. Stat. (AISTATS11) (2011)

74.113 Y.K. Hwang, K.J. Choi, D.S. Hong: Self-learning control of cooperative motion for a humanoid robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 475–480

74.114 B. Jansen, T. Belpaeme: A computational model of intention reading in imitation, Robotics Auton. Syst. **54**(5), 394–402 (2006)

74.115 A. Billard, K. Dautenhahn: Grounding communication in autonomous robots: An experimental study, Robotics Auton. Syst. **24**(1/2), 71–81 (1998)

74.116 P. Abbeel, A. Ng: Apprenticeship learning via inverse reinforcement learning, Int. Conf. Mach. Learn. (2004)

74.117 N. Ratliff, A.J. Bagnell, M. Zinkevich: Maximum margin planning, Int. Conf. Mach. Learn. (2006)

74.118 A. Billard, S. Calinon, F. Guenter: Discriminative and adaptive imitation in uni-manual and bimanual tasks, Robotics Auton. Syst. **54**, 370–384 (2006)

74.119 J. Choi, K. Kim: Nonparametric Bayesian inverse reinforcement learning for multiple reward functions, Adv. Neural Inf. Process. Syst. **25**, 305–313 (2012)

74.120 A.K. Tanwani, A. Billard: Transfer in inverse reinforcement learning for multiple strategies, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2013)

74.121 D.H. Grollman, A. Billard: Donut as i do: Learning from failed demonstrations, IEEE Int. Conf. Robotics Autom. (2011)

74.122 A. Rai, G. de Chambrier, A. Billard: Learning from failed demonstrations in unreliable systems, IEEE-RAS Int. Conf. Humanoid Robots (2013)

74.123 M. Goodrich, A. Schultz: Human-robot interaction: A survey, Found. Trend. Human-Comput. Int. **1**(3), 203–275 (2007)

74.124 T. Fong, I. Nourbakhsh, K. Dautenhahn: A survey of socially interactive robots, Robotics Auton. Syst. **42**(3/4), 143–166 (2003)

74.125 C. Breazeal, B. Scassellati: Robots that imitate humans, Trends Cogn. Sci. **6**(11), 481–487 (2002)

74.126 B. Scassellati: Imitation and mechanisms of joint attention: A developmental structure for building social skills on a humanoid robot, Lect. Notes Comput. Sci. **1562**, 176–195 (1999)

74.127 H. Kozima, H. Yano: A robot that learns to communicate with human caregivers, Int. Workshop Epigenet. Robotics (2001)

74.128 H. Ishiguro, T. Ono, M. Imai, T. Kanda: Development of an interactive humanoid robot Robovie – An interdisciplinary approach, Springer Tracts Adv. Robotics **6**, 179–192 (2003)

74.129 K. Nickel, R. Stiefelhagen: Pointing gesture recognition based on 3d-tracking of face, hands and head orientation, Int. Conf. Multimodal Interfaces (ICMI) (2003) pp. 140–146

74.130 M. Ito, J. Tani: Joint attention between a humanoid robot and users in imitation game, Int. Conf. Dev. Learn. (ICDL) (2004)

74.131 V.V. Hafner, F. Kaplan: Learning to interpret pointing gestures: Experiments with four-legged autonomous robots, Lect. Notes Comput. Sci. **3575**, 225–234 (2005)

74.132 C. Breazeal, D. Buchsbaum, J. Gray, D. Gatenby, B. Blumberg: Learning from and about others:

Towards using imitation to bootstrap the social understanding of others by robots, Artif. Life **11**(1/2), 31–62 (2005)
74.133 P.F. Dominey, M. Alvarez, B. Gao, M. Jeambrun, A. Cheylus, A. Weitzenfeld, A. Martinez, A. Medrano: Robot command, interrogation and teaching via social interaction, Proc. IEEE-RAS Int. Conf. Humanoid Robots (Humanoids) (2005) pp. 475–480
74.134 A.L. Thomaz, M. Berlin, C. Breazeal: Robot science meets social science: An embodied computational model of social referencing, Workshop Toward Soc. Mech. Android Sci. (CogSci) (2005) pp. 7–17
74.135 C. Breazeal, L. Aryananda: Recognition of affective communicative intent in robot-directed speech, Auton. Robots **12**(1), 83–104 (2002)
74.136 S. Calinon, A. Billard: Teaching a humanoid robot to recognize and reproduce social cues, Proc. IEEE Int. Symp. Robot Human Int. Commun. (RO-MAN) (2006) pp. 346–351
74.137 Y. Yoshikawa, K. Shinozawa, H. Ishiguro, N. Hagita, T. Miyamoto: Responsive robot gaze to interaction partner, Proc. Robotics Sci. Syst. (RSS) Philadelphia (2006)
74.138 S. Calinon, A. Billard: What is the teacher's role in robot programming by demonstration? – Toward benchmarks for improved learning, Int. Stud. Spec. Issue Psychol, Benchmarks Human-Robot Int. **8**(3), 441–464 (2007)
74.139 S. Chernova, M. Veloso: Interactive policy learning through confidence-based autonomy, J. Artif. Intell. Res. **34**, 1–25 (2009)
74.140 B.D. Argall, E.L. Sauser: Tactile guidance for policy adaptation, Found. Trend. Robotics **1**(2), 79–133 (2010)
74.141 S. Calinon, A. Billard: Active teaching in robot programming by demonstration, Proc. IEEE Int. Symp. Robot Human Int. Commun. (RO-MAN), Jeju (2007) pp. 702–707
74.142 D. Silver, A. Bagnell, A. Stentz: Active learning from demonstration for robust autonomous navigation, IEEE Conf. Robot. Autom. ICRA'12 (2012)
74.143 M. Riley, A. Ude, C. Atkeson, G. Cheng: Coaching: An approach to efficiently and intuitively create humanoid robot behaviors, Proc. IEEE-RAS Int. Conf. Humanoid Robots (Humanoids) (2006) pp. 567–574
74.144 H. Bekkering, A. Wohlschlaeger, M. Gattis: Imitation of gestures in children is goal-directed, Q. J. Exp. Psychol. **53A**(1), 153–164 (2000)
74.145 M. Nicolescu, M.J. Mataric: Task learning through imitation and human-robot interaction. In: *Models and Mechanisms of Imitation and Social Learning in Robots, Humans and Animals*, (MIT Press, Cambridge 2006) pp. 407–424
74.146 J. Demiris, G. Hayes: Imitative learning mechanisms in robots and humans, 5th Eur. Workshop Learn. Robots, ed. by V. Klingspor (1996) pp. 9–16
74.147 J. Aleotti, S. Caselli: Robust trajectory learning and approximation for robot programming by demonstration, Robotics Auton. Syst. **54**(5), 409–413 (2006)
74.148 M. Ogino, H. Toichi, Y. Yoshikawa, M. Asada: Interaction rule learning with a human partner based on an imitation faculty with a simple visuomotor mapping, Robotics Auton. Syst. **54**, 414–418 (2006)
74.149 A. Billard, M. Matarić: Learning human arm movements by imitation: Evaluation of a biologically-inspired connectionist architecture, Robotics Auton. Syst. **941**, 1–16 (2001)
74.150 A.J. Ijspeert, J. Nakanishi, S. Schaal: Movement imitation with nonlinear dynamical systems in humanoid robots, IEEE Int. Conf. Robotics Autom. (ICRA2002) (2002) pp. 1398–1403
74.151 R.H. Cuijpers, H.T. van Schie, M. Koppen, W. Erlhagen, H. Bekkering: Goals and means in action observation: A computational approach, Neural Netw. **19**(3), 311–322 (2006)
74.152 M.W. Hoffman, D.B. Grimes, A.P. Shon, R.P.N. Rao: A probabilistic model of gaze imitation and shared attention, Neural Netw. **19**(3), 299–310 (2006)
74.153 A. Billard: Drama, a connectionist architecture for on-line learning and control of autonomous robots: Experiments on learning of a synthetic proto-language with a doll robot, Ind. Robot **26**(1), 59–66 (1999)
74.154 P. Gaussier, S. Moga, J.P. Banquet, J. Nadel: Learning and communication via imitation: An autonomous robot perspective systems, IEEE Trans. Man Cybern. A **31**(5), 431–442 (2001)

Multimedia Contents

75. Biologically Inspired Robotics

Fumiya Iida, Auke Jan Ijspeert

Throughout the history of robotics research, nature has been providing numerous ideas and inspirations to robotics engineers. Small insect-like robots, for example, usually make use of reflexive behaviors to avoid obstacles during locomotion, whereas large bipedal robots are designed to control complex human-like leg for climbing up and down stairs. While providing an overview of bio-inspired robotics, this chapter particularly focus on research which aims to employ robotics systems and technologies for our deeper understanding of biological systems. Unlike most of the other robotics research where researchers attempt to develop robotic applications, these types of bio-inspired robots are generally developed to test unsolved hypotheses in biological sciences. Through close collaborations between biologists and roboticists, bio-inspired robotics research contributes not only to elucidating challenging questions in nature but also to developing novel technologies for robotics applications. In this chapter, we first provide a brief historical background of this research area and then an overview of ongoing research methodologies. A few representative case studies will detail the successful instances in which robotics technologies help identifying biological hypotheses. And finally we discuss challenges and perspectives in the field.

Biologically inspired robotics (or bio-inspired robotics in short) is a very broad research area because almost all robotic systems are, in one way or the other, inspired from biological systems. Therefore, there is no clear distinction between bio-inspired robots and the others, and there is no commonly agreed definition [75.1]. For example, legged robots that walk, hop, and run are usually regarded as bio-inspired robots because many biological systems rely on legged locomotion for their survival. On the other hand, many robotics researchers implement biological models of motion control and navigation onto wheeled platforms, which could also be regarded as bio-inspired robots [75.2].

75.1 **General Background** 2016
75.1.1 Brief History and Conceptual Background 2016
75.2 **Methodology** 2017
75.2.1 Similarities and Differences in Robotics and Biology 2017
75.2.2 Modeling Biological Systems 2018
75.2.3 Research Methods and Tools 2020
75.3 **Case Studies** 2021
75.3.1 Bio-Inspired Legged Locomotion 2021
75.3.2 Reflexes and Central Pattern Generators 2023
75.3.3 Bio-Inspired Navigation 2024
75.4 **Landscape of Bio-Inspired Robotics Research and Challenges** 2026
75.4.1 Bio-Inspired Climbing 2026
75.4.2 Flapping Flight and Swimming Mechanisms 2027
75.4.3 Artificial Hands, Haptics, and Whiskers 2027
75.4.4 Self-Reconfigurable and Evolutionary Robotics 2028
75.4.5 Bio-Inspired Soft Robots 2028
75.4.6 Neuroprosthetics and Social Interactions 2028
75.5 **Conclusion** 2028
Video-References 2028
References 2029

75.1 General Background

The broad spectrum of bio-inspired robotics research is reflected to a variety of synonyms used in different scientific communities. For example, biomimetics and bionics are usually used to represent the types of research in which researchers observe biological systems and extract design principles for robotic applications. The terms bio-robotics and bio-engineering are also used interchangeably to biomimetics and bionics, but they often refer to engineered solutions specifically for biomedical applications. Another approach, usually classified as artificial life, biological cybernetics, or biophysics, investigates biological systems by using synthetic approaches. Here biological systems are typically viewed as mechanical and chemical entities and characterized by mechanistic models that are often very similar to the bio-inspired robots. The definitions and types of research studies conducted in these research areas are often overlapping, and many research projects provide results across the different areas.

While some of these research areas are also covered in the other chapters of this handbook, one of the most prominent, and more importantly useful, aspects of bio-inspired robotics lies in the contributions of robotics research for biological sciences. In contrast to the other robotics research, a significant number of bio-inspired robots were developed for the purpose of testing hypotheses concerning biological systems and for identifying the underlying mechanisms of biological systems that are very difficult to be clarified otherwise. In this chapter, we specifically focus on this aspect of bio-inspired robotics.

75.1.1 Brief History and Conceptual Background

There is a long history of the engineers' desire to replicating biological systems into artificial ones, including the famous examples of Japanese Karakuri dolls and Swiss automata several centuries ago (Fig. 75.1). Similarly, scientific efforts to use robotic systems to understand biological systems have also a relatively long history, which goes back even before the modern robotics started. One of the most influential examples in the earlier ages can be represented by the works in the field of cybernetics in which a number of system theories such as the concepts of feedback and feedforward were applied to understand phenomena in biological systems [75.3, 4].

The rise of neuroscience in the mid-twentieth century influenced bio-inspired robotics research significantly. In the 1950s, one of the first bio-inspired robots, Tortoises, was built by Grey Walter, a neurophysiologist. The work was mainly driven by biological questions, such as how a simple neuron connectivity can result in richness of sensory-motor behaviors, and a turtle-like robot consisting of analog electronics, sensors, and electric motors demonstrated reflexive behaviors as well as basic motor learning in autonomous robots [75.5]. Another physiologist, *Valentino Braitenberg*, also explored the power of *understanding by building* approach, which led to the famous thought experiments of Braitenberg Vehicles, i. e., a series of imaginary mobile robots that are still often used to teach the relationship between neural connectivity and sensory-motor behaviors [75.6].

The invention of digital computers was also another historical milestone in bio-inspired robotics. The pioneers of digital computers such as *John von Neumann* and *Alan Turing* provided significant influences in the biological studies through their seminal works on self-replication and self-organization [75.7, 8], and the subsequent foundation of the field of Artificial Intelligence in the 1950s was driven based on the understanding of human intelligence especially from the computational standpoint [75.9].

While the power of digital computers dominated bio-inspired robotics for a while, the study of insect-like reflexive behaviors became popular again in the 1980s with the emergence of behavior-based robotics. The intensive studies of this approach shed light on the massively parallel nature of motion control processes, and demonstrated behaviors that cannot be fully explained by the conventional *sense-think-act* style control approach [75.10, 11]. One of the main contributions of this line of studies lies in the fact that the diversity and flexibility of control architecture is essential in adaptive behaviors in real-world systems, and the complexity of behaviors is not necessarily originated in the complexity of controller but in physical system–environment interactions [75.12].

Since then the research on physical system–environment interactions has been very popular in the interdisciplinary community of embodied cognitive science and artificial intelligence in which roboticists, biologists, computer scientists, and physicists work together to look into further details of underlying mechanisms of adaptivity in biological systems. Here the body as a physical entity is not only considered as a necessary *container* of intelligent adaptive behaviors but it also plays a central role that induces self-organization of patterns and structures in adaptive intelligent behaviors [75.13–15]. In this context, robotics methodologies and technologies are used to investigate hypotheses about the roles of embodiment that cannot be easily tested in animals.

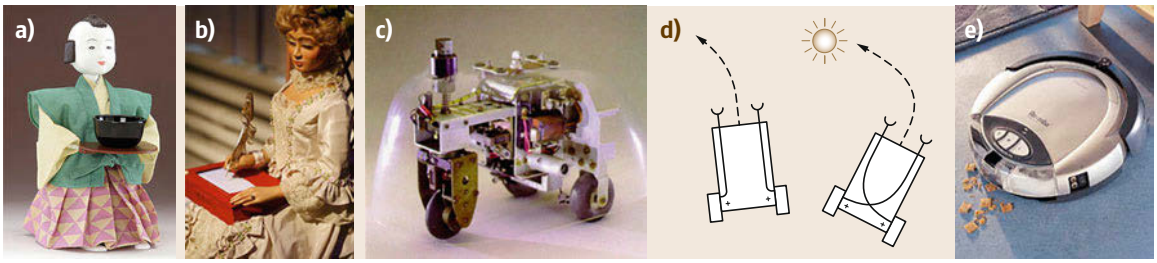


Fig.75.1a–e Examples of bio-inspired robots in the history. **(a)** Japanese Karakuri doll, **(b)** Swiss Automaton, **(c)** Grey Walter's Tortoises Robot (after [75.5]), **(d)** Braitenberg Vehicle (after [75.6]), and **(e)** a cleaning robot Roomba as an example of behavior-based robotics application

75.2 Methodology

In order to provide contributions to both robotics and biological sciences, bio-inspired robotics research usually follows a series of unique research processes which are not necessarily common to the other areas of robotics. This section first explains similarities and differences in robotics and biology research, and then introduces a set of important concepts, methodology, and research tools often used in the bio-inspired robotics.

75.2.1 Similarities and Differences in Robotics and Biology

Both robotics and biological research aim to obtain the basic understanding about autonomous and adaptive behaviors of complex systems, thus the scientific methodologies of these fields are usually very similar. For example, biologists usually start with finding an interesting problem in animals, that can be comparable to roboticists building robot prototypes of their interests as the first step of research. Second, the animals are carefully observed such that the underlying mechanisms can be efficiently and effectively analyzed, a step which roboticists also follow in the case of their robots. And finally, both biologists and roboticists develop hypothetical models to explain the mechanisms of their interests and test them through additional experiments and analyses. Here the modeling processes are also similar in both robotics and biology in a sense that they typically employ similar scientific methods, such as dynamics modeling tools, computational optimization techniques, and the other analysis methods of system engineering, for example.

There are, however, a number of fundamental differences in biological and robotics research, which are mostly originated in the fact that biological systems are *constructed* by considerably different design principles in nature. Although it is very difficult to cover all differences, the following aspects of biological systems characterize some of the major discrepancies from today's robotic systems.

Multipurpose Systems in Unstructured and Uncertain Environment

In contrast to most of the robotic systems that are designed to perform one type of task in a well-defined environment, all biological systems are intrinsically multipurpose systems that are designed for many tasks in undefined and uncertain environments. For example, animals need to process tasks such as regulating metabolism for self-sufficiency, protecting themselves from predators, mating, and reproducing. It is important for roboticists to know that (i) animals are not optimized for one of these tasks, but they are designed to conduct all of them, and (ii) they do so in a way that is not optimal in any sense, but just *good enough* to survive and reproduce. Therefore, it is often not a good idea to blindly copy a part of animals' designs and mechanisms into robotic systems because there could be an alternative optimal solution if the system has to do a single task in a well-defined environment. Robotics researchers are, however, able to learn many principles and mechanisms from biological systems, if they are interested in the systems that need to deal with many tasks in unstructured and uncertain environments.

Massively Parallel, Modular, and Redundant Structures

Biological systems are composed of highly redundant structures in their bodies [75.13]. For example, most of multicellular organisms have massively parallel muscle fibers constituting a muscle group, a skeletal joint controlled through multiple muscle groups, millions of nerve cells conducting parallel signal processing, and countless receptors sensing changes in the environment. If compared to our robotic systems today, biological systems have orders of magnitude larger numbers of

such sensory, actuation, and computational units that have to be carefully considered in bio-inspired robotics research. In addition, the processes necessary for autonomy and adaptivity are generally highly decentralized in biological systems. The control of animals' arms and legs, for example, involves mechanical interactions of musculoskeletal structures, reflexive sensory-motor pathways in spinal cord, and more complex motion control and planning in the higher centers of nervous systems, which are running in parallel. The redundant structures of biological systems also provide an important discrepancy between animals and today's robots: most of the subsystems in a living organisms have great autonomy and adaptivity by themselves, as exemplified by the fact that individual cells in skins and bones as well as receptors and muscles have their own regulation mechanisms as metabolism and growth.

Self-Organization and Dynamic Changes of Entire Organisms

Another important discrepancy between animals and today's robots lies in the fact that there is no *human designers* behind animals, and all components in an organism have to be designed, assembled and repaired by itself. Consequently, every individual animal has to start its life smaller and gradually grow larger over time; every part of their bodies is continuously changing; sensory-motor control has to be continuously updated to reflect the changes in the bodies. It is particularly important for robotics researchers to consider that there are different timescales in these dynamic processes in biological systems (Fig. 75.2 [75.16]). Some of the different timescales can be represented by the continuous changes of body plans at the evolutionary processes, the changes of body sizes and muscle strength through ontogenetic timescale, and update of sensory-motor loops in here and now timescale.

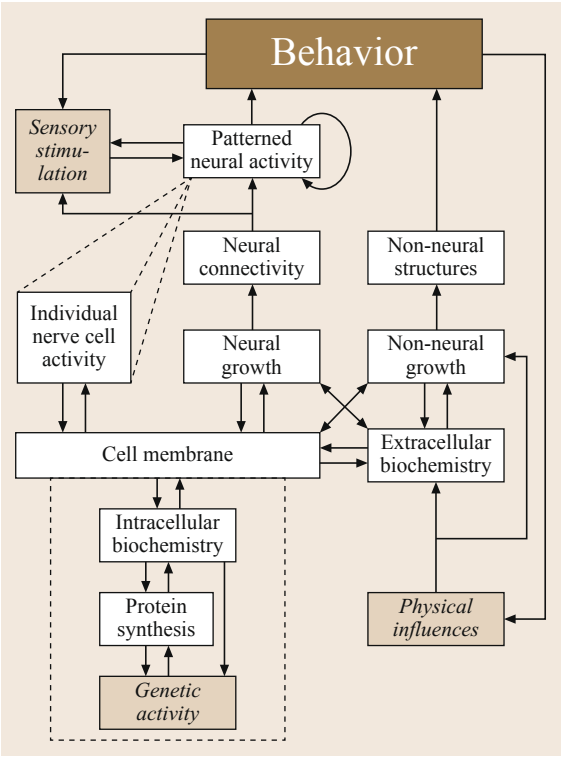


Fig. 75.2 Elements and their interactions that influence behaviors in biological systems. The model includes both neural and non-neural elements such as hormones (which constitute part of the extracellular biochemistry), bones, muscles, sensors, and their ontogenetic processes (after [75.16])

75.2.2 Modeling Biological Systems

Modeling plays an essential role in biological sciences. Biological models are the representation of knowledge which is not only used to communicate scientific discoveries among researchers but also structuring research areas by labeling knowns and unknowns. To make research activities efficient and effective, there are many different ways to model biological systems such as descriptive/illustrative explanations, mathematical, physical, or chemical representations. From this perspective, the researchers in bio-inspired robotics have been exploring whether robotic platforms can be used as a scientific tool to develop models of biological systems, which leads to an alternative approach to elucidating biological hypotheses (Fig. 75.3). Robots are useful because they are physical entities as compared to simulated ones. First of all, by building models that can be implemented in a physical system, the process ensures whether the model in question is physically meaningful or not. In contrast, computationally simulated models always involve many approximations and simplifications which might affect the realism and therefore usefulness of the model (e.g., correctly modeling the hydrodynamics of a swimming fish is very hard in a simulation while it *comes for free* in a fish robot). Second, building robots directly contributes to the development of unconventional technological components.

Having said that, the use of robots as a scientific tool for biology is challenging because of the considerable discrepancies between animals and robots as explained in the previous subsection. A superficial copy of biological systems into a robotic counterpart is not a good idea because robots usually functions based on a completely different set of mechanisms, and the developed robot would most likely not explain much about the underlying mechanisms of biological

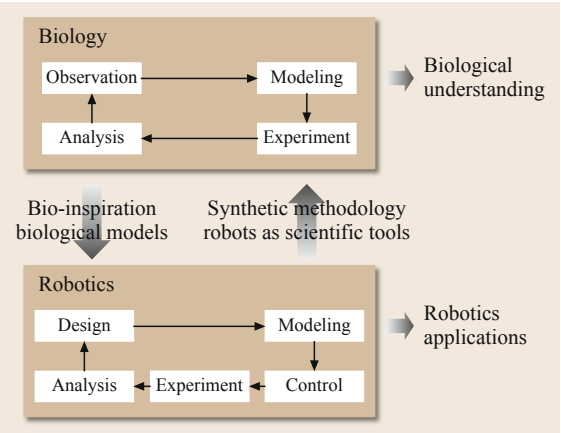


Fig. 75.3 Overview of bio-inspired robotics. Both biology and robotics follow similar research methods and interact to each other through models and tools, while they contribute to different objectives, i. e., understanding of biological systems or development of robotic applications

systems. Instead, it is particularly important to carefully examine what biological hypothesis we are interested in and we are testing by building robotic platforms. As an extreme example, if one is interested in understanding navigation mechanisms of animals, it is usually better to start with wheeled robot platforms than legged or flying ones, because the latter platforms would introduce unnecessary complexity in the research to examine a given hypothesis.

It is, however, not a trivial problem to find good biological hypotheses that can be applied in the bio-inspired robotics research. Biological systems are generally very complex, and hypotheses are not clearly separable from one question to another. Behaviors of animals are, for example, a result from neuronal activities of short and long terms, musculoskeletal dynamics, genetic and social interactions, and the mechanisms could also significantly vary in different individuals or species.

In order to deal with such a complexity in nature, *Full* and *Koditchek* have proposed an insightful methodology [75.17]. As shown in Fig. 75.4, they proposed a two-level modeling process which could benefit bio-inspired robotics research greatly: in the first level, a model should be simplified as much as possible such that it can be generalized over many species or in a large scale without being bothered too much by the details of complex animal structures. These models are called *templates*. A good example of this kind is the so-called spring-mass model that characterizes running behaviors of many different types of legged animals, even though it is an extremely simple model consisting of a point mass and a linear spring only. This approach enables researchers to examine the basic principles in nature, and despite its simplicity, the model can explain behavioral characteristics in a wide variety of animals even without considering detailed anatomical discrepancies between species. In the second level of modeling processes, templates should be enhanced by more details, which are called *anchors*. The investigations of the anchors are generally intended for more specific questions by adding redundancies of muscles or limbs or implementing more complex neuromuscular circuitry, for example. Through the template–anchor research approach, a research area can be effectively structured, and, for bio-inspired robotics research in particular, the simplified models can benefit robotics engineers to replicate behaviors of biological systems in a conceptual level.

Independently from the simplicity, it is also important to consider the purposes of models which

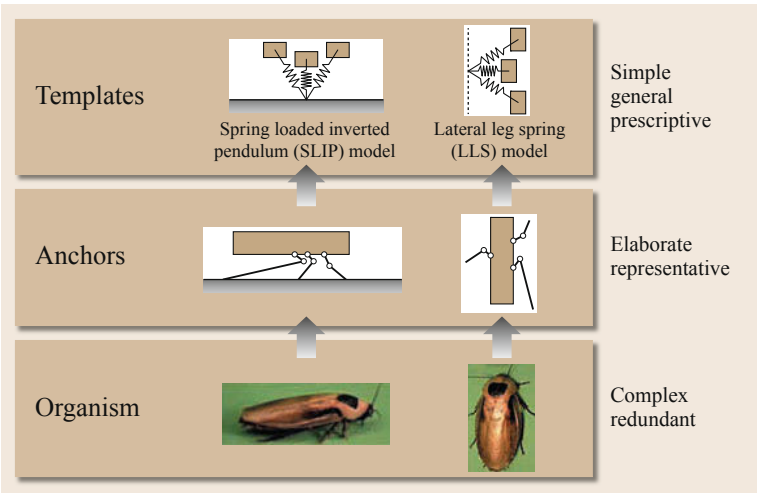


Fig. 75.4 Modeling hierarchy exemplified by the legged locomotion models. Animals can be abstracted into more elaborate and representative models (anchors) and/or simpler and more general models (templates) (after [75.17])

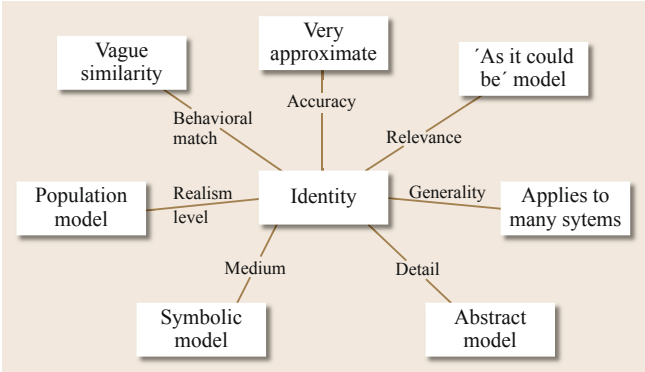


Fig. 75.5 Dimensions for describing models (after [75.18])

also influence how to structure bio-inspired robotics projects. After all, there is no correct model, but there are good or bad models with respect to the goal and hypothesis that are investigated. *Webb* has, for example, pointed out that there are seven major criteria with which we could evaluate models of bio-inspired robotics (Fig. 75.5 [75.18]):

1. Relevance to biology: as not all robotics research contributes to biological studies, it is important to clarify the degree to which a target robotic system and model are relevant.
2. Level: what are the base units of the model, and on what level of hierarchy in biological systems does the model attempt to represent?
3. Generality: a model could be developed for elucidating a mechanism of a specific system of interest, or for a more general mechanism that can be applied to many others. Therefore, the generality criterion considers how many systems the model intends to represent?
4. Abstraction: this criterion concerns the number and complexity of mechanisms included in the model.
5. Structural accuracy: there are many ways of representing systems but a question is the degree to which the model explains the internal mechanism of the target systems. In an extreme case, a model can behave the same at the level of input/output relationship, but it does not necessarily mean that the internal mechanism is the same.
6. Behavioral match: to what extent does the model behave like the target animal?
7. Medium of model: what is the model built from? A model can be made of mechanical, electrical, hydraulic, etc. or alternatively iconic, analog, symbolic, for example. Properly identifying these questions and the purpose of a robot is important. Indeed, there is a risk in bio-inspired robotics that a project is not useful to biology (i. e., it does not properly address a scientific question) nor to robotics (i. e., it does not perform better than a more conventional robot).

75.2.3 Research Methods and Tools

As it might have been already noticed, robotics technologies are used for many different purposes in the context of bio-inspired robotics. Robots are often used to *explore* new research areas and questions; many platforms were built and examined for the purpose of *testing biological hypotheses*; some of the other platforms were developed specifically for *application discovery* based on the identified principles in biological studies; and more recently, robotics technologies were *interfaced with or integrated into biological systems* to understand or enhance animals capabilities.

Even though there are many successful contributions of robotics technologies to biological studies, it is often not trivial to identify what are the good methods and tools for the given specific hypothesis or problems. In particular, one of the most critical questions is probably whether it is necessary to build a physical robotic platform or it is sufficient to investigate pseudo-robots in simulation. Building physical robots is usually very costly and requires significant amount of additional knowledge and know how; thus there are many case studies of bio-inspired robotics research conducted only in simulation in the past [75.19, 20]. This type of research generally makes use of physics models or physically realistic simulation environments that allow us to explore fairly complex artificial creatures. This approach is useful to explore a large parameter space which cannot be optimized in the real-world platforms. Also, the use of virtual creatures in the bio-inspired robotics research is also extremely important to investigate concepts and hypotheses that are not possible to test technologically such as robots with point masses, frictionless joints, self-replication, and robotic hardware evolution that cannot be possible with our today's technologies.

However, building physical robots is a *must* in some cases. For example, there are often target concepts or hypotheses involved in physical processes that are difficult to theoretically model, such as friction, impacts, thermodynamics, and hydro/aerodynamics. In addition, there has been an increasing interest in complex robotic systems that contain a considerable number of physical elements such as joints and actuators, simulation models of which tend to be vulnerable against accumulated errors. Another aspect of robots used as a scientific tool is to explore hypotheses that cannot be tested in nature. This approach is often called synthetic methodology (an understanding-by-building approach [75.13]), meaning

that, by building and using artificial systems, we test biological principles that cannot be tested in conventional biological methods. The most representative example is the Braitenberg Vehicles, which we introduced in Sect. 75.1.1. Every vehicle has a set of oversimplified neural connections of an *animal* which does not exist in nature, but these creatures can explain important principles in biological systems. A similar approach was also extended to the studies of *life as it could be* [75.21, 22], in which, by creating biological phenomena in simulation or robots, a broader and more universal understanding of living systems. This methodology is particularly effective in the studies related to evolutionary biology, in which verifications of hypotheses are usually very challenging ([75.20, 23]; see also the chapter of evolutionary robotics).

75.3 Case Studies

This section introduces three research areas that have been particularly active in bio-inspired robotics in the last few decades. Here we highlight the case studies in which robotics researches contributed to our further understanding of biological systems, and we illustrate how the concepts and methods we introduced in the previous sections were applied to these studies.

75.3.1 Bio-Inspired Legged Locomotion

Legged locomotion has been one of the most popular research topics in bio-inspired robotics for a long time because it characterizes the salient differences between biological and artificial systems. Although legged locomotion is seemingly easy, useful, and efficient for biological systems to move around in complex environments, it is surprisingly difficult to understand the underlying mechanisms hence challenging to implement into robotic systems. The legged locomotion research has a relatively long history that goes back to the foundation of biomechanics in the seventeenth century [75.24], and it became particularly popular when the modern robotics was established in the 1970s [75.25, 26]. One of the main reasons that many robotics researchers were attracted by the issue of legged locomotion lies in the fact that it covers many of the discrepancies between biological and artificial systems that we discussed in Sect. 75.2.1. More specifically, legged systems have to carry out many different tasks in unstructured environments such as establishing sturdy footholds, avoiding obstacles, dealing with variations of payload and velocities, and changing gait patterns, for example; Legged systems have to deal with massive parallel processes which are required for coordinating many joints and muscles, as well as local reflexes and high-level decision and planning; And adaptivity is essential for legged systems because they need to maintain mobility in different environments as well as under significant changes of its body plan over growth processes, for example. While there are countless case studies in the past, this subsection focuses on the four key challenges of bio-inspired legged locomotion, i. e., stability, gait, energetics, and actuation, because they have been recognized as the long-standing challenges in bio-inspired legged locomotion. More comprehensive robotics research can be found in the section of legged locomotion.

Stability and Gait in Legged Locomotion

One of the main challenges in the study of legged locomotion is to uncover the mechanisms of motion stabilization against disturbances [75.27]. Biological systems usually make use of a variety of mechanisms including mechanical self-stabilization [75.28, 29], spinal reflexes [75.30, 31], central pattern generators [75.32], or sensory-motor control originated in the higher center of nervous system [75.33]. Because of the complexity of stabilization mechanisms in animals, bio-inspired robotics played an important role to systematically investigate the issues of stabilization through modeling of locomotion dynamics and reproducing the behaviors in legged robots of various kinds in the past.

Among others, bipedal walking was one of the most intensively studied topic areas which nicely illustrates how a synthetic approach could structure a research area of a complex biological problem. The backbone of this research area is the so-called *inverted pendulum walking model* which was originally investigated in biomechanics, and later used in the bio-inspired robotics research. The model considers the simplest physics representation of walking dynamics, i. e., a point mass is attached on a massless link, and simulates walking dynamics as the mass vaulting over the link [75.24]. This model can be, for example, physically implemented by the so-called rimless wheel that is the simplest *robotic* representation of the model (Fig. 75.6a; [75.34, 43]), and then, a slightly more complex configuration, the *compass gait model*, was proposed (Fig. 75.6b, [75.44]; VIDEO 111 shows an experiment of a compass gait bipedal robot locomotion on rough terrains). In contrast to the rimless wheel model that considers only stance leg dynamics during walking, the compass gait model has three masses with a passive hip joint, which allow us to investigate swing leg dynamics in addition to the stance

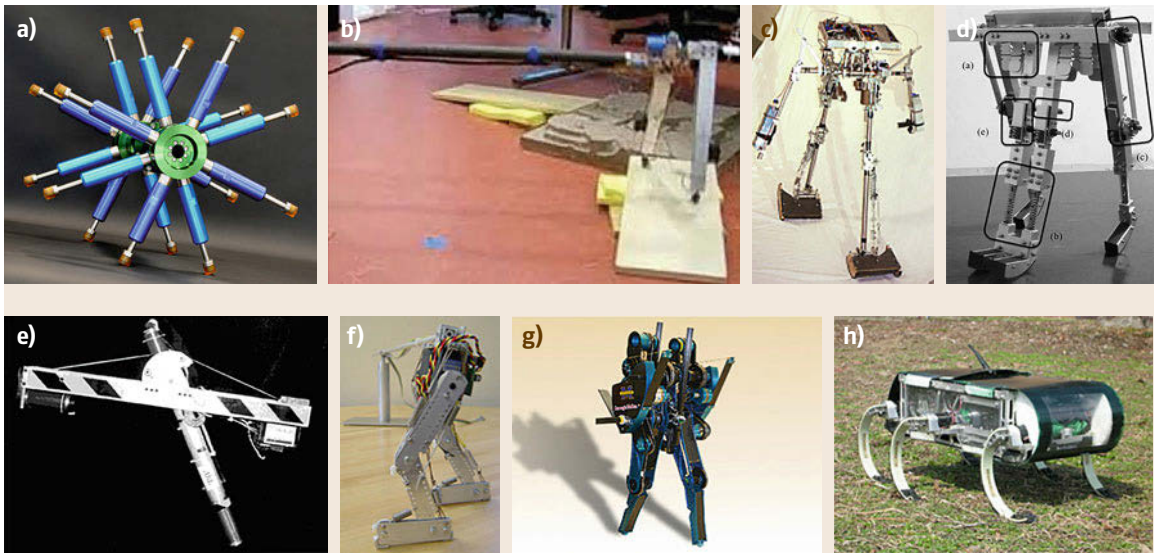


Fig.75.6a–h Examples of bio-inspired legged robots. **(a)** A physical implementation of rimless wheel walking model (after [75.34]), **(b)** A physical implementation of compass gait walking model (after [75.35, 36]), **(c)** passivity based bipedal walking robot (after [75.37]), **(d)** passive dynamic runner (after [75.38]), **(e)** energy efficient hopping robot (after [75.39]), **(f)** bipedal walking and running robot based on biarticlar springs (after [75.40]), **(g)** biped robot based on variable stiffness actuators (after [75.41]), and **(h)** hexapod robot based on spring mass dynamics (after [75.42])

leg [75.45]. An important implication of this line of research lies in the fact that, because of the simple formulations of complex dynamics, researchers were able to systematically investigate different aspects of the complex behaviors while keeping an overarching structure of research issues. These simple models were, for example, gradually and systematically enhance by integrating knee and ankle joints (Fig. 75.6c; [75.37, 43]), foot segments with variations of shapes, influences of mass distributions [75.46], influence of lateral motions, as well as a variety of advanced motion control architectures to demonstrate actuated locomotion in more complex environments [75.35, 36, 47, 48].

Although walking dynamics is a highly interesting challenge, the basic locomotion stability in walking is not sufficient to understand legged locomotion in nature, but stability has to be also maintained in different gait patterns, such as running because most of biological systems exhibit a rich variety of gaits. For this reason, running dynamics has also been studied intensively by investigating a simple model, i. e., the so-called spring-loaded inverted pendulum (SLIP model; [75.28, 49, 50]). The model consists of a point mass and a massless linear spring, on top of which many variations were proposed such as running models with nonlinear spring, segmented legs [75.51], swing leg dynamics, upper torso, lateral balancing [75.52], and wheel-like configuration (Fig. 75.6h [75.42]), for example. The SLIP model was also used to study walking dynamics and gait transitions between walking and running (Fig. 75.6f [75.40, 53]; VIDEO 110 shows an example of a biped robot walking and running). It is important to note that many of these models and robots were developed for the hypotheses difficult to test in biological systems. The biological legs are hardly linear springs but they consist of numerous active and nonlinear components. However, by examining these models and robots, we are able to learn the basic underlying principles such as the degree to which a spring-like behavior of legs could contribute to the stability of walking and running locomotion, for example. In addition, such an abstraction of biological body structures is very practical for robotics research as we are able to design and construct robots based on the underlying principles without replicating complex anatomical structures consisting of organic components.

Energy Efficiency and Bio-Inspired Actuation

Another considerable challenge in the study of bio-inspired legged locomotion is the principles for energy efficiency. It has been known that the locomotion efficiency of biological systems is known to be at least an order of magnitude better than most of the legged robots today but it is not fully understood why biological locomotion is so efficient. The complexity of the energetic problem in legged locomotion is originated in the many possible sources of energy dissipation such as frictional and damping losses in joints and mus-

cles, mechanical impact losses at foot touchdown to the ground, metabolic costs, and energy required for acceleration and deceleration of body parts. Because of such a complexity, the bio-inspired robotics research has been significantly contributing to this problem by building and analyzing, for example, purely mechanical locomotion systems [75.38, 43], underactuated locomotion control [75.35, 37], the use of passive spring and self-excited vibration [75.39, 54], and exoskeleton devices [75.55]. All of these case studies were contributing to a comprehensive understanding of energy efficiency in biological locomotion [75.56], and some of the hypotheses have been analyzed and tested in biological systems.

In addition to the whole body dynamics, energy efficient locomotion has also been investigated at the level of actuation because the muscle-tendon systems play a major role in animals' efficient locomotion. Inspired from the biological models of muscles, this research trend started from the so-called series elastic actuators, which is an actuator unit containing a mechanical spring being installed in series to an electric motor [75.62]. The implementation of mechanical spring in an electric motor explained the unique characteristics such as storage of kinetic energy to elastic energy, shock absorption to protect mechanical transmission, and force-based feedback control, all of which are favorable for both biological and artificial legged systems. More recently, many researchers have been attempting to enhance the actuation mechanisms with variable stiffness and damping capabilities [75.41, 63] that are also expected to provide valuable insights into the roles of muscle properties in efficient legged locomotion.

75.3.2 Reflexes and Central Pattern Generators

Agility and adaptability of animals' locomotion are not only originated in mechanical dynamics as explained in the previous subsection, but there are also highly complex control systems regulating animals' motions. Usually animals control systems are labeled into four

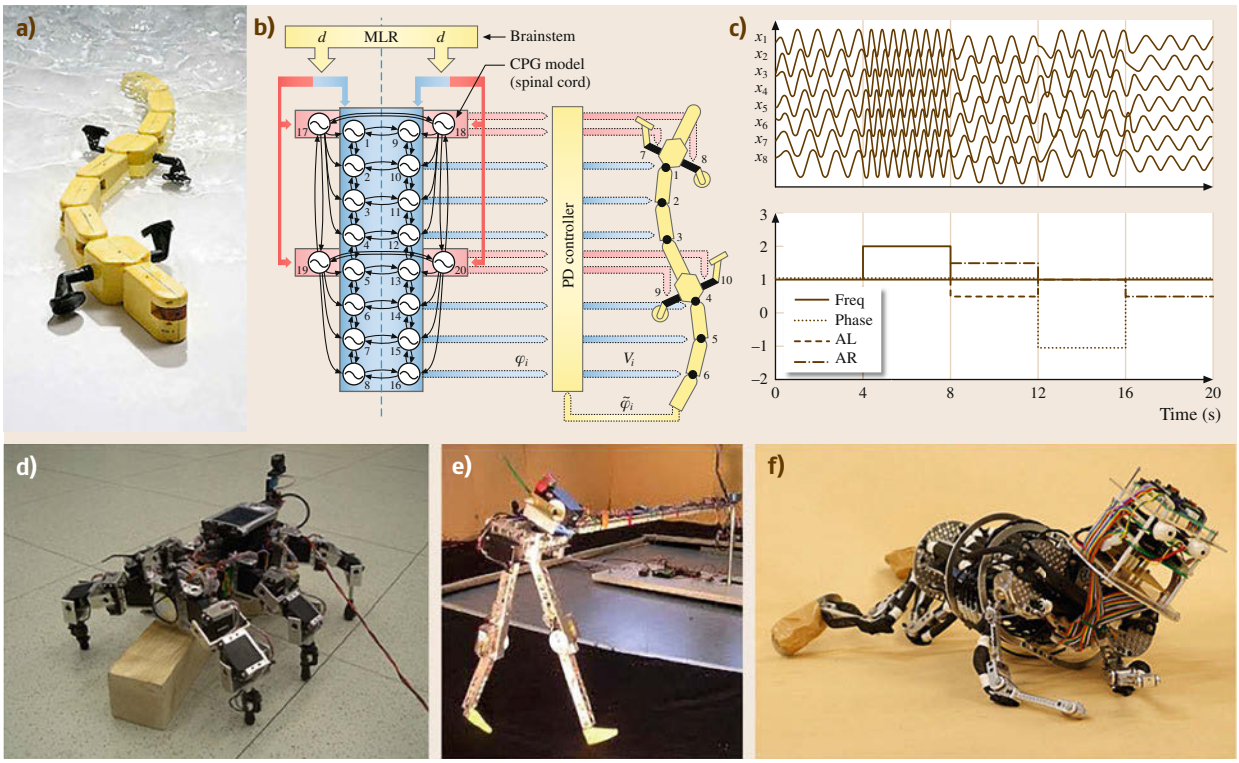


Fig.75.7a–f Robotic implementation of CPG models. **(a)** A salamander robot consisting of motorized modules, and **(b)** its control architecture. Every module has a servomotor which is controlled through a simulated CPG model. **(c)** Typical behavioral response of the CPG model (*upper figure*) with respect to a control signal (*lower figure*). One control input is sufficient to control smooth oscillations of multiple body segments because of coupled dynamics originated in the CPG model (after [75.57]). **(d,e)** A hexapod and biped robots that use CPG models to adapt to changes in the environment (after [75.58–60]). **(f)** A musculoskeletal humanoid robot that simulate developmental processes of human babies (after [75.61])

components: the musculoskeletal system, reflexes, central pattern generators (CPGs), and modulation by higher control centers. Bio-inspired robotics has been investigating these components individually or in combination, and resulted in a number of demonstrations of surprisingly robust robot locomotion (Fig. 75.7).

This research area has been historically investigated by the two distinguished approaches, i. e., the CPG-based approach and the reflex-based one. The former approach usually considers voluntary oscillations of neural circuits, that is, CPGs, and the output of these circuits triggers locomotion cycles. Because these behaviors are found in the spinal cord of vertebrate animals and in ganglions in invertebrates [75.64], this approach has been very popular in the bio-inspired robotics. In contrast, the reflex-based approach does not incorporate intrinsic oscillators and generate periodic behavior as a chain of reflex-mediated events. While conceptually different, most of the CPG-based approach considers reflexes in the neural processes; thus these two approaches are often overlapping and reflex-based approaches can be seen as a subset of CPG-based approaches.

One of the pioneering works in the reflex-based approach was based on the neural circuits underlying walking identified in stick insects [75.65, 66]. In this study, it was identified that a series of reflexes for each leg use information related to leg postures and ground contact to generate movements. Coordination between legs (i. e., specific gaits) is obtained with direct neural couplings between individual legs circuits and also through mechanical couplings (e.g., the movement of one limb affecting the load on other limbs). This leads to a decentralized control mechanism, similar in spirit to those developed in behavior-based robotics, that has been validated on simulated and real hexapod robots [75.65, 67].

The reflex-based approach was also investigated in human walking, in which a series of neuromechanical models were developed to demonstrate how the combination of muscle properties and low-level reflexes can lead to stable locomotion in simulated bipeds [75.68, 69]. In particular, Geyer and Herr [75.69] present a simulated bipedal walker which manages to walk in the sagittal plane and to be stable against slight slopes. This model captures principles of neuromuscular feedbacks and predicts muscle activation patterns observed in leg muscles. Similar reflex-based controllers (without muscle models but with simulated synaptic plasticity) have successfully been ported to real robots such as the Runbot [75.60] and the dynamic walkers reported in [75.37].

One of the first examples of the CPG approach has been done in simulation by *Taga* and colleagues [75.70, 71]. They developed a series of 2-D models of biped locomotion that combine a simple musculoskeletal model with a CPG modeled as a system of coupled *Matsuoka* oscillators [75.72]. The work showed how bidirectional couplings between the CPG and the musculoskeletal model could lead to entrainment (i. e., frequency locked regimes) between the two as well as to robust locomotion. Since then a large number of CPG-based controllers have been implemented in robots for different types of locomotion. Examples include hexapod and octopod robots [75.73–75] (see also Robot Roach VIDEO 112), swimming robots [75.76–78] (see also Salamandra Robotica VIDEO 113), quadruped robots [75.57, 79, 80], and biped robots [75.81–85].

As discussed in [75.64], a CPG-based approach has several interesting properties:

i) Stable limit cycle behavior that provides robustness against perturbations.
ii) Suitability for a distributed implementation.
iii) Possibility to modulate gaits with a few control parameters.
iv) Integration of sensory feedback signals in order to obtain mutual entrainment between the CPG and the mechanical body.
v) Suitable substrate for learning and optimization algorithms. These properties were particularly difficult to investigate by using animals; hence the bio-inspired robotics has provided significant contributions to the nature of motion control in biological systems.

75.3.3 Bio-Inspired Navigation

Biological systems use a variety of cognitive processes for their navigation in complex environment: it is known, for example, that animals use both proprioceptive and exteroceptive receptors for sensing environments; the obtained sensory information is passed to massively parallel processes distributed over many hierarchical levels in central nervous systems; animals' perception of the world is coordinated with low-level sensory-motor processes; and in addition to these mechanisms, long-term planning and learning processes are continuously running to achieve more advanced tasks such as goal-directed navigation. In order to tackle such a complex problem of animals' navigation, bio-inspired robotics also provided a set of effective tools to apply the synthetic methodology some of which are outlined in this subsection.

Sensor Morphology and Sensory–Motor Coordination

A significant number of researchers in bio-inspired robotics have been working on relatively simple animals

such as insects because their central nervous systems are far more tractable than other animals. Despite the simplicity of their brains, insects are incredible navigators being capable of avoiding obstacles while running or flying with enormous speed, recognizing landmarks in unstructured environment, and traveling long distance for foraging. Moreover, some of the social insects can even learn to go back to their own nests, and communicate with co-workers for efficient *community management*. Although biologists have been investigating these fascinating animals for centuries, there are still a number of issues that are not fully clarified yet, and among others, robotics platforms were used to investigate the mechanisms in which physical system–environment interactions play central roles [75.86].

One of the most successful case studies in bio-inspired navigation research was on the mechanisms of sensory-motor coordination in flying insects such as flies and bees. These insects are known to rely on visual sensory information, more specifically, optic flow to detect ego motions, stabilize body posture, measure distance to various objects and landing spots, and track traveling distance, for example. While the visual information processing is usually regarded as computationally expensive, many insects, which have very limited computational resources, make use of this modality to achieve these behavioral functions. The underlying mechanisms are found in the *hardware setups* of animals, in which sensor morphology (i. e., how receptors are distributed) and low-level sensory information processing are exploited in the coordination of sensory-motor processes [75.87]. More specifically, the photoreceptors of these insects are usually distributed to almost all directions which give rise to surprisingly informative stimuli about the environment and ego motions, and low-level neural circuits are configured such that extremely low-processing power is necessary. To test these hypotheses, a number of robotic platforms were developed and tested previously, which showed the feasibility of these mechanisms such as the optic flow to detect nearby objects [75.88], visual odometry [75.89], flight altitude [75.90], and flight stabilization [75.91].

Technological advances are essential to gain additional insights into the complex sensory-motor processes in the animals. At the beginning of the investigations, many researchers developed omni-directional vision based on specifically shaped mirrors attached to regular cameras, while recently more advanced technologies are being developed to flexibly adjust photoreceptors [75.92]. Neuromorphic engineering also provides an additional enabling technology to explore physical foundations of biological nervous systems [75.93, 94]. Neuromorphic silicon retinas are, unlike conventional visual sensors, able to process sensory information extremely fast and computationally less demanding owing to the event driven and asynchronous processing architectures, while keeping sensitivity very high (i. e., the receptors can be sensitive in very dark environments as well as in a very bright one, [75.95]). With the technological progress, we will be able to reproduce more precise landscape of the world from insects' viewpoint for more comprehensive investigation of bio-inspired navigation.

Goal-Directed Navigation

Compared to reflexive behaviors, goal-directed ones are significantly more complex where much less is known even in biology. In nature, goal-directed behaviors such as navigation to a nest from a distant location require learning of routes and locations, short- and long-term memories, episodic memories, while flexibly adapting to unstructured and often dynamically changing environments. The underlying mechanisms are related to many different locations in the central nervous systems and they vary one species to another, thus the ongoing researches are essentially driven on the basis of the important findings in biology, rather than developing a unified and generalized framework.

One of the representative case studies on bio-inspired goal-directed navigation was again conducted in relation to insect behaviors (Fig. 75.8): some social insects such as desert ant Cataglyphis are known to exhibit the so-called *visual homing* behaviors, in which the animals go back to their nests by using visual cues nearby [75.96]. These insects usually walk randomly when searching for food sources, while they go back straight to the nest by using visual cues. Although the neural basis of these behaviors is not yet identified, the biologists argue that an abstract model, the so-called *average landmark vector* method, explain the insects' behavior fairly accurately. Here it is assumed that insects know the global orientation in the world, and every once in a while, they perceive the direction of foraging behavior stored as vector information. Over time, the animals sum up the vectors so that they can keep track on the direction to the nest while randomly search for the food sources. The biologists have been exploring these behaviors for decades, and the accumulated knowledge and hypothetical models were implemented and tested in physical robot platforms. Unlike most of the simulation experiments, the implementation to the robots helped to test the hypothesis in the real-world desert environment [75.96], or physically implemented in an analog circuitry (see also multimedia material of Analog Robot navigation in VIDEO 242; [75.97]).

A significantly more challenging problem is to identify the mechanisms of goal-directed behaviors in more complex animals, especially in mammals. There

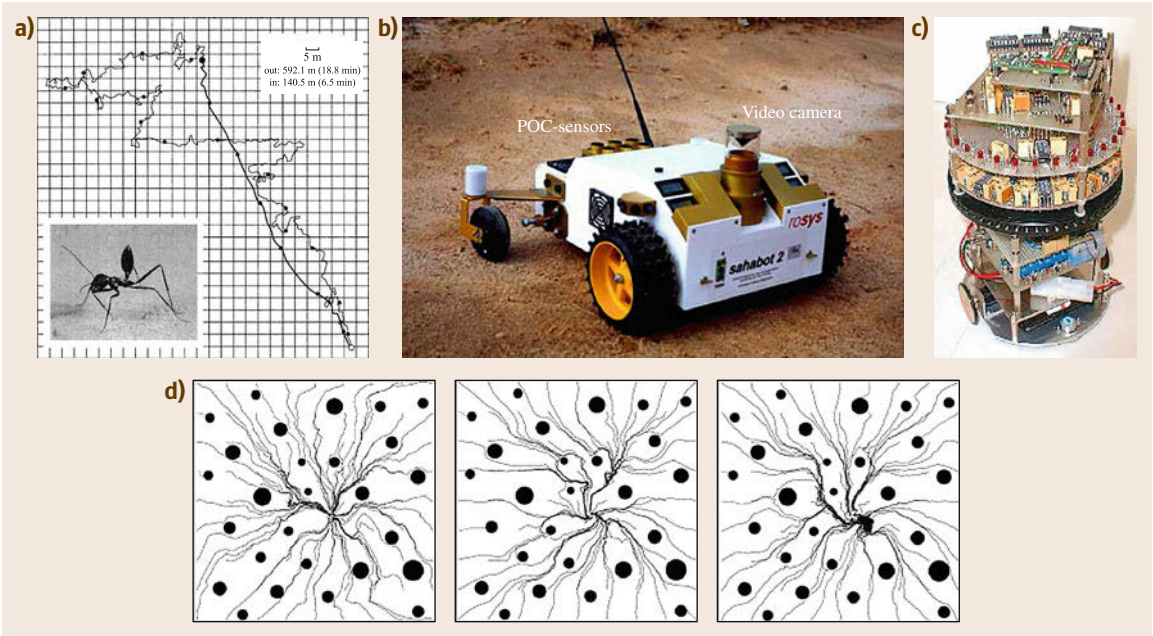


Fig.75.8a–d Visual homing of insect-inspired robots. **(a)** An result of navigation experiment of desert ant Cataglyphis. **(b)** A mobile robot Sahabot II that was developed to investigate navigation mechanisms of Cataglyphis. The robot is equipped with an omni-directional camera and a digital compass that can be used for the bio-inspired landmark navigation (after [75.96]). **(c)** A fully analog implementation of the visual homing algorithm. **(d)** Simulation results of visual homing algorithm (after [75.97])

is a large body of literature about this issue including cognitive science and brain science, but one of the most prominent contributions of robotic platforms in this research area was to explore neural dynamics during physical system–environment interactions. The research was originally motivated by a discovery in physiology such as the so-called place cells, i. e., a group of neurons in hippocampus exhibit unique behaviors whenever the animal is in a specific location in an environment [75.98]. This hard evidence in neuroscience has been widely used to analyze how brains function in the context of spatial cognition and navigation in general, including those investigating computational neuroscience and bio-inspired robotics. Essentially, it is still a challenge to explain the behaviors of place cells because they involve sensory-motor activities as well as temporal changes of neural activities (i. e., learning of sensory motor activities) thus a synthetic methodology is extremely helpful. So far it has been shown that the computational models of hippocampus were implemented onto some mobile robot platforms to replicate the behaviors of place cells in navigation tasks [75.99, 100] as well as some more complex goal-oriented behaviors and learning [75.101].

75.4 Landscape of Bio-Inspired Robotics Research and Challenges

So far we introduced only a few representative and ongoing case studies of bio-inspired robotics research, but there are many other active topic areas in the field. Although many of these studies are covered also in the other chapters of this handbook, this section provides a brief overview of the relevant topics in which robotic technologies are being used as scientific tools for biology.

75.4.1 Bio-Inspired Climbing

When legged systems are reduced down to smaller scales, adhesion forces become more dominant than the gravitational, and for this reason, the small-sized animals in nature such as insects, amphibians, and lizards tend to climb terrains rather than walk on level grounds. While the governing physics in the climbing

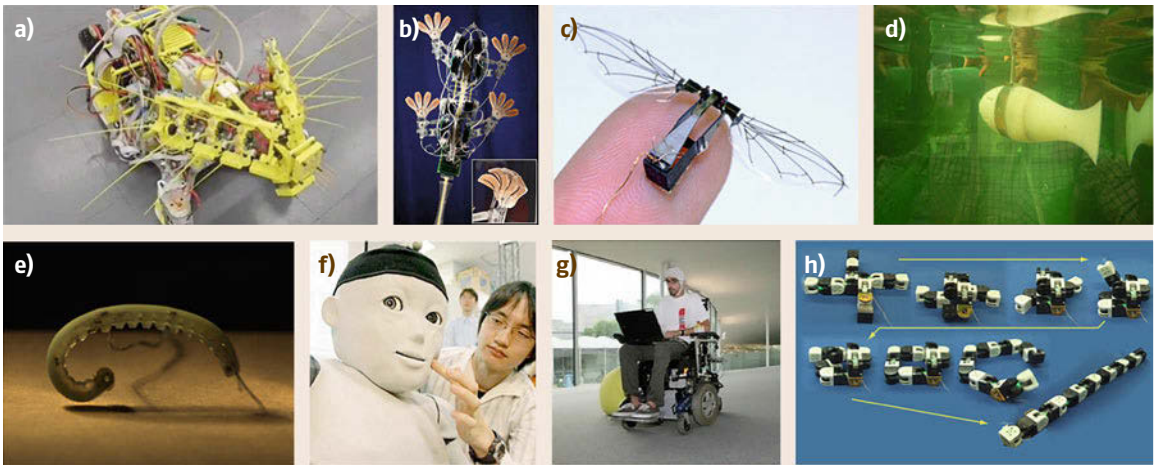


Fig.75.9a–h Examples of recent bio-inspired robots. **(a)** A mobile robot that uses active whisker arrays for navigation (after [75.102]). **(b)** A climbing robot based on feet made of dry adhesives (after [75.103]). **(c)** Micro robot for flapping flight developed by the micro-fabrication techniques (after [75.104]). **(d)** A fish-like swimming robot that exploits soft continuum body structure (after [75.105]). **(e)** A worm-like robot that exhibit rolling-locomotion of soft body structure (after [75.106]). **(f)** A humanoid robot that is equipped with soft skin for interactions with human partners (after [75.107]). **(g)** A wheel-chair controlled by brain signals of the user (after [75.108]). **(h)** Self-reconfigurable robot that is capable of autonomously changing its own body structure (after [75.109])

locomotion is different from that of gravity-oriented legged locomotion, robotic platforms are also useful because the dynamics during the locomotion is similarly complex. One of the most representative case studies in this line of research was the use of dry adhesives in climbing robots that are inspired from geckos. Many research topics focused on the fabrication techniques of micro hair-like structures that can generate adhesive forces for a series of small-sized robots (Fig. 75.9b, [75.103, 110]). Similarly a few other approaches were also proposed to explore the different climbing strategies of animals including the use of material-dependent adhesion [75.111], rough-surface locomotion by using feet with micro spine structures [75.112, 113], and climbing strategy based on force closure of relatively long legs [75.114]. There still exist many challenges in fabrication techniques of micro structures in order to replicate the sophisticated climbing mechanisms of animals, which requires continuing close collaborations between researchers in biology and robotics.

75.4.2 Flapping Flight and Swimming Mechanisms

Another complex, yet popular, dynamics used in animal kingdom is fluid/aerodynamics that are typically observed in flying and swimming systems. Fluid/aerodynamics are also dynamics difficult to model and simulate thus robotic platforms are intensively employed for exploring underlying mechanisms [75.115, 116]. As is the case of walking and running on land, mechanical dynamics also play an important role in flying and swimming locomotion and a number of underactuated robots were developed to understand the nature of locomotion in fluid Fig. 75.9d [75.105, 117]. As the microfabrication techniques evolved in the recent years, roboticists and biologists also started collaborating to investigate small-sized flying robots (Fig. 75.9c, [75.104]; Chap. 26).

75.4.3 Artificial Hands, Haptics, and Whiskers

Haptic perception is known to be one of the most important sensor modalities in biological systems, although the biological nature is far from a comprehensive understanding because animals make use of complex sensory-motor interactions for the purpose of tactile sensing [75.118, 119]. Haptic sensing can be defined as sensing of mechanical environment through *touch* although there are many different variations in nature including tactile sensing through fingers and skins [75.119], active whisking (Fig. 75.9a) [75.102], or more specifically targeted sensing such as slippage detection [75.120]. Exploration on haptic technologies is also crucial in this research area as the biological tactile sensing involves an enormous number of mechanoreceptors each of which has a large sensitivity range. Currently, a number of researchers are actively inves-

tigating technological solutions through haptic devices and soft and stretchable electronics for tactile sensing [75.121, 122].

75.4.4 Self-Reconfigurable and Evolutionary Robotics

Animals' adaptivity in nature relies significantly on their capability of changing their body sizes and structures. Animals are, for example, able to start their lives smaller and gradually grow larger and more complex; they are able to self-repair or regenerate when encountering failures in body parts; and muscles and skins are able to strengthen themselves if necessary [75.123]. Synthetic methodology has also been employed to investigate these fascinating capabilities of biological systems by using, for example, modular robots Fig. 75.9h [75.109, 124, 125], redundant body structures for snake-like motion control [75.126, 127], and self-repair and self-assembly of structures [75.128, 129]. Due to the technological limitations, many researchers take advantage of simulation-based methods to explore the ontogenetic and evolutionary processes to uncover the characteristics of optimization strategies in nature [75.19, 20, 130].

75.4.5 Bio-Inspired Soft Robots

Unlike the conventional robots that are usually made of rigid materials articulated into discrete pieces, animals' body structures mostly consist of soft, continuum, and elastic components such as muscles, tendons, skins, organs surrounded by smooth membrane [75.1]. Recently, there has been an increasing interest in the use of soft deformable materials in robotic systems to enhance capabilities of, for example, soft locomotion [75.106, 131] (VIDEO 109 shows and example of soft robot locomotion), manipulation [75.132, 133], shape adaptation [75.134], and soft human–robot interactions [75.63]. Despite its demand in the robotics and biological studies, there are still a number of technological challenges in this field such as soft actuation and sensing [75.135], simulation of soft deformable structures [75.136], and control of flexible continuum bodies [75.137].

75.4.6 Neuroprosthetics and Social Interactions

As we develop more technological components compatible to biological systems, there are more possibilities to implant artificial devices into biological bodies. Although most of the case studies in this research area aim at bio-medical applications as exemplified by visual/auditory prosthetics, pain relief, and motor prosthetics, there are also intensive investigations on the use of prosthetic devices to gain additional insights into the nature of motion control [75.108, 138] and perception [75.139]. There is also an increasing interest in the use of robotic platforms in the studies of social interactions where robots are used to study communications with humans [75.107] and the other animals [75.140].

75.5 Conclusion

This chapter introduced a class of bio-inspired robotics research that is specifically targeted to deepen our understanding of biological systems. Through the representative case studies, we explained how bio-inspired robotics research can be useful not only for developing innovative robots, but also for exploring uncovered challenges in biology by employing the understanding-by-building approach. There are, however, a set of important concepts that need to be considered for successful collaborations between robotics and biology. Specifically:

1. It is necessary to take the similarities and differences in robots and animals into account.
2. There are different goals and methods to develop models of biological systems.
3. The use of robots as a scientific tool has both advantages and disadvantages.
4. There are types of hypotheses in biology that bio-inspired robotics can be particularly beneficial for.

And finally, we also introduced a concise landscape of trends and challenges in bio-inspired robotics. As mentioned earlier, the field of bio-inspired robotics is very broad, and the outline introduced in this chapter is by no means complete. For example, although this chapter only focused on the types of research which contribute to biological studies, there is a large body of the literature on *bio-inspired robotic applications* which are mostly ignored in this chapter. The interested readers should refer to the other chapters in this handbook as well as the other review articles in the field. Also, there are significantly more case studies available in literature which reported on the different species or the other aspects of animals which were summarized in [75.15, 18, 64].

Video-References

VIDEO 109 Dynamic rolling locomotion of GoQBot
available from http://handbookofrobotics.org/view-chapter/75/videodetails/109

VIDEO 110 JenaWalker – Biped robot with biologically-inspired bi-articular springs
available from http://handbookofrobotics.org/view-chapter/75/videodetails/110

VIDEO 111 MIT Compass Gait Robot – Locomotion over rough terrain
available from http://handbookofrobotics.org/view-chapter/75/videodetails/111

VIDEO 112 RobotRoach with adaptive gait pattern variations
available from http://handbookofrobotics.org/view-chapter/75/videodetails/112

VIDEO 113 Salamandra Robotica II – Swimming to walking transition
available from http://handbookofrobotics.org/view-chapter/75/videodetails/113

VIDEO 242 Analog Robot
available from http://handbookofrobotics.org/view-chapter/75/videodetails/242

References

75.1 R. Pfeifert, M. Lungarella, F. Iida: The challenges ahead for bio-inspired 'soft' robotics, Communications ACM **55**(11), 76–87 (2012)

75.2 G. Becky: *Autonomous Robots* (MIT Press, Cambridge 2001)

75.3 W.R. Ashby: *An Introduction to Cybernetics* (Chapman Hall, London 1956)

75.4 N. Wiener: *Cybernetics, or Communication and Control in the Animal and the Machine* (MIT Press, Cambridge 1948)

75.5 O.E. Holland: Grey Walter: The pioneer of real artificial life, Proc. 5th Int. Workshop Artif. Life, ed. by C. Langton (MIT Press, Cambridge 1997) pp. 34–44

75.6 V. Braitenberg: *Vehicles: Experiments in Synthetic Psychology* (MIT Press, Cambridge 1984)

75.7 J. von Neumann: *The Theory of Self-Reproducing Automata* (Univ. of Illinois Press, Illinois 1966)

75.8 A.M. Turing: The chemical basis of morphogenesis, Philos. Trans. R. Soc. **237**(641), 37–72 (1952)

75.9 N. Nilsson: The physical symbol system hypothesis: Status and prospects, Lect. Notes Artif. Intell. **4850**, 9–17 (2007)

75.10 R. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1998)

75.11 R.A. Brooks: A robust layered control system for a mobile robot, IEEE Robot. Autom. **2**(1), 14–23 (1986)

75.12 R.A. Brooks: Intelligence without representation, Artif. Intell. **47**, 139–159 (1991)

75.13 R. Pfeifer, C. Scheier: *Understanding Intelligence* (MIT Press, Cambridge 2001)

75.14 M. Lungarella, G. Metta, R. Pfeifer, G. Sandini: Developmental robotics: A survey, Connect. Sci. **15**, 151–190 (2003)

75.15 R. Pfeifer, M. Lungarella, F. Iida: Self-organization, embodiment, and biologically inspired robotics, Science **318**, 1088–1093 (2007)

75.16 T.D. Johnston, L. Edwards: Genes, interactions, and the development of behavior, Psychol. Rev. **109**(1), 26–34 (2002)

75.17 R.J. Full, D.K. Koditschek: Templates and anchors: Neuromechanical hypotheses of legged locomotion on land, J. Exp. Biol. **202**, 3325–3332 (1999)

75.18 B. Webb: Can robots make good models of biological behaviour?, Behav. Brain Sci. **24**, 1033–1050 (2001)

75.19 K. Sims: Evolving 3D morphology and behavior by competition. In: *Proc. Artificial Life IV*, ed. by R. Brooks, P. Maes (MIT Press, Cambridge 1994) pp. 28–39

75.20 J. Bongard: Morphological change in machines accelerates the evolution of robust behavior, Proc. Natl. Acad. Sci. USA **108**(4), 1234–1239 (2011)

75.21 C.G. Langton: *Artificial Life: An Overview* (MIT Press, Cambridge 1995)

75.22 A. Adamatzky, M. Komosinski: *Artificial Life Models in Hardware* (Springer, Berlin, Heidelberg 2009)

75.23 H. Lipson, J. Pollack: Automatic design and manufacture of robotic lifeforms, Nature **406**, 974–978 (2000)

75.24 G. Borelli: *De Motu Animalium* (Apud Petrum vander Aa, Cornelium Boutesteyn, Johannem de Vivie, Danielem à Gaesbeeck, Lugduni in Batavis 1685)

75.25 H. Miura, I. Shimoyama: Dynamic walk of a biped, Int. J. Robotics Res. **3**(2), 60–74 (1984)

75.26 M. Vukobratovic, B. Borovac: Zero-moment point – Thirty years of its life, Int. J. Humanoid Robotics **1**(1), 157–174 (2004)

75.27 M. Raibert: *Legged Robots that Balance* (MIT Press, Cambridge 1987)

75.28 R. Blickhan, A. Seyfarth, H. Geyer, S. Grimmer, H. Wagner, M. Guenther: Intelligence by mechanics, Philos. Trans. R. Soc. A **365**(1850), 199–220 (2007)

75.29 I.E. Brown, G.E. Loeb: A reductionist approach to creating and using neuromusculoskeletal models. In: *Biomechanical and Neurological Control of Posture and Movements*, ed. by P.E. Winters (Springer, New York 2000) pp. 148–163

75.30 T.A. McMahon: *Muscles, Reflexes, and Locomotion* (Princeton Univ. Press, Princeton 1984)

75.31 S. Yakovenko, V. Gritsenko, A. Prochazka: Contribution of stretch reflexes to locomotor control: a modeling study, Biol. Cybern. **90**(2), 146–155 (2004)

75.32 G. Orlovsky, G.N. Orlovskii, S. Grillner: *Neuronal Control of Locomotion From Mollusc to Man* (Oxford Univ. Press, Oxford 1999)

75.33 M. Bear, M. Paradiso, B.W. Connors: *Neuroscience: Exploring the Brain* (Lippincott Williams and Wilkins, Baltimore 2006)

75.34 F. Asano, M. Suguro: Limit cycle walking, running, and skipping of telescopic-legged rimless wheel, Robotica **30**(6), 989–1003 (2012)

75.35 F. Iida, R. Tedrake: Minimalistic control of biped walking in rough terrain, Auton. Robots **28**(3), 355–368 (2010)

75.36 I.R. Manchester, U. Mettin, F. Iida, R. Tedrake: Stable dynamic walking over uneven terrain, Int. J. Robotics Res. **30**(3), 265–279 (2011)

75.37 S.H. Collins, A. Ruina, R. Tedrake, M. Wisse: Efficient bipedal robots based on passive-dynamic walkers, Science **307**, 1082–1085 (2005)

75.38 D. Owaki, M. Koyama, S. Yamaguchi, S. Kubo, A. Ishiguro: A 2-D passive dynamic running biped with elastic elements, IEEE Trans. Robotics **27**(1), 156–162 (2011)

75.39 M. Ahmadi, M. Buehler: Controlled passive dynamic running experiments with ARL monopod II, IEEE Trans. Robotics **22**(5), 974–986 (2006)

75.40 F. Iida, J. Rummel, A. Seyfarth: Bipedal walking and running with spring-like biarticular muscles, J. Biomech. **41**, 656–667 (2008)

75.41 J.W. Hurst, A.A. Rizzi: Series compliance for an efficient running gait: Lessons learned from the electric cable differential leg, IEEE Robotics Autom, Mag. **15**(3), 42–51 (2008)

75.42 R. Altendorfer, N. Moore, H. Komsuoglu, H.B. Brown Jr., D. McMordie, U. Saranli, R. Full, D.E. Koditschek: RHex: A biologically inspired hexapod runner, Auton. Robots **11**, 207–213 (2001)

75.43 T. McGeer: Passive dynamic walking, Int. J. Robotics Res. **9**(2), 62–82 (1990)

75.44 A. Goswami, B. Thuilot, B. Espiau: A study of the passive gait of a compass-like biped robot: Symmetry and chaos, Int. J. Robotics Res. **17**(12), 1282–1301 (1998)

75.45 D.G.E. Hobbelen, M. Wisse: Swing-leg retraction for limit cycle walkers improves disturbance rejection, IEEE Trans. Robotics **24**(2), 377–389 (2008)

75.46 J. Hass, J.M. Herrmann, T. Geisel: Optimal mass distrib- ution for passivity-based bipedal robots, Int. J. Robotics Res. **25**(11), 1087–1098 (2006)

75.47 K. Byl, R. Tedrake: Metastable walking machines, Int. J. Robotics Res. **28**(8), 1040–1064 (2008)

75.48 R.D. Gregg, M.W. Spong: Reduction-based control of three-dimensional bipedal walking robots, Int. J. Robotics Res. **29**(6), 680–702 (2010)

75.49 R.M. Alexander: Three uses for springs in legged locomotion, Int. J. Robotics Res. **9**(2), 53–61 (1990)

75.50 X. Zhou, S. Bi: A survey of bio-inspired compliant legged robot designs, Bioinspir. Biomim. **7**(4), 1–20 (2012)

75.51 J. Rummel, A. Seyfarth: Stable running with segmented legs, Int. J. Robotics Res. **27**(8), 919–934 (2008)

75.52 T.M. Kubow, R.J. Full: The role of the mechanicalvsystem in control: A hypothesis of self-stabilization in hexapedal runners, Philos. Trans. R. Soc. B **354**, 849–862 (1999)

75.53 H. Geyer, A. Seyfarth, R. Blickhan: Compliant leg behaviour explains basic dynamics of walking and running, Proc. R. Soc. B **273**, 1471–2954 (2006)

75.54 M. Reis, F. Iida: An energy efficient hopping robot based on free vibration of a curved beam, IEEE/ASME Trans. Mechatron. **19**(1), 300–311 (2013)

75.55 A.M. Grabowski, H.M. Herr: Leg exoskeleton reduces the metabolic cost of human hopping, J. Appl. Physiol. **107**, 670–678 (2009)

75.56 A.D. Kuo: Choosing your steps carefully: Trade-offs between economy and versatility in dynamic walking bipedal robots, IEEE Robotics Autom. Mag. **14**, 18–29 (2007)

75.57 A.J. Ijspeert, A. Crespi, D. Ryczko, J.M. Cabelguen: From swimming to walking with a salamander robot driven by a spinal cord model, Science **315**(5817), 1416–1420 (2007)

75.58 S. Steingrube, M. Timme, F. Wörgötter, P. Manoonpong: Self-organized adaptation of simple neural circuits enables complex robot behavior, Nat. Phys. **6**, 224–230 (2010)

75.59 T. Geng, B. Porr, F. Wörgötter: Fast biped walking with a sensor-driven neuronal controller and real-time online learning, Int. J. Robotics Res. **25**(3), 243–259 (2006)

75.60 P. Manoonpong, T. Geng, T. Kulvicius, B. Porr, F. Wörgötter: Adaptive, fast walking in a biped robot under neuronal control and learning, PLoS Comput. Biol. **3**(7), 1305–1320 (2007)

75.61 K. Narioka, K. Hosoda: Motor development of an pneumatic musculoskeletal infant robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2011) pp. 963–968

75.62 G.A. Pratt, M.M. Williamson: Series elastic actuators, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (1995) pp. 399–406

75.63 A. Albu-Schaffer, O. Eiberger, M. Grebenstein, S. Haddadin, C. Ott, T. Wimbock, S. Wolf, G. Hirzinger: Soft robotics, IEEE Robotics Autom, Mag. **15**(3), 20–30 (2008)

75.64 A.J. Ijspeert: Central pattern generators for locomotion control in animals and robots: A review, Neural Netw. **21**(4), 642–653 (2008)

75.65 H. Cruse, T. Kindermann, M. Schumm, J. Dean, J. Schmitz: Walknet: A biologically inspired network to control six- legged walking, Neural Netw. **11**, 1435–1447 (1998)

75.66 V. Dürr, A.F. Krause, J. Schmitz, H. Cruse: Neuroethological concepts and their transfer to walking machines, Int. J. Robotics Res. **22**(3/4), 151–167 (2003)

75.67 K.S. Espenschied, R.D. Quinn, H.J. Chiel, R.D. Beer: Biologically-based distributed control and local reflexes improve rough terrain locomotion in a hexapod robot, Robotics Auton. Syst. **18**, 59–64 (1996)

75.68 H. Geyer, A. Seyfarth, R. Blickhan: Positive force feedback in bouncing gaits, Proc. R. Soc. B **270**, 2173–2183 (2003)

75.69 H. Geyer, H.M. Herr: A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities, IEEE Trans. Neural. Syst. Rehabil. Eng. **18**(3), 263–273 (2010)

75.70 G. Taga, Y. Yamaguchi, H. Shimizu: Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment, Biol. Cybern. **65**, 147–159 (1991)

75.71 G. Taga: A model of the neuro-musculo-skeletal system for anticipatory adjustment of human locomotion during obstacle avoidance, Biol. Cybern. **78**(1), 9–17 (1998)

75.72 K. Matsuoka: Mechanisms of frequency and pattern control in the neural rhythm generators, Biol. Cybern. **56**, 345–353 (1987)

75.73 P. Arena, L. Fortuna, M. Frasca, G. Sicurella: An adaptive, self-organizing dynamical system for hierarchical control of bio-inspired locomotion, IEEE Trans. Syst. Man Cybern. B **34**(4), 1823–1837 (2004)

75.74 S. Inagaki, H. Yuasa, T. Suzuki, T. Arai: Wave CPG model for autonomous decentralized multi-legged robot: Gait generation and walking speed control, Robotics Auton. Syst. **54**(2), 118–126 (2006)

75.75 B. Klaassen, R. Linnemann, D. Spenneberg, F. Kirchner: Biomimetic walking robot scorpion: Control and modeling, Auton. Robots **41**, 69–76 (2002)

75.76 A. Crespi, A. Ijspeert: Online optimization of swimming and crawling in an amphibious snake robot, IEEE Trans. Robotics **24**(1), 75–87 (2008)

75.77 C. Stefanini, G. Orlandi, A. Menciassi, Y. Ravier, G.L. Spina, S. Grillner: A mechanism for biomimetic actuation in lamprey-like robots, Proc. 1st IEEE/RAS-EMBS Int. Conf. Biomed. Robotics Biomechatron. (2006) pp. 579–584

75.78 C. Wilbur, W. Vorus, Y. Cao, S.N. Currie: A lamprey-based undulatory vehicle. In: *Neurotechnology for Biomimetic Robots*, ed. by J. Ayers, J.L. Davis, A. Rudolph (MIT Press, Cambridge 2002)

75.79 H. Kimura, S. Akiyama, K. Sakurama: Realization of dynamic walking and running of the quadruped using neural oscillators, Auton. Robots **7**(3), 247–258 (1999)

75.80 H. Kimura, Y. Fukuoka, A.H. Cohen: Adaptive dynamic walking of a quadruped robot on natural ground based on biological concepts, Int. J. Robotics Res. **26**(5), 475–490 (2007)

75.81 J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, M. Kawato: Learning from demonstration and adaptation of biped locomotion, Robotics Auton. Syst. **47**, 79–91 (2004)

75.82 S. Aoi, K. Tsuchiya: Locomotion control of a biped robot using nonlinear oscillators, Auton. Robots **19**, 219–232 (2005)

75.83 G. Endo, J. Nakanishi, J. Morimoto, G. Cheng: Experimental studies of a neural oscillator for biped locomotion with QRIO, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2005) pp. 598–604

75.84 L. Righetti, A.J. Ijspeert: Programmable central pattern generators: An application to biped locomotion control, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2006) pp. 1585–1590

75.85 R. Héliot, B. Espiau: Multisensor input for CPG-based sensorymotor coordination, IEEE Trans. Robotics **24**(1), 191–195 (2008)

75.86 D. Floreano, J.C. Zufferey, M.V. Srinivasan, C. Ellington (Eds.): *Flying Insects and Robots* (Springer, Berlin, Heidelberg 2009)

75.87 N. Franceschini, J.M. Pichon, C. Blanes: From insect vision to robot vision, Philos. Trans. R. Soc. B **337**, 283–294 (1992)

75.88 J.C. Zufferey, D. Floreano: Fly-inspired visual steering of an ultralight indoor aircraft, IEEE Trans. Robotics **22**(1), 137–146 (2006)

75.89 F. Iida: Biologically inspired visual odometer for navigation of a flying robot, Robotics Auton. Syst. **44**(3/4), 201–208 (2003)

75.90 M.V. Srinivasan, S.W. Zhang, J.S. Chahl, E. Barth, S. Venkatesh: How honeybees make grazing landings on flat surfaces, Biol. Cybern. **83**, 171–183 (2000)

75.91 S. Viollet, N. Franceschini: Aerial minirobot that stabilizes and tracks with a bio-inspired visual scanning sensor. In: *Biorobotics. Methods and Applications*, ed. by B. Webb, T. Consi (MIT Press, Cambridge 2001) pp. 67–83

75.92 M. Dobrzynski, R.P. Camara, D. Floreano: Vision Tape –A flexible compound vision sensor for motion detection and proximity estimation, IEEE Sens. J. **12**(5), 1131–1139 (2012)

75.93 C. Eliasmith, C.H. Anderson: *Neural engineering: Computation, Representation, and Dynamics in Neurobiological Systems* (MIT Press, Cambridge 2004)

75.94 S.C. Liu, T. Delbruck: Neuromorphic sensory systems, Curr. Opin. Neurobiol. **20**, 1–8 (2010)

75.95 R. Berner, T. Delbruck: Event-based pixel sensitive to changes of color and brightness, IEEE Trans. Circuits Syst. I **58**(7), 1581–1590 (2011)

75.96 D. Lambrinos, R. Möller, T. Labhart, R. Pfeifer, R. Wehner: A mobile robot employing insect strategies for navigation, Robotics Auton. Syst. **30**(1/2), 39–64 (2000)

75.97 R. Möller: Insect visual homing strategies in a robot with analog processing, Biol. Cybern. **83**(3), 231–243 (2000)

75.98 E. Moser, E. Kropff, M. Moser: Place cells, grid cells, and the brain's spatial representation system, Annu. Rev. Neurosci. **31**, 69–89 (2008)

75.99 J.L. Krichmar, D.A. Nitz, J.A. Gally, G.M. Edelman: Characterizing functional hippocampal pathways in a brain-based device as it solves a spatial memory task, Proc. Natl. Acad. Sci. USA **102**, 2111–2116 (2005)

75.100 M.J. Milford, J. Wiles, G.F. Wyeth: Solving navigational uncertainty using grid cells on robots, PLoS Comput. Biol. **6**(11), e1000995 (2010)

75.101 G. Edelman: Learning in and from brain-based devices, Science **318**, 1103–1105 (2007)

75.102 T.J. Prescott, M.J. Pearson, B. Mitchinson, T. Pipe: Whisking with robots: From rat vibrissae to biomimetic technology for active touch, IEEE Robotics Autom. Mag. **16**, 42–50 (2009)

75.103 S. Kim, A. Asbeck, M. Cutkosky, W. Provancher: SpinybotII: Climbing hard walls with compliant microspines, Proc. 12th Int. Conf. Adv. Robotics (ICAR) (2005) pp. 601–606

75.104 R.J. Wood: The first biologically inspired at-scale robotic insect, IEEE Trans. Robotics **24**(2), 341–347 (2008)

75.105 T. Salumäe, M. Kruusmaa: A flexible fin with bio-inspired stiffness profile and geometry, J. Bionic Eng. **8**(4), 418–428 (2011)

75.106 H.T. Lin, G.G. Leisk, B. Trimmer: GoQBot: A caterpillar-inspired soft-bodied rolling robot, Bioinspir. Biomim. **6**, 026007 (2011)

75.107 M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, C. Yoshida: Cognitive developmental robotics: A survey, IEEE Trans. Auton. Mental Dev. **1**(1), 12–34 (2009)

75.108 T. Carlson, J.D.R. Mill: Brain-controlled wheelchairs: A robotic architecture, IEEE Robotics Autom, Mag. **20**(1), 65–73 (2013)

75.109 S. Murata, H. Kurokawa: Self-reconfigurable robots, IEEE Robotics Autom, Mag. **14**(1), 71–78 (2007)

75.110 M.R. Cutkosky, S. Kim: Design and fabrication of multi-material structures for bioinspired robots, Philos. Trans. R. Soc. **367**, 1799–1813 (2009)

75.111 O. Unver, M. Sitti: Tankbot: A palm-size, tank like climbing robot on rough and smooth surfaces, Int. J. Robotics Res. **29**(14), 1761–1777 (2010)

75.112 A.T. Asbeck, S. Kim, M.R. Cutkosky, W.R. Provancher, M. Lanzetta: Scaling hard vertical surfaces with compliant microspine arrays, Int. J. Robotics Res. **25**(12), 1165–1179 (2006)

75.113 M.J. Spenko, G.C. Haynes, J.A. Saunders, M.R. Cutkosky, A.A. Rizzi, R.J. Full, D.E. Koditschek: Biologically inspired climbing with a hexapedal robot, J. Field Robotics **25**(4/5), 223–242 (2008)

75.114 T. Lam, Y. Xu: Climbing strategy for a flexible tree climbing robot Treebot, IEEE Trans. Robotics **27**(6), 1107–1117 (2011)

75.115 M.S. Triantafyllou, G.S. Triantafyllou: An efficient swimming machine, Sci. Am. **272**(3), 40–48 (1995)

75.116 M.H. Dickinson, C.T. Farley, R.J. Full, M.A. Koehl, R. Kram, S. Lehman: How animals move: An integrative view, Science **288**(5463), 100–106 (2000)

75.117 G.V. Lauder, E.J. Anderson, J. Tangorra, P.G. Madden: Fish biorobotics: Kinematics and hydrodynamics of self-propulsion, J. Exp. Biol. **210**(16), 2767–2780 (2007)

75.118 R. Fearing, J. Hollerbach: Basic solid mechanics for tactile sensing, Int. J. Robotics Res. **1**, 266–275 (1984)

75.119 V. Hayward, O.R. Astley, M. Cruz-Hernandez, D. Grant, G. Robles-De-La-Torre: Haptic interfaces and devices, Sens. Rev. **24**(1), 16–29 (2004)

75.120 D. Gunji, Y. Mizoguchi, S. Teshigawara, A. Ming, A. Namiki, M. Ishikawaand, M. Shimojo: Grasping force control of multi-fingered robot hand based on slip detection using tactile sensor, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2008) pp. 2605–2610

75.121 T. Someya, Y. Kato, T. Sekitani, S. Iba, Y. Noguchi, Y. Murase, H. Kawaguchi, T. Sakurai: Conformable, flexible, large-area networks of pressure and thermal sensors with organic transistor active matrixes, Proc. Natl. Acad. Sci. USA **102**, 12321–12325 (2005)

75.122 G. Cannata, M. Maggiali, G. Metta, G. Sandini: An embedded artificial skin for humanoid robots, Int. Conf. Multi-Sensor Fusion Integr. (2008) pp. 434–438

75.123 S.F. Gilbert: *Developmental Biology* (Palgrave Macmillan, London 2010)

75.124 F. Hara, R. Pfeifer (Eds.): *Morpho-Functional Machines: The New Species: Designing Embodied Intelligence* (Springer, Berlin, Heidelberg 2003)

75.125 M. Yim, W.M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, G.S. Chirikjian: Modular self-reconfigurable robot systems: Challenges and opportunities for the future, IEEE Robotics Autom, Mag. **14**(1), 43–52 (2007)

75.126 S. Hirose: *Biologically inspired robots: Snake-like locomotors and manipulators* (Oxford Univ. Press, Oxford 1993)

75.127 T. Kano, T. Sato, R. Kobayashi, A. Ishiguro: Local reflexive mechanisms essential for snakes' scaffold-based locomotion, Bioinspir. Biomim. **7**, 046008 (2012)

75.128 P.J. White, K. Kopanski, H. Lipson: Stochastic self-reconfigurable cellular robotics, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2004) pp. 2888–2893

75.129 E. Klavins, R. Ghrist, D. Lipsky: A grammatical approach to self-organizing robotic systems, IEEE Trans. Autom. Control **51**(6), 949–962 (2006)

75.130 S. Nolfi, D. Floreano: *Evolutionary Robotics. The Biology, Intelligence, and Technology of Self-Organizing Machines* (MIT Press, Cambridge 2000)

75.131 S. Seok, C.D. Onal, K.J. Cho, R.J. Wood, D. Rus, S. Kim: Meshworm: A peristaltic soft robot with antagonistic nickel titanium coil actuators, IEEE/ASME Trans. Mechatron. **18**, 1485–1497 (2013)

75.132 D. Trivedi, C.D. Rahn, W.M. Kier, I.D. Walker: Soft robotics: Biological inspiration, state of the art, and future research, Appl. Bionics Biomech. **5**(3), 99–117 (2008)

75.133 F. Ilievski, A.D. Mazzeo, R.F. Shepherd, X. Chen, G.M. Whitesides: Soft robotics for chemists, Angew. Chem. Int. Ed. **50**, 1890–1895 (2011)

75.134 L. Brodbeck, F. Iida: Enhanced robotic body extension with modular units, Proc. 2012 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) (2012) pp. 1428–1433

75.135 A.T. Conn, J.M. Rossiter: Towards holonomic electro-elastomer actuators with six degrees of

freedom, Smart Mater. Struct. **21**, 035012–035020 (2012)

75.136 J. Rieffel, D. Knox, S. Smith, B.A. Trimmer: Growing and evolving soft robots, Artif. Life **20**(1), 143–162 (2014)

75.137 C. Laschi, B. Mazzolai, V. Mattoli, M. Cianchetti, P. Dario: Design of a biomimetic robotic octopus arm, Bioinspir. Biomim. **4**, 1–8 (2009)

75.138 A.B. Schwartz, X.T. Cui, D.J. Weber, D.W. Moran: Brain-controlled interfaces: Movement restoration with neural prosthetics, Neuron **52**(1), 205–220 (2006)

75.139 J.E. O'Doherty, M.A. Lebedev, K.Z. Zhuang, S. Shokur, H. Bleuler, M.A.L. Nicolelis: Active tactile exploration enabled by a brain-machine-brain interface, Nature **479**, 228–231 (2011)

75.140 J. Halloy, G. Sempo, G. Caprari, C. Rivault, M. Asadpour, F. Tache, I. Said, V. Durier, S. Canonge, J.M. Ame, C. Detrain, N. Correll, A. Martinoli, F. Mondada, R. Siegwart, J.L. Deneubourg: Social integration of robots into groups of cockroaches to control self-organized choices, Science **318**(5853), 1155–1158 (2007)

Multimedia Contents

76. Evolutionary Robotics

Stefano Nolfi, Josh Bongard, Phil Husbands, Dario Floreano

Evolutionary Robotics is a method for automatically generating artificial brains and morphologies of autonomous robots. This approach is useful both for investigating the design space of robotic applications and for testing scientific hypotheses of biological mechanisms and processes. In this chapter we provide an overview of methods and results of Evolutionary Robotics with robots of different shapes, dimensions, and operation features. We consider both simulated and physical robots with special consideration to the transfer between the two worlds.

76.1 **Method** 2036
76.2 **First Steps** 2036
76.2.1 Evolution of Neural Controllers for Walking 2038
76.3 **Simulation and Reality** 2040
76.4 **Behavior as a Complex Adaptive System** 2041
76.4.1 Behavior Recombination and Re-Use 2043
76.4.2 Sensory-Motor Coordination 2043
76.5 **Evolving Bodies** 2044
76.5.1 Co-Evolving Body and Brains 2044
76.5.2 Self-Modeling 2045
76.6 **Seeing the Light** 2046
76.6.1 Coevolution of Active Vision and Feature Selection 2047
76.7 **Computational Neuroethology** 2049
76.7.1 Emergence of Place Cells 2049
76.7.2 Spiking Neurons 2051
76.7.3 GasNets 2052
76.8 **Evolution and Learning** 2054
76.8.1 Learning to Adapt to Fast Environmental Variations 2055
76.8.2 Evolution of Learning 2056
76.9 **Evolution of Social Behavior** 2057
76.9.1 Coevolving Predator and Prey Robots 2057
76.9.2 Evolving Cooperative Behavior 2058
76.9.3 Evolution of Communication 2059
76.10 **Evolutionary Hardware** 2060
76.10.1 Evolvable Hardware Robot Controllers 2060
76.11 **Closing Remarks** 2061
Video-References 2061
References 2062

Evolutionary robotics is a method for the automatic creation of autonomous robots [76.1]. It is inspired by the Darwinian principle of selective reproduction of the fittest, captured by evolutionary algorithms [76.2]. In evolutionary robotics, robots are considered as autonomous artificial organisms that develop their own control system and body configuration in close interaction with the environment without human intervention. Drawing inspiration from principles of biological self-organization, evolutionary robotics includes elements of evolutionary, neural, developmental, and morphological systems. The idea that an evolutionary process could drive the generation of control systems dates back to at least the 1950s [76.3] with a more explicit form appearing in the mid 1980s with the ingenious thought experiments by neuroscientist *Valentino Braitenberg* on neurally driven vehicles [76.4]. In the early 1990s, the first generation of simulated artificial organisms with a genetic code describing the neural circuitry and morphology of a sensory motor system began evolv-

ing on computer screens [76.5–8]. At that time, real robots were still complicated and expensive machines that required specialized programming techniques and skillful manipulation. Towards the end of that period, a new generation of robots started to emerge that shared important characteristics with simple biological systems: robustness, simplicity, small size, flexibility, and modularity [76.9, 10]. Above all, those robots were designed so that they could be programmed and manipulated by people without engineering training. Those technological achievements, together with the growing influence of biological inspiration in artificial intelligence [76.11], coincided with the first evolutionary experiments on real robots [76.12–14] (VIDEO 39 and VIDEO 371), and the term evolutionary robotics was coined [76.15].

76.1 Method

The major methodological steps in evolutionary robotics proceed as follows (Fig. 76.1). An initial population of different artificial chromosomes, each encoding the control system (and possibly the morphology) of a robot, is randomly created. Each of these chromosomes is then decoded into a corresponding controller, for example a neural network (NN), and downloaded into the processor of the robot. The robot is then let free to act (move, look around, manipulate the environment) according to a genetically specified controller while its performance for a given task is automatically evaluated. Performance evaluation is done by a fitness function that measures how fast and straight the robot moves, how frequently it collides with obstacles, etc. This procedure is repeated for all chromosomes of the population. The fittest individuals (those that have received more fitness points) are allowed to reproduce by generating copies of their chromosomes with the addition of random modifications introduced by genetic operators (e.g., mutations and exchange of genetic material). The newly obtained population is then tested again on the same robot. This process is repeated for a number of generations until an individual is born which satisfies the fitness function set by the user. The control system of evolved robots, encoded in an artificial genome, is therefore generated by a repeated process of selective reproduction, random mutation, and genetic recombination, similarly to what happens in natural evolution (VIDEO 119).

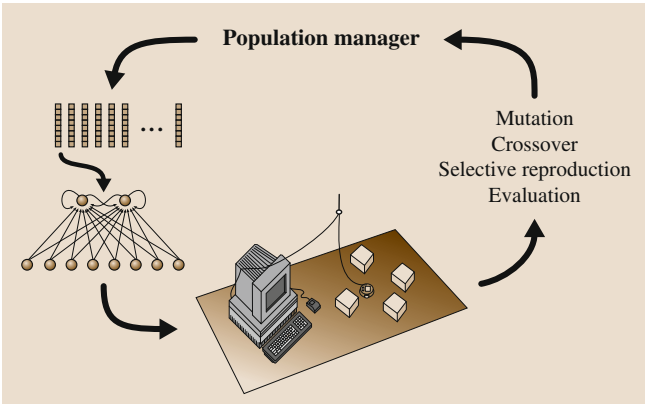


Fig. 76.1 Evolutionary experiments on a single robot. Each individual of the population is decoded into a corresponding neurocontroller which reads sensory information and sends motor commands to the robot every 300 ms while its fitness is automatically evaluated and stored away for reproductive selection

76.2 First Steps

In an early experiment on robot evolution without human intervention, carried out at Ecole Polytechnique Fédérale de Lausanne (EPFL) [76.12], a small wheeled robot was evolved for navigation in a looping maze (Fig. 76.2). The Khepera robot has a diameter of 55 mm and two wheels with controllable velocities in both directions of rotation. It also has eight infrared sensors, six on one side and two on the other side, that can function either in active mode to measure distance from obstacles or in passive mode to measure the amount of (infrared) light in the environment. The robot was connected to a desktop computer through rotating contacts that provided both power supply and data exchange through a serial port (VIDEO 39).

A simple genetic algorithm [76.16] was used to evolve the synaptic strengths of a neural network composed of eight sensory neurons and two motor neurons. Each sensory unit was clamped to one of the eight active infrared sensors whose value was updated every 300 ms. Each motor unit received weighted signals from the sensory units and from the other motor unit, plus a recurrent connection with itself with a 300 ms

delay. The net input of the motor units was offset by a modifiable threshold and passed through a logistic squashing function. The resulting outputs, in the range $[0, 1]$, were used to control the two motors so that an output of 1 generated maximum rotation speed in one direction, an output of 0 generated maximum rotation speed in the opposite direction, and an output of 0.5 did not generate any motion in the corresponding wheel. A population of 80 individuals, each coding the synaptic strengths and threshold values of the neural controllers, was initialized with all weights set to small random values centered around zero. Each individual was tested on the physical robot for 80 sensorimotor cycles (approximately 24 s) and evaluated at every cycle according to a fitness function with three components measured onboard the robot

$$\phi = V(1 - \sqrt{\Delta v})(1 - i) \,, \tag{76.1}$$

where V is the average rotation speed of the two wheels, Δv is the absolute value of the algebraic difference between the signed speed values of the wheels (positive is one direction, negative the other), and i is the normalized activation value of the infrared sensor with the highest activity. The first component is maximized by speed, the second by straight motion, and the third by distance from objects.

During the first 100 generations, both average and best fitness values grew steadily, as shown in Fig. 76.3. A fitness value of 1.0 would correspond to a robot moving straight at maximum speed in an open space and therefore was not attainable in the looping maze shown in Fig. 76.2, where some of the sensors were often active and where several turns were necessary to navigate. Fig. 76.4 shows the trajectory of the best individual of the last generation.

Although the fitness function did not specify in what direction the robot should navigate (given that it was perfectly circular and that the wheels could rotate in both directions), after a few generations all the best individuals moved in the direction corresponding to the side with the highest number of sensors. Individuals moving in the other direction had a higher probability of colliding into corners without detecting them and thus disappeared from the population. Furthermore, the cruising speed of the best evolved robots was approximately half of the maximum speed that could be technically achieved and did not increase even when the evolutionary experiment was continued up to 200 generations. Further analysis revealed that this self-limitation of the navigation speed had an adaptive function because, considering the sensory and motor

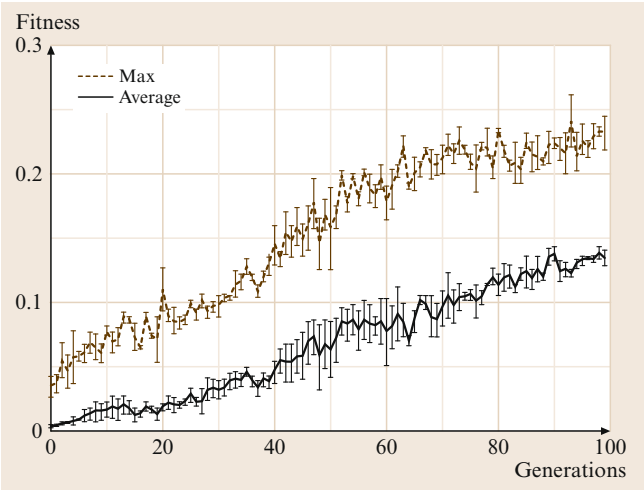


Fig. 76.3 Average fitness of the population and fitness of the best individual at each generation (*error bars* show standard error over three runs from different initial populations)

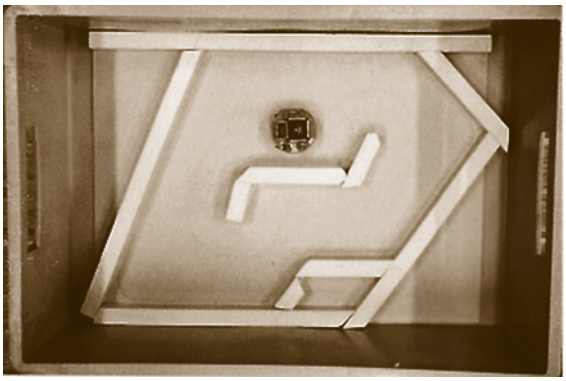


Fig. 76.2 Bird's-eye view of the desktop Khepera robot in the looping maze

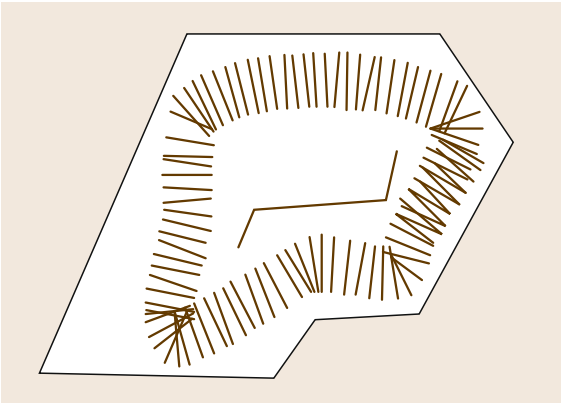


Fig. 76.4 Trajectory of the robot with the best neural controller of the last generation. Segments represent the axis between the two wheels. Data were recorded and plotted every 300 ms using an external laser positioning device

refresh rate together with the response profile of the distance sensors, robots that traveled faster had a higher risk of colliding with walls before detecting them; they gradually disappeared from the population.

Despite its simplicity, this experiments shows that evolution can discover solutions that match not only the computational requirements of the task to be solved, but also the morphological and mechanical properties of the robot in relation to its physical environment.

76.2.1 Evolution of Neural Controllers for Walking

Over the past 20 years or so, there has been a growing body of work on evolving controllers for various kinds of walking robots – a nontrivial sensorimotor coordination task. Early work in this area concentrated on evolving dynamical network controllers for simple (abstract) simulated insects (often inspired by cockroach studies) which were required to walk in simple environments [76.17, 18]. Earlier, *Beer* et al. had introduced a neural architecture for locomotion based on studies of cockroaches [76.19], which is shown in Fig. 76.5. The promise of this work soon led to versions of this methodology being used on real robots. Probably the first success in this direction was by *Lewis* et al. [76.14, 20] who evolved a neural controller for a simple hexapod robot using coupled oscillators built from continuous-time, leaky-integrator, artificial neurons. All evaluations were done on the actual robot with each leg connected to its own pair of coupled neurons, leg swing being driven by one neuron and leg elevation by the other. These pairs of neurons were cross connected, in a manner similar to that used by *Beer* and *Gallagher* [76.18] (Fig. 76.5), to allow coordination between the legs. In order to speed up the process, they employed staged evolution where first an oscillator capable of moving a leg was evolved and then an architecture based on these oscillators was further evolved to develop walking. The robot was able to execute an efficient tripod gait on flat surfaces.

Gallagher et al. [76.21] described experiments where neural networks controlling locomotion in an artificial insect were evolved in simulation and then successfully downloaded onto a real hexapod robot. This machine was more complex than Lewis et al.'s, with a greater number of degrees of freedom per leg. In this approach, each leg was controlled by a fully connected network of five continuous-time, leaky-integrator neurons, each receiving a weighted sensory input from that leg's angle sensor. Initially the architecture shown in Fig. 76.5 was used, with the connection weights and neuron time constants and biases under genetic control. This produced efficient tripod gaits for walking on flat surfaces. In order to produce a wider range of gaits operating at a number of speeds such that rougher terrain could be successfully negotiated, a different distributed architecture, more inspired by stick insect studies, was found to be more effective [76.22].

Galt et al. [76.23] used a genetic algorithm to derive the optimal gait parameters for a Robug III robot, an eight-legged, pneumatically powered walking and climbing robot. The individual genotypes represented parameters defining each leg's support period and the timing relationships between leg movements. These parameters were used as inputs to a mechanistic finite-state machine pattern-generating algorithm that drove the locomotion. Such algorithms, which are often used in conventional walking machines, rely on relatively simple control dynamics and do not have the same potential for the kind of sophisticated multigait coordination that complex dynamical neural network archi-

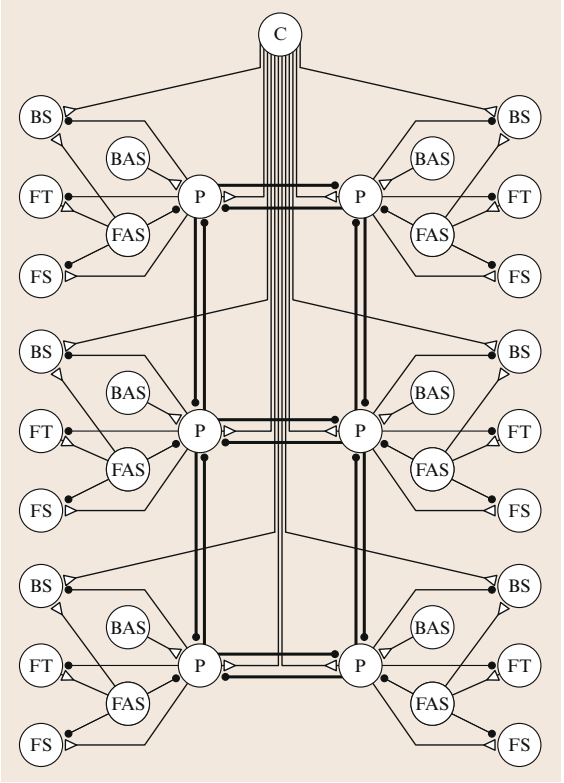


Fig. 76.5 Schematic diagram of a distributed neural network for the control of locomotion as used by *Beer* et al. [76.19]. Excitatory connections are denoted by *open triangles* and inhibitory connections are denoted by *filled circles*. C, command neuron; P, pacemaker neuron; FT, foot motor neuron; FS and BS, forward swing and backward swing motor neurons; FAS and BAS, forward and backward angle sensors

tectures, such as those described in this section, have been shown to produce. However, controllers were successfully evolved for a wide range of environments and to cope with damage and systems failure (although an individual controller had to be tuned to each environment; they were not able to self-adapt across a wide range of conditions). *Gomi* and *Ide* [76.24] evolved the gaits of an eight-legged robot (Fig. 76.6) using genotypes made of eight similarly organized sets of genes, each gene coding for leg motion characteristics such as the amount of delay after which the leg begins to move, the direction of the leg's motion, the end positions of both vertical and horizontal swings of the leg, and the vertical and horizontal angular speed of the leg. After a few dozen generations, where evaluation was on the robot, a mixture of tetrapod and wave gaits was obtained. Using the cellular encoding [76.25] developmental approach – which genetically encodes a grammar-tree program that controls the division of cells growing into a dynamical recurrent neural network of the kind used by Beer et al. – *Gruau* and *Quatramaran* [76.26] evolved a single-leg neural controller for the same eight-legged robot used by Gomi and Ide. This generated a smooth and fast quadrupod locomotion gait (VIDEO 372 and VIDEO 373). *Kodjabachian* and *Meyer* [76.27] extended this work to develop more sophisticated locomotion behaviors. *Jakobi* [76.28] successfully used his minimal simulation techniques (described in Sect. 76.3) to evolve controllers for the same eight-legged robot as Gruau. Evolution in simulation took less than 2 h on what would today be regarded as a very slow computer, and was then successfully transferred to the real robot. Jakobi evolved modular controllers based on Beer's continuous recurrent networks to control the robot as it engaged in walking about its environment, avoiding obstacles and seeking out goals depending on the sensory input. The robot could smoothly change gait, move backward and forward, and even turn on the spot. More recent work has used similar architectures to those explored by the researchers mentioned above, to control more mechanically sophisticated robots such as the Sony Aibo [76.29].

Recently there has been successful work on evolving coupled oscillator style neural controllers for the highly unstable dynamic problem of biped walking. *Reil* and *Husbands* [76.30] showed that accurate physics based simulations employing physics-engine software could be used to develop controllers able to generate successful bipedal gaits (VIDEO 374). *Reil* et al. have now significantly developed this technology to exploits its commercial possibilities, in the animation and games industries, for the real-time control of physically simulated three-dimensional (3-D)

Fig. 76.6 The octopod robot built by Applied AI Systems Inc.

humanoid characters engaged in a variety of motor behaviors (refer to [76.31] for further details). Coupled neural oscillators have been evolved also to control the swimming pattern of articulated, snake-like, underwater robots using physics-based simulations [76.32].

Vaughan has taken related work in another direction. He has successfully applied evolutionary robotics techniques to evolve a simulation of a 3-D ten-degree-of-freedom bipedal robot. This machine demonstrates many of the properties of human locomotion. By using passive dynamics and compliant tendons, it conserves energy while walking on a flat surface. Its speed and gait can be dynamically adjusted and it is capable of adapting to discrepancies in both its environment and its body's construction [76.33]. Parameters describing the body shape (leg segment lengths, hip width, etc.) and properties of a continuous dynamical neural network controller were under genetic control. The machine started out as a passive dynamic walker [76.34] on a slope, and then throughout the evolutionary process the slope was gradually lowered to a flat surface. The machine demonstrated resistance to disturbance while retaining passive dynamic features such as a passive swing leg. *Wischmann* and *Passemann* independently took a very similar approach [76.35]. Vaughan's original machine did not have a torso, but he has also successfully applied the method to a simplified two-dimensional (2-D) machine with a torso above the hips. When pushed, this dynamically stable bipedal machine walks either forward or backwards just enough to release the pressure placed on it. It is also able to adapt to external and internal perturbations as well as variations in body size and mass [76.36]. These biped examples make use of the co-evolution of body morphology and neural controller, an idea also used in

earlier, more abstract, work on evolved bipedal locomotion by *Endo* et al. [76.37]. Although possible changes to the body morphology are quite tightly constrained, nonetheless this aspect was important. This theme is covered in more detail in the later section on Evolving Bodies, which also describes recent examples of evolved walking behaviors in the context of body-brain co-evolution.

McHale and *Husbands* [76.38, 39] have compared many forms of evolved neural controllers for bipedal and quadrupedal walking machines. Recurrent dynamical continuous time networks and GasNets (described in Sect. 76.7.3) were shown to have advantages in most circumstances. The vast majority of the studies mentioned above were conducted for relatively benign environments. Notwithstanding this observation, we can conclude that the more complex dynamical neural network architectures, with their intricate dynamics, generally produce a wider range of gaits and generate smoother, more adaptive locomotion than the more standard use of systems based on finite-state machines employing parameterized rules governing the timing and coordination of individual leg movements [76.40].

76.3 Simulation and Reality

Few of the experiments in the previous section were carried out entirely on physical robots because:

1. Evolution may take a long time, especially if it is carried out on a single robot that incarnates the bodies of all the individuals of the evolving population.
2. The physical robot can be damaged because populations always contain a certain number of poorly performing individuals (for example, colliding against walls) by effect of random mutations.
3. Restoring the environment to initial conditions between trails of different individuals or populations (for example, replenishing the arena with objects) may not always be feasible without human intervention.
4. Evolution of morphologies and evolution of robots that can grow during their lifetime is almost impossible with today's technology without some level of human intervention.

For those reasons, researchers often resort to evolution in simulation and transfer the evolved controllers to the physical robot. However, it is well known that programs that work well in simulations may not function properly in the real world because of differences in sensing, actuation, and in the dynamic interactions between robot and environment [76.41]. This *reality gap* is even more evident in adaptive approaches, such as evolutionary robotics, where the control system and morphology are gradually crafted through the repeated interactions between the robot and the environment. Therefore, robots will evolve to match the specificities of the simulation, which differ from the real world. Although these issues clearly rule out any simulation based on grid worlds or pure kinematics, over the last 10 years simulation techniques have dramatically improved and resulted in software libraries that model reasonably well dynamical properties such as friction, collision, mass, gravity, and inertia [76.42]. These software tools allow one to simulate articulated robots of variable morphology and their environment as fast as, or faster than, real time in a desktop computer.

Nonetheless, even physics-based simulations include small discrepancies that can accumulate over time and result in very different behavior from reality (for example, a robot may get stuck against a wall in simulation whereas it can get free in reality, or vice versa). Also, physics-based simulations cannot account for diversity of response profiles of the individual sensors, motors, and gears of a physical robot. Several methods can be used to cope with these problems and improve the quality of the transfer from simulation to reality.

A widely used method consists of adding independent noise to the values of the sensors provided by the model and to the end position of the robot computed by the simulator [76.43]. Some software libraries allow the introduction of noise at several levels of the simulation. This solution prevents evolution from finding solutions that rely on the specificities of the simulation model. Another method consists in sampling the actual sensor values of the real robot positioned at several angles and distances from objects of different texture. Those values are then stored in a look-up table and retrieved with the addition of noise according to the position of the robot in the environment [76.44]. This method proved to be very effective for generating controllers that transfer smoothly from simulation to reality. A drawback of this sampling method is that it does not scale up well to high-dimensional sensors (e.g., vision) or geometrically complicated objects.

Another method, also known as minimal simulations, consists of modeling only those characteristics of the robot and environment that are relevant for the emergence of desired behaviors [76.45]. These char-

acteristics, which are referred to as base-set features, should be accurately modeled in simulation. Instead, all the other characteristics, which are referred to as implementation aspects, should be randomly varied across several trials of the same individual in order to ensure that evolving individuals do not rely on implementation aspects, but rely on base-set features only. Base-set features must also be varied to some extent across trials in order to ensure some degree of robustness of the individual with respect to base-set features, but this variation should not be so large that reliably fit controllers fail to evolve at all. This method allows very fast evolution of complex robot–environment situations, as in the example of the hexapod walk described in Sect. 76.2.1. A drawback of minimal simulations is that it is not always easy to tell in advance which are the base-set features that are relevant for the desired behavior.

Yet another method consists of the coevolution of the robot (control and/or morphology) and of the simulator parameters that are most likely to differ from the real world and that may affect the quality of the transfer [76.46]. This method consists of coevolving two populations, one encoding the properties of the robot and one encoding the parameters of the simulator. Coevolution happens in several passes through a two-stage process. In stage one, a randomly generated population of robots are evolved in the default simulator and the best individual is tested on the real robot while the time series of sensory values are recorded. In stage two, the population of simulators is evolved to reduce the difference between the time series recorded on the real robot and the time series obtained by testing evolved robots within the simulator. The best evolved simulator is then used for stage one where a new randomly generated population is evolved and the best individual is tested on the real robot to generate the time series for stage two of simulator evolution. This two-stage coevolution is repeated several times until the error between simulated and real robot behavior is the smallest possible. It has been shown that approximately 20 passes of the two-stage process are sufficient to evolve a good control system that could be transferred to an articulated robot. In that case, the real robot was used to test only 20 individuals.

A recent approach tackles the simulation to reality transfer problem by using a multi-objective formulation of ER in which two main objectives are optimized via a Pareto-based multi-objective evolutionary algorithm: (1) the fitness and (2) the transferability [76.47]. To evaluate the transferability a simulation-to-reality disparity measure was defined in terms of the difference in behavior between simulation and reality for any given controller. This measure is approximated for each member of the population and the method has successfully been demonstrated for walking behaviors [76.47].

Finally, another method consists of genetically encoding and evolving the learning rules of the control system, rather than its parameters (e.g., connection strengths). The parameters of the decoded control system are always initialized to small random values at the beginning of an individual lifetime and must self-organize using the learning rules [76.48]. This method prevents evolution from finding a set of control parameters that fit the specificities of the simulation model, and encourages emergence of control systems that remain adaptive to partially unknown environments. When such an evolved individual is transferred to the real robot, it will develop online its control parameters according to the genetically evolved learning rules and taking into account the specificities of the physical world. This method is described in more detail in Sect. 76.7.2 on evolution of learning.

76.4 Behavior as a Complex Adaptive System

Behavior is a dynamical process resulting from nonlinear interactions (occurring at a fast time rate) between the agent's control system, its body, and the environment [76.49, 50]. At any time step, the environment and the agent–environment relation influence the body and the motor reaction of the agent, which in turn influences the environment and/or the agent–environmental relation (Fig. 76.7). Sequences of these interactions lead to a dynamical process where the contributions of the different aspects (i. e., the robot's control system, the robot's body, and the environment) cannot be separated. This implies that even complete knowledge of the elements governing the interactions provides little insight into the behavior emerging from these interactions [76.51, 52].

An interesting property of evolutionary robotics is that it can enable to synthesize robots displaying a certain behavioral capacity without specifying the manner in which such capacity should be realized and/or the combination of elementary behaviors that should be produced and combined to achieve the desired overall capacity. This allows the evolving robots to discover and exploit behaviors that emerge from the interactions between the robot control system, the robot body and

the environment and/or from the interaction between previously developed behavioral capacities [76.52]. The synthesis and exploitation of emergent properties, in turn, often allows evolving robots to discover solutions that rely on relatively parsimonious control policy and/or body structures.

For an example of how evolving robots can solve an adaptive task on the basis of a simple control policy, thanks to the possibility to exploit properties emerging from the agent/environmental interactions, let us consider the case of a Khepera robot placed in an arena surrounded by walls (Fig. 76.8) that should evolve an ability to forage by finding and remaining close to a food object (i. e., a cylindrical object) [76.53]. The robot is provided with eight infrared sensors and two motors controlling the desired speed of the two corresponding wheels. From the point of view of an external observer, solving this problem requires robots able to:

1. Explore the environment until an obstacle is detected.
2. Discriminate whether the obstacle detected is a wall or a cylindrical object.
3. Approach or avoid the object depending on the object type.

A detailed analysis of the sensory patterns experienced by the robot indicated that the task of discriminating the two objects is far from trivial since the two classes of sensory patterns experienced by robots close to a wall and close to cylindrical objects overlap significantly. However, robots evolved for the ability to solve this task resorted to a strategy that does not require to explicitly discriminate of the two types of objects [76.53]. This solution (VIDEO 116) consists in reacting to sensory states so that the robot/environmental dynamics converge into a limit cycle near the cylindrical object, in which the robot keep moving forth and back and left and right, and not near a wall (Fig. 76.9).

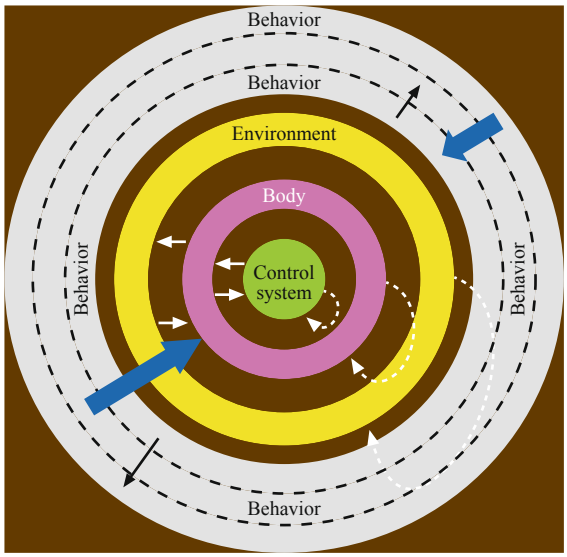


Fig. 76.7 Schematization of how: (1) behavior can emerge from several nonlinear interactions, occurring at fast time rates, between the agent's control system, its body, and the environment, and (2) behavior can display a multi-level organization in which the robot/environmental interactions and the interaction between lower-level behaviors give rise to higher level behaviors that later affect the interactions from which they originate

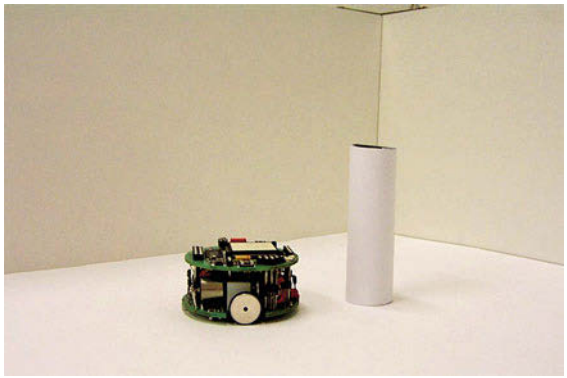


Fig. 76.8 The environment and the robot. The environment consists of an arena of 60×35 cm and contains a cylindrical objects placed at a randomly selected location

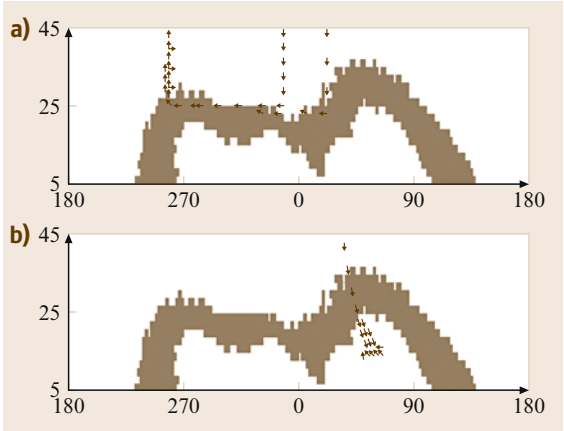


Fig.76.9a,b Angular trajectories of an evolved robot close to a wall **(a)** and to a cylinder **(b)**. The picture was obtained by placing the robot at a random position in the environment, leaving it free to move for 500 cycles, and recording its relative movements with respect to the two types of objects for distances smaller than 45 mm. For sake of clarity, *arrows* are used to indicate the relative direction, but not the amplitude of movements

The possibility to discover and rely on these forms of emergent behavior allows evolving robots to find computationally simple solutions to apparently complex problems. Indeed, the foraging task described above can be solved by a Khepera robot provided with a simple reactive controller (i. e., a feedforward neural network with eight sensory neurons that encode the state of the corresponding infrared sensors directly connected to two motor neurons that set the desired speed of the two wheels).

76.4.1 Behavior Recombination and Re-Use

Evolving robots can recombine and re-use acquired elementary behavioral capacities to produce higher level behaviors. This has been demonstrated in a series of simulation experiments in which a population of humanoid robots provided with an articulated arm, a camera, and a touch sensor on the palm of the hand, have been evolved for the ability to execute two-words imperative sentences constituted by the combination of three action and three object words (reach, touch, move, red-object, green-object, and blue-object) encoded by six corresponding binary sensors [76.54].

During the evolutionary process the robots were evaluated for the ability to *comprehend* seven out of the nine possible sentences (that can be generated by combining the three action and the three object words) by executing the seven corresponding behaviors. The robots were then post-evaluated also on the other two sentences not experienced during the evolutionary process.

Some of the evolved robots were able to develop the required skills and to generalize their capacities to the two new sentences by executing the appropriate corresponding behaviors (VIDEO 41). Differently from the other individuals, the robots able to generalize where characterized by a hierarchical organization in which the nine behaviors were produced by combining over time a set of elementary behaviors and in which the same elementary behaviors was re-used to produce different high-level behaviors. More specifically the robots able to generalize displayed a reach-X behavior (that consisted in moving the arm toward a red, or green, or blue object), a touch behavior (that consisted in moving the hand until the object is touched irrespectively from the color), and a move behavior (that consists in keep moving the hand also after the object has been touched irrespectively from the color) and combined these lower-level behaviors in a compositional manner to produce the nine required higher-level actions. This means that, for example, the same *reach red-object* behavior was used incombination with the *touch* or the *push* behavior to produce a *touch the red-object* and a *push the red-object* behavior. For other works discussing the emergence and the role of multi-level behavioral organizations see [76.52].

76.4.2 Sensory-Motor Coordination

By acting robots inevitably modify the robot-environmental relation and/or the environment and consequently the stimuli that they will experience next. By exploiting the possibility to actively influence the perceived stimuli through actions, robots can find adaptive solutions based on parsimonious control policies. Artificial evolution constitutes an effective method for discovering such type of solutions that are often hard to imagine from the point of view of a human observer. Indeed, examples of clever use of sensory-motor coordination abounds in the evolutionary robotics literature.

Let us consider, for instance, the case of a Khepera robot endowed with infrared and wheels speed sensors, that can forage by ramaining close to large cylindrical objects (food) while avoiding small cylindrical objects (dangers) [76.55]. From a passive perspective, that does not take into account the fat that the robot can self-select useful stimuli through action, the ability to discriminate between sensory stimuli experienced near small and large cylindrical objects requires a relatively complex control policy since the two classes of stimuli strongly overlap in the robot's perceptual space. On the other hand, the exploitation of sensory-motor coordination can can allow the robots to simplify the discrimination problem.

Indeed, evolving robots tend to converge on a rather simple solution that consists in circling around the cylindrical objects, as soon as an object is perceived, and in using the differential speed of the left and right wheels sensed during the execution of the object-circling behavior to decide to keep circling around the object (in the case of small differential speeds) or to abandon the object (in the case of large differential speeds). Indeed, the execution of the object-circling behavior allows the robots to experience sensory stimuli on the wheel sensors that are well differentiated for small and large objects. This, in turn, allows them to solve the object discrimination problem with a rather simple but reliable control policy. For other examples see [76.53, 56, 57].

76.5 Evolving Bodies

Most evolutionary robotics experiments – and most robotics experiments in general – assume that the body plan of the robot has already been designed; an optimization method is used to improve the control policy only (The term *body plan* is here used to denote all aspects of a robot's design other than its control policy. Such design considerations include the robot's mechanical layout and material properties as well as its sensor and motor distributions.). This emphasis belies an assumption within the field of robotics, which is that control policy design is non-intuitive and thus should be automated, while choosing an appropriate robot body plan is intuitive and thus can be manually designed.

However, it has been shown that the careful design of the robot's body can have large and desirable impacts on its resulting behavior. For example, proper curvature on the underside of a biped robot's feet (along with other settings) can allow it to walk down a declined plane with no control policy at all [76.58]. Or, that modifications to an anthropomorphic robot arm and hand can facilitate the evolution of active categorical perception [76.59] (Active categorical perception occurs when a robot or animal actively interacts with objects of interest, and the sensory stimulation resulting from this physical interaction allows for categorization of those objects.). These results fit with the view of embodied behavior, as outlined in Fig. 76.7: because behavior arises from the interaction between a robot's body and its environment, alterations to the robot's body will alter the resulting behavior.

This suggests that it is useful to automatically improve not just a robot's control policy but also its body plan. Evolutionary algorithms are a uniquely well suited tool for this task because, unlike many learning methods, they do not make assumptions about the structure of the system being optimized: the length of a robot's leg – or the number of legs – can be evolved just as easily as can the strength of a synaptic connection in an artificial neural network.

76.5.1 Co-Evolving Body and Brains

The Sussex group was the first to demonstrate the evolution of robot morphology: they evolved sensor placements on a physical robot [76.15], although the other aspects of the robot's body plan remain fixed. A year later *Sims* [76.61] demonstrated an evolutionary algorithm that improved the structure and parameters of the robots' body plans and control policies. Although Sim's *creatures* were virtual and operated in a simulated environment, the robots exhibited a wide range of intuitive and non-intuitive body plans that allowed them to swim, walk or compete over a limited resource [76.62].

Funes and *Pollack* [76.63] demonstrated that it was possible to evolve three-dimensional forms in simulation, build them in reality, and have the physical structure act similarly to the originally-evolved simulated structure. This was followed by work from the same group in which robots evolved in simulation were manufactured as physical robots using 3-D printing technology [76.60] (Fig. 76.10). Although only the plastic frame of the robot was printed and the electronics and battery had to be manually added, this served as a demonstration that, in principle, robot design could be automated using evolutionary algorithms and robot manufacture could be automated using rapid prototyping [76.64].

Since then, a number of research groups have evolved robot body plans and control policies simultaneously for various purposes. Some researchers have adapted this approach for studying biological questions. For example *Long* et al. [76.65] have evolved the stiffness of artificial tails attached to physical swimming robots: robots with tails of differing stiffness have differing abilities to swim fast or turn well. This provides a unique experimental tool for investigating how backbones originally evolved in early vertebrates. *Clark* et al. [76.66] also evolved the material properties of part of a robot fish, but focussed in this case on evolving the stiffness and shape of its fins in simulation. This project had an engineering aim: the evolved fins were

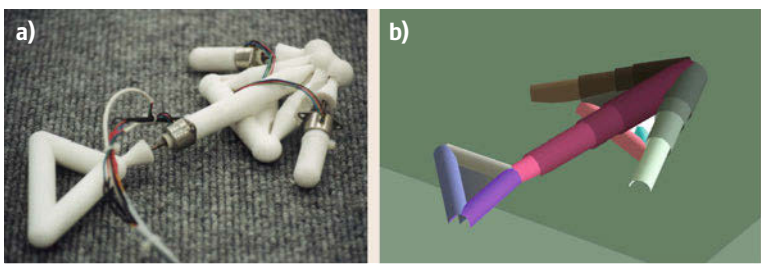


Fig.76.10a,b An example evolved robot from the GOLEM project [76.60]. **(a)** The virtual robot, as originally evolved in the simulated environment. **(b)** The physical robot comprised of a 3-D-printed plastic frame and manually-added electronics

manufactured and tested on a physical robot fish and were found to aid desirable swimming behavior.

Evolutionary algorithms can be used to explore the space of possible robot body plans, but they can also be used to explore metamorphosis, or how a robot's body plan might change over its lifetime. In recent work *Bongard* [76.67] compared two approaches to evolving walking behaviors for upright legged robots (VIDEO 771). In the first approach, upright legged robots were evolved until successful walking was discovered. In the second approach, locomotion was first evolved for legless, anguilliform robots that gradually grew legs while moving. As evolution proceeded in this second approach, later generations of robots gradually lost this infant legless body plan and instead were born with the upright, legged body plan. It was found that walking evolved for the upright legged robots more rapidly in the second approach, and that the evolved controllers were more robust. The explanation for this result is that the legless robot provides a form of scaffolding that accelerates search: it is easier for evolution to generate locomotion for the legless robot because with the anguilliform body plan the robot cannot fall over (Scaffolding is the phenomenon in which a teacher introduces some aspect into the learner's environment that helps the learner to grasp a concept and then later refine the concept when the scaffold is removed [76.68]. The canonical example of scaffolding is training wheels for bicycles.). This locomotion strategy is then refined subsequently by evolution to successfully control the upright and legged (and thus unstable) robot.

The simultaneous evolution of robot body plans and control policies offers other avenues for investigating the relationship between body, brain and environment. In [76.69] it was found that more complexly-shaped robots were produced when evolved to walk over rough terrain than when evolved to walk over flat terrain (VIDEO 772). This was due to the fact that robots evolved in rough terrain evolved appendages and hooks to gain purchase between outcroppings and then pull or push themselves forward.

Most recently *Hiller* and *Lipson* [76.70] have demonstrated the evolution of soft robots: this requires evolving not just the control policy of the robot and its physical shape, but also the material properties of each voxel comprising the machine. This allows for complex three-dimensional patterning of soft and rigid material throughout the robot, which can be exploited by evolution to produce locomotion [76.71]. Soft robots are an ideal vehicle for demonstrating the power of evolutionary algorithms: the design and control of such machines is highly nonintuitive, making manual design extremely difficult.

76.5.2 Self-Modeling

The evolution of robot body plans can be useful not just for robot design but also for increasing the adaptivity of a physical robot once it has been designed and deployed. For example in [76.72] it was shown that a physical robot could be equipped with an onboard simulator that the robot could use to continuously evolve models of itself. These self-models reflect the mechanical construction of the robot. This method was found useful for robots that might sustain unanticipated damage such as the mechanical separation of a leg: the robot diagnoses the damage; it then evolves a simulated damaged robot that accurately reflects the physical damage; it evolves a compensatory control policy internally using the simulated, damaged robot; and finally the physical robot uses the internally evolved compensatory control policy to continue moving despite its injury. Figure 76.11 outlines this method.

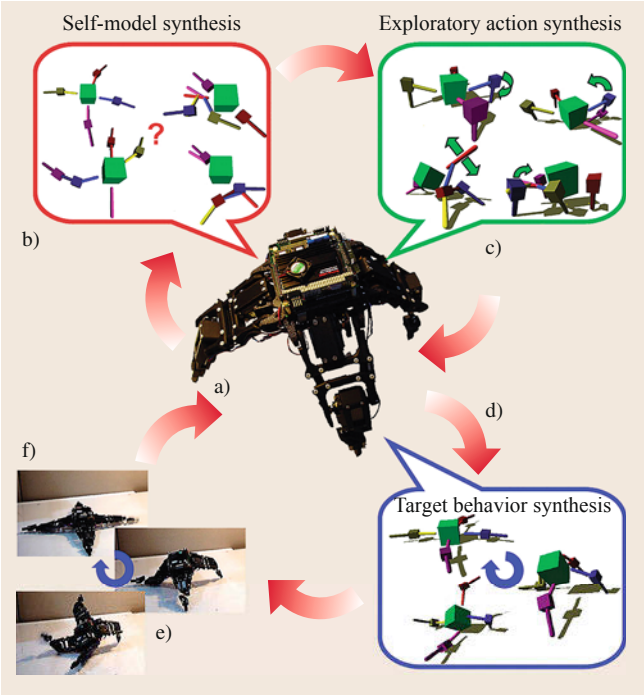


Fig.76.11a–f The machine begins by performing a random action and collects the resulting sensor-motor data **(a)**. An evolutionary algorithm evolves a population of simulated robots that, when they move, produce similar sensor-motor data as the physical machine **(b)**. Another algorithm searches for a new action for the physical robot to perform **(c)**. The physical robot performs the new action, and re-evolves self-models to explain the result of the first action and the new action (**(a)**). After several cycles of this self-modeling, the best self-model is used to evolve new behaviors **(d)**. Finally, the physical robot executes these newly-evolved behaviors **(e,f)**

A method for generalizing this to robot swarms was demonstrated in [76.73]. Each robot in the swarm maintained its own self-modeling engine, but would periodically export its best self-model and control policy to others in the swarm. The result of this was that if one robot was damaged and recovered, a second robot that suffered similar damage recovered more rapidly.

Finally, instead of modeling the self, a robot could create a model of another robot in its vicinity. Even better, the robot could evolve a model of the other robot's intentions and use this information to aid or thwart the other robot's actions, as demonstrated in [76.74]. Although the robots did not model each other's body plans, this ability to model others in general is known as Theory of Mind. One could imagine how increasing levels of recursion of such embedded mind reading could provide continued evolutionary pressure toward increasingly intelligent machines.

76.6 Seeing the Light

Pioneering experiments on evolving visually guided behaviors were performed at Sussex University [76.75] on a specially designed gantry robot (Fig. 76.12, see also VIDEO 371). Discrete-time dynamical recurrent neural networks and visual sampling morphologies were concurrently evolved: the brain was developed in tandem with the visual sensor [76.13, 76, 77]. The robot was designed to allow real-world evolution by having *off-board* power and processing so that the robot could be run for long periods while being monitored by automatic fitness evaluation functions. A charge-coupled device (CCD) camera points down towards a mirror angled at 45° as shown in Fig. 76.12. The mirror can rotate around an axis perpendicular to the camera's image plane. The camera is suspended from the gantry, allowing motion in the X, Y, and Z dimensions. This effectively provides an equivalent to a wheeled robot with a forward-facing camera when only the X and Y dimensions of translation are used. The additional dimension allows flying behaviors to be studied.

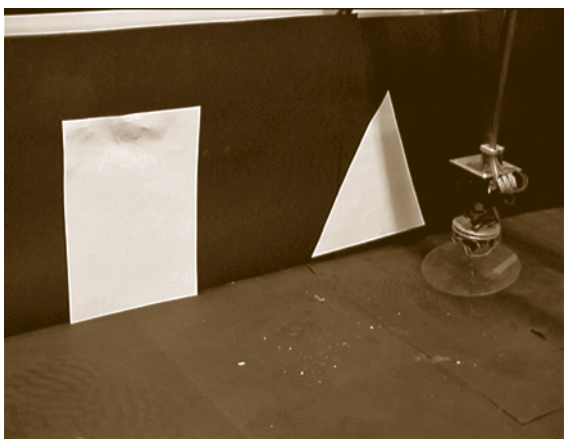


Fig. 76.12 The gantry robot used in the visual discrimination task. The camera inside the top box points down at the inclined mirror, which can be turned by the stepper motor beneath. The lower plastic disk is suspended from a joystick to detect collisions with obstacles

The apparatus was initially used in a manner similar to the real-world experiments on navigation in the looping maze with the miniature mobile robot described in Sect. 76.2. A number of visually guided navigation behaviors were successfully achieved, including navigating around obstacles, tracking moving targets, and discriminating between different objects [76.76]. The evolutionary process was incremental. The ability to distinguish between two different targets was evolved on top of the single target-finding behavior. The chromosome was of dynamic length so the neurocontroller was structurally further developed by evolution to achieve the new task (neurons and connections added). In the experiment illustrated in Figs. 76.12 and 76.13,

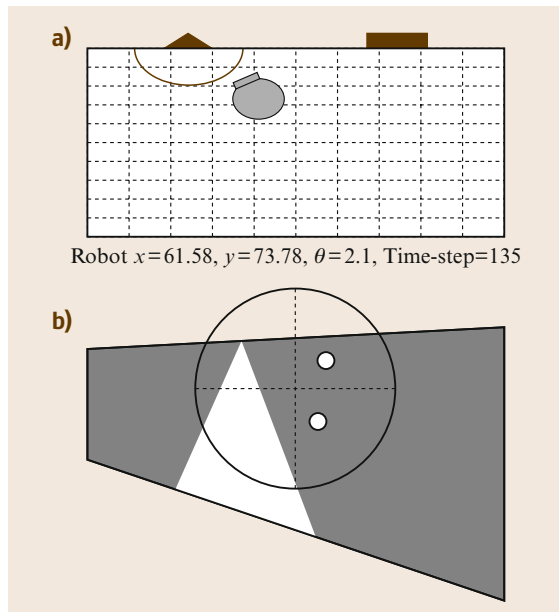


Fig.76.13a,b The shape discrimination task. **(a)** The position of the robot in the arena, showing the target area in front of the triangle. **(b)** The robot camera's field of view showing the visual patches selected by evolution for sensory input

starting from a random position and orientation, the robot had to move to the triangle rather than the rectangle. This had to be achieved irrespective of the relative positions of the shapes and under very noisy lighting conditions. Recurrent neural network controllers were evolved in conjunction with visual sampling morphologies. Only genetically specified patches from the camera image were used (by being connected to input neurons according to the genetic specification). The rest of the image was thrown away. This resulted in extremely minimal systems using only two or three pixels of visual information, yet still able to perform the task reliably under highly variable lighting conditions [76.13, 76].

This was another example of staged, or incremental, evolution to obtain control systems capable of solving problems that are either too complex or may profit from an evolutionary methodology that discovers, preserves, and builds upon subcomponents of the solution. For an evolutionary method that incorporate strategies to explicitly address this issue, interested readers may refer to [76.78]. However, staged evolution remains a poorly explored area of evolutionary robotics that deserves further study and a more principled approach [76.79] in order to achieve increasingly complex robotic systems.

76.6.1 Coevolution of Active Vision and Feature Selection

Machine vision today can hardly compete with biological vision despite the enormous power of computers. One of the most remarkable – and often neglected – differences between machine vision and biological vision is that computers are often asked to process an entire image in one shot and produce an immediate answer whereas animals take time to explore the image over time, searching for features and dynamically integrating information over time.

Active vision is the sequential and interactive process of selecting and analyzing parts of a visual scene [76.80–82]. *Feature selection* instead is the development of sensitivity to relevant features in the visual scene to which the system selectively responds, e.g., [76.83]. Each of these processes has been investigated and adopted in machine vision. However, the combination of active vision and feature selection is still largely unexplored. An intriguing hypothesis is that coevolution of active vision and feature selection could greatly simplify the computational complexity of vision-based behavior by facilitating each other's task.

This hypothesis was investigated in a series of experiments [76.84] on coevolution of active vision and feature selection for behavioral systems equipped with a primitive moving retina and a deliberately simple neural architecture (Fig. 76.14). The neural architecture was composed of an artificial retina and two sets of output units. One set of output units determined the movement and zooming factor of the retina, and the other set of units determined the behavior of the system, such as the response of a pattern-recognition system, the control parameters of a robot, or the actions of a car driver. The neural network was embedded in a behavioral system and its input/output values were updated every 300 ms while its fitness was computed. Therefore, the synaptic weights of this network were responsible for both the visual features on which the system based its behavior and for the motor actions necessary to search for those features.

In a first set of experiments, the neural network was embedded in a simulated pan–tilt camera and asked to discriminate between triangles and squares of different size that could appear at any location of a screen (Fig. 76.15a), a perceptual task similar to that explored with the gantry robot described in Sect. 76.5. The visual system was free to explore the image for 60 s while continuously reporting whether the current screen showed a triangle or a square. The fitness was proportional to

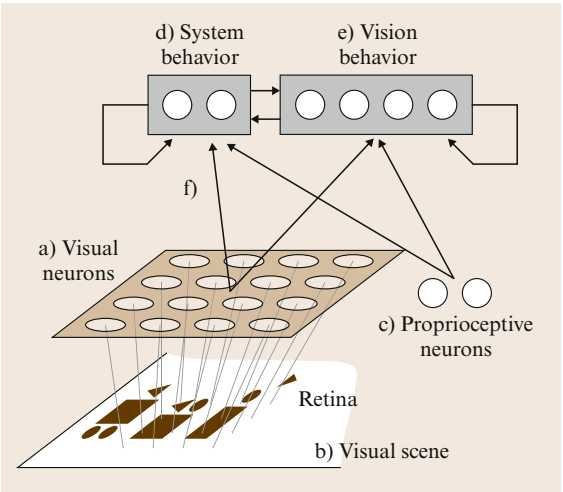


Fig.76.14a–f The neural architecture of the active vision system is composed of: **(a)** a grid of visual neurons with nonoverlapping receptive fields whose activation is given by **(b)** the grey level of the corresponding pixels in the image; **(c)** a set of proprioceptive neurons that provide information about the movement of the vision system; **(d)** a set of output neurons that determine the behavior of the system (pattern recognition, car driving, robot navigation); **(e)** a set of output neurons that determine the behavior of the vision system; and **(f)** a set of evolvable synaptic connections. The number of neurons in each subsystem can vary according to the experimental settings

the amount of correct responses accumulated over the 60 s for several screenshots containing various instances of the two shapes. Evolved systems were capable of correctly identifying the type of shape with 100% accuracy after a few seconds despite the fact that this recognition problem is not linearly separable and that the neural network does not have hidden units, which in theory are necessary to solve nonlinearly separable tasks. Indeed, the same neural network presented with the same set of images and trained with supervised learning, but without the possibility to actively explore the scene, was not capable of solving the task. The evolved active vision system developed sensitivity to vertical edges, oriented edges and corners, and used its movement to search for these features in order to tell whether the shape was a triangle or a square. These features, which are also found in the early visual system of almost all animals, are invariant to size and location.

In a second set of experiments, the neural network was embedded in a simulated car and was asked to drive over several mountain circuits (Fig. 76.15b). The simulator was a modified version of a car race video game. The neural network could move the retina across the scene seen through the windscreen at the driver's seat and control the steering, acceleration, and braking of the car. The fitness was inversely proportional to the time taken to complete the circuits without exiting the road. Evolved networks completed all circuits with time laps competitive to those of well-trained students controlling the car with a joystick. The evolved network started by searching for the edge of the road and tracked its relative position with respect to the edge of the windscreen in order to control steering and acceleration. This behavior was supported by the development of sensitivity to oriented edges.

In a third set of experiments, the neural network was embedded in a real mobile robot with a pan–tilt camera that was asked to navigate in a square arena with low walls located in an office (Fig. 76.16). The fitness was proportional to the amount of straight motion measured over two minutes. Robots that hit the walls because they watched people or other irrelevant features of the office had lower fitness than robots that could perform long straight paths and avoid walls of the arena. Evolved robots tended to fixate the edge between the floor and the walls of the arena, and turned away from the wall when the size of its retinal projection became larger than a threshold (VIDEO 36). This combination of sensitivity to oriented edges and looming is also found in the visual circuits of several insects and birds.

In a further set of experiments [76.85], the visual pathway of the neural network was augmented by an intermediate set of neurons whose synaptic weights could be modified by Hebbian learning [76.86] while the robot moved in the environment. All the other synaptic weights were genetically encoded and evolved. The results showed that lifelong development of the receptive fields improved the performance of evolved robots and allowed robust transfer of evolved neural controllers from simulated to real robots, because the receptive fields developed sensitivity to features encountered in the environment where they happen to be born (see also the section above on simulation and reality). Furthermore, the results showed that the development of visual receptive fields was significantly and consistently affected by active vision as compared to the development of receptive fields passively exposed to the same set of sample images. In other words, robots evolved with active vision developed sensitivity to a smaller subset of features in the environment and actively tracked those features to maintain a stable behavior.

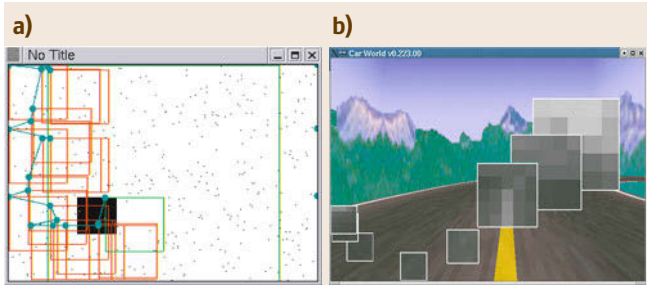


Fig. 76.15 **(a)** An evolved individual explores the screen searching for the shape and recognizes it by the presence of a vertical edge. **(b)** Search for the edge of the road at the beginning of a drive over a mountain road

Fig. 76.16 A mobile robot with a pan–tilt camera is asked to move within the walled arena in the office environment

76.7 Computational Neuroethology

Evolutionary robotics is also used to investigate open questions in neuroscience and cognitive science [76.87–90] because it offers the vantage point of a behavioral system that interacts with its environment [76.91]. Although the results should be carefully considered when drawing analogies with biological organisms, evolutionary robotics can generate and test hypotheses that could be further investigated with mainstream neuroscience methods.

For example, the active vision system with Hebbian plasticity described in the previous section was used to answer a question raised by *Held* and *Hein* [76.92] in the 1960s. The authors devised the apparatus shown in Fig. 76.17 where the free movements of a kitten (*active kitten*) were transmitted to a second kitten that was carried in a gondola (*passive kitten*). The second kitten could move its head, but its feet did not touch the ground. Consequently, the two kitten received almost identical visual stimulation, but only one of them received that stimulation as a result of body self-movement. After a few days in that environment, only the active kitten displayed normal behavior in several visually guided tasks. The authors suggested the hypothesis that proprioceptive motor information resulting from generation of actions was necessary for the development of normal, visually guided behavior.

The kitten experiments were replicated by cloning an evolved robot controller and randomly initializing the synaptic values of the adaptive visual pathways in both clones. One cloned robot was then left free to move in a square environment while the other cloned robot was forced to move along imposed trajectories, but was free to control its camera position, just like the passive kitten [76.94]. The results indicated that the visual receptive fields and behaviors of passive robots differ significantly from those of active robots. Furthermore, passive robots that were later left free to move were no longer capable of properly avoiding walls. A thorough analysis of neural activation correlated with behavior of the robot and even transplantation of neurons across active and passive robots revealed that the poor performance was due to the fact that passive robots could not completely select the visual features they were exposed to. Consequently, passive robots developed sensitivity to features that were not functional to their normal behavior and interfered with other dominant features in the visual field. Whether this explanation also hold for living animals remains to be further investigated, but at least these experiments indicated that motor feedback is not necessary to explain the pattern of pathological behavior observed in animals and robots.

76.7.1 Emergence of Place Cells

Let us now consider the case of an animal exploring an environment and periodically returning to its nest to feed. It has been speculated that this type of situation requires the formation of spatial representations of the environment that allow the animal to find its way home [76.95]. Different neural models with various degrees of complexity and biological detail that could provide such functionality have been proposed [76.96, 97].

Would a robot evolved under similar survival conditions develop a spatial representation of the environment and, if so, what type of representation would that be? These questions were explored using the same Khepera robot and evolutionary methodology described in Sect. 76.2 for reactive navigation in the looping maze. The environment was a square arena with a small patch on the floor in a corner where the robot could instantaneously recharge its (simulated) battery (Fig. 76.18). The environment was located in a dark room with a small light tower over the *recharging station*.

The sensory system of the robot was composed of eight distance sensors, two ambient-light sensors (one

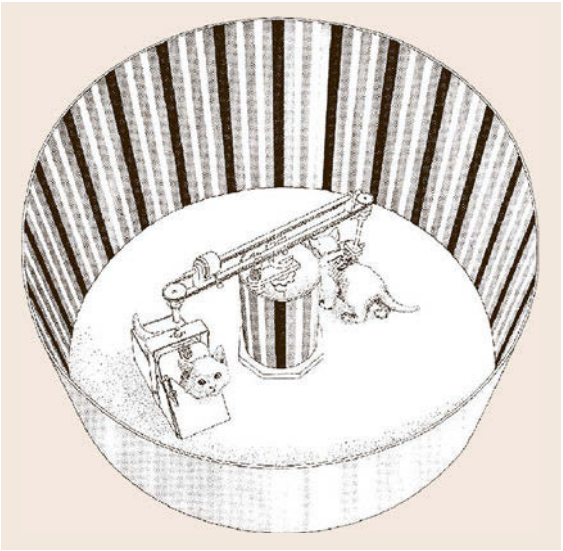


Fig. 76.17 The original apparatus in [76.92], where the gross movements of a kitten moving almost freely were transmitted to a second kitten that was carried in a gondola. Both kittens were allowed to move their head. They received essentially the same visual stimulation because of the unvarying pattern on the walls and the center post of the apparatus (after [76.93], with permission)

on each side), one floor-color sensor, and a sensor for battery charge level. The battery lasted only 20 s and had a linear discharge. The evolutionary neural network included five fully connected internal neurons between sensory and motor neurons. The same fitness function described in Sect. 76.2 for navigation in the looping maze was used, except for the middle term which had been used to encourage straight navigation in the looping maze. The fitness value was computed every 300 ms and accumulated over the life span of the individual. Therefore, individuals who discovered where the charger was could live longer and accumulate more fitness by exploring the environment (individuals were killed if they survived longer than 60 s to limit the experimentation time).

The same physical robot evolved for 10 days and nights as both the fitness and life span of individuals continued to increase (Fig. 76.19). After approximately 200 generations, the robot was capable of navigating around the environment, covering long trajectories while avoiding both walls and the recharging area (VIDEO 118). When the battery was almost discharged it initiated a straight navigation towards the recharging area and exited immediately after battery recharge to resume navigation. Best evolved individuals always entered the recharging area one or two seconds before full discharge of the battery. That implies that robots must somehow calibrate the timing and trajectory of their homing behavior depending on where they happened to be in the environment.

In order to understand how that behavior could possibly be generated, a set of neuroethological measures were performed using a laser positioning device that provided exact position and orientation of the robot every 300 ms. By correlating the robot position and behavior with the activation of the internal neurons in real time while the evolved individual freely moved in the environment, it was possible to see that some neurons specialized for reactive behaviors, such as obstacle avoidance, forward motion, and battery monitoring. Other neurons instead displayed more complex activation patterns. One of them revealed a pattern of activation levels that depended on whether the robot was oriented facing the light tower or facing the opposite direction (Fig. 76.20). In the former case, the activation pattern reflected zones of the environment and paths typically followed by the robot during exploration and homing. For example, the robot trajectory towards the recharging area never crossed the two *gate walls* visible in the activation maps around the recharging station. When the robot faced the opposite direction, the same neuron displayed a gradient field orthogonally aligned with the recharging area. This gradient provides an indication of the distance from the recharging area. Interestingly, this pattern of activity is not significantly affected by the charge level of the battery.

The functioning of this neuron reminds of the classic findings on the hippocampus of the rat brain where some neurons (also known as *place cells*) selectively fire when the rat is in specific areas of the environment [76.98]. Also, the orientation-specific pattern of neural activation measured on the evolved robot is rem-

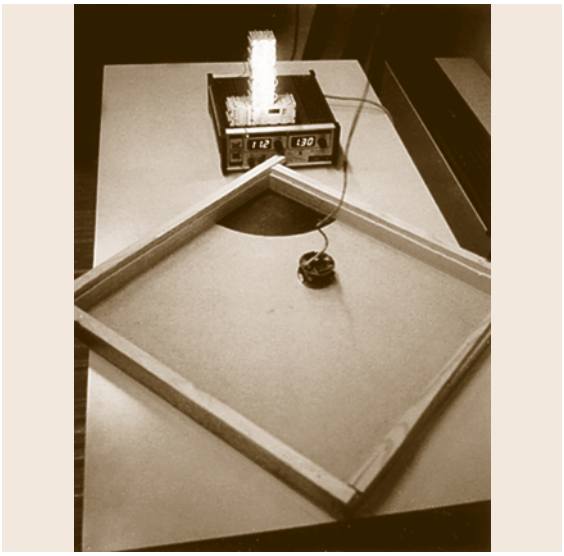


Fig. 76.18 Bird's eye view of the arena with the light tower over the recharging station and the Khepera robot

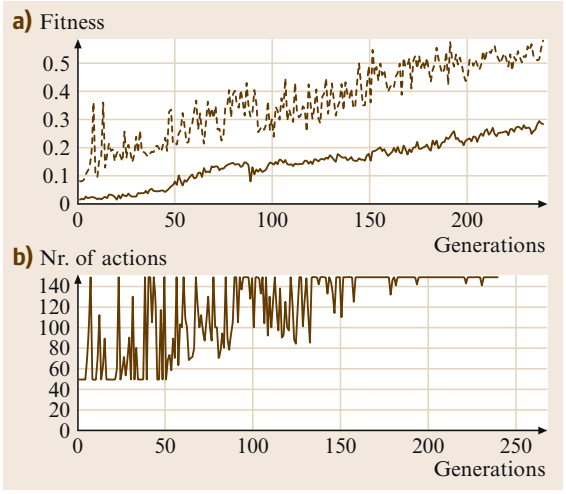


Fig. 76.19 **(a)** Average population fitness (*continuous line*) and fitness of the best individual (*dotted line*). **(b)** Life span of the best individuals measured as number of sensorimotor cycles, or actions. Individuals start with a full battery which lasts 50 actions (20 s), if not recharged. The maximum life span is 150 actions

iniscent of the so-called *head-direction neurons* in the rat hippocampus, which are positioned nearby place cells, whose firing patterns depend on the rat heading direction with respect to an environmental landmark [76.99]. Although the analogy between brains of evolved robots and of biological organisms should not be taken too literally, these results indicate that the two organisms converge towards a functionally similar neural strategy, which may be more efficient to address this type of situation than a strategy that does not rely on representations (but only on reactive strategies such as random motion, light following, or dead reckoning).

76.7.2 Spiking Neurons

The great majority of biological neurons communicate using self-propagating electrical pulses called *spikes*, but from an information-theoretic perspective it is not yet clear how information is encoded in the spike train. Connectionist models [76.100], by far the most widespread, assume that what matters is the *firing rate* of a neuron, that is, the average quantity of spikes emitted by the neuron within a relatively long time window (for example, over 100 ms). Alternatively, what matters is the average number of spikes of a small population of neurons at a give point. In these models the real-valued output of an artificial neuron represents the firing rate, possibly normalized relatively to the maximum attainable value. Pulsed models [76.101], instead, are based on the assumption that the *firing time*, that is, the precise time of emission of a single spike, may convey important information [76.102]. Spiking neuron models have slightly more complicated dynamics of synaptic and membrane integration. Depending on one's theory of what really matters, connectionist or spiking models are used.

However, designing circuits of spiking neurons that display a desired functionality is still a challenging task. The most successful results in the field of robotics obtained so far focused on the first stages of sensory processing and on relatively simple motor control [76.103, 104]. Despite these implementations, there are not yet methods for developing complex spiking circuits that could display minimally cognitive functions or learn behavioral abilities through autonomous interaction with a physical environment.

Artificial evolution represents a promising methodology to generate networks of spiking circuits with desired functionalities expressed as behavioral criteria (fitness function). Evolved networks could then be examined to detect what communication modality is used and how that correlates with observed behavior of the robot.

Floreano and *Mattiussi* [76.105] evolved a fully connected network of spiking neurons for driving a vision-based robot in an arena painted with black stripes of variable size against a white background (Fig. 76.21). The Khepera robot used in these experiments was equipped with a vision turret composed of one linear array of grayscale photoreceptors spanning a visual field of 36°. The output values of a bank

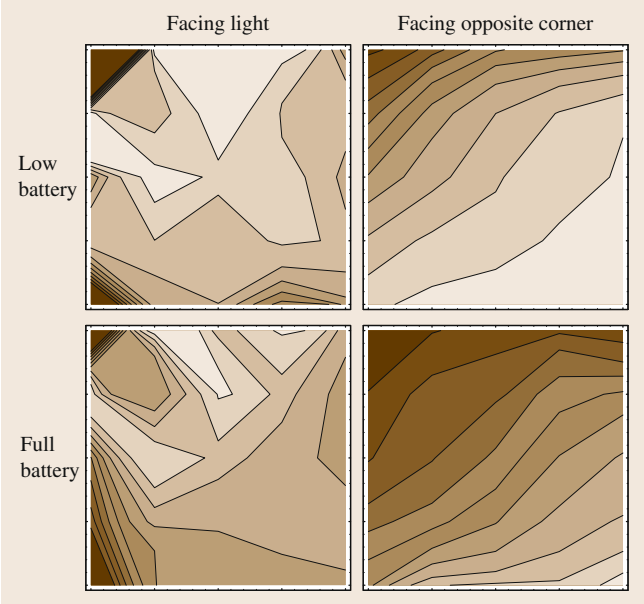


Fig. 76.20 Activation levels (brightness proportional to activation) of an internal neuron plotted over the environment while the robot was positioned at various locations in each of the four conditions (facing recharging area or not, discharged battery or not). The recharging area is located at the *top left corner* of each map

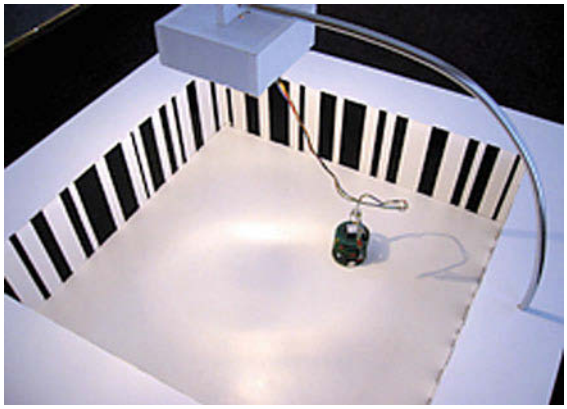


Fig. 76.21 A network of spiking neurons is evolved to drive the vision-based robot in the arena. The light below the rotating contacts allows continuous evolution also overnight

of local contrast detection filters were converted in spikes (the stronger the contrast, the larger the number of spikes per second) sent to ten fully connected spiking neurons implemented according to the spike response model [76.106]. The spike series of a subset of these neurons was translated into motor commands (more spikes per second corresponded to faster rotation of the wheel). The fitness function was the amount of forward translation of the robot measured over 2 min. Consequently robots that turned in place or hit the walls had comparatively lower fitness than robots that could move straight and turn only to avoid walls. The genome of these robots was a bit string that encoded only the sign of the neurons and the presence of synaptic connections. Existing connections were set to 1 and could not change during the lifetime of the robot.

Evolution reliably discovered very robust spiking controllers in approximately 20 generations, approximately 30 h of evolution on the real robot (VIDEO 37). Evolved robots could avoid not only the walls, but any object positioned in front of them. Detailed analysis of the best evolved controllers revealed that neurons did not exploit time differences between spikes, which one would have expected if optic flow was used to detect distance from walls. Instead, they simply used the number of incoming spikes (firing rate) as an indication of when to turn. When the robot perceived a lot of contrast it would go straight, but when the contrast decreased below a certain threshold (indicating that it approached an object), it started to turn away. This extremely efficient and simple result seems to be in contrast with theories of optic flow detection in insects and may be worth considering as an alternative hypothesis for vision-based behavior.

Spiking neural networks turned out to be more evolvable than connectionist models (at least for this task). One possible explanation is that spiking neurons have subthreshold dynamics that, to some extent, can be affected by mutations without immediately affecting the output of the network.

The robust results and compact genetic encoding encouraged the authors to use an even simpler model of spiking neuron so that the entire neural network could be mapped in less than 50 bytes of memory. The evolutionary algorithm was also reduced to a few lines of code and the entire system was implemented within a programmable intelligent computer (PIC) microcontroller without the need for any external computer for data storage. The system was used for a sugar-cube robot (Fig. 76.22) that autonomously and reliably developed the ability to navigate around a maze in less than an hour [76.107]. Interestingly, evolved spiking controllers developed a pattern of connections where spiking neurons received connections from a small patch of neighboring sensors, but not from other sensors, and were connected only to neighboring spiking neurons. This pattern of connectivity is also observed in biological systems and encourages specialization of neurons to sensory features.

76.7.3 GasNets

This section describes another style of artificial neural network strongly inspired by those parts of contemporary neuroscience that emphasize the complex electrochemical nature of real nervous systems. In particular, they make use of an analogue of *volume signaling*, whereby neurotransmitters freely diffuse into a relatively large volume around a nerve cell, potentially affecting many other neurons [76.108, 109]. This exotic form of neural signaling does not sit easily with classical pictures of brain mechanisms and is forcing a radical rethink of existing theory [76.110–113]. The class of artificial neural networks developed to explore artificial volume signaling are known as GasNets [76.114]. These are essentially standard neural networks augmented by a chemical signaling system comprising a diffusing *virtual* gas which can modulate the response of other neurons. A number of GasNet variants, inspired by different aspects of real nervous systems, have been explored in an evolutionary robotics context as artificial nervous systems for mobile autonomous robots. They have been shown to be significantly more evolvable, in terms of speed to a good solution, than other forms of neural networks for a variety of robot tasks and behaviors [76.38, 114–116]. They are being investigated as potentially useful engineering tools and as a way of gaining helpful insights into biological systems [76.112, 117–119].

By analogy with biological neuronal networks, GasNets incorporate two distinct signaling mechanisms, one *electrical* and one *chemical*. The underlying *electrical* network is a discrete-time-step recurrent neural network with a variable number of nodes. These nodes

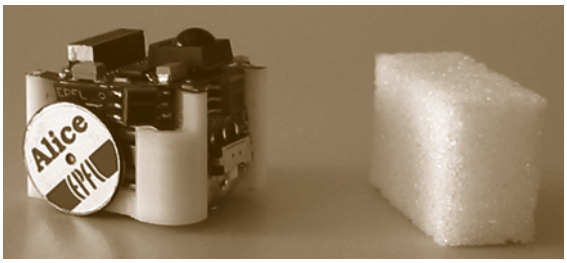


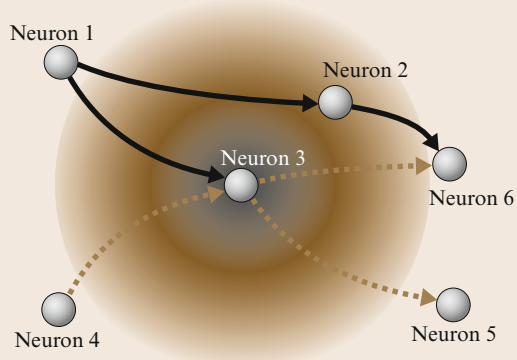
Fig. 76.22 The Alice sugar-cube robot equipped with the evolutionary spiking neural network implemented within its PIC microcontroller

are connected by either excitatory or inhibitory links (Fig. 76.23).

In addition to this underlying network in which positive and negative *signals* flow between units, an abstract process loosely analogous to the diffusion of gaseous modulators is at play. Some units can emit virtual *gases* which diffuse and are capable of modulating the behavior of other units by changing the profile of their output functions. The networks occupy a 2-D space; the diffusion processes mean that the relative positioning of nodes is crucial to the functioning of the network. Spatially, the gas concentration varies as an inverse exponential of the distance from the emitting node with a spread governed by a parameter r with the concentration set to zero for all distances greater than r. The total concentration of gas at a node is determined by summing the contributions from all other emitting nodes.

For mathematical convenience, in the original GasNet there are two *gases*, one whose modulatory effect is to increase the transfer function gain parameter and one whose effect is to decrease it. Thus the gas does not alter the electrical activity in the network directly but rather acts by continuously changing the mapping between input and output for individual nodes, either directly of by stimulating the production of further virtual gas. The general form of the diffusion is based on the properties of a (real) single-source neuron as modeled in detail by *Philippides* et al. [76.112, 117]. The modulation chosen is motivated by what is known of NO modulatory effects at synapses [76.120]. For full details see [76.114].

Various extensions of the basic GasNet have been produced. Two in particular are strongly inspired by contemporary neuroscience. The *plexus model* is directly inspired by a type of signaling seen in the mammalian cerebral cortex in which the NO signal is generated by the combined action of many fine NO-producing fibers, giving a targeted *cloud* which is distant from the neurons from which the fiber plexus emanates [76.118]. In the plexus GasNet, which models this form of signaling at an abstract level, the spatial distribution of gas concentration has been modified to be uniform over the area of affect. The center of this gas diffusion cloud is under genetic control and can be distant from the controlling node (which, by analogy, is the source of the *plexus*) [76.116]. All other details of the models are identical to the original GasNet model, as described earlier. The *receptor* GasNet incorporates an aspect of biological neuronal networks that has no analog in the vast majority of artificial neural networks (ANNs): the role of receptor molecules. Although neuroscience is a long way from a full understanding of receptor mechanisms, a number of powerful systems level ideas can be abstracted.



A GasNet. Neuron 3 is emitting gas, and modulating neuron 2 despite there being no synaptic connection.

Fig. 76.23 A basic GasNet showing positive (*solid*) and negative (*dashed*) *electrical* connections and a diffusing virtual gas creating a *chemical* gradient

Details of the receptor variant are similar to the basic GasNet except there is now *only one* virtual gas and each node in the network can have one of three discrete quantities (zero, medium, maximum) of a number of possible receptors. The modulation the diffusing neurotransmitter affects at a neuron depends on which receptors are present. The strength of a modulation at a node is proportional to the product of the gas concentration at the node and the relevant receptor quantity. In the original GasNet, any node that was in the path of a diffusing transmitter would be modulated in a fixed way. The receptor model allows site-specific modulations, including no modulation (no receptors) and multiple modulations at a single site ([76.116] for further details).

Although most of the GasNet variants described in this section have been successfully used in a number of robotic tasks, their evolvability and other properties were thoroughly compared on a version of the (gantry) robot visual discrimination task described in Sect. 76.5, VIDEO 375. All aspects of the networks were under genetic control: the number of nodes, the connectivity and, in the case of the GasNets, all parameters governing volume signaling (including the position of the nodes and whether or not they were *virtual gas* emitters). The visual sampling morphology was also under evolutionary control. The original basic GasNet was found to be significantly more evolvable than a variety of other styles of connectionist neural networks as well as a GasNet with the volume signaling disabled. Successful GasNet controllers for this task tended to be rather minimal, in terms of numbers of nodes and connections, while possessing complex dynamics [76.114].

Later experiments comparing the basic GasNet with the plexus and receptor variants showed the latter two to be considerably more evolvable than the former, with the receptor GasNet being particularly successful [76.116]. These GasNet experiments demonstrated that the intricate network dynamics made possible by the artificial volume signaling mechanisms can be readily harnessed to generate adaptive behaviors in autonomous agents. They also throw up such questions as why GasNets are more evolvable than many other forms of ANN and why there is a difference in evolvability between GasNet variants. In order to gain insight into what factors are most important in GasNet evolvability, several other varieties were studied, including non-spatial GasNets where the diffusion process is replaced by explicit gas connections with complex dynamics and version with other forms of modulation and diffusion [76.119]. Detailed comparative studies of these variants with each other, and with other forms of ANN, were performed using the visual discrimination task described above [76.116, 119].

The comparative studies revealed that the rich dynamics and additional timescales introduced by the gas played an important part in enhanced evolvability, but were not the whole story [76.116, 119]. The particular form of modulation was also important – multiplicative or exponential modulation (in the form of changes to the transfer function) were found to be effective, but additive modulations were not. The former kind of modulations may well confer evolutionary advantages by allowing nodes to be sensitive to different ranges of input (internal and sensory) in different contexts. The spatial embedding of the networks also appears to play a role in producing the most effective coupling between the two distinct signalling processes (*electrical* and *chemical*). By exploiting a loose, flexible coupling between the two processes, it is possible to significantly reduce destructive interference between them, allowing one to be *tuned* against the other while searching for good solutions [76.115, 116, 121]. Similar forces may be at play in spiking neural networks, where sub-threshold and spiking dynamics interact with each other, which although not yet compared to GasNets, were shown to be more evolvable than connectionist networks. Measurements of the degree of coupling in the GasNets variants versus speed of evolution supported this view [76.116]; the receptor GasNet, for which the evolutionary search process has the most direct control over the degree of coupling between the signaling processes, and which has a bias towards a loose coupling, was by far the most evolvable [76.116].

Analysis of GasNet solutions often reveals high levels of degeneracy, with functionally equivalent subnetworks occurring in many different forms, some involving gas and some not [76.121]. Their genotype to phenotype mapping (where the phenotype is robot behavior) is also highly degenerate with many different ways of achieving the same outcome (e.g., moving node positions, changing gas diffusion parameters or adding new connections can all have the same effect). This is especially true when variable length genotypes are used to efficiently sculpt solutions in a search space of variable dimensions. The levels of degeneracy are generally significantly higher than when using connectionist networks. These properties partly explain the robustness and adaptability of GasNets in noisy environments and are another important factor in their evolvability (there are many paths to the same phenotypical outcome with reduced probabilities of lethal mutations) [76.119, 122]. In the most successful varieties of GasNet, multi-scale dynamics, modulation and spatial embedding act in concert to produce highly evolvable degenerate networks.

These and ongoing investigations indicate that explicitly dealing with the electrochemical nature of nervous systems is likely to be an increasingly fruitful area of research, both for evolutionary robotics and for neuroscience, that will likely force us to broaden our notions of what behavior-generating mechanisms might look like.

Because of its ability to explore whole classes of underspecified models, ER is being increasingly used to develop or explore neural models aimed at answering specific questions in neuroscience [76.88–90] or to probe new theories about possible neural mechanisms [76.90]. One intriguing recent hypothesis is that one of the forms of plasticity on which the brain relies is itself a form of evolution via natural selection acting within neural tissue [76.123, 124]. The units of selection in this case are activity and connection patterns which are copied between groups of neurons. Irrespective of whether or not it occurs in nature (and it might), this kind of mechanism could be employed in a whole new kind of evolutionary robotics.

76.8 Evolution and Learning

Evolution and learning (or phylogenetic and ontogenetic adaptation) are two forms of biological adaptation that differ in space and time. Evolution is a process of selective reproduction and substitution based on the existence of a population of individuals displaying variability at the genetic level. Learning, instead,

is a set of modifications taking place within each single individual during its own life time. Evolution and learning operate on different time scales. Evolution is a form of adaptation capable of capturing relatively slow environmental changes that might encompass several generations (e.g., the perceptual characteristics of food sources for a given species). Learning, instead, allows an individual to adapt to environmental modifications that are unpredictable at the generational level. Learning might include a variety of mechanisms that produce adaptive changes in an individual during its lifetime, such as physical development, neural maturation, variation of the connectivity between neurons, and synaptic plasticity. Finally, whereas evolution operates on the genotype, learning affects only the phenotype, and phenotypic modifications cannot directly modify the genotype.

Researchers have combined evolutionary techniques and learning techniques (supervised or unsupervised learning algorithm such us reinforcement learning or Hebbian learning; for a review see [76.125]). These studies have been conducted with two different purposes:

1. Identifying the potential advantage of combining these two methods from the point of view of developing robust and effective robots.
2. Understanding the role of the interaction between learning and evolution in nature.

Within an evolutionary perspective, learning has several different adaptive functions. First, it might allows individuals to adapt to changes that occur too quickly to be tracked by evolution [76.126]. Secondly, learning might allows robots to use information extracted during their interaction with environment to develop adaptive characters ontogenetically without necessarily discovering these characters through genetic variations and without encoding these characters in their genome. To understand the importance of this aspect, we should consider that evolutionary adaptation is based on an explicit but concise indication of how well an individual robot coped with its environment – the fitness value of a robot. Ontogenetic adaptation, on the contrary, is based on extremely rich information – the state of the sensors while the robot interacts with its environment. This huge amount of information encodes very indirectly how well an individual is doing in different phases of its lifetime or how it should modify its behavior to increase its fitness. However, evolving robots that have acquired a predisposition to exploit this information to produce adaptive changes during their lifetime might be able to develop adaptive characteristics on the fly, thus leading to the possibility to produce *complex* phenotypes on the basis of parsimonious genotypes. Finally, learning can help and guide evolution. Although physical changes of the phenotype, such as strengthening of synapses during learning, cannot be written back into the genotype, *Baldwin* [76.127] and *Waddington* [76.128] suggested that learning might indeed affect the evolutionary course in subtle but effective ways. Baldwin's argument was that learning accelerates evolution because suboptimal individuals can reproduce by acquiring during life necessary features for survival. However, variation occurring during successive generation might lead to the discovery of genetic traits that lead to the establishment of the same characteristics that were previously acquired thorough lifetime learning. This latter aspect of Baldwin's effect, namely indirect genetic assimilation of learned traits, has been later supported by scientific evidence and defined by *Waddington* [76.128] as a canalization effect.

Learning however, also has costs such as: (1) a delay in the ability to acquire fitness (due to the need to develop fit behavior ontogenetically), and (2) increased unreliability due to the fact that the possibility to develop certain abilities ontogenetically is subjected to partially unpredictable characteristics of the robot–environment interaction [76.129]. In the next two subsections we describe two experiments that show some of the potential advantages of combining evolution and learning.

76.8.1 Learning to Adapt to Fast Environmental Variations

Consider the case of a Khepera robot that should explore an arena surrounded by black or white walls to reach a target placed in a randomly selected location [76.126]. Evolving robots are provided with eight sensory neurons that encode the state of the four corresponding infrared sensors and two motor neurons that control the desired speed of the two wheels. Since the color of the walls change every generation and since the color significantly affects the intensity of the response of the infrared sensors, evolving robots should develop an ability to *infer* whether they are currently located in an environment with white or black walls and learn to modify their behavior during lifetime. That is, robots should avoid walls only when the infrared sensors are almost fully activated in the case of arenas with white walls, while they should avoid walls even when the infrared sensors are slightly activated in the case of arenas with black walls.

Robots were provided with a neural controller (Fig. 76.24) including four sensory neurons that encoded the state of four corresponding infrared sensors;

two motors neurons that encoded the desired speed of the two wheels; and two teaching neurons that encoded the teaching values used to modify the connection weights from the sensory neurons to the motor neurons during the robots' lifetime. This special architecture allows evolving robots to transform the sensory states experienced by the robots during their lifetime into teaching signals that might potentially lead to adaptive variations during lifetime. Analysis of evolved robots revealed that they developed two different behaviors that are adapted to the particular arena where they happen to be *born* (surrounded by white or black walls). Evolving robots did not inherit an ability to behave effectively, but rather a predisposition to learn to behave. This predisposition to learn involves several aspects such as a tendency to experience useful learning experiences, a tendency to acquire useful adaptive characters through learning, and a tendency to channel variations toward different directions in different environmental conditions [76.126].

76.8.2 Evolution of Learning

In the previous example, the evolutionary neural network learned using a standard learning rule that was applied to all synaptic connections. *Floreano* and *Mondada* [76.130] explored the possibility of genetically encoding and evolving the learning rules associated to the different synaptic connections of a neural network embedded in a real robot. The main motivation of this line of work was to evolve robots capable of adapting to a partially unknown environment, rather than robots adapted to the environment(s) seen during evolution. In order to prevent evolutionary tuning of the neural network to the specificities of the evolutionary environment (which would limit transfer to different environments or transfer from simulation to reality), the synaptic weight values were not genetically encoded. Instead, each synaptic connection in the network was described by three genes that defined its sign, its learning rule, and its learning rate (Fig. 76.25). Every time a genome was decoded into a neural network and downloaded onto the robot, the synaptic strengths were initialized to small random values and could change according to the genetically specified rules and rates while the robot interacted with the environment. Variations of this methodology included a more compact genetic encoding where the learning properties were associated to a neuron instead of a synapse. All synapses afferent to a neuron used its genetically specified rules and rates. Genes could encode four types of Hebbian learning that were modeled upon neurophysiological data and were complementary to each other [76.131].

Experimental results in a nontrivial, multitask environment (Fig. 76.26, VIDEO 40) indicated that this methodology has a number of significant advantages with respect to the evolution of synaptic strengths without learning [76.48]. Robots evolved faster and obtained better fitness values. Furthermore, evolved behaviors were qualitatively different, notably in that they did not exploit minimal solutions tuned to the environment (such as turning only on one side, or turning in circles tuned to the dimensions of the evolutionary arena). Most important, these robots displayed remarkable adaptive properties after evolution. Best evolved individuals: (1) transferred perfectly from simulated

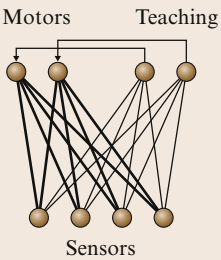


Fig. 76.24 A self-teaching network. The output of the two teaching neurons is used as a teaching value for the two motor neurons. The weights that connect the sensory neurons to the teaching neurons do not vary during the robots' lifetime while the weights that connect the sensory neurons to the motor neurons are modified with an error-correction algorithm

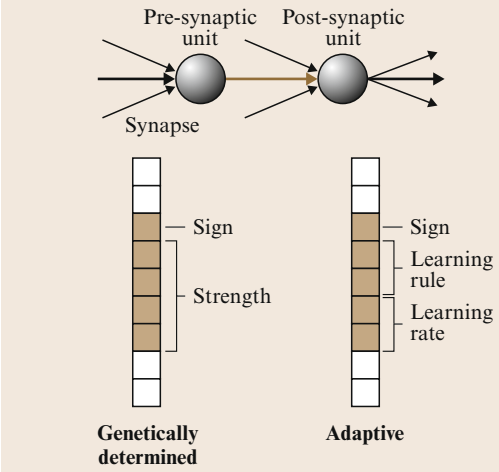


Fig. 76.25 Two methods for genetically encoding a synaptic connection. Genetically determined synapses cannot change during the lifetime of the robot. Adaptive synapses instead are randomly initialized and can change during lifetime of the robot according to the learning rules and rates specified in the genome

to physical robots, (2) accomplished the task when the light and reflection properties of the environment were modified, (3) accomplished the task when key landmarks and target areas of the environment were displaced, and (4) transferred well across morphologically different robotic platforms. In other words, these robots were selected for their ability to solve a partially unknown problem by adapting on the fly, rather than for being a solution to the problem seen during evolution.

In further experiments where the genetic code for each synapse of the network included one gene whose value caused its remaining genes to be interpreted as connection strengths or learning rules and rates, 80% of the synapses *made the choice* of using learning, reinforcing the fact that this genetic strategy has a comparatively stronger adaptive power [76.131]. This methodology could also be used to evolve the morphology of neural controllers were synapses are created at runtime and therefore their strengths cannot be genetically specified [76.132]. Recently, the adaptive properties of this type of adaptive genetic encoding were confirmed also in the context of evolutionary spiking neurons for robot control [76.133].

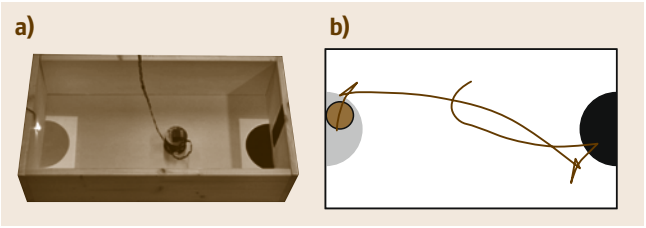


Fig. 76.26 (a) A mobile robot Khepera equipped with a vision module can gain fitness points by staying on the *grey area* only when the light is on. The light is normally off, but it can be switched on if the robot passes over the *black area* positioned on the other side of the arena. The robot can detect ambient light and wall color, but not the color of the floor. (b) Behavior of an individual evolved in simulation with genetic encoding of learning rules

76.9 Evolution of Social Behavior

In the previous sections, we limited our analysis to individual behaviors, i. e., to the evolution of robots placed in an environment that does not include other robots. The evolutionary method, however, can also be applied to evolve social behaviors in which multiple robots situated in the same environment interact between themselves in cooperative or competitive manners.

As we will see, competitive co-evolution is particularly interesting from the point of view of synthesizing progressively more complex capacities and from the point of view of developing solutions that are robust with respect to environmental variations. Cooperative evolution instead is particularly interesting for the possibility to solve problems that cannot be handled by a single robot, because of physical constraints or limited behavioral capabilities [76.134] and to develop solutions that are robust.

76.9.1 Coevolving Predator and Prey Robots

Competitive coevolution, for example the coevolution of two populations of predator and prey robots that are evolved for the ability to catch prey and to escape predators, respectively, has two characteristics that are particularly interesting from an evolutionary robotics perspective. The first aspect is that the competition between populations with different interests might spontaneously lead to a sort of incremental evolutionary process where evolving individuals are faced with progressively more complex challenges (although this is not necessarily the case). Indeed, in initial generations the task of the two populations is relatively simple because opponents have simple and poorly developed abilities on average. After a few generations, however, the abilities of the two populations increase and, consequently, the challenges for each population become more difficult. The second aspect consists of the fact that the environment varies across generations because it includes other coevolving individuals. This implies that coevolving individuals should be able to adapt to ever-changing environments and to develop behaviors that are robust with respect to environmental variations [76.135].

The potential advantages of competitive coevolution for evolutionary robotics have been demonstrated by a set of experiments conducted by *Floreano* and *Nolfi* [76.136, 137] where two populations of robots were evolved for the ability to catch prey and escape predators, respectively (Fig. 76.27).

The results indicated that both predator and prey robots tended to vary their behavior throughout generations without converging on a stable strategy. The behavior displayed by individuals at each generation tended to be tightly adapted to the counter-strategy exhibited by the opponent of the same generation (VIDEO 38). This evolutionary dynamic however does not really lead to long-lasting progress because, after an initial evolutionary phase, the coevolutionary

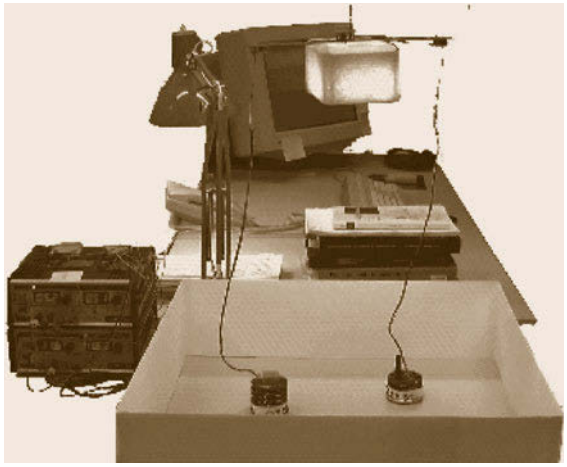


Fig. 76.27 Experimental setup. The predator and prey robot (*from left to right*) are placed in an arena surrounded by walls and are allowed to interact for several trials starting at different randomly generated orientations. Predators are selected on the basis of the percentage of trials in which they are able to catch (i. e., to touch) the prey, and prey on the basis of the percentage of trials in which they were able to escape (i. e., to not be touched by) predators. Predators have a vision system, whereas the prey have only short-range distance sensors, but can go twice as fast as the predator. Collision between the robots is detected by a conductive belt at the base of the robots

Fig. 76.28 An s-bot and a simulated swarm-bot consisting of four s-bots assembled in chain formation

process led to a limit cycle dynamic where the same small set of behavioral strategies recycled over and over again along generations [76.137]. This limit cycle dynamic can be explained by considering that prey robots tended to vary their behavior in order to disorient predators as soon as predators become effective against the current behavioral strategies exhibited by prey robots.

However, experiments [76.138] where robots were allowed to change their behavior on the fly on the basis of unsupervised Hebbian learning rules showed that the evolutionary phase where coevolving robots were able to produce real progress was significantly longer, and evolved predators displayed an ability to effectively cope with prey exhibiting different behavioral strategies by adapting their behavior on the fly to the prey's behavior. Prey instead tended to display behavior that changed in unpredictable ways.

Further experiments showed that competitive coevolution can solve problem that the evolution of a single population cannot. *Nolfi* and *Floreano* [76.137] demonstrated that the attempt to evolve predators robot for the ability to catch a fixed pre-evolved prey produced lower performance with respect to control experiments where predators and prey were coevolved at the same time.

76.9.2 Evolving Cooperative Behavior

As testified by social insects, colonies of simple cooperating individuals can display remarkable capacities and exhibit self-organising behaviors in which the spatio-temporal pattern observed at the system level emerge from numerous interactions among the individual robots. On the other hand, designing collective robotic systems of this sort constitutes a difficult problem due to the indirect relationship between the desired group behavior and the characteristics of the individual robots. By evaluating the robotic system as a whole (i. e., by selecting the robots on the basis of the global behavior that emerge from a large number of robot/environmental and robot/robot interactions), Evolutionary Robotics provides a means for discovering effective behavioral solutions and simple and robust control policies [76.139].

Recent research showed that teams of evolved robots can:

1. Develop robust and effective coordinated behavior [76.140, 141]
2. Collaborate by assuming complementary role [76.141, 142] (VIDEO 376)
3. Display self-organizing properties [76.143]
4. Develop and use communicative capabilities [76.144–146].

Moreover, some of the research carried in this area demonstrated how evolutionary robotics experiments can contribute to model biological phenomena, e.g., to identify the evolutionary conditions that enable the emergence of cooperative communicative behaviors [76.146] or the mechanisms enabling the evolution of effective division of labour strategies [76.147]

Here we briefly review a series of experiments where swarm-bots [76.148], i. e., groups of autonomous robots capable of assembling by physically connecting together, were evolved for the ability to display coordinated motion (VIDEO 115). Each individual robot consisted of a main platform (chassis) and turret that could actively rotate with respect to each other (Fig. 76.28). The chassis included tracks with teethed wheels for navigation on both rough and flat terrain, and infrared sensors pointing to the ground. The turret included a gripper, sixteen light-sensors distributed around the body, a loudspeaker, three microphones, and a traction sensor placed between the turret and the chassis to detect the direction and the intensity of the traction force that the turret exerts on the chassis. Swarm-bots were formed by several robots provided with identical neural controllers and assembled together so to form a single physical entity.

By evolving the neural controllers of these Swarm-bots, *Baldassare* et al. [76.140] demonstrated how the robots can display a robust and effective coordinated capacities that allow the individuals to negotiate and converge on a coherent direction and to keep moving along that direction by compensating the disalignments originating during motion. Such behavioral capacity was robust enough to allow a smooth transfer from simulation to reality and to allow the robots to to generalize their capacity to rugged terrains. In an extended experiments in which the s-bots were also equipped with infrared sensors, speakers and microphones, the evolved swarm-bots also showed a capacity to avoid dangers (e.g., holes) by coordinately changing direction as soon as one s-bot detected a hole [76.149].

Evolved swarm-bots generalized their coordinated motion capabilities also when they were tested in different conditions (e.g., when they were assembled in much more numerous groups and/or in different topologies, or when they had to also carry heavy objects by pushing and pulleding them in a coordinated manner). Finally, when placed in new environmental conditions (e.g., in environment with obstacle and walls), the swarm-bots spontaneously displayed new behavioral skills (related to acquired skills), such as the ability to cooperatively avoid obstacles, without any further adaptation. This ability to display new related behaviors, in new behavioral conditions, emerged as a result of the dynamical process originating from the interaction of the same robots with the new environmental conditions [76.142, 150].

76.9.3 Evolution of Communication

Communication represents a key aspect in collective behaviors. Recent research in evolutionary robotics has demonstrated how sophisticated communication capabilities can emerge and evolve in population of robots selected for the ability to perform tasks requiring coordination and/or cooperation.

The analysis of these experiments indicate that communicative interactions often originate as the result of cues, that provide useful information to other robots, produced inadvertently during the execution of specific behaviors [76.146, 151]. The presence of these cues create the basis for the development of an ability to react to them in an adaptive way thus leading to the establishment of adaptive communicative interactions in which robots produce signals and react to detected signals adaptively. The establishment of these forms of communicative interactions then create the adaptive conditions for the co-evolution of signalling and response strategies [76.152] (VIDEO 117).

The reliability and stability of the resulting communication system depend on the level of relatedness (i. e., genetic similarity) between robots and the level at which they were selected [76.146]. Robots that are genetically higly related or that are selected on the basis of the behavior exhibited by the group evolve reliable signals and stable communicative conventions. In contrast, when relatedness between robots is low and selection is acting at the level of the individuals, the evolutionary process might lead to the emergence of instable, ineffective and in some case deceptive communication forms [76.153, 154].

The evolution of communication is strongly interlinked with the evolution of other behavioral capacities [76.155]. Indeed, after all, robots need to develop appropriate behaviors to access and/or generate the information to be communicated and/or to react to detected signals appropriately. The co-adaptation of behavioral and communicative skills might lead to prolonged innovation phases in which the development of behavioral capacities create the adaptive conditions for the development of communication capacities and vice versa [76.151, 152]. Moreover, the co-adaptation of behavioral and communication capacities tend to lead to highly contingent evolutionary processes in which the capacities possessed by the population at a certain evolutionary phase strongly influence the outcome of the successive phases [76.152, 156] (VIDEO 117).

76.10 Evolutionary Hardware

In recent years, technology advancements have allowed researchers to explore evolution of electronic circuits. In this section, we briefly summarize some foundational work in this direction.

76.10.1 Evolvable Hardware Robot Controllers

In most of the work discussed so far some form of genetically specified neural network, implemented in software, has been at the center of the robot control system. Work on a related approach of evolving control systems directly onto hardware dates back to *Thompson*'s work in the mid 1990s [76.157]. In contrast to hardware controllers that are designed or programmed to follow a well-defined sequence of instructions, evolved hardware controllers are directly configured by evolution and then allowed to behave in real time according to semiconductor physics. By removing standard electronics design constraints, the physics can be exploited to produce highly nonstandard and often very efficient and minimal systems [76.158].

Thompson [76.157] used artificial evolution to design an onboard hardware controller for a two-wheeled autonomous mobile robot engaged in simple wall-avoidance behavior in an empty arena. Starting from a random orientation, and position near the wall, the robot had to move to the center of the arena and stay there using limited sensory input (Fig. 76.29). The direct current (DC) motors driving the wheels were not allowed to run in reverse and the robot's only sensors were a pair of time-of-flight sonars rigidly mounted on the robot, pointing left and right.

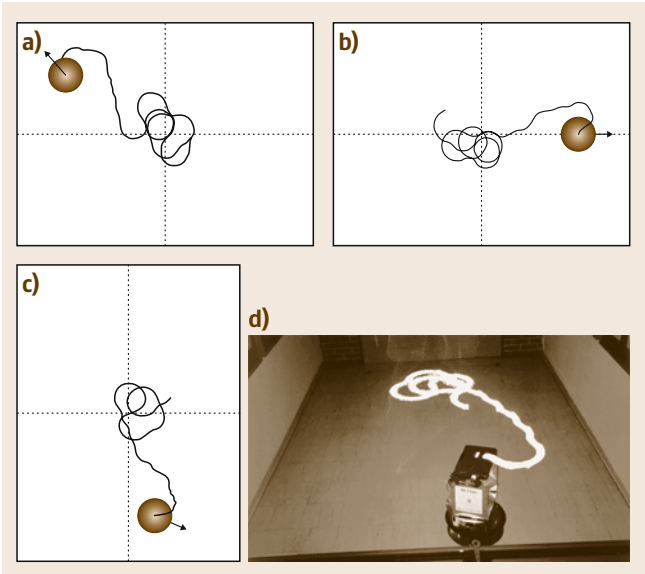


Fig.76.29a–d Wall-avoidance behavior of a robot with an evolved hardware controller in virtual reality (**a–c**) and the real world (**d**)

Thompson's approach made use of a so-called dynamic state machine (DSM) – a kind of generalized read-only memory (ROM) implementation of a finite-state machine where the usual constraint of strict synchronization of input signals and state transitions are relaxed (in fact put under evolutionary control). The system had access to a global clock whose frequency was also under genetic control. Thus evolution determined whether each signal was synchronized to the clock or allowed to flow asynchronously. This allowed the evolving DSM to be tightly coupled to the dynamics of interaction between the robot and environment and for evolution to explore a wide range of systems dynamics. The process took place within the robot in a kind of *virtual reality* in the sense that the real evolving hardware controlled the real motors, but the wheels were just spinning in the air. The movements that the robot would have actually performed if the wheels had been supporting it were then simulated and the sonar echo signals that the robot was expected to receive were supplied in real time to the hardware DSM. Excellent performance was attained after 35 generations, with good transfer from the virtual environment to the real world (Fig. 76.29).

Shortly after this research was performed, particular types of field programmable gate arrays (FPGAs) which were appropriate for evolutionary applications became available. FPGAs are reconfigurable systems allowing the construction of circuits built from basic logic elements. Thompson exploited their properties to demonstrate evolution directly in the chip. By again relaxing standard constraints, such as synchronizing all elements with a central clock, he was able to develop very novel forms of functional circuits, including a controller for a Khepera robot using infrared sensors to avoid obstacles [76.158, 159].

Following Thompson's pioneering work, *Keymeulen* et al. evolved a robot control system using a Boolean function approach implemented on gate-level evolvable hardware [76.160]. This system acted as a navigation system for a mobile robot capable of locating and reaching a colored ball while avoiding obstacles. The robot was equipped with infrared sensors and an vision system giving the direction and distance to the target. A programmable logic device (PLD) was

used to implement a Boolean function in its disjunctive form. This work demonstrated that such gate-level evolvable hardware was able to take advantage of the correlations in the input states and to exhibit useful generalization abilities, thus allowing the evolution of robust behavior in simulation followed by a good transfer into the real world.

In a rather different approach, *Ritter* et al. used an FPGA implementation of an onboard evolutionary algorithm to develop a controller for a hexapod robot [76.161]. *Roggen* et al. devised a multicellular reconfigurable circuit capable of evolution, self-repair, and adaptation [76.162], and used it as a substrate for evolving spiking controllers of a wheeled robot [76.163]. Although evolved hardware controllers are not widely used in evolutionary robotics, they still hold out the promise of some very useful properties, such as robustness to faults, which make them interesting for extreme condition applications such as space robotics.

76.11 Closing Remarks

Evolutionary robotics is a young and integrated approach to robot development without human intervention where machines change and adapt by capitalizing on the interactions with their environment. Despite initial skepticism by mainstream and applied robotics practitioners and even by pioneers of this approach [76.164], over the years the field has been constantly growing with new methods and approaches for evolving more complex, efficient, and sometimes surprising robotic systems. In some areas, such as morphology and self-assembly, evolutionary robotics is still the most widely used and powerful approach.

Evolutionary robotics is not only a method for automatic robot development inspired by biology, but also a tool for investigating open questions in biology concerning evolutionary, developmental, and brain dynamics. Its richness and fecundity make us believe that this approach will continue to grow and progress towards the creation of a new species of machines capable of self-evolution.

To gain a pratical knowledge, interested readers might use software libraries such as framework for autonomous robotics simulation and analysis (FARSA) [76.165], an open-software tool that permit to carry on evolutionary robotics experiments based on a variety of robotic platforms and to replicate and vary some of the experiments described in this chapter [76.166].

Video-References

VIDEO 36 Visual navigation of mobile robot with pan-tilt camera
available from http://handbookofrobotics.org/view-chapter/76/videodetails/36

VIDEO 37 Visual navigation with collision avoidance
available from http://handbookofrobotics.org/view-chapter/76/videodetails/37

VIDEO 38 Coevolved predator and prey robots
available from http://handbookofrobotics.org/view-chapter/76/videodetails/38

VIDEO 39 Evolution of collision-free navigation
available from http://handbookofrobotics.org/view-chapter/76/videodetails/39

VIDEO 40 Online learning to adapt to fast environmental variations
available from http://handbookofrobotics.org/view-chapter/76/videodetails/40

VIDEO 41 iCub language comprehension
available from http://handbookofrobotics.org/view-chapter/76/videodetails/41

VIDEO 114 Resilent machines through continuous self-modeling
available from http://handbookofrobotics.org/view-chapter/76/videodetails/114

VIDEO 115 A swarm-bot of eight robots displaying coordinated motion
available from http://handbookofrobotics.org/view-chapter/76/videodetails/115

VIDEO 116 Discrimination of objects through sensory-motor coordination
available from http://handbookofrobotics.org/view-chapter/76/videodetails/116

VIDEO 117 Evolution of cooperative and communicative behaviors
available from http://handbookofrobotics.org/view-chapter/76/videodetails/117

VIDEO 118 Exploration and homing for battery recharge
available from http://handbookofrobotics.org/view-chapter/76/videodetails/118

VIDEO 119 Introduction to evolutionary robotics at EPFL
available from http://handbookofrobotics.org/view-chapter/76/videodetails/119
VIDEO 371 Evolution of visually guided behavior on Sussex gantry robot
available from http://handbookofrobotics.org/view-chapter/76/videodetails/371
VIDEO 372 Evolved walking in an Octpod
available from http://handbookofrobotics.org/view-chapter/76/videodetails/372
VIDEO 373 Evolved homing walk on rough ground
available from http://handbookofrobotics.org/view-chapter/76/videodetails/373
VIDEO 374 Evolved bipedal walking
available from http://handbookofrobotics.org/view-chapter/76/videodetails/374
VIDEO 375 Evolved GasNet visualization
available from http://handbookofrobotics.org/view-chapter/76/videodetails/375
VIDEO 376 Evolved group coordination
available from http://handbookofrobotics.org/view-chapter/76/videodetails/376
VIDEO 771 Morphological change in an autonomous robot
available from http://handbookofrobotics.org/view-chapter/76/videodetails/771
VIDEO 772 More complex robots evolve in more complex environments
available from http://handbookofrobotics.org/view-chapter/76/videodetails/772

References

76.1 S. Nolfi, D. Floreano: *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines* (MIT/Bradford, Cambridge 2000)
76.2 J.H. Holland: *Adaptation in Natural and Artificial Systems* (Univ. of Michigan Press, Ann Arbor 1975)
76.3 A.M. Turing: Computing machinery and intelligence, Mind **LIX 236**, 433–460 (1950)
76.4 V. Braitenberg: *Vehicles. Experiments in Synthetic Psychology* (MIT, Cambridge 1984)
76.5 R.D. Beer: *Intelligence as Adaptive Behavior: An Experiment in Computational Neuroethology* (Academic, Boston 1990)
76.6 D. Parisi, F. Cecconi, S. Nolfi: Econets: Neural networks that learn in an environment, Network **1**, 149–168 (1990)
76.7 P. Husbands, I. Harvey: Evolution versus design: Controlling autonomous robots, Integrating Percept. Plan. Action, Proc. 3rd IEEE Annu. Conf. Artif. Intell. Simul. Plan. (1992) pp. 139–146
76.8 D. Floreano, O. Miglino, D. Parisi: Emergent complex behaviors in ecosystems of neural networks. In: *Parallel Architectures and Neural Networks*, ed. by E. Caianiello (World Scientific, Singapore 1991)
76.9 R.A. Brooks: Intelligence without representation, Artif. Intell. **47**, 139–159 (1991)
76.10 F. Mondada, E. Franzi, P. Ienne: Mobile robot miniaturization: A tool for investigation in control algorithms, Proc. 3rd Int. Symp. Exp. Robotics, Tokyo, ed. by T. Yoshikawa, F. Miyazaki (1993) pp. 501–513
76.11 L. Steels (Ed.): *The Biology and Technology of Intelligent Autonomous Agents*, NATO ASI (Springer, Berlin, Heidelberg 1995)
76.12 D. Floreano, F. Mondada: Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot, Proc. 3rd Int. Conf. Simul. Adapt. Behav.: Anim. Animat. 3, ed. by D. Cliff, P. Husbands, J.A. Meyer, S.W. Wilsonpages (MIT, Cambridge 1994) pp. 402–410
76.13 I. Harvey, P. Husbands, D.T. Cliff: Seeing the light: Artificial evolution, real vision, Proc. 3rd Int. Conf. Simul. Adapt. Behav.: Anim. Animat. 3, ed. by D.T. Cliff, P. Husbands, J.-A. Meyer, S. Wilson (MIT, Cambridge 1994) pp. 392–401
76.14 M.A. Lewis, A.H. Fagg, A. Solidum: Genetic programming approach to the construction of a neural network for a walking robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (1992) pp. 2618–2623
76.15 D. Cliff, I. Harvey, P. Husbands: Explorations in evolutionary robotics, Adapt. Behav. **2**, 73–110 (1993)
76.16 D.E. Goldberg: *Genetic Algorithms in Search, Optimization and Machine Learning* (Addison-Wesley, Reading City 1989)
76.17 H. de Garis: Genetic programming: Evolution of time dependent neural network modules which teach a pair of stick legs to walk, Proc. 9th Eur. Conf. Artif. Intell. (ECAI), Stock. (1990) pp. 204–206
76.18 R.D. Beer, J.C. Gallagher: Evolving dynamical neural networks for adaptive behavior, Adapt. Behav. **1**, 94–110 (1992)
76.19 R.D. Beer, H.J. Chiel, L.S. Sterling: Heterogeneous neural networks for adaptive behavior in dynamic environments. In: *Neural Information Processing Systems*, Vol. 1, ed. by D. Touretzky (Morgan Kauffman, San Mateo 1989) pp. 577–585
76.20 M.A. Lewis, A.H. Fagg, G. Bekey: Genetic algorithms for gait synthesis in a hexapod robot. In: *Recent Trends in Mobile Robots*, ed. by Y. Zheng (World Scientific, Singapore 1994) pp. 317–331
76.21 J. Gallagher, R. Beer, M. Espenschiel, R. Quinn: Application of evolved locomotion controllers to a hexapod robot, Robotics Auton. Syst. **19**(1), 95–103 (1996)
76.22 R.D. Beer, R.D. Quinn, H.J. Chiel, R.E. Ritzmann: Biologically inspired approaches to robotics, Commun. ACM **40**, 31–38 (1997)

76.23 S. Galt, B.L. Luk, A.A. Collie: Evolution of smooth and efficient walking motions for an 8-legged robot, Proc. 6th Eur. Workshop Learn. Robots, Brighton (1997)

76.24 T. Gomi, K. Ide: Emergence of gaits of a legged robot by collaboration through evolution, IEEE World Congr. Comput. Intell. (IEEE Press, New York 1998)

76.25 F. Gruau: Automatic definition of modular neural networks, Adapt. Behav. **3**(2), 151–183 (1995)

76.26 F. Gruau, K. Quatramaran: Cellular encoding for interactive evolutionary robotics, Proc. 4th Eur. Conf. Artif. Life, ed. by P. Husbands, I. Harvey (MIT, Cambridge 1997) pp. 368–377

76.27 J. Kodjabachian, J.A. Meyer: Evolution and development of neural networks controlling locomotion, gradient following and obstacle avoidance in artificial insects, IEEE Trans. Neural Netw. **9**, 796–812 (1998)

76.28 N. Jakobi: Running across the reality gap: Octopod locomotion evolved in a minimal simulation, Lect. Notes Comput. Sci. **1468**, 39–58 (1998)

76.29 R. Téllez, C. Angulo, D. Pardo: Evolving the walking behavior of a 12 DOF quadruped using a distributed neural architecture, Lect. Notes Comput. Sci. **3853**, 5–19 (2006)

76.30 T. Reil, P. Husbands: Evolution of central pattern generators for bipedal walking in real-time physics environments, IEEE Trans. Evol. Comput. **6**(2), 10–21 (2002)

76.31 NaturalMotion: http://www.naturalmotion.com

76.32 B. von Haller, A.J. Ijspeert, D. Floreano: Co-evolution of structures and controllers for Neubot underwater modular robots, Lect. Notes Comput. Sci. **3630**, 189–199 (2005)

76.33 E. Vaughan, E.A. Di Paolo, I. Harvey: The evolution of control and adaptation in a 3D powered passive dynamic walker, Proc. 9th Int. Conf. Simul. Synth. Living Syst. Artif. Life IX, ed. by J. Pollack, M. Bedau, P. Husbands, T. Ikegami, R. Watson (MIT, Cambridge 2004) pp. 139–145

76.34 T. McGeer: Passive walking with knees, Proc. IEEE Conf. Robotics Autom. (ICRA) (1990) pp. 1640–1645

76.35 S. Wischmann, F. Passeman: From passive to active dynamic 3D bipedal walking – An evolutionary approach, Proc. 7th Int. Conf. Climbing Walk. Robots (CLAWAR 2004), ed. by M. Armada, P. González de Santos (Springer, Berlin, Heidelberg 2005) pp. 737–744

76.36 E. Vaughan, E.A. Di Paolo, I. Harvey: The tango of a load balancing biped, Proc. 7th Int. Conf. Climbing Walk. Robots (CLAWAR), ed. by M. Armada, P. González de Santos (2005)

76.37 K. Endo, F. Yamasaki, T. Maeno, H. Kitano: A method for co-evolving morphology and walking pattern of biped humanoid robot, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2002) pp. 2775–2780

76.38 G. McHale, P. Husbands: Quadrupedal locomotion: Gasnets, CTRNNs and hybrid CTRNN/PNNs compared, Proc. 9th Int. Conf. Simul. Synth. Living Syst. (Artif. Life IX), ed. by J. Pollack, M. Bedau, P. Husbands, T. Ikegami, R. Watson (MIT, Cambridge 2004) pp. 106–112

76.39 G. McHale, P. Husbands: GasNets and other evolvable neural networks applied to bipedal locomotion, Proc. 8th Int. Conf. Simul. Adapt. Behav.: Anim. Animat. 8, ed. by S. Schaal (MIT, Cambridge 2004) pp. 163–172

76.40 J.F. Laszlo, M. van de Panne, E. Fiume: Limit cycle control and its application to the animation of balancing and walking, Proc. 23rd Annu. Conf. Comp. Graph. Interact. Tech., ACM (1996) pp. 155–162

76.41 R.A. Brooks: Artificial life and real robots, Proc. 1st Eur. Conf. Artif. Life., Toward a Pract. Auton. Syst., ed. by F.J. Varela, P. Bourgine (MIT, Cambridge 1992) pp. 3–10

76.42 R. Featherstone, D. Orin: Robot dynamics: Equations and algorithms, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 826–834

76.43 N. Jakobi, P. Husbands, I. Harvey: Noise and the reality gap: The use of simulation in evolutionary robotics, Lect. Notes Comput. Sci. **929**, 704–720 (1995)

76.44 O. Miglino, H.H. Lund, S. Nolfi: Evolving mobile robots in simulated and real environments, Artif. Life **2**, 417–434 (1996)

76.45 N. Jakobi: Half-baked, ad-hoc and noisy: Minimal simulations for evolutionary robotics, Proc. 4th Eur. Conf. Art. Life, ed. by P. Husbands, I. Harvey (MIT, Cambridge 1997) pp. 348–357

76.46 J.C. Bongard, H. Lipson: Nonlinear system identification using coevolution of models and tests, IEEE Trans. Evol. Comput. **9**(4), 361–384 (2005)

76.47 S. Koos, J. Mouret, S. Doncieux: Crossing the reality gap in evolutionary robotics by promoting transferable controllers, Proc. 12th Annu. Conf. Genetic Evol. Comput. ACM (2010) pp. 119–126

76.48 J. Urzelai, D. Floreano: Evolution of adaptive synapses: Robots with fast adaptive behavior in new environments, Evol. Comput. **9**, 495–524 (2001)

76.49 H.R. Maturana, F.J. Varela: *Autopoiesis and Cognition: The Realization of the Living* (Reidel, Dordrecht 1980)

76.50 R.D. Beer: A dynamical systems perspective on agent-environment interaction, Artif. Intell. **72**, 173–215 (1995)

76.51 P. Funes, B. Orme, E. Bonabeau: Evolving emergent group behaviors for simple humans agents, Proc. 7th Eur. Conf. Artif. Life, ed. by J. Dittrich, T. Kim (Springer, Berlin, Heidelberg 2003) pp. 76–89

76.52 S. Nolfi: Behavior and cognition as a complex adaptive system: Insights from robotic experiments. In: *Philosophy of Complex Systems*, ed. by C. Hooker (Elsevier, Amsterdam 2009) pp. 443–466

76.53 S. Nolfi: Power and limits of reactive agents, Neurocomputing **42**, 119–145 (2002)

76.54 E. Tuci, T. Ferrauto, A. Zeschel, G. Massera, S. Nolfi: An Experiment on behaviour generalisation and the emergence of linguistic compositionality in

evolving robots, IEEE Trans. Auton. Mental Dev. **3**, 176–189 (2011)
76.55 C. Scheier, R. Pfeifer, Y. Kunyioshi: Embedded neural networks: Exploiting constraints, Neural Netw. **11**, 1551–1596 (1998)
76.56 S. Nolfi, D. Marocco: Active perception: A sensorimotor account of object categorization, Proc. 7th Int. Conf. Simul. Adapt. Behav.: Anim. Animat. 7, ed. by B. Hallam, D. Floreano, J. Hallam, G. Hayes, J.-A. Meyer (MIT, Cambridge, MA 2002) pp. 266–271
76.57 E. Tuci, G. Massera, S. Nolfi: Active categorical perception of object shapes in a simulated anthropomorphic robotic arm, IEEE Trans. Evol. Comput. **14**, 885–899 (2010)
76.58 S. Collins, A. Ruina, R. Tedrake, M. Wisse: Efficient bipedal robots based on passive-dynamic walkers, Science **307**(5712), 1082–1085 (2005)
76.59 J.C. Bongard: Innocent until proven guilty: Reducing robot shaping from polynomial to linear time, IEEE Trans. Evol. Comput. **15**(4), 571–585 (2011)
76.60 H. Lipson, J.B. Pollack: Automatic design and manufacture of artificial lifeforms, Nature **406**, 974–978 (2000)
76.61 K. Sims: Evolving 3D morphology and behaviour by competition, Artif. Life **1**(4), 28–39 (1994)
76.62 Karl Sims: Evolved virtual creatures, evolution simulation, https://www.youtube.com/watch?v=JBgG_VSP7f8 (1994)
76.63 P. Funes, J. Pollack: Evolutionary body building: Adaptive physical designs for robots, Artif. Life **4**(4), 337–357 (1998)
76.64 Golem Evolutionary Robotics: https://www.youtube.com/watch?v=sLtXXFw_q8c&playnext=1&list=PL396A15596535B451&feature=results_video
76.65 J. Long: *Darwin's devices: What evolving robots can teach us about the history of life and the future of technology* (Basic Books, New York 2012)
76.66 A.J. Clark, J.M. Moore, J. Wang, X. Tan, P.K. McKinley: Evolutionary design and experimental validation of a flexible caudal fin for robotic fish, Artif. Life **13**, 325–332 (2012)
76.67 J. Bongard: Morphological change in machines accelerates the evolution of robust behavior, Proc. Natl. Acad. Sci. **108**(4), 1234–1239 (2011)
76.68 M. Dorigo, M. Colombetti: *Robot shaping: An experiment in behavior engineering* (MIT, Cambridge 1997)
76.69 J.E. Auerbach, J.C. Bongard: On the relationship between environmental and morphological complexity in evolved robots, Proc. 14th Int. Conf. Genetic Evol. Comput. Conf., ACM (2012) pp. 521–528
76.70 J. Hiller, H. Lipson: Automatic design and manufacture of soft robots, IEEE Trans. Robotics **28**(2), 457–466 (2012)
76.71 Evolved Soft Robots: https://www.youtube.com/watch?v=RrgZoo1-z_Y
76.72 J. Bongard, V. Zykov, H. Lipson: Resilient machines through continuous self-modeling, Science **314**(5802), 1118–1121 (2006)
76.73 J.C. Bongard: Accelerating self-modeling in cooperative robot teams, IEEE Trans. Evol. Comput. **13**(2), 321–332 (2009)
76.74 K.J. Kim, H. Lipson: Towards a theory of mind in simulated robots, Proc. 11th Annual Conf. Companion Genetic Evol. Comput. Conf. Late Break. Pap. ACM (2009) pp. 2071–2076
76.75 I. Harvey, P. Husbands, D.T. Cliff, A. Thompson, N. Jakobi: Evolutionary robotics: The Sussex approach, Robotics Auton. Syst. **20**, 205–224 (1997)
76.76 P. Husbands, I. Harvey, D. Cliff, G. Miller: Artificial evolution: A new path for AI?, Brain Cogn. **34**, 130–159 (1997)
76.77 N. Jakobi: Evolutionary robotics and the radical envelope of noise hypothesis, Adapt. Behav. **6**, 325–368 (1998)
76.78 K.O. Stanley, R. Miikkulainen: Evolving neural networks through augmenting topologies, Evol. Comput. **10**(2), 99–127 (2002)
76.79 M.A. Arbib: Self-reproducing automata – Some implications for theoretical biology. In: *Towards a Theoretical Biology*, 2nd edn., ed. by C.H. Waddington (Edinburgh Univ. Press, Edinburgh 1969) pp. 204–226
76.80 J. Aloimonos, I. Weiss, A. Bandopadhay: Active vision, Int. J. Comput. Vis. **1**(4), 333–356 (1987)
76.81 R. Bajcsy: Active perception, Proc. IEEE **76**(8), 996–1005 (1988)
76.82 D.H. Ballard: Animate vision, Artif. Intell. **48**(1), 57–86 (1991)
76.83 P.J. Hancock, R.J. Baddeley, L.S. Smith: The principal components of natural images, Network **3**, 61–70 (1992)
76.84 D. Floreano, T. Kato, D. Marocco, E. Sauser: Coevolution of active vision and feature selection, Biol. Cybern. **90**(3), 218–228 (2004)
76.85 D. Floreano, M. Suzuki, C. Mattiussi: Active vision and receptive field development in evolutionary robots, Evol. Comput. **13**(4), 527–544 (2005)
76.86 T.D. Sanger: Optimal unsupervised learning in a single-layer feedforward neural network, Neural Netw. **2**, 459–473 (1989)
76.87 I. Harvey, E.A. Di Paolo, R. Wood, M. Quinn, E. Tuci: Evolutionary robotics: A new scientific tool for studying cognition, Artif. Life **11**(1-2), 79–98 (2005)
76.88 A. Seth: Causal connectivity of evolved neural networks during Behaviour, Netw. Comput. Neural Syst. **16**(1), 35–54 (2005)
76.89 E. Izquierdo, S. Lockery: Evolution and analysis of minimal neural circuits for klinotaxis in Caenorhabditis elegans, J. Neurosci. **30**, 12908–12917 (2010)
76.90 P. Husbands, R.C. Moioli, Y. Shim, A. Philippides, P.A. Vargas, M. O'Shea: Evolutionary robotics and neuroscience. In: *The Horizons of Evolutionary Robotics*, ed. by P.A. Vargas, E.A. Di Paolo, I. Harvey, P. Husbands (MIT, Cambridge 2013) pp. 17–64
76.91 D.T. Cliff: Computational neuroethology: A provisional manifesto, Proc. 1st Int. Conf. Simul. Adapt. Behav.: Anim. Animat., ed. by J.-A. Meyer, S.W. Wilson (MIT, Cambridge 1991) pp. 29–39

76.92 R. Held, A. Hein: Movement-produced stimulation in the development of visually guided behavior, J. Comp. Physiol. Psychol. **56**(5), 872–876 (1963)

76.93 R. Held: Plasticity in sensory-motor systems, Sci. Am. **213**(5), 84–94 (1965)

76.94 M. Suzuki, D. Floreano, E.A. Di Paolo: The contribution of active body movement to visual development in evolutionary robots, Neural Netw. **18**(5/6), 656–665 (2005)

76.95 S. Healy (Ed.): *Spatial Representations in Animals* (Oxford Univ. Press, Oxford 1998)

76.96 N.A. Schmajuk, H.T. Blair: Place learning and the dynamics of spatial navigation: A neural network approach, Adapt. Behav. **1**, 353–385 (1993)

76.97 N. Burgess, J.G. Donnett, K.J. Jeffery, J. O'Keefe: Robotic and neuronal simulation of the hippocampus and rat navigation, Philos. Trans. R. Soc. **352**, 1535–1543 (1997)

76.98 J. O'Keefe, L. Nadel: *The Hippocampus as a Cognitive Map* (Clarendon, Oxford 1978)

76.99 J.S. Taube, R.U. Muller, J.B. Ranck Jr.: Head-direction cells recorded from the postsubiculum in freely moving rats. I. Description and quantitative analysis, J. Neurosci. **10**, 420–435 (1990)

76.100 D.E. Rumelhart, J. McClelland, P.D.P. Group: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (MIT, Cambridge 1986)

76.101 W. Maas, C.M. Bishop (Eds.): *Pulsed Neural Networks* (MIT, Cambridge 1999)

76.102 F. Rieke, D. Warland, R. van Steveninck, W. Bialek: *Spikes:: Exploring the Neural Code* (MIT, Cambridge 1997)

76.103 G. Indiveri, P. Verschure: Autonomous vehicle guidance using analog VLSI neuromorphic sensors, Lect. Notes Comput. Sci. **1327**, 811–816 (1997)

76.104 M.A. Lewis, R. Etienne-Cummings, A.H. Cohen, M. Hartmann: Toward biomorphic control using custom aVLSI CPG chips, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2000) pp. 494–500

76.105 D. Floreano, C. Mattiussi: Evolution of spiking neural controllers for autonomous vision-based robots. In: *Evolutionary Robotics. From Intelligent Robotics to Artificial Life*, ed. by T. Gomi (Springer, Tokyo 2001) pp. 38–61

76.106 W. Gerstner, J.L. van Hemmen, J.D. Cowan: What matters in neuronal locking?, Neural Comput. **8**, 1653–1676 (1996)

76.107 D. Floreano, Y. Epars, J.C. Zufferey, C. Mattiussi: Evolution of spiking neural circuits in autonomous mobile robots, Int. J. Intell. Syst. **21**(9), 1005–1024 (2006)

76.108 J.A. Gally, P.R. Montague, G.N. Reeke, G.M. Edelman: The NO hypothesis: Possible effects of a short-lived, rapidly diffusible signal in the development and function of the nervous system, Proc. Natl. Acad. Sci. **87**(9), 3547–3551 (1990)

76.109 J. Wood, J. Garthwaite: Models of the diffusional spread of nitric oxide: Implications for neural nitric oxide signaling and its pharmacological properties, Neuropharmacology **33**, 1235–1244 (1994)

76.110 T.M. Dawson, S.N. Snyder: Gases as biological messengers: Nitric oxide and carbon monoxide in the brain, J. Neurosci. **14**(9), 5147–5159 (1994)

76.111 J. Garthwaite, C.L. Boulton: Nitric oxide signaling in the central nervous system, Annu. Rev. Physiol. **57**, 683–706 (1995)

76.112 A.O. Philippides, P. Husbands, M. O'Shea: Four-dimensional neuronal signaling by nitric oxide: A computational analysis, J. Neurosci. **20**(3), 1199–1207 (2000)

76.113 C. Hölscher: Nitric oxide, the enigmatic neuronal messenger: Its role in synaptic plasticity, Trends Neurosci. **20**, 298–303 (1997)

76.114 P. Husbands, T. Smith, N. Jakobi, M. O'Shea: Better living through chemistry: Evolving GasNets for robot control, Connect. Sci. **10**(4), 185–210 (1998)

76.115 T.M.C. Smith, P. Husbands, M. O'Shea: Local evolvability, neutrality, and search difficulty in evolutionary robotics, Biosystems **69**, 223–243 (2003)

76.116 A.O. Philippides, P. Husbands, T. Smith, M. O'Shea: Flexible couplings: Diffusing neuromodulators and adaptive robotics, Artif. Life **11**(1–2), 139–160 (2005)

76.117 A.O. Philippides, P. Husbands, T. Smith, M. O'Shea: Structure based models of NO diffusion in the nervous system. In: *Computational Neuroscience: A Comprehensive Approach*, ed. by J. Feng (CRC, Boca Raton 2004) pp. 97–130

76.118 A.O. Philippides, S.R. Ott, P. Husbands, T. Lovick, M. O'Shea: Modeling co-operative volume signaling in a plexus of nitric oxide synthase-expressing neurons, J. Neurosci. **25**(28), 6520–6532 (2005)

76.119 P. Husbands, A. Philippides, P. Vargas, C. Buckley, P. Fine, E.A. Di Paolo, M. O'Shea: Spatial, temporal and modulatory factors affecting GasNet evolvability in a visually guided robotics task, Complexity **16**(2), 35–44 (2010)

76.120 D. Barañano, C. Ferris, S. Snyder: A typical neural messenger, Trends Neurosci. **24**(2), 99–106 (2001)

76.121 T.M.C. Smith, P. Husbands, A. Philippides, M. O'Shea: Neuronal plasticity and temporal adaptivity: Gasnet robot control networks, Adapt. Behav. **10**(3/4), 161–184 (2002)

76.122 G. Edelman, J. Gally: Degeneracy and complexity in biological systems, Proc Natl. Acad. Sci. USA **98**, 13763–13768 (2001)

76.123 C. Fernando, K. Karishma, E. Szathmáry: Copying and evolution of neuronal topology, PLoS ONE **3**(11), e3775 (2008)

76.124 C. Fernando, E. Szathmáry, P. Husbands: Selectionist and evolutionary approaches to brain function: A critical appraisal, Front. Comput. Neurosci. **6**, 24 (2012)

76.125 S. Nolfi, D. Floreano: Learning and evolution, Auton. Robots **7**, 89–113 (1999)

76.126 S. Nolfi, D. Parisi: Learning to adapt to changing environments in evolving neural networks, Adapt. Behav. **1**, 75–98 (1997)

76.127 J.M. Baldwin: A new factor in evolution, Am. Nat. **30**, 441–451 (1896)

76.128 C.H. Waddington: Canalization of development and the inheritance of acquired characters, Nature **150**, 563–565 (1942)

76.129 G. Mayley: Landscapes, learning costs, and genetic assimilation, Evol. Comput. **4**, 213–234 (1997)

76.130 D. Floreano, F. Mondada: Evolution of plastic neurocontrollers for situated agents, Proc. 4th Int. Conf. Simul. Adapt. Behav.: Anim. Animat. 4, ed. by P. Maes, M. Matarić, J.A. Meyer, J. Pollack, H. Roitblat, S. Wilson (MIT, Cambridge 1996) pp. 402–410

76.131 D. Floreano, J. Urzelai: Evolutionary robots with online self-organization and behavioral fitness, Neural Netw. **13**, 431–443 (2000)

76.132 D. Floreano, J. Urzelai: Neural morphogenesis, synaptic plasticity, and evolution, Theory Biosci. **120**(3–4), 225–240 (2001)

76.133 E. Di Paolo: Evolving spike-timing-dependent plasticity for single-trial learning in robots, Philos. Trans. R. Soc. Lond. **361**, 2299–2319 (2003)

76.134 Y.U. Cao, A.S. Fukunaga, A. Kahng: Cooperative mobile robotics: Antecedents and directions, Auton. Robots **4**, 7–27 (1997)

76.135 S. Nolfi: Co-evolving predator and prey robots, Adapt. Behav. **20**, 10–15 (2012)

76.136 D. Floreano, S. Nolfi: God save the red queen! Competition in co-evolutionary robotics, Proc. 2nd Conf. Genetic Program., ed. by J.R. Koza, K. Deb, M. Dorigo, D. Foegel, B. Garzon, H. Iba, R.L. Riolo (Morgan Kaufmann, San Francisco, CA 1997) pp. 398–406

76.137 S. Nolfi, D. Floreano: Co-evolving predator and prey robots: Do *arm races* arise in artificial evolution?, Artif. Life **4**(4), 311–335 (1998)

76.138 D. Floreano, S. Nolfi: Evolution versus design: Controlling autonomous robots, Proc. 4th Eur. Conf. Artif. Life, ed. by P. Husbands, I. Harvey (MIT, Cambridge 1997) pp. 378–387

76.139 V. Trianni, S. Nolfi: Evolving collective control, cooperation and distributed cognition. In: *Handbook of Collective Robotics – Fundamentals and Challenges*, ed. by S. Kernbach (CRC, Boca Raton 2012) pp. 246–276

76.140 G. Baldassarre, V. Trianni, M. Bonani, F. Mondada, M. Dorigo, S. Nolfi: Self-organised coordinated motion in groups of physically connected robots, IEEE Trans. Syst. Man Cybern. **37**, 224–239 (2007)

76.141 M. Quinn, L. Smith, G. Mayley, P. Husbands: Evolving controllers for a homogeneous system of physical robots: Structured cooperation with minimal sensors, Philos. Trans. R. Soc. Lond. **361**, 2321–2344 (2003)

76.142 G. Baldassarre, D. Parisi, S. Nolfi: Coordination and behavior integration in cooperating simulated robots, Proc. 8th Int. Conf. Simul. Adapt. Behav.: Anim. Animat. 8 (MIT, Cambridge 2003) pp. 385–394

76.143 V. Sperati, V. Trianni, S. Nolfi: Self-organised path formation in a swarm of robots, Swarm Intell. **5**, 97–119 (2011)

76.144 M. Quinn: Evolving communication without dedicated communication channels, Proc. 6th Eur. Conf. Artif. Life, ed. by J. Kelemen, P. Sosik (Springer, Berlin, Heidelberg 2001) pp. 357–366

76.145 D. Marocco, S. Nolfi: Self-organization of communication in evolving robots, Proc. 10th Int. Conf. Artif. Life, ed. by L. Rocha, L. Yeager, M. Bedau, D. Floreano, R. Goldstone, A. Vespignani (MIT, Cambridge 2006) pp. 178–184

76.146 D. Floreano, S. Mitri, S. Magnenat, L. Keller: Evolutionary conditions for the emergence of communication in robots, Curr. Biol. **17**, 514–519 (2007)

76.147 M. Waibel, D. Floreano, S. Magnenat, L. Keller: Division of labour and colony efficiency in social insects: Effects of interactions between genetic architecture, colony kin structure and rate of perturbations, Proc. Royal Soc. B Biol. Sci. **273**, 1815–1823 (2006)

76.148 F. Mondada, G. Pettinaro, A. Guignard, I. Kwee, D. Floreano, J.L. Deneubourg, S. Nolfi, L.M. Gambardella, M. Dorigo: Swarm-bot: A new distributed robotic concept, Auton. Robots **17**, 193–221 (2004)

76.149 V. Trianni, S. Nolfi, M. Dorigo: Cooperative hole-avoidance in a swarm-bot, Robotics Auton. Syst. **54**, 97–103 (2006)

76.150 G. Baldassarre, S. Nolfi, D. Parisi: Evolving mobile robots able to display collective behavior, Artif. Life **9**, 255–267 (2003)

76.151 S. Nolfi: Evolution of communication and language in evolving robots. In: *Current Perspective on the origin of language*, ed. by C. Lefebvre, B. Comrie, H. Cohen (Cambridge Univ. Press, Cambridge 2013)

76.152 J. De Greef, S. Nolfi: Evolution of implicit and explicit communication in a group of mobile robots. In: *Evolution of Communication and Language in Embodied Agents*, ed. by S. Nolfi, M. Mirolli (Springer, Berlin, Heidelberg 2010) pp. 179–214

76.153 S. Mitri, D. Floreano, L. Keller: The evolution of information suppression in communicating robots with conflicting interests, Proc. Natl. Acad. Sci. **106**, 15786–15790 (2009)

76.154 S. Mitri, D. Floreano, L. Keller: Relatedness influences signal reliability in evolving robots, Proc. Royal Soc. B Biol. Sci. **278**, 378–383 (2011)

76.155 S. Nolfi: Emergence of communication in embodied agents: Co-adapting communicative and non-communicative behaviours, Connect. Sci. **3–4**, 231–248 (2005)

76.156 S. Wischmanna, D. Floreano, L. Keller: Historical contingency affects signaling strategies and competitive abilities in evolving populations of simulated robots, Proc. Natl. Acad. Sci. **109**, 864–868 (2011)

76.157 A. Thompson: Evolving electronic robot controllers that exploit hardware resources, Lect. Notes Artif. Intell. **929**, 640–656 (1995)

76.158 A. Thompson: *Hardware Evolution: Automatic Design of Electronic Circuits in Reconfigurable Hardware by Artificial Evolution*, Distinguished Dissertation Series (Springer, Berlin, Heidelberg 1998)

76.159 A. Thompson: Artificial evolution in the physical world. In: *Evolutionary Robotics. From Intelligent Robots to Artificial Life (ER'97)*, ed. by T. Gomi (AAI Books, Ottawa 1997) pp. 101–125
76.160 D. Keymeulen, M. Durantez, M. Konaka, Y. Kuniyoshi, T. Higuchi: An evolutionary robot navigation system using a gate-level evolvable hardware, Lect. Notes Comput. Sci. **1259**, 193–209 (1996)
76.161 G. Ritter, J.-M. Puiatti, E. Sanchez: Leonardo and discipulus simplex: An autonomous, evolvable six-legged walking robot, Lect. Notes Comput. Sci. **1586**, 688–696 (1999)
76.162 D. Roggen, D. Floreano, C. Mattiussi: A morphogenetic evolutionary system: Phylogenesis of the POETIC circuit, Lect. Notes Comput. Sci. **2606**, 153–164 (2003)
76.163 D. Roggen, S. Hofmann, Y. Thoma, D. Floreano: Hardware spiking neural network with run-time reconfigurable connectivity in an autonomous robot, NASA/DoD Conf. Evolv. Hardw., ed. by J. Lohn, R. Zebulum, J. Steincamp, D. Keymeulen, A. Stoica, M.I. Fergusonpages (2003) pp. 189–198
76.164 M. Matarić, D. Cliff: Challenges in evolving controllers for physical robots, Robotics Auton. Syst. **19**(1), 67–83 (1996)
76.165 G. Massera, T. Ferrauto, O. Gigliotta, S. Nolfi: FARSA: An open software tool for embodied cognitive science, Proc. 12th Eur. Conf. Artif. Life, ed. by P. Lio, O. Miglino, G. Nicosia, S. Nolfi, M. Pavone (MIT, Cambridge 2013) pp. 454–538
76.166 Framework for Autonomous Robotics Simulation and Analysis: http://laral.istc.cnr.it/farsa

Multimedia Contents

77. Neurorobotics: From Vision to Action

Patrick van der Smagt, Michael A. Arbib, Giorgio Metta

The lay view of a robot is a mechanical human, and thus robotics has always been inspired by attempts to emulate biology. In this chapter, we extend this biological motivation from humans to animals more generally, but with a focus on the central nervous systems in its relationship to the bodies of these creatures. In particular, we investigate the sensorimotor loop in the execution of sophisticated behavior. Some of these sections concentrate on cases where vision provides key sensory data. Neuroethology is the study of the brain mechanisms underlying animal behavior, and Sect. 77.2 exemplifies the lessons it has to offer robotics by looking at optic flow in bees, visually guided behavior in frogs, and navigation in rats, turning then to the coordination of behaviors and the role of attention. Brains are composed of diverse subsystems, many of which are relevant to robotics, but we have chosen just two regions of the mammalian brain for detailed analysis. Section 77.3 presents the cerebellum. While we can plan and execute actions without a cerebellum, the actions are no longer graceful and become uncoordinated. We reveal how a cerebellum can provide a key ingredient in an adaptive control system, tuning parameters both within and between motor schemas. Section 77.4 turns to the mirror system, which provides shared representations which bridge between the execution of an action and the observation of that action when performed by others. We develop a neurobiological model of how learning may forge mirror neurons for hand movements, provide a Bayesian view of a robot mirror system, and discuss what must be added to a mirror system to support robot imitation. We conclude by emphasizing that, while neuroscience can inspire novel robotic designs, it is also the case that robots can be used as embodied test beds for the analysis of brain models.

77.1 **Definitions and History** 2070
77.1.1 History and Definitions 2070
77.2 **The Case for Vision** 2071
77.2.1 Optic Flow in Bees and Robots 2071
77.2.2 Visually Guided Behavior in Frogs and Robots 2072
77.2.3 Navigation in Rat and Robot 2072
77.2.4 Salience and Visual Attention 2074
77.3 **Vertebrate Motor Control** 2075
77.3.1 The Flat Hierarchy of Neurocontrol 2076
77.3.2 On Spinal Cord and Muscle 2077
77.3.3 Models of Cerebellar Control 2078
77.3.4 Cerebellar Models and Robotics 2081
77.4 **The Role of Mirror Systems** 2082
77.4.1 Mirror Neurons and the Recognition of Hand Actions 2082
77.4.2 Computational Models 2084
77.4.3 Mirror Neurons and Imitation 2085
77.4.4 Mirror System and Speech 2086
77.5 **Conclusion and Further Reading** 2089
77.5.1 Further Reading 2090
References 2090

77.1 Definitions and History

Neurorobotics may be defined as:

> *the design of computational structures for robots inspired by the study of the nervous systems of humans and other animals.*

We note the success of (deep) *artificial* neural networks – networks of simple computing elements whose connections change with *experience* – as providing a medium for parallel adaptive computation that has seen application in robot vision systems and controllers but here we emphasize neural networks derived from the study of specific *neurobiological* systems. Neurorobotics has a twofold aim: creating better machines which employ the principles of natural neural computation; and using the study of bio-inspired robots to improve understanding of the functioning of the brain. Chapter 75, *Biologically Inspired Robots*, complements our study of *brain design* with work on *body design*, the design of robotic control and actuator systems based on careful study of the relevant biology.

77.1.1 History and Definitions

Science has long been playing with technical replicas of biological behavior. As a famous example, *Walter* [77.1] described two *biologically inspired* robots, the electromechanical tortoises *Machina speculatrix* and *M. docilis* (though each body has wheels, not legs). *M. speculatrix* has a steerable photoelectric cell, which makes it sensitive to light, and an electrical contact, which allows it to respond when it bumps into obstacles. The photoreceptor rotates until a light of moderate intensity is registered, at which time the organism orients itself towards the light and approaches it. However, very bright lights, material obstacles, and steep gradients are repellent to the *tortoise*. The latter stimuli convert the photoamplifier into an oscillator, which causes alternating movements of butting and withdrawal, so that the robot pushes small objects out of its way, goes around heavy ones, and avoids slopes. The *tortoise* has a *hutch*, which contains a bright light. When the machine's batteries are charged, this bright light is repellent. When the batteries are low, the light becomes attractive to the machine and the light continues to exert an attraction until the tortoise enters the hutch, where the machine's circuitry is temporarily turned off until the batteries are recharged, at which time the bright hutch light again exerts a negative tropism. The second robot, *M. docilis* was produced by grafting onto *M. speculatrix* a circuit designed to form conditioned reflexes. In one experiment, Walter connected this circuit to the obstacle-avoiding device in *M. speculatrix*. Training consisted of blowing a whistle just before bumping the shell.

Although Walter's controllers are simple and not based on neural analysis, they do illustrate an attempt to gain inspiration from seeking the simplest mechanisms that will yield an interesting class of biologically inspired robot behaviors, and then showing how different additional mechanisms yield a variety of enriched behaviors. *Braitenberg*'s book [77.2] is very much in this spirit and has entered the canon of neurorobotics. While their work provides a historical background for the studies surveyed here, we instead emphasize studies inspired by the computational neuroscience of the mechanisms serving vision and action in the human and in animal brains. We seek lessons from linking behavior to the analysis of the internal workings of the brain (1) at the relatively high level of characterizing the functional roles of specific brain regions (or the functional units of analysis called schemas, Sect. 77.2.4), and the behaviors which emerge from the interactions between them, and (2) at the more detailed level of models of neural circuitry linked to the data of neuroanatomy and neurophysiology. There are lessons for neurorobotics to be learned from even finer-scale analysis of the biophysics of individual neurons and the neurochemistry of synaptic plasticity, but these are beyond the scope of this chapter (see *Segev* and *London* [77.3] and *Fregnac* [77.4], respectively, for entry points into the relevant computational neuroscience).

The plan of this chapter is as follows. We will start with explaining how the higher-level *cognitive* functionality of vision-based planning and navigation is realized in biology, and how this relates to robotic systems (Sect. 77.2). We then (Sect. 77.3) explain vertebrate movement generation itself, and put forth a theory on what role the cerebellum plays in tuning and coordinating actions. This is followed by a section on the mirror system and its roles in action recognition and imitation (Sect. 77.4). The extroduction will then invite readers to explore the many other areas in which neurorobotics offers lessons from neuroscience to the development of novel robot designs. What follows, then, can be seen as a contribution to the continuing dialog between robot behavior and animal and human behavior in which particular emphasis is placed on the search for the neural underpinnings of vision, visually guided action, and cerebellar control.

77.2 The Case for Vision

Before we turn to vertebrate brains for much of our inspiration for neurorobotics, we briefly sample the rich literature on insect-inspired research. Among the founding studies in computational neuroethology were a series of reports from the laboratory of Werner Reichardt in Tübingen, which linked the delicate anatomy of the fly's brain to the extraction of visual data needed for flight control. More than 40 years ago, *Reichardt* [77.5] published a model of motion detection inspired by this work that has long been central to discussions of visual motion in both the neuroscience and robotics literatures. *Borst* and *Dickinson* [77.6] provide a recent study of continuing biological research on visual course control in flies. Such work has inspired a large number of robot studies, including those of *van der Smagt* and *Groen* [77.7], *van der Smagt* [77.8], *Liu* and *Usseglio-Viretta* [77.9], *Ruffier* et al. [77.10], and *Reiser* and *Dickinson* [77.11].

77.2.1 Optic Flow in Bees and Robots

Here, however, we look in a little more detail at honeybees. *Srinivasan* et al. [77.15] continued the tradition of studying image motion cues in insects by investigating how optic flow (the flow of pattern across the eye induced by motion relative to the environment) is exploited by honeybees to guide locomotion and navigation. They analyzed how bees perform a smooth landing on a flat surface: image velocity is held constant as the surface is approached, thus automatically ensuring that flight speed is close to zero at touchdown. This obviates any need for explicit knowledge of flight speed or height above the *ground*. This landing strategy was then implemented in a robotic gantry to test its applicability to autonomous airborne vehicles. *Barron* and *Srinivasan* [77.14] investigated the extent to which ground speed is affected by headwinds. Honeybees were trained to enter a tunnel to forage at a sucrose feeder placed at its far end (Fig.77.1a). The bees used visual cues to maintain their ground speed by adjusting their airspeed to maintain a constant rate of optic flow, even against headwinds which were, at their strongest, 50% of a bee's maximum recorded forward velocity.

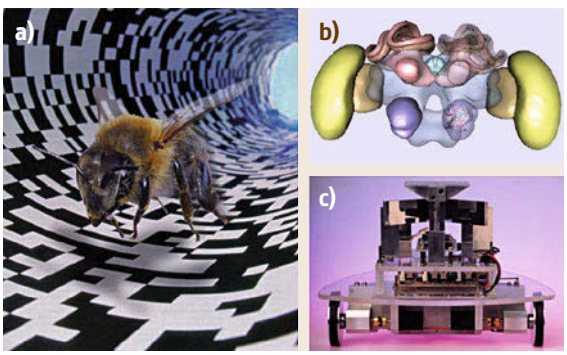


Fig. 77.1 **(a)** Observation of the trajectories of honeybees flying in visually textured tunnels has provided insights into how bees use optic flow cues to regulate flight speed and estimate distance flown, and balance optic flow in the two eyes to fly safely through narrow gaps (images courtesy of *Srinivasan* et al. [77.12]). This information has been used to build autonomously navigating robots. **(b)** Schematic illustration of a honeybee brain, carrying about a million neurons within $\approx 1\,\mathrm{mm}^3$ (after [77.13]). **(c)** A mobile robot guided by an optic flow algorithm based on the studies exemplified in [77.14]

Vladusich et al. [77.16] studied the effect of adding goal-defining landmarks. Bees were trained to forage in an optic-flow-rich tunnel with a landmark positioned directly above the feeder. They searched much more accurately when both odometric and landmark cues were available than when only odometry was available. When the two cue sources were set in conflict, by shifting the position of the landmark in the tunnel during tests, bees overwhelmingly used landmark cues rather than odometry. This, together with other such experiments, suggests that bees can make use of odometric and landmark cues in a more flexible and dynamic way than previously envisaged. In earlier studies of bees flying down a tunnel, *Srinivasan* and *Zhang* [77.17] placed different patterns on the left and right walls. They found that bees balance the image velocities in the left and right visual fields. This strategy ensures that bees fly down the middle of the tunnel, without bumping into the side walls, enabling them to negotiate narrow passages or to fly between obstacles. This strategy has been applied to a corridor-following robot (Fig. 77.1c). By holding constant the average image velocity as seen by the two eyes during flight, the bee avoids potential collisions, slowing down when it flies through a narrow passage. The movement-sensitive mechanisms underlying these various behaviors differ qualitatively as well as quantitatively, from those that mediate the optomotor response (e.g., turning to track a pattern of moving stripes) that had been the initial target of investigation of the Reichardt laboratory. The lesson for robot control is that flight appears to be coordinated by a number of visuomotor systems acting in concert, and the same lesson can apply to a whole range of tasks that must convert vision to action. Of *course*, vision is but one of the sensory systems that play a vital role in insect behavior. *Webb* [77.18] uses her own work on robot

design inspired by the auditory control of behavior in crickets to anchor a far-ranging assessment of the extent to which robotics can offer good models of animal behaviors.

77.2.2 Visually Guided Behavior in Frogs and Robots

Lettvin et al. [77.19] treated the frog's visual system from an ethological perspective, analyzing circuitry in relation to the animal's ecological niche to show that different cells in the retina and the visual midbrain region known as the tectum were specialized for detecting predators and prey. However, in much visually guided behavior, the animal does not respond to a single stimulus, but rather to some property of the overall configuration. We thus turn to the question *what does the frog's eye tell the frog?*, stressing the embodied nervous system or, perhaps equivalently, an action-oriented view of perception. Consider, for example, the snapping behavior of frogs confronted with one or more fly-like stimuli. *Ingle* [77.20] found that it is only in a restricted region around the head of a frog that the presence of a fly-like stimulus elicits a snap, that is, the frog turns so that its midline is pointed at the stimulus and then lunges forward and captures the prey with its tongue. There is a larger zone in which the frog merely orients towards the target, and beyond that zone the stimulus elicits no response at all. When confronted with two *flies* within the snapping zone, either of which is vigorous enough that it could elicit a snapping response alone, the frog exhibits one of three reactions: it snaps at one of the flies, it does not snap at all, or it snaps in between at the *average fly*. *Didday* [77.21] offered a simple model of this choice behavior which may be considered as the prototype for a *winner-take-all* (WTA) model, which receives a variety of inputs and (under ideal circumstances) suppresses the representation of all but one of them; the one that remains is the *winner* that will play the decisive role in further processing. This was the beginning of *Rana computatrix* (see *Arbib* [77.22, 23] for overviews).

Studies on frog brains and behavior inspired the successful use of potential fields for robot navigation strategies. Data on the strategies used by frogs to capture prey while avoiding static obstacles (*Collett* [77.24]) grounded the model by *Arbib* and *House* [77.25], which linked systems for depth perception to the creation of spatial maps of both prey and barriers. In one version of their model, they represented the map of prey by a potential field with long-range attraction and the map of barriers by a potential field with short-range repulsion, and showed that summation of these fields yielded a field that could guide the frog's detour around the barrier to catch its prey. *Corbacho* and *Arbib* [77.26] later explored a possible role for learning in this behavior. Their model incorporated learning in the weights between the various potential fields to enable adaptation over trials as observed in the real animals. The success of the models indicated that frogs use reactive strategies to avoid obstacles while moving to a goal, rather than employing a planning or cognitive system. Other work, *Cobas* and *Arbib* [77.27], studied how the frog's ability to catch prey and avoid obstacles was integrated with its ability to escape from predators. These models stressed the interaction of the tectum with a variety of other brain regions such as the pretectum (for detecting predators) and the tegmentum (for implementing motor commands for approach or avoidance).

Arkin [77.28] showed how to combine a computer vision system with a frog-inspired potential field controller to create a control system for a mobile robot that *could* successfully navigate in a fairly structured environment using camera input. The resultant system thus enriched other roughly contemporaneous applications of potential fields in path planning with obstacle avoidance for both manipulators and mobile robots (*Khatib* [77.29], *Krogh* and *Thorpe* [77.30]). The work on *Rana computatrix* proceeded at two levels – both biologically realistic neural networks and in terms of functional units called *schemas*, which compete and cooperate to determine behavior. Section 77.2.4 will show how more general behaviors can emerge from the competition and cooperation of perceptual and motor schemas, as well as more abstract coordinating schemas. Such ideas were, of course, developed independently by a number of authors, and so entered the robotics literature by various routes, of which the best known may be the subsumption architecture of *Brooks* [77.31] and the ideas of Braitenberg cited above, whereas *Arkin's* work on behavior-based robotics [77.32] is, indeed, rooted in schema theory. *Arkin* et al. [77.33] present a recent example of the continuing interaction between robotics and ethology, offering a novel method for creating high-fidelity models of animal behavior for use in robotic systems based on a behavioral systems approach (i. e., based on a schema-level model of animal behavior, rather than analysis of biological circuits in animal brains), and describe how an ethological model of a domestic dog can be implemented with AIBO, the Sony entertainment robot.

77.2.3 Navigation in Rat and Robot

The tectum, the midbrain visual system which determines how the frog turns its whole body towards it prey or orients it for escape from predators (Sect.77.2.2), is

homologous with the superior colliculus of the mammalian midbrain. The rat superior colliculus has been shown to be *frog like*, mediating approach and avoidance (*Dean* et al. [77.34]), whereas the best-studied role of the superior colliculus of cat, monkey, and human is in the control of saccades, rapid eye movements to acquire a visual target. Moreover, the superior colliculus can integrate auditory and somatosensory information into its visual frame (*Stein* and *Meredith* [77.35]), and this inspired *Strosslin* et al. [77.36] to use a biologically inspired approach based on the properties of neurons in the superior colliculus to learn the relation between visual and tactile information in control of a mobile robot platform. More generally, then, the comparative study of mammalian brains has yielded a rich variety of computational models of importance in neurorobotics. In this section, we further introduce the study of *mammalian neurorobotics* by looking at studies of mechanisms of the rat brain for spatial navigation.

The frog's detour behavior is an example of what *O'Keefe* and *Nadel* [77.37] called the *taxon (behavioral orientation) system* (as in *Braitenberg*, [77.38] a *taxis* (plural *taxes*) is an organism's response to a stimulus by movement in a particular direction). They distinguished this from a system for *map-based navigation* and proposed that the latter resides in the hippocampus, though *Guazzelli* et al. [77.39] qualified this assertion, showing how the hippocampus may function as *part of* a cognitive map. The taxon versus map distinction is akin to the distinction between reactive and deliberative control in robotics (*Arkin* et al. [77.33]). It will be useful to relate taxis to the notion of an *affordance* (*Gibson* [77.40]), a feature of an object or environment relevant to action, for example, in picking up an apple or a ball, the identity of the object may be irrelevant, but the size of the object is crucial. Similarly, if we wish to push a toy car, recognizing the make of car copied in the toy is irrelevant, whereas it is crucial to recognize the placement of the wheels to extract the direction in which the car can be readily pushed. Just as a rat may have basic taxes for approaching food or avoiding a bright light, say, so does it have a wider repertoire of affordances for possible actions associated with the immediate sensing of its environment. Such affordances include *go straight ahead* for visual sighting of a corridor, *hide* for a dark hole, *eat* for food as sensed generically, *drink* similarly, and the various turns afforded by, e.g., the sight of the end of the corridor. It also makes rich use of olfactory cues. In the same way, a robot's behavior will rely on a host of reactions to local conditions in fulfilling a plan, e.g., knowing that it must go to the end of a corridor it will nonetheless use local visual cues to avoid hitting obstacles or to determine through which angle to turn when reaching a bend in the corridor.

Both normal and hippocampal-lesioned rats can learn to solve a simple T-maze (e.g., learning whether to turn left or right to find food) in the absence of any consistent environmental cues other than the T-shape of the maze. If anything, the lesioned animals learn this problem faster than normal ones. After the criterion was reached, probe trials with an eight-arm radial maze were interspersed with the usual T-trials. Animals from both groups consistently chose the side to which they were trained on the T-maze. However, many did not choose the 90° arm but preferred either the 45° or 135° arm, suggesting that the rats eventually solved the T-maze by learning to rotate within an egocentric orientation system at the choice point through *approximately* 90°. This leads to the hypothesis of an *orientation vector* being stored in the animal's brain but does not tell us where or how the orientation vector is stored. One possible model would employ coarse coding in a linear array of cells, coding for turns from −180° to +180°. From the behavior, one might expect that only the cells close to the preferred *behavioral* direction are excited, and that learning *marches* this peak from the old to the new preferred direction. To *unlearn* −90°, say, the array must reduce the peak there, while at the same time *building* a new peak at the new direction of +90°. If the old peak has *massp(t)* and the new peak has *massq(t)*, then as $p(t)$ declines toward 0 while $q(t)$ increases steadily from 0, the center of mass will progress from −90° to +90°, fitting the behavioral data.

The determination of movement direction was modeled by *rat-ification* of the *Arbib* and *House* [77.25] model of frog detour behavior. There, prey was represented by excitation coarsely coded across a population, while barriers were encoded by inhibition whose extent closely matched the retinotopic extent of each barrier. The sum of excitation was passed through a winner-takes-all circuit to yield the choice of movement direction. As a result, the direction of the gap closest to the prey, rather than the direction of the prey itself, was often chosen for the frog's initial movement. The same model serves for behavioral orientation once we replace the direction of the prey (frog) by the direction of the orientation vector (rat), while the barriers correspond to the presence of walls rather than alley ways.

To approach the issue of how a cognitive map can extend the capability of the affordance system, *Guazzelli* et al. [77.39] extended the *Lieblich* and *Arbib* [77.41] approach to building a cognitive map as a *world graph*, a set of nodes connected by a set of edges, where the nodes represent recognized places or situations, and the links represent ways of moving from one situation to another. A crucial notion is that a place encountered in different circumstances may be repre-

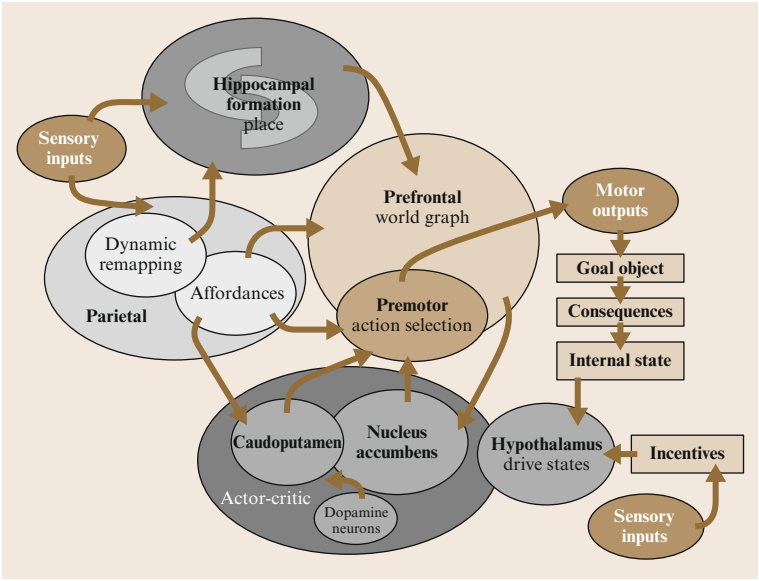


Fig. 77.2 The TAM-WG model has at its basis a system, TAM (the taxon affordance model), for exploiting affordances. This is elaborated by a system, WG (the world graph), which can use a cognitive map to plan paths to targets which are not currently visible. Note that the model processes two different kinds of sensory inputs. At the *bottom right* are those associated with, e.g., hypothalamic systems for feeding and drinking and that may provide both incentives and rewards for the animal's behavior, contributing both to behavioral choices, and to the reinforcement of certain patterns of behavior. The nucleus accumbens and caudoputamen mediate an actor-critic style of reinforcement learning based on the hypothalamic drive of the dopamine system. The sensory inputs at the *top left* are those that allow the animal to sense its relation with the external world, determining both where it is (the hippocampal place system), as well as the affordances for action (the parietal recognition of affordances can shape the premotor selection of an action). The TAM model focuses on the parietal–premotor reaction to immediate affordances; the WG model places action selection within the wider context of a cognitive map (after *Guazzelli* et al. [77.39])

sented by multiple nodes, but that these nodes may be merged when the similarity between these circumstances is recognized. They model the process whereby the animal decides where to move next, on the basis of its current drive state (hunger, thirst, fear, etc.). The emphasis is on spatial maps for guiding locomotion into regions not necessarily currently visible, rather than retinotopic representations of immediately visible space, and yields exploration and latent learning without the introduction of an explicit exploratory drive. The model shows:

1. How a route, possibly of many steps, may be chosen that leads to the desired goal.
2. How short cuts may be chosen.
3. Through its account of node merging why, in open fields, place cell firing does not seem to depend on direction.

The overall structure and general mode of operation of the complete model is shown in Fig. 77.2, which gives a vivid sense of the lessons to be learned by studying not only specific systems of the mammalian brain but also their patterns of large-scale interaction. This model is but one of many inspired by the data on the role of the hippocampus and other regions in rat navigation. Here, we just mention as pointers the wider literature the papers by *Girard* et al. [77.42] and *Meyer* et al. [77.43], which are part of the *Psikharpax* project, which does for rats what *Rana computatrix* did for frogs and toads.

77.2.4 Salience and Visual Attention

Discussions of how an animal (or robot) grasps an object assume that the animal or robot is attending to the relevant object. Thus, whatever the subtlety of processing in the canonical and mirror systems for grasping, its success rests on the availability of a visual system coupled to an oculomotor control system that bring foveal vision to bear on objects to set the parameters needed for successful interaction. Indeed, the general point is that attention greatly reduces the processing load for animal and robot. The catch, of course, is that reducing the computing load is a Pyrrhic victory unless

the moving focus of attention captures those aspects of behavior that are relevant for the current task – or supports necessary *priority interrupts*. Indeed, directing attention appropriately is a topic for which there is a great richness of both neurophysiological data and robotic application (see *Deco* and *Rolls* [77.44] and *Choi* et al. [77.45]).

In their neuromorphic model of the bottom-up guidance of attention in primates, *Itti* and *Koch* [77.46] decompose the input video stream into eight feature channels at six spatial scales. After surround suppression, only a sparse number of locations remain active in each map, and all maps are combined into a unique *saliency map*. This map is scanned by the focus of attention in order of decreasing saliency through the interaction between a winner-takes-all mechanism (which selects the most salient location) and an *inhibition-of-return* mechanism (which transiently suppresses recently attended locations from the saliency map). Because it includes a detailed low-level vision front-end, the model has been applied not only to laboratory stimuli, but also to a wide variety of natural scenes, predicting a wealth of data from psychophysical experiments.

When specific objects are searched for, low-level visual processing can be biased both by the gist (e.g., *outdoor suburban scene*) and also for the features of that object. This top-down modulation of bottom-up processing results in an ability to guide search towards targets of interest (*Wolfe* [77.47]). Task affects eye movements (*Yarbus* [77.48]), as do training and general expertise. *Navalpakkam* and *Itti* [77.49] propose a computational model which emphasizes four aspects that are important in biological vision: determining the task relevance of an entity, biasing attention for the low-level visual features of desired targets, recognizing these targets using the same low-level features, and incrementally building a visual map of task relevance at every scene location. It attends to the most salient location in the scene, and attempts to recognize the attended object through hierarchical matching against object representations stored in long-term memory. It updates its working memory with the task relevance of the recognized entity and updates a topographic task-relevance map with the location and relevance of the recognized entity; for example, in one task the model forms a map of likely locations of cars from a video clip filmed while driving on a highway. Such work illustrates the continuing interaction between models based on visual neurophysiology and human psychophysics with the tackling of practical robotic applications.

Orabona et al. [77.50] implemented an extension of the Itti–Koch model on a humanoid robot with moving eyes, using log-polar vision as in *Sandini* and *Tagliasco* [77.51], and changing the feature construction pyramid by considering proto-object elements (blob-like structures rather than edges). The inhibition-of-return mechanism has to take into account a moving frame of reference, the resolution of the fovea is very different from that at the periphery of the visual field, and head and body movements need to be stabilized. The control of movement might thus have a relationship with the structure and development of the attention system. *Rizzolatti* et al. [77.52] proposed a role for the feedback projections from premotor cortex to the parietal lobe, assuming that they form a tuning signal that dynamically changes visual perception. In practice, this can be seen as an implicit attention system that *selects* sensory information while the action is being prepared and subsequently executed (*Flanagan* and *Johansson* [77.53], *Flanagan* et al. [77.54], and *Mataric* and *Pomplun* [77.55]). The early responses, before action onset, of many premotor and parietal neurons suggest a premotor mechanism of attention that deserves exploration in further work in neurorobotics.

77.3 Vertebrate Motor Control

The body of literature on primate motor control is, of course, vast, and gives a patchy view on the principles behind it. Getting a clear view of how limb and general body control functions is difficult; moreover, there are no clear proofs of whether any of the existing views on motor control are correct.

But there exist a few observations of the human central and peripheral nervous systems from which clear conclusions can be drawn. The first observation is the presence of neural communication delays. How does the system know the position of limbs? There are two principled methods: (1) through proprioceptive signals, consisting of muscle spindles and Golgi tendon organs (GOs); and (2) through skin information. It is, however, not very likely that information from muscle spindles and GOs are accurate enough to code limb position. Tendon organs are sensitive to forces along in-series motor units and there is no physiological evidence that Golgi tendon organs signal muscle length (but, of course, force changes with muscle length, so during movement a correlation is found). There is another problem with respect to limb position, which is particularly clear for fingers: flexibilities and nonlinear relationships between finger position and muscle force, in combination with the imprecise receptors, makes the relationship between GO/spindle data and finger posi-

Table 77.1 Classification of sensory fibers from muscle (after [77.56])

Type	Receptor	Axon diameter [μm]	Transmission speed [m/s]	Sensitive to
Ia	Primary spindle endings	12–20	60–100	Muscle length and rate of change
Ib	Golgi tendon organs	12–20	60–100	Muscle tension
II	Secondary spindle endings	6–12	30–70	Muscle length
III	Free nerve endings	2–6	10–30	Skin signals

tion too complex and variable to be a likely candidate to code finger position; after all, the sensors are in the forearm rather than in the fingers; and information on finger position is not available in muscle movement or tendon force. Furthermore, muscle spindle data is noisy [77.57]. It has conversely been shown [77.58, 59] that the receptors in hairy skin code information that can be related to finger position; furthermore, similar data have been found for the knee joint [77.60]. Table 77.1 lists nerve transmission speeds for these signals.

Since neurons are only to be found in the spinal cord and the brain, for hand skin, therefore, we can expect signal transfer delays to the spine of around 30–100 ms. Round-trip muscle activation is, therefore, around 70 ms for signals based on skin data [77.61], or around 25 ms for spindle-based signals (we have verified these delays by measuring hand skin-based reflexes by measuring the corresponding electromyography(EMG) signals, and found a round-trip delay of around 75 ms. Spindle-based feedback for the wrist was measured at around 25 ms.

Of course, when a sensory signal has to be processed in the brain, the delays are correspondingly longer. At any rate: error-correcting feedback control has delays of several tens of milliseconds; feedback control based on such delays cannot lead to any acceptable accuracy with the movement speeds that humans typically display. This means that large portions of our movement, over time frames in the order of 100 ms or more, need to be controlled open loop.

A second important observation is our generalizing capabilities. Consider the case of playing fast and accurate sports, e.g., table tennis. During play, we obtain sensory visual, haptic, and tactile sensory data, the result of which must lead to an accurate movement of the bat in order to score a point. Even a player with little training is able to do this rather accurately: at ball flight times between 200 and 500 ms, the brain does not have much time to plan an accurate whole-body movement for each and every possible sensory state, but we are usually capable of returning the ball. Training helps, but we do not need to exhaustively learn many states in the very high-dimensional sensor space in which our observations move.

Generalization can only be done with reasonably accurate models of the sensor/motor behavior. However, models of our motor system are difficult to obtain: variations such as including payloads, wearing heavy clothing, muscle fatigue, etc., do not influence our accuracy considerably.

77.3.1 The Flat Hierarchy of Neurocontrol

How is an open-loop movement generated? In this paper we concentrate on voluntary vertebrate motor control; the only reason for any animal to have a brain is to generate movement. Moreover, despite differences in brain structures, there is a large correspondence in movement patterns among the whole animal kingdom, irrespective of the presence of a cortical structure or a cerebellum. What parts of the brain are directly involved in movement?

The major role of the cerebral cortex seems to be unsupervised learning to establish relationships between sensory and action patterns [77.62]. The neocortex is only to be found in mammals; experiments with decorticated cats [77.63] clearly show that the cortex is not necessary to generate movement; rather, it is likely that the motor cortex models and weighs movements, to subsequently make decisions based thereon.

The major role of the basal ganglia seems to be reinforcement learning to filter out unwanted movements [77.62]. They play a dominant role in movement generation or gating (filtering) of generated movement patterns. The effect of Parkinson's disease (the inability to initiate movement) and Huntington's disease (the inability to prevent unwanted movement) on the basal ganglia is well known and clearly indicates their function.

The major role of the cerebellum seems to be supervised learning of motor patterns [77.62]. Moreover, decerebellation does not lead to complete movement loss. An individual with cerebellar lesions may be able to move the arm to successfully reach a target and to successfully adjust the hand to the size of an object. However, the action cannot be made swiftly and accurately, and the ability to coordinate the timing of the two subactions is lacking. The behavior will thus exhibit decomposition of movement – first the hand is moved until the thumb touches the object, and only then is the hand shaped appropriately to grasp the object [77.64].

Robot control usually favors a strict hierarchical approach. A typical robot works as follows. At the low-

est level, a very fast ($\approx 100\,\mu s$) current control loop controls the rotation of the dc motor. On top of that, a torque controller (running typically at 1 kHz) controls the torque of all joints, and is in its turn controlled by an impedance or position controller. On top of that, typically, a Cartesian path planner forms the slowest loop. An error in any of these elements will disable the robot.

A look into evolutionary development of neural control thus makes immediately clear that a strict hierarchical approach is not viable in neural control. Although any of the above-mentioned brain regions is important in movement control, and similar structures can be found in any vertebrate, their dysfunction leads to movement degradation but not to movement loss (this is, of course, not true for the spinal cord, which (combines and) transmits the controls to the muscles). Also, the development of the neural system shows that animals were always capable of movement – irrespective of their brain structure. However, the cerebellum is usually rightly focused upon when analyzing vertebrate movement. How do the parts of the brain collaborate towards smooth goal-directed movement?

In placing the function of the cerebellum in the loop, a normal distinction is to consider the cerebellum as representing (a) a forward or direct model which represents the path from motor command to motor output, or (b) an inverse model of motor function, i. e., going from a desired motor outcome to a set of motor commands likely to achieve it. As we have just suggested, the action plan unfolds as if it were feedforward or open loop when the actual parameters of the situation match the stored parameters, while a feedback component is employed to counteract disturbances (current feedback) and to learn from mistakes (learning from feedback). This is obtained by relying on a forward model that predicts the outcome of the action as it unfolds in real time. The accuracy of the forward model can be evaluated by comparing the output generated by the system with the signals derived from sensory feedback (*Miall* et al. [77.65]). Also, delays must be accounted for to address the different propagation times of the neural pathways carrying the predicted and actual outcome of the action. Note that the forward model in this case is relatively simple, predicting only the motor output in advance; since motor commands are generated internally it is easy to imagine a predictor for these signals (known as an efference copy). The inverse model, on the other hand, is much more complicated since it maps sensory feedback (e.g., vision) back into motor terms.

We suggest a much simpler approach to the vertebrate control system. However, let us first look into the functionality of the lower-level apparatus: muscle, spinal cord, and cerebellum.

77.3.2 On Spinal Cord and Muscle

The key element in movement generation is given by two building blocks: (a) our muscles, and (b) the spinal cord. Muscle behavior is strongly nonlinear; the exerted force decreases nonlinearly with velocity (Fig. 77.3) and varies nonlinearly with length (Fig. 77.4).

Limb movement, however, is caused by a complex of muscles – for instance, the human arm uses a total of 19 muscle groups for planar motion of the elbow and shoulder alone (*Nijhof* and *Kouwenhoven* [77.67]) with altogether highly nonlinear dynamics. How can this large number of actuators be controlled without feedback error control?

The concept is simple and was first well described by *Bernstein* [77.68]: skeletal muscles are always controlled in functional groups, leading to *synergies* of movement. Rather than activating muscles indepen-

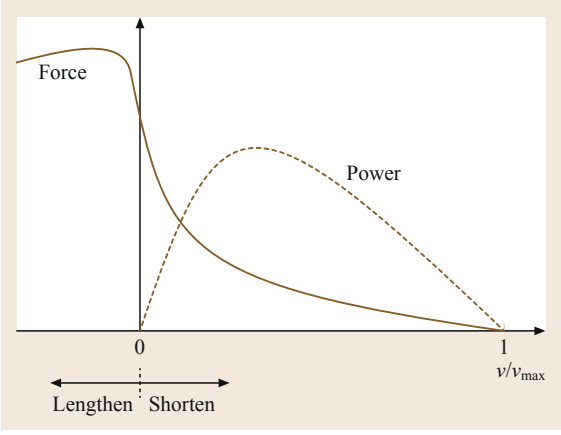


Fig. 77.3 The force/velocity and power/velocity relationship of muscle (after [77.56])

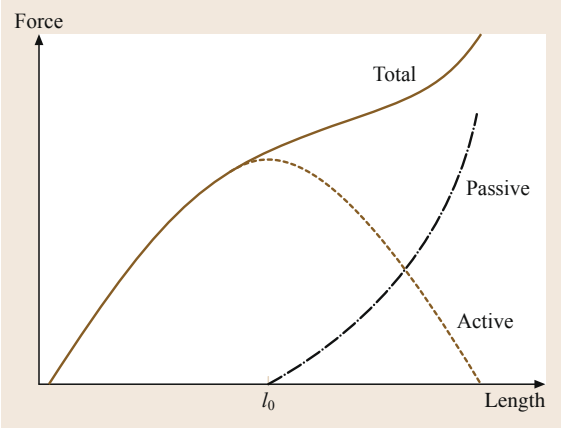


Fig. 77.4 The force/length relationship of muscle (after [77.56])

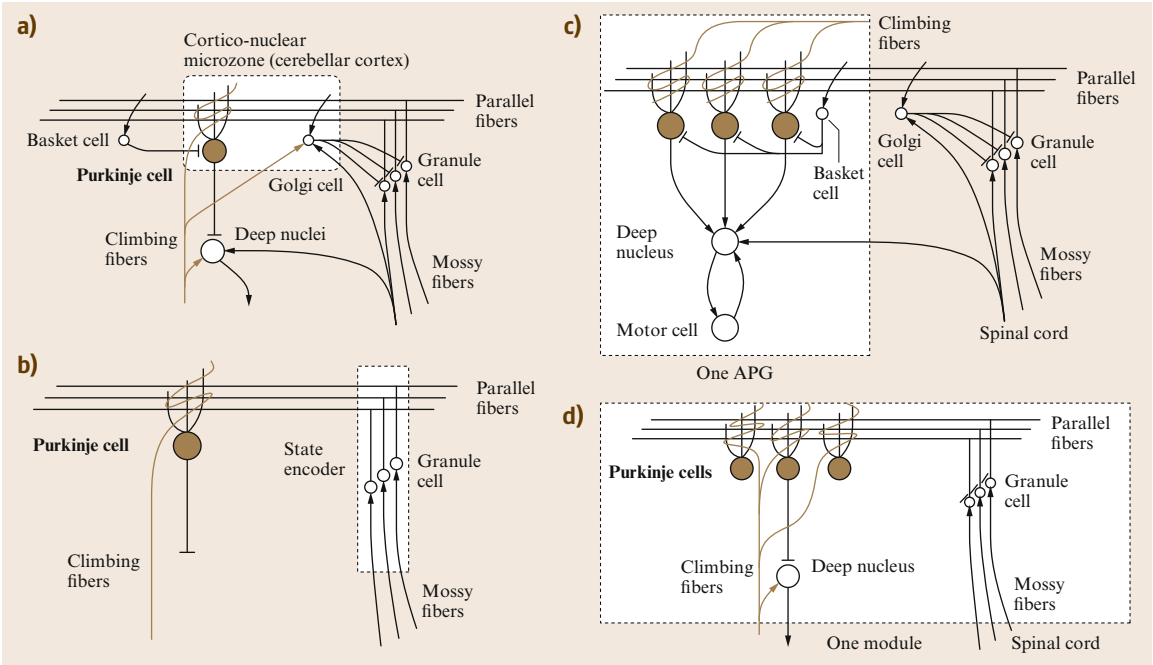


Fig. 77.5 **(a)** Major cells in the cerebellum. **(b)** Cells in the Marr–Albus model. The granule cells are state encoders, feeding system state, and sensor data into the PC. PC/PF synapses are adjusted using the Widrow–Hoff rule. The output of the PC are steering signals for the robotic system. **(c)** The APG model, using the same state encoder as in **(b)**. **(d)** The MPFIM model. A single module corresponds to a group of Purkinje cells: predictor, controller, and responsibility estimator. The granule cells generate the necessary basis functions of the original information (after [77.66])

dently, a neural signal controls groups of muscles that perform (a part of) an action. Linear dimension reduction methods [77.69] (e.g., principal component analysis, (PCA), independent component analysis (ICA), or non-negative matrix factorization (NMF)) have been used to establish synergies in EMG data, and this can be used [77.70] to linearly combine single-finger movement to whole-hand movement in EMG space. So, we cannot control single muscles (i. e., coherent groups of muscle fibers) but rather control muscle groups, the linear combination of which can be used to span a decent part of our voluntary movement.

There are currently still open questions as to the nature of movement synergies: how much of the synergies are defined by the biomechanical structure of our muscles and tendons; how much of it is laid out in the spinal cord; and which part of it is learned in the higher movement control regions?

77.3.3 Models of Cerebellar Control

The cerebellum can be divided into two parts: the cortex and the deep nuclei. There are two systems of fibers bringing input to the both the cortex and nuclei: the mossy fibers and the climbing fibers. The only output from the cerebellar nucleus comes from cells called Purkinje cells, and they project only to the cerebellar nuclei, where their effect is inhibitory. This inhibition sculpts the output of the nuclei which (the effect varies from nucleus to nucleus) may act by modulating activity in the spinal cord, the mid-brain, or the cerebral cortex. We now turn to models that make explicit use of the cellular structure of the *cerebellar* cortex (see *Eccles* et al. [77.71] and *Ito* [77.72], and also Fig. 77.5a). The human cerebellum has 7−14 million Purkinje cells (PCs), each receiving about 200 000 synapses. Mossy fibers (MFs) arise from the spinal cord and brainstem. They synapse onto granule cells and deep cerebellar nuclei. Granule cells have axons which each project up to form a T, with the bars of the T forming the parallel fibers (PFs). Each PF synapses on about 200 PCs. The PCs, which are grouped into microzones, inhibit the deep nuclei. PCs with their target cells in cerebellar nuclei are grouped together in microcomplexes [77.72]. Microcomplexes are defined by a variety of criteria to serve as the units of analysis of cerebellar influence on specific types of motor activity. The climbing fibers (CFs) arise from the inferior olive (IO). Each PC receives synapses from only one CF, but a CF makes about 300 excitatory synapses on each PC that it con-

tacts. This powerful input alone is enough to fire the PC, though most PC firing depends on subtle patterns of PF activity. The cerebellar cortex also contains a variety of inhibitory interneurons. The basket cell is activated by PF afferents and makes inhibitory synapses onto PCs. Golgi cells receive input from PFs, MFs, and CFs, and inhibit granule cells.

The Marr–Albus Model

In the Marr–Albus model (*Marr* [77.73] and *Albus* [77.74]) the cerebellum functions as a classifier of sensory and motor patterns received through the MFs. Only a small fraction of the parallel fibers (PF) are active when a Purkinje cell (PC) fires and thus influence the motor neurons. Both Marr and Albus hypothesized that the error signals for improving PC firing in response to PF, and thus MF input were provided by the climbing fibers (CF), since only one CF affects a given PC. However, Marr hypothesized that CF activity would strengthen the active PF/PC synapses using a Widrow–Hoff learning rule, whereas Albus hypothesized they would weaken them. This is an important example of a case where computational modeling inspired important experimentation. Eventually, *Masao Ito* was able to demonstrate that Albus was correct – the weakening of active synapses is now known to involve a process called long-term depression [77.72]. However, the rule with weakening of synapses is still known as the Marr–Albus model, and remains the reference model for studies of synaptic plasticity of the cerebellar cortex. However, both Marr and Albus viewed each PC as functioning as a perceptron whose job it was to control an elemental movement, contrasting with more plausible models in which PCs serve to modulate the involvement of microcomplexes (which include cells of the deep nuclei) in motor pattern generators (e.g., the APG model described below).

Since the development of the Marr–Albus model several cerebellar models have been introduced in which cerebellar plasticity plays a key role. Limiting our overview to computational models, we will describe:

1. The CMAC (cerebellar model articulation controller).
2. The adjustable pattern generator (APG).
3. The Schweighofer–Arbib model.
4. The multiple paired forward-inverse models [77.75, 76].

The Cerebellar Model Articulation Controller

One of the first well-known computational models of the cerebellum is the CMAC (*Albus* [77.77]; Fig. 77.5b). The algorithm was based on Albus' understanding of the cerebellum, but it was not proposed as a biologically plausible model. The idea has its origins in the BOXES approach, in which for n variables an n-dimensional hypercube stores function values in a lookup table. BOXES suffers from the curse of dimensionality: if each variable can be discretized into D different steps, the hypercube has to store D^n function values in memory. Albus assumed that the mossy fibers provided discretized function values. If the signal on a mossy fiber is in the receptive field of a particular granule cell, it fires onto a parallel fiber. This mapping of inputs onto binary output variables is often considered to be the generalization mechanism in CMAC. The learning signals are provided by the climbing fibers.

Albus' CMAC can be described in terms of a large set of overlapping, multidimensional receptive fields with finite boundaries. Every input vector falls within the range of some local receptive fields. The response of CMAC to a given input is determined by the average of the responses of the receptive fields excited by that input. Similarly, the training for a given input vector affects only the parameters of the excited receptive fields.

The organization of the receptive fields of a typical Albus CMAC with a two-dimensional input space can be described as follows. The set of overlapping receptive fields is divided into C subsets, commonly referred to as *layers*. Any input vector excites one receptive field from each layer, for a total of C excited receptive fields for any input. The overlap of the receptive fields produces input generalization, while the offset of the adjacent layers of receptive fields produces input quantization. The ratio of the width of each receptive field (input generalization) to the offset between adjacent layers of receptive fields (input quantization) must be equal to C for all dimensions of the input space. This organization of the receptive fields guarantees that only a fixed number, C, of receptive fields is excited by any input.

If a receptive field is excited, its response equals the magnitude of a single adjustable weight specific to that receptive field. The CMAC output is the average of the weights of the excited receptive fields. If nearby points in the input space excite the same receptive fields, they produce the same output value. The output only changes when the input crosses one of the receptive field boundaries. The Albus CMAC thus produces piecewise-constant outputs. Learning takes place as described above.

CMAC neural networks have been applied in various control situations *Miller* [77.78], starting from adaptation of PID (proportional–integral–derivative) control parameters for an industrial robot arm and

hand–eye systems up to biped walking (see, for instance, *Sabourin* and *Bruneau* [77.79]).

The Adjustable Pattern Generator APG

The APG model (*Houk* et al. [77.80]) got its name because the model can generate a burst command with adjustable intensity and duration. The APG is based on the same understanding of the mossy fiber–granule cell–parallel fiber structure as CMAC, using the same state encoder, but has the crucial difference (Fig. 77.2c) that the role of the nuclei is crucial. In the APG model, each nucleus cell is connected to a motor cell in a feedback circuit. Activity in the loop is then modulated by Purkinje cell inhibition, a modeling idea introduced by *Arbib* et al. [77.81].

The learning algorithm determines which of the PF–PC synapses will be updated in order to improve movement generation performance. This is the traditional credit assignment problem: *which* synapse (the structural credit assignment) must be updated based on a response issued *when* (temporal credit assignment). While the former is solved by the CFs, which are considered binary signals, for the latter eligibility traces on the synapses are introduced, serving as memory for recent activity to determine which synapses are eligible for updates. The motivation for the eligibility signal is this: each firing of a PC cell will take some time to affect the animal's movement, and a further delay will occur before the CF can signal an error in the movement in which the PC is involved. Thus the error signal should not affect those PF–PC synapses that are currently active, but should instead act upon those synapses that affected the activity whose error is now being registered.

The Schweighofer–Arbib Model

The Schweighofer–Arbib model was introduced in *Schweighofer* [77.82]. It does not use the CMAC state encoder but tries to copy the anatomy of the cerebellum. All the cells, fibers, and axons in Fig. 77.2a are included. Several assumptions are made:

1. There are two types of mossy fibers, one type reflecting the desired state of the controlled plant and another carrying information on the current state. A mossy fiber diverges into approximately 16 branches.
2. Granule cells have an average of four dendrites, each of which receive input from different mossy fibers through a synaptic structure called the glomerulus.
3. Three Golgi cells synapse on a granule cell through the glomerulus and the strength of their influence depends on the simulated geometric distance between the glomerulus and the Golgi cell.
4. The climbing fiber connection on nuclear cells as well as deep nuclei is neglected.

Learning in this model depends on directed error information given by the climbing fibers from the inferior olive (IO). Here, long-term depression is performed when the IO firing rate provides an error signal for an eligible synapse, while compensatory but slower increases in synaptic strength can occur when no error signal is present. Schweighofer applied the model to explain several acknowledged cerebellar system functions:

1. Saccadic eye movements
2. Two-link limb movement control *Schweighofer* et al. [77.83, 84]
3. Prism adaptation (*Arbib* et al. [77.85]).

Furthermore, control of a simulated human arm was demonstrated.

Multiple Paired Forward–Inverse Models (MPFIM)

Building on a long history of cerebellar modeling, *Wolpert* and *Kawato* [77.86] proposed a functional model of the cerebellum, which uses multiple coupled predictors and controllers that are trained for control, each being responsible for a small state-space region. The MPFIM model is based on the indirect/direct model approach by Kawato, and is also based on the microcomplex theory. We noted earlier that a microzone is a group of PCs, while a microcomplex combines the PCs of a microzone with their target cells in cerebellar nuclei. In MPFIM, a microzone consists of a set of modules controlling the same degree of freedom and is learned by only one particular climbing fiber. The modules in this microzone compete to control this particular synergy. Inside such a module there are three types of PC, which perform the computations of a forward model, an inverse model, or a responsibility predictor, but all receiving the same input. A single internal model i is considered to be a controller that generates a motor command τ_i and a predictor that predicts the current acceleration. Each predictor is a forward model of the controlled system, while each controller contains an inverse model of the system in a region of specialization. The responsibility signal weights the contribution that this model will make to the overall output of the microzone. Indeed, MPFIM further assumes that each microzone contains n internal models of situations occurring in the control task. Model i generates motor command τ_i, and estimates its own responsibility r_i. The feedforward motor command τ_{ff} consists only of the output of the single models adjusted by the sum of responsibility signals: $\tau_{\mathrm{ff}} = \sum r_i \tau_i / \sum r_i$.

The PCs are considered to be roughly linear. The MF inputs carry all necessary information including state information, efference copies of the last motor commands, as well as desired states. Granule cells, and eventually the inhibitory interneurons as well, nonlinearly transform the state information to provide a rich set of basis functions through the PFs. A climbing fiber carries a scalar error signal while each Purkinje cell encodes a scalar output – responsibilities, predictions, and controller outputs are all one-dimensional values. MPFIM has been introduced with different learning methods: its first implementations were done using gradient descent methods; subsequently, expectation maximization (EM) batch-learning, and hidden Markov chain EM learning were applied.

Comparison of the Models

Summing up, we can categorize the cerebellar models CMAC, APG, Schweighofer–Arbib, and MPFIM as follows:

- *State-encoder-driven models*: This kind of model assumes that the granule cells are on–off types of entities that split up the state space. This kind of model is best suited for, e.g., simple function approximation, and suffers strongly from the curse of dimensionality.
- *Cellular-level models*: Obviously, the most realistic simulations would be at the cellular level. Unfortunately, modeling only a few Purkinje cells at realistic conditions is an immense computational challenge, and other relevant neurons are even less well understood. Still, from the biological point of view this kind of model is the most important since it allows obtaining insight into cerebellar function on cellular level. The first steps in this direction were taken by the Schweighofer–Arbib model.
- *Functional models*: From the computer-science point of view, the most interesting models are based on functional understanding of the cells. In this case, we obtain only a basic insight of the functions of the parts and apply it as a crude approximation. This kind of approach is very promising and MPFIM, with its emphasis on the use of responsibility signals to combine models appropriately, provides an interesting example of this approach.

Proprioceptive feedback is used for adaptation of the motor programs as well as for updating the forward model stored in the cerebellum. However, the Schweighofer–Arbib model is based on the view that the cerebellum offers not so much a total forward model of the skeletomuscular system as a forward model of the difference between the crude model of the skeletomuscular system available to the motor planning circuits of the cerebral cortex, and the more intricately parameterized forward model of the skeletomuscular system needed to support fast, graceful movements with minimal use of feedback. This hypothesis is reinforced by the fact that cerebellar lesions do not prohibit motion but substantially reduce its quality, since the forward model of the skeletomuscular system is of lesser quality.

77.3.4 Cerebellar Models and Robotics

From the previous discussions, it is clear that a popular view is that the function of the cerebellum within the motor control loop is to represent a forward model of the skeletomuscular system; but how can these models be used in control?

Our assumption is that the cerebellum stores motor primitive relationships, which can be recalled through a certain state (i. e., sensor plus cerebrum-directed goal) input. These motor primitives perform certain coordinated movements (*synergies*) to, e.g., intercept a ball with a tennis racket. A key property of the underlying spine-controlled musculoskeletal system, however, is that *voluntary movement can be easily interpolated* within the control realm of the spinal cord. With this we mean that the combination of two movement primitives that are *nearby* in the relevant sensor domain will lead to a good prediction. *In one possible interpretation, the spinal cord-based control of our muscular system is approximately linear* or *linearized through internal models* [77.87]. It allows the cerebellum to store or recall movements at any level of granularity, and get good enough results in unlearned areas. There are various papers which, in part, confirm this theory (e.g., *Osu* and *Gomi* show the linear relationship between muscle activation and joint stiffness [77.88] or Höppner et al. between grip force and stiffness [77.89]).

Does this understanding of the human control system help robotics? Biological control algorithms are certainly a result of slow feedback loops and the flexibility of the actuators. One may argue that, as robotic systems move towards their biological counterparts, the control approaches can or must do the same. There are many lines of research investigating the former part; Chaps. 11 and 75. It should be noted that the drive principle that is used to move the joints does not necessarily have a major impact on the outer control loop. Whether McKibben muscles, which are intrinsically flexible but bulky (*van der Smagt* et al. [77.90]), low-dynamics polymer linear actuators, or direct-current (DC) motors with spindles and added elastic components are used does not affect the control approach at the cerebellar level, but rather at the motor control level (cf. the spinal

cord level). Of key importance, however, are the resulting dynamic properties of the system, which are, of course, influenced by its actuators. Linearity of the low-level control system, as we find in biology, is a goal to strive for. Yet technical systems can benefit from advanced modeling approaches, and equally good results can be obtained – yet at the cost of more complex sensing, computation, and less generalizability.

77.4 The Role of Mirror Systems

Area F5 (frontal area 5) in the premotor cortex of the macaque contains, among others, neurons which fire when the monkey executes a specific manual action, e.g., one neuron might fire when the monkey performs a precision pinch, another when it executes a power grasp. (In discussing neurorobotics, it seems unnecessary to explain in any detail the areas like F5, AIP (anterior intraparietal sulcus), and STS (superior temporal sulcus) described here – they will function as labels for components of functional systems. To fill in the missing details see, e.g., *Rizzolatti* et al. [77.91, 92].)

77.4.1 Mirror Neurons and the Recognition of Hand Actions

A subset of these neurons, the so-called *mirror neurons*, also discharge when the monkey observes meaningful hand movements made by the experimenter, which are similar to those whose execution is associated with the firing of the neuron. In contrast, the *canonical neurons* are those belonging to the complementary, anatomically segregated subset of grasp-related F5 neurons, which fire when the monkey performs a specific action and also when it sees an object as a possible target of such an action – but do not fire when the monkey sees another monkey or human perform the action. Finally, F5 contains a large population of motor neurons that are active when the monkey grasps an object (either with the hand or mouth) but do not possess any visual response. F5 is clearly a motor area although the details of the muscular activation are abstracted out – F5 neurons can be effector-independent. In contrast, the primary motor cortex (F1) formulates the neural instructions for lower motor areas and motor neurons.

Moreover, macaque mirror neurons encode transitive actions and do not fire when the monkey sees the hand movement unless it can also see the object or, more subtly, if the object is not visible but is appropriately *located* in working memory because it has recently been placed on a surface and has then been obscured by a screen behind which the experimenter is seen to be reaching (*Umiltà* et al. [77.93]). All mirror neurons show visual generalization. They fire when the instrument of the observed action (usually a hand) is large or small, far from or close to the monkey. They may also fire even when the action instrument has shapes as different as those of a human or monkey hand. Some neurons respond even when the object is grasped by the mouth. When naive monkeys first see small objects grasped with a pair of pliers, mirror neurons do not respond, but after extensive training some precision pinch mirror neurons do show activity, also with this new grasp type [77.94].

Mirror neurons for grasping have also been found in parietal areas of the macaque brain and, recently, it was shown that parietal mirror neurons are sensitive to the context of the observed action being predictive of the outcome as a function of contextual cues – e.g., some grasp-related parietal mirror neurons may fire for a grasp that precedes eating the grasped object, while others fire for a grasp that precedes placing the object in a container (*Fogassi* et al. [77.95]). In practice, the parieto-frontal circuitry seems to encode action execution and simultaneously action recognition by taking into account a large set of potential candidate actions, which are selected on the basis of a range of cues such as vision of the relation of the effector to the object and certain sounds (when relevant for the task). Further, feedback connections (frontal to parietal) are thought to be part of a stimulus selection process that refines the sensory processing by *attending* to stimuli relevant for the ongoing action (*Rizzolatti* et al. [77.52] and recall the discussion in Sect. 77.2.4). Recognition is then supported by the activation of the same circuitry in the absence of overt movement.

We clarify these ideas by briefly presenting the FARS model of the canonical F5 neurons and the MNS model of the F5 mirror neurons. In each case, the F5 neurons function effectively only because of the interaction of F5 with a wide range of other regions. We have stressed (Sect. 77.2.3) the distinction between recognition of the category of an object and recognition of its *affordances*. The parietal area AIP processes visual information to extract *affordances*, in this case properties of the object relevant to grasping it (*Taira* et al. [77.96]). AIP and F5 are reciprocally connected, with AIP being more visual and F5 more motoric.

The Fagg–Arbib–Rizzolatti–Sakata (FARS) model (*Fagg* and *Arbib* [77.97] and Fig. 77.6) embeds F5 canonical neurons in a larger system. The dorsal stream

(which passes through AIP) can only analyze the object as a set of possible affordances, whereas the ventral stream (via the inferotemporal cortex, IT) is able to recognize what the object is. The latter information is passed to the prefrontal cortex (PFC) which can then, on the basis of the current goals of the organism, bias the choice of affordances appropriate to the task at hand. Neuroanatomical data (as analyzed by *Rizzolatti* and *Luppino* [77.98]) suggest that PFC and IT may modulate action selection at the level of the parietal cortex. Figure 77.6 gives a partial view of the FARS model updated to show this modified pathway. The affordance selected by AIP activates F5 neurons to command the appropriate grip once they receive a *go* signal from another region, F6, of the prefrontal cortex. F5 also accepts signals from other PFC areas to respond to working memory and instruction stimuli in choosing among the available affordances. Note that this same pathway could be implicated in tool use, bringing in semantic knowledge as well as perceptual attributes to guide the dorsal system (*Johnson–Frey* [77.99]).

With this, we turn to the *mirror* system. Since grasping a complex object requires careful attention to motion of, e.g., fingertips relative to the object, we hold that the primary evolutionary impetus for the mirror system was to facilitate feedback control of dexterous movement. We now show how *parameters relevant to such feedback* could be crucial in enabling the monkey to associate the visual appearance of what it is doing with the task at hand. The key side-effect will be that this feedback-serving self-recognition is so structured as to also support recognition of the action when performed by others – and it is this *recognition of the actions of others* that has created the greatest interest in mirror neurons and systems.

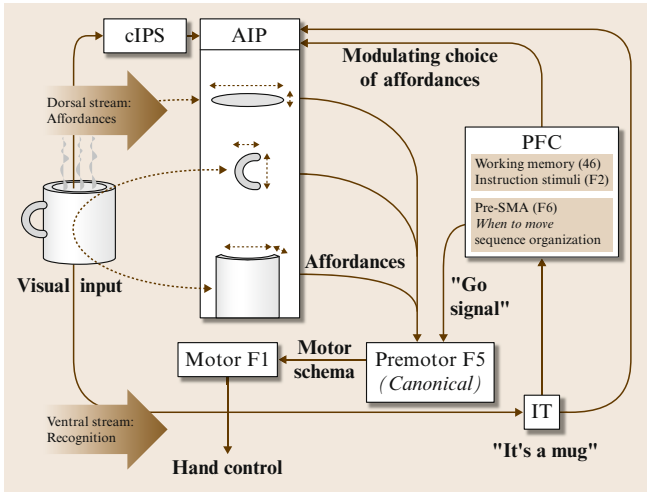


Fig. 77.6 The original FARS diagram (after *Fagg* and *Arbib* [77.42]) is here modified to show PFC acting on AIP rather than F5. The idea is that the prefrontal cortex uses the IT identification of the object, in concert with task analysis and working memory, to help the AIP select the appropriate affordance from its *menu*

The MNS model of *Oztop* and *Arbib* [77.101] provides some insight into the anatomy while focusing on the learning capacities of mirror neurons. Here, the task is to determine whether the shape of the hand and its trajectory are *on track* to grasp an observed affordance of an object using a known action. The model is organized around the idea that the AIP → $F5_{canonical}$ pathway emphasized in the FARS model (Fig. 77.6) is

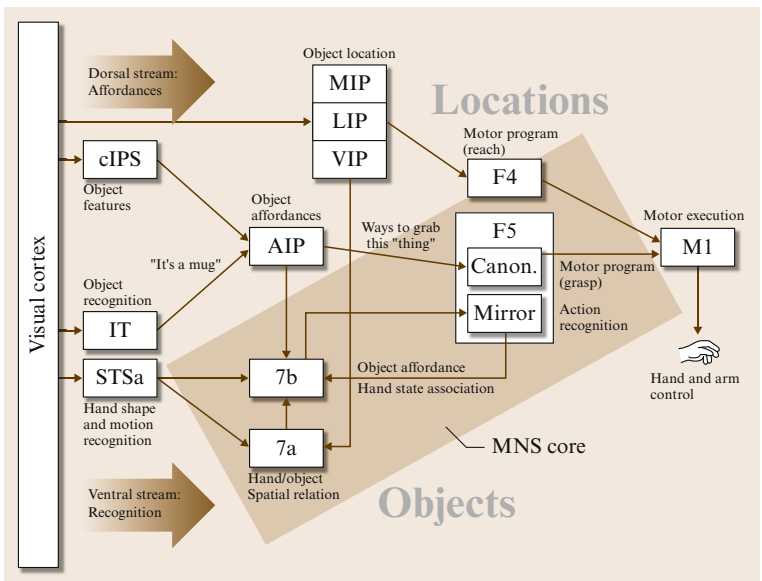


Fig. 77.7 The mirror neuron system (MNS) model (after *Oztop* and *Arbib* [77.100]). Note that this basic mirror system for grasping crucially links the visual process of the STS to the parietal regions (b) and premotor regions (F5), which have been shown to contain mirror neurons for manual actions

complemented by another pathway 7b → $F5_{mirror}$. As shown in Fig. 77.7 (middle diagonal), object features are processed by AIP to extract grasp affordances; these are sent on to the canonical neurons of F5 that choose a particular grasp. Recognizing the location of the object (top diagonal) provides parameters to the motor programming area F4 which computes the reach. The information about the reach and the grasp is taken by the motor cortex M1 (= F1) to control the hand and the arm. The rest of the figure provides components that can learn and apply key criteria for activating a mirror neuron, recognizing that the preshape of the observed hand corresponds to the grasp that the mirror neuron encodes and is appropriate to the object, and that the hand is moving on an appropriate trajectory. Making crucial use of input from the superior temporal sulcus (*Perrett* et al. [77.102] and *Carey* et al. [77.103]), schemas at the bottom left recognize the shape of the observed hand and how that hand is moving. Other schemas implement hand–object spatial relation analysis and check how object affordances relate to hand state. Together with F5 canonical neurons, this last schema (in parietal area 7b) provides the input to the F5 mirror neurons.

In the MNS model, the *hand state* was defined as a vector whose components represented the movement of the wrist relative to the location of the object and of the hand shape relative to the affordances of the object. *Oztop* and *Arbib* showed that an artificial neural network corresponding to PF and $F5_{mirror}$ could be trained to recognize the grasp type from the *hand state trajectory*, with correct classification often being achieved well before the hand reached the object, using activity in the F5 canonical neurons that commands a grasp as training signal for recognizing it visually; this basically shows that there is a causal relationship. Crucially, this training prepares the F5 mirror neurons to respond to hand–object relational trajectories even when the hand is of the *other* rather than the *self* because the hand state is based on the view of movement of a hand relative to the object, and thus only *indirectly* on the retinal input of seeing the hand and object, which can differ greatly between observation of self and other. *Bonaiuto* et al. [77.104] have developed MNS2, a new version of the MNS model to address data on audiovisual mirror neurons that respond to the sight and sound of actions with characteristic sounds such as paper tearing and nut cracking *Kohler* et al. [77.93], and on the response of mirror neurons when the target object was recently visible but is currently hidden *Umiltà* et al. [77.93]. Such learning models, and the data they address, make it clear that:

> *mirror neurons are not restricted to recognition of an innate set of actions but can be recruited to recognize and encode an expanding repertoire of novel actions.*

The discussion of this section avoided any reference to imitation (Sect. 77.4.3). On the other hand, even without considering imitation, mirror neurons provide a new perspective for tackling the problem of robotic perception by incorporating action (and motor information) into a plausible recognition process. The role of the fronto-parietal system in relating affordances, plans, and actions shows the crucial role of motor information and embodiment. We argue that this holds lessons for neurorobotics: the richness of the *motor* system should strongly influence what the robot can learn, proceeding autonomously via a process of exploration of the environment rather than overly relying on the intermediary of logic-like formalisms. When recognition exploits the ability to act, then the breadth of the action space becomes crucially related to the precision, quality, and robustness of the robot's perception.

77.4.2 Computational Models

Roboticists have been fascinated by the discovery of mirror neurons and the purported link to imitation that exists in the human nervous system, for they can help to teach robots new tasks with relative ease. The literature on the topic extends from models of the monkey's (nonimitative) action recognition system (*Oztop* and *Arbib* [77.101]) to models of the putative role of the mirror system in imitation (*Demiris* and *Johnson* [77.105] and *Arbib* et al. [77.106]), and in real and virtual robots (*Schaal* et al. [77.107]). *Oztop* et al. [77.108] propose a taxonomy of the models of the mirror system for recognition and imitation, and it is interesting to note how different the computational approaches that have now been framed as mirror system models are, including recurrent neural networks with parametric bias (*Tani* et al. [77.109]), behavior-based modular networks (*Demiris* and *Johnson* [77.105]), associative memory-based methods (*Kuniyoshi* et al. [77.110]), and the use of multiple direct-inverse models as in the MOSAIC architecture (*Wolpert* et al. [77.111]; cf. the multiple paired forward-inverse models of Sect. 77.3.2).

Following [77.112], we can cast much that is known about the mirror system into a controller-predictor model [77.65, 113] and analyze the resulting model as a Bayesian classifier. As shown by the FARS model, the decision to initiate a particular grasping action is attained by the convergence in area F5 of several factors, including contextual and object-related information; similarly many factors affect the recognition of an action. All this depends on learning both direct (from decision to executed action) and inverse models

(from observation of an action to activation of a motor command that could yield it). Similar procedures are well known in the computational motor control literature [77.114, 115]. Learning of the affordances of objects with respect to grasping can also be achieved autonomously by learning from the consequences of applying many different actions to different parts of different objects.

However, how is the decision made to classify an observed behavior as an instance of one action or another? Many comparisons could be performed in parallel with the models for one action to become predominantly activated. There are plausible implementations of this mechanism using a gating network [77.105, 116]. A gating network learns to partition an input space into regions; for each region a different model can be applied or a set of models can be combined through an appropriate weight function. The design of the gating network can encourage collaboration between models (e.g., linear combination of models) or competition (choosing only one model rather than a combination). Reference [77.117] offers a similar approach to the estimation of the mental states of the observed actor, using some additional circuitry involving the frontal cortex.

On the other hand, if we take the Bayesian view of the predictor-controller formulation, then affordances are simply the priors in the action recognition process where the evidence is conveyed by the visual information of the hand, providing the data for finding the posterior probabilities as mirror neuron-like responses which automatically activate for the most probable observed action. Recall that the presence of a goal (at least in working memory) is needed to elicit mirror neuron responses in the macaque. We believe it is also particularly important during the ontogenesis of the human mirror system. For example, [77.118] has shown that even at 9 months of age, infants recognized an action as being novel if it was directed toward a novel object rather than just having a different kinematics – showing that the goal is more fundamental than the enacted trajectory. Similarly, if one sees someone drinking from a coffee mug then one can hypothesize that a particular action (that one already knows in motor terms) is used to obtain that particular effect. The association between the canonical response (object-action) and the mirror one (including vision) is made when the observed consequences (or goal) are recognized as similar in the two cases. Similarity can be evaluated following criteria ranging from kinematic to social consequences.

In a similar experiment *Lopes* et al. [77.119] compared action recognition performance (a) when using the output of an inverse visuo-motor model and thus employing motor features to aid classification during the training phase, and (b) when only visual data were available for recognition. Overall, their interpretation of the results is that by mapping in motor space through inverse model mapping, they allow the classifier to choose features that are much better suited for performing optimally with respect to the task of recognizing actions, which in turn facilitates generalization. The same is not true when recognition is performed purely in visual space using generic visual features, since a given action is viewed from different viewpoints. One may compare this to the viewpoint-invariant hand state adopted in the MNS model – which has the weakness of being built in rather than emerging from training.

Along the same line, the work of *Gijsberts* et al. [77.120] included motorically-derived affordance information, which was recorded using a data-glove-based system and a set of cameras. In this case though, motor information was not much for action recognition but rather used to *simulate* the response of F5's canonical neurons by generating discrete grasping types from the time-varying set of postures recorded with the data glove. After training the original motor information is removed and only reconstructed using an inverse model. Furthermore, this motoric information was combined with a simulation of the brain ventral pathway which extracts pictorial features from images (e.g., SIFT (scale-invariant feature transform), H-Max). The dorsal and ventral features were combined through a special kernel function in a simple least squares classifier, showing a significant improvement at recognizing objects in comparison to a purely visual classification. A machine learning framework to address the question of learning from multimodal signals (some of which can even be intermittent) is presented in [77.121].

We can speculate that this computational advantage (better recognition rates) makes the presence of mixed sensory and motor information compelling in the brain (i. e., the fronto-parietal system); this may not necessarily lead to mirror neurons although it seems plausible that any clear advantage of using information at best is eventually selected during evolution. These experiments, using robots, simulations, and computational arguments can thus explain the whys of certain brain structures and mechanisms.

77.4.3 Mirror Neurons and Imitation

*Fitzpatrick*and *Metta* [77.122] also addressed the question of what is further required for interpreting observed actions. Whereas in observing its own actions, the robot identifies them from the effects on the objects, later it could backtrack and derive the type of action needed to replicate a certain observed effect on a given object.

Therefore, imitation can be framed into the identification of a common goal between the observed action and various possible actions in the *motor repertoire* of the robot. In [77.122] the robot used the same visual processing algorithms both in observing its own hand and the hand of a person (although they were different in appearance). One might argue that observation alone can be used for learning, never relying on active exploration of objects and actions. This is possibly true to the extent that passive vision is reliable and action is not required. The advantage of the active approach, at least for the robot, is that it allows controlling the amount of information impinging on the visual sensors by, for instance, controlling the speed and type of action. This strategy might be especially useful given the limitations of artificial perceptual systems. Thus, observations can be converted into interpreted actions. The action whose effects are closest to the observed consequences on the object (which we might translate into the goal of the action) is selected as the most plausible interpretation given the observation. Most importantly, the interpretation reduces to the interpretation of the *simple* kinematics of the goal and consequences of the action rather than to understanding the *complex* kinematics of the human manipulator. The robot understands only to the extent it has learned to act. One might note that a more refined model should probably include visual cues from the appearance of the manipulator into the interpretation process. Indeed, the *hand state* that was central to the *Oztop–Arbib* model was based on an object-centered view of the hand's trajectory in a coordinate frame based on the object's affordances. The last question to address is whether a robot can imitate the *goal* of the action. The step is indeed small, since most of the complexity is actually in interpreting observations. Imitation can be generated by replicating the latest observed human movement with respect to the object utilizing one of the many approximation methods for motion generation such as, e.g., a mixture of Gaussians [77.123], dynamic motion primitives [77.124], or reinforcement learning [77.125]. More generally, following the work of *Schaal* et al. [77.107] and *Oztop* et al. [77.108] we can propose a set of schemas required to produce imitation:

- Determining what to imitate, inferring the goal of the demonstrator
- Establishing a metric for imitation (correspondence; see *Nehaniv* [77.126])
- Map between dissimilar bodies (mapping).
- Imitating behavior formation.

These are also discussed in greater detail by *Nehaniv* and *Dautenhahn* [77.127]. In practice, computational and robotic implementations have tackled these problems with different approaches and emphasizing different parts or specific subproblems of the whole, for example, in the work of *Demiris* and *Hayes* [77.128], the rehearsal of the various actions (akin to the aforementioned theory of motor perception) was used to generate hypotheses to be compared with the actual sensory input. It is then remarkable how more recently a modified approach of this paradigm has been used in comparison with real human transcranial magnetic stimulation (TMS) data.

Ito et al. [77.129] (not Masao Ito of cerebellar fame) took a dynamical systems approach using a recurrent neural network with parametric bias (RNNPB) to teach a humanoid robot to manipulate certain objects. In this approach the parametric bias (PB) encodes (tags) certain sensorimotor trajectories. Once learning is complete, the neural network can be used either to recall a given trajectory by setting the PB externally or provide input for the sensory data only and observe the PB vector that would represent in that case the recognition of the situation on the basis of the sensory input only (no motor information available). It is relatively easy to interpret these two situations as the motor generation and the observation in a mirror neurons model.

The problem of building useful mappings between dissimilar bodies (consider a human imitating a bird's flapping wings) was tackled by *Nehaniv* and *Dautenhahn* [77.127] where an algebraic framework for imitation is described and the correspondence problem formally addressed. Any system implementing imitation should clearly provide a mapping between either dissimilar bodies or even in the case of similar bodies when either the kinematics or dynamics is different depending on the context of the imitative action.

Sauser and *Billard* [77.130] modeled the *ideomotor principle*, according to which observing the behavior of others influences our own performances. The ideomotor principle points directly to one of the core issues of the mirror system, that is, the fact that watching somebody else's actions changes something in the activation of the observer, thus facilitating certain neural pathways. The work in question also gives a model implemented in terms of neural fields (see *Sauser* and *Billard* [77.130] for details) and tries to explain the imitative cortical pathways and the behavior formation.

77.4.4 Mirror System and Speech

Already in the 1960s *Liberman* et al. [77.131] started to discuss the possible links between production and perception in speech: in other words the contribution of articulation into the perception of utterances. Later he commented [77.132]:

A result in all cases is that there is not, first, a cognitive representation of the proximal pattern that is modality-general, followed by a translation to a particular distal property; rather, perception of the distal property is immediate, which is to say that the module has done all the hard work.

Liberman argued that there is no such a thing as a modality-aspecific representation which then becomes speech as an effect of a translation to a specific set of articulators (the vocal apparatus in this case), rather, he claimed that perception of speech is immediate and effected by the same speech module (the same that generates speech); speech remains a motor fact. Lately, theories of the motor involvement in speech perception have gained credit because of the discovery of the mirror neurons. It has been postulated that the mirror system in humans *controls* jointly speech production and perception, whereby the actions in speech are the articulation of appropriate segments of the utterances [77.133].

We recall this line of reasoning in the following [77.133], that is:

- Mirror neurons (or a mirror system) exist in humans [77.134].
- The human mirror system is identified in Broca's area, a cytoarchitectonical homolog of area F5 in the macaque's brain.
- Speech articulation is coded/controlled in/by the areas of the human mirror system (Broca's) [77.135].
- The recognition of the intention of the speaker by the listener owing to a mirror mechanism leads to the first seed of true communication (via, e.g., orofacial gestures) [77.133].
- The combinatorial properties of F5/Broca and the precise control of the effectors are needed to generate speech (the evolutionarily older animal calls are too stereotyped to grant this flexibility that eventually leads to speech proper) [77.136].

To establish that this is the case, however, more empirical evidence is required. Recently, two experiments improved the plausibility of the mirror neurons theory of speech perception. In a first TMS experiment, *Fadiga* and colleagues [77.137] established that MEPs (motor evoked potentials) in the tongue muscles directly correlate with high specificity to the perception of particular sounds (these were *rr* and *ff* in Italian). The listener was delivered TMS (single pulse) and the observed MEPs correlated in amplitude with the different use of the tongue muscles for the pronunciation of either the *rr* or *ff* sounds (*rr* in Italian requires a strong mobilization of the tongue). Albeit convincing, this experiment leaves open the question of specificity, since it can still be the case of a diffuse/generic activation of Broca's area.

A second experiment also by *Fadiga* et al. [77.137] was designed to set the issue. In this case, the TMS was delivered to the primary motor cortex with the aim of establishing a specific motor involvement into the perception of different sounds/phones. Two areas were individuated in the primary motor cortex as responsible to the lip and tongue movement, respectively (e.g., *p/b* sounds versus t/d). The data show a double dissociation pattern, that is, when the lip motor area is stimulated there is a decrease of the reaction times (RTs) of the subject in perceiving the *p/b* (labial sounds) and vice-versa an increase for the perception of the t/d (dental sounds). The opposite happens when the tongue motor area is stimulated. This experiment clearly relates a very specific (small) region of the primary motor cortex with the perception of certain specific (and related) sounds.

Clearly, this is only part of the story; to complicate matters, for example, the semantic content of words related to actions (e.g., kick, pick, lick) activate both motor and pre-motor brain areas somatotopically. Object features, odors, etc., instead have been shown to generate responses in the corresponding cortices. For a review of these and other results, see [77.136].

Theories and models such as the perception for action control theory (PACT) [77.138] take a more moderate interpretation by including both a motor component and *perceptual shaping*, that is, the filtering of certain linguistic combinations because of purely perceptual characteristics (e.g., separation of vowel formants). In PACT, it is hypothesized that the motor system is activated more in *adverse conditions*, while it is perhaps under-threshold for normal speech understanding in good signal-to-noise conditions.

Indeed, we can recognize speech in a foreign accent, and recognizing what is being said can then be decoiupled from being able to articulate how it is being said – but both possibilities are available. This has led to a new view of the integration of mirror systems with other systems [77.139] which downplays the motor theory of speech perception while preserving many other features of the mirror system hypothesis of *Rizzolatti* and *Arbib* [77.133].

Armed with these results *Castellini* and colleagues [77.140] conducted a computational experiment that mimic some of the TMS results of *D'Ausilio* et al. [77.141]. All processing employed a database of synchronized recordings of Italian speakers with acoustic, articulograph, camera, ultrasound, and electroglottograph data [77.142]. For the experiments, only the articulograph and electroglottograph signals were used together with speech sound. These identify the position of the tongue and teeth versus the lips in

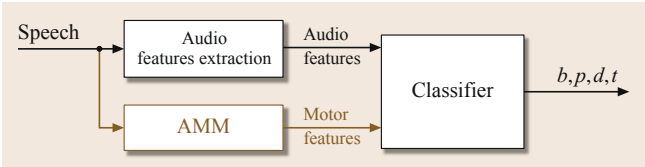


Fig. 77.8 Conceptual schema of the classifiers used in the experiments

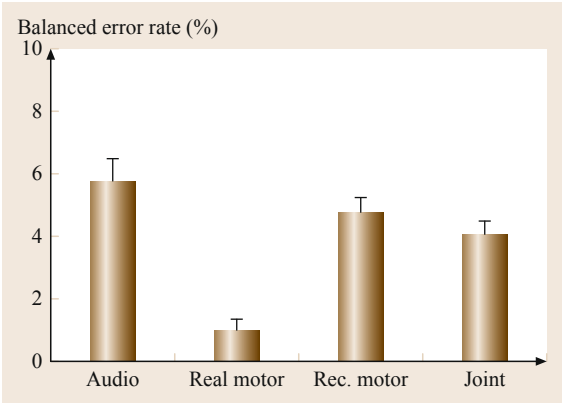


Fig. 77.9 Baseline experiment comparing the performance of acoustic versus motor data (or jointly acoustic plus motor data) in classifying b/p versus d/t

real time (200 Hz) in addition to the activation of the vocal folds (voicing signal). The conceptual schema of all experiments and learning follows some previous work as by *Metta* et al. [77.112], and which as is shown in Fig. 77.8. In particular, acoustic data are mapped into motoric features and these are used for classifying phones. Similarly to the PACT model, it was found useful to incorporate also a purely acoustic classifier. Acoustic features were the standard Mel cepstral coefficients with similar parameters and frequency bands of conventional automatic speech recognition (ASR).

The mapping from acoustic to motor data was performed using either an artificial neural network or support vector machine for regression with indistinguishable results. The classifier was always a support vector machine with Gaussian kernel and parameters optimized through grid search.

In order to compare it with the TMS experiments, phones were divided into two classes, the b/p and t/d, respectively, as representing the bi-labial and dental (movement of the tongue toward the teeth) phones. Fivefold cross-validation was employed on all results by either random splits of the data or by selecting data from various participants (e.g., training on 1–5 participants, testing on 1–5 participants). Gaussian white noise was added to the stimuli (at increasing levels from 0 to 150%) to replicate the conditions of the TMS.

Figure 77.9 shows these results. The baseline experiment shows an improved performance where either the real motoric or jointly motoric and acoustic features are used. The comparison of *audio* versus *joint* features is statistically significant ($p < 0.01$) and verifies the claim as no new information is added to the system when the reconstructed motor features are employed. Motor features are *reconstructed* by the audio-motor map (AMM) of Fig. 77.8 and replicates previous results obtained in the classification of hand gestures [77.119] or handwriting characters [77.143].

A second experiment from the same work of *Castellini* and colleagues [77.140] shows the behavior of the same system in various conditions of increased difficulty ranging from running classification on speakers not included in the training sets to co-articulation. Figure 77.10 shows a number of variants where N vs M indicates N speakers for training versus M speakers for testing given the size of the database (6 speakers). Ex-

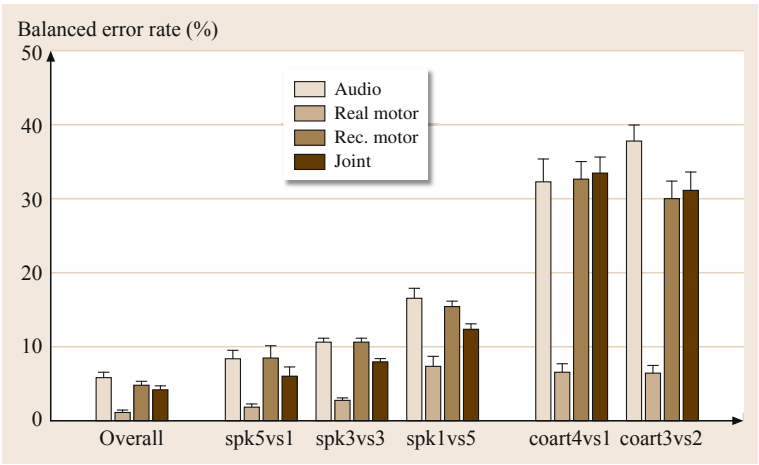


Fig. 77.10 Comparison across various conditions. In all cases apart from coart4vs1 the use of motor features improve classification with statistical significance ($p < 0.01$)

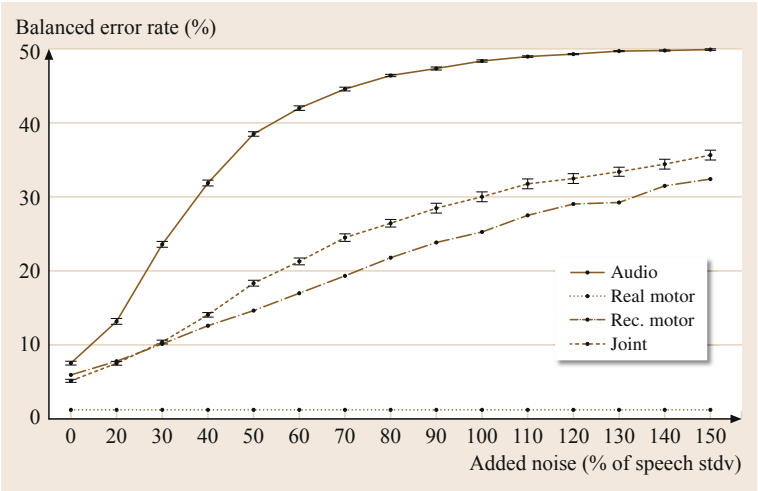


Fig. 77.11 Comparison of acoustic versus motor features under increasing level of added Gaussian white noise for the same classifier of the previous experiments

periments with co-articulation were conducted also on five speakers, albeit the number of identifiable examples in the database was smaller.

In a final experiment, the classifier was tested on acoustic data corrupted by Gaussian white noise. Results show a consistent improvement with the motor information gain increasing with the increase of the noise level (up to 150% of the speech standard deviation): Fig. 77.11.

More recently a full phone classifier was built using similar principles [77.144] together with a combination of deep belief networks (DBNs) and more standard hidden Markov models (HMMs). The results show improvement with respect to the state of the art, continuing the long tradition of neurorobotics and bringing models very close to concrete applications on robots that bear resemblance to the exquisite human performance in speech recognition in noisy environments.

77.5 Conclusion and Further Reading

As the foregoing makes clear, robotics has much to learn from neuroscience and much to teach neuroscience. Neurorobotics can learn from the ways in which the brains and bodies of different creatures adapt to diverse ecological niches – as computational neuroethology helps us understand how the brain of a creature has evolved to serve *action-oriented perception*, and the attendant processes of learning, memory, planning, and social interaction.

We have sampled the *design* of just a few subsystems (both functional and structural) in just a few animals – optic flow in the bee, approach, escape, and barrier avoidance in frogs and toads, and navigation in the rat, as well as the control of eye movements in visual attention, the role of the mammalian cerebellum in handling the nonlinearities and time delays of flexible motor systems, and the mirror systems of primates in action recognition and of humans in imitation. There are many more creatures with lessons to offer the roboticist than we can sample here.

Moreover, if we just confine attention to the brains of humans, this chapter has mentioned at least 7a, 7b, AIP, lateral, medial and ventral intraparietal sulcus (LIP, MIP and VIP), area 46, basal ganglia, caudoputamen, cerebellum, F2, F4, F5, hippocampus, hypothalamus, inferotemporal cortex, motor cortex, nucleus accumbens, parietal cortex, prefrontal cortex, premotor cortex, pre-SMA (F6), spinal cord, STS, and – and it is clear that there are many more details to be understood for each region, and many more regions whose interactions hold lessons for roboticists. We say this not to depress the reader, but rather to encourage further exploration of the literature of computational neuroscience and to note that the exchange with neurorobotics proceeds both ways: neuroscience can inspire novel robotic designs; conversely, robots can be used to test whether brain models still work when they make the transition from disembodied computer simulation to meeting the challenge of guiding the interactions of a physically embodied system with the complexities of its environment.

Nonetheless, a thorough study of the spinal cord and its effect on muscle behavior is where a roboticist, who is interested in replicating some of the functionality of vertebrate movement, may want to start looking.

77.5.1 Further Reading

- *Arbib* (2006) [77.145]: This volume provides 16 articles on the mirror system, written by diverse experts. Of particular relevance to this chapter are articles on dynamical systems: brain, body and imitation; attention and the minimal subscene; the development of grasping and the mirror system; and development of goal-directed imitation, object manipulation, and language in humans and robots.
- *Bell* (1996) [77.146]: This somewhat older BBS special issue provides what was, back then, a rather definitive number of articles on the cerebellum, including an overview of models in a paper by *Houk* et al.
- *van der Smagt* and *Bullock* (2002) [77.147]: This special issue is focused on the application of cerebellar and other models to robotics tasks, and lists some successful and – between the lines – more unsuccessful applications thereof.
- *Gallese* et al. (1996) [77.148]: This paper provides a detailed account of the neurophysiological evidence for mirror neurons. It is good reading to get the real data unbiased from further interpretation on the role of mirror neurons and it is complete and accurate. Although it is a technical paper it is easy to read also for a general audience.
- *Fadiga* et al. 2002 [77.149]: This work extends the mirror system concept with an interesting perspective on its role into language. This paper is interesting reading by providing evidence in humans (the other references above are about monkey experiments). In this case, it has been shown that speech listening facilitates the activation of tongue muscles which match the specific phoneme being listened to.

References

77.1 W. G. Walter: *The Living Brain* (Duckworth, London 1953), reprinted by Pelican Books, Harmondsworth, 1961

77.2 V. Braitenberg: *Vehicles: Experiments in Synthetic Psychology* (Bradford Books/The MIT, Cambridge 1984)

77.3 I. Segev, M. London: Dendritic processing. In: *The Handbook of Brain Theory and Neural Networks*, 2nd edn., ed. by M.A. Arbib (Bradford Books/MIT Press, Cambridge 2003) pp. 324–332

77.4 Y. Fregnac: Hebbian synaptic plasticity. In: *The Handbook of Brain Theory and Neural Networks*, 2nd edn., ed. by M.A. Arbib (Bradford Books/MIT Press, Cambridge 2003) pp. 515–522

77.5 W. Reichardt: Autocorrelation, a principle for the evaluation of sensory information by the central nervous system. In: *Sensory Communication*, ed. by W.A. Rosenblith (MIT Press/Wiley, New York, London 1961) pp. 303–317

77.6 A. Borst, M. Dickinson: Visual course control in flies. In: *The Handbook of Brain Theory and Neural Networks*, 2nd edn., ed. by M.A. Arbib (Bradford Books/MIT Press, Cambridge 2003) pp. 1205–1210

77.7 P. van der Smagt, F. Groen: Visual feedback in motion. In: *Neural Systems for Robotics*, ed. by O. Omidvar, P. van der Smagt (Morgan Kaufmann, San Francisco 1997) pp. 37–73

77.8 P. van der Smagt: Teaching a robot to see how it moves. In: *Neural Network Perspectives on Cognition and Adaptive Robotics*, ed. by A. Browne (Institute of Physics Publishing, Bristol 1997) pp. 195–219

77.9 S.C. Liu, A. Usseglio-Viretta: Fly-like visuomotor responses of a robot using aVLSI motion-sensitive chips, Biol. Cybern. **85**(6), 449–457 (2001)

77.10 F. Ruffier, S. Viollet, S. Amic, N. Franceschini: Bio-inspired optical flow circuits for the visual guidance of micro air vehicles, Int. Symp. Circuits Syst. (ISCAS) 2003, Vol. 3 (2003)

77.11 M.B. Reiser, M.H. Dickinson: A test bed for insect-inspired robotic control, Philos. Trans. Math. Phys. Eng. Sci. **361**(1811), 2267–2285 (2003)

77.12 M. V. Srinivasan, S. W. Zhang, M. Altwein, J. TAUTZ: Honeybee navigation: Nature and calibration of the odometer, Science **287**, 851–853 (2000)

77.13 S. Funke: *Virtual atlas of the honeybee brain* (see Univ. Berlin, Institute of Biology – Neurobiology, Berlin 2015), http://www.neurobiologie.fu-berlin.de/beebrain/Bee/VRML/SnapshotCosmoall.jpg

77.14 A. Barron, M.V. Srinivasan: Visual regulation of ground speed and headwind compensation in freely flying honey bees, *Apis mellifera L*, J. Exp. Bio. **209**(5), 978–984 (2006)

77.15 M.V. Srinivasan, S. Zhang, J.S. Chahl: Landing strategies in honeybees, and possible applications to autonomous airborne vehicles, Biol. Bull. **200**(2), 216–221 (2001)

77.16 T. Vladusich, J.M. Hemmi, M.V. Srinivasan, J. Zeil: Interactions of visual odometry and landmark guidance during food search in honeybees, J. Exp. Biol. **208**, 4123–4135 (2005)

77.17 M.V. Srinivasan, S.W. Zhang: Visual control of honeybee flight. In: *Orientation and Communication in Arthropods*, Experientia Supplementum, Vol. 84, ed. by M. Lehres (Birkhäuser, Basel 1997) pp. 95–113

77.18 B. Webb: Can robots make good models of biological behavior?, Behav. Brain Sci. **24**, 1033–1094 (2001)

77.19 J.Y. Lettvin, H. Maturana, W.S. McCulloch, W.H. Pitts: What the frog's eye tells the frog brain, Proc. IRE **47**, 1940–1951 (1959)

77.20 D. Ingle: Visual releasers of prey catching behavior in frogs and toads, Brain Behav. Evol. **1**, 500–518 (1968)

77.21 R.L. Didday: A model of visuomotor mechanisms in the frog optic tectum, Math. Biosci. **30**, 169–180 (1976)

77.22 M.A. Arbib: Levels of modeling of visually guided behavior, Behav. Brain Sci. **10**, 407–465 (1987)

77.23 M.A. Arbib: Visuomotor coordination: Neural models and perceptual robotics. In: *Visuomotor Coordination: Amphibians, Comparisons, Models, and Robots*, 2nd edn., ed. by J.P. Ewert, M.A. Arbib (Plenum, New York 1989) pp. 121–171

77.24 T. Collett: Do toads plan routes? A study of detour behavior of *B. viridis*, J. Comp. Physiol. **146**, 261–271 (1982)

77.25 M.A. Arbib, D.H. House: Depth and detours: An essay on visually guided behavior. In: *Vision, Brain and Cooperative Computation*, ed. by M.A. Arbib, A.R. Hanson (Bradford Books/MIT Press, Cambridge 1987) pp. 129–163

77.26 F.J. Corbacho, M.A. Arbib: Learning to detour, Adapt. Behav. **4**, 419–468 (1995)

77.27 A. Cobas, M.A. Arbib: Prey-catching and predator-avoidance in frog and toad: Defining the schemas, J. Theor. Biol. **157**, 271–304 (1992)

77.28 R.C. Arkin: Motor schema-based mobile robot navigation, Int. J. Robot. Res. **8**, 92–112 (1989)

77.29 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robot. Res. **5**, 90–98 (1986)

77.30 B.H. Krogh, C.E. Thorpe: Integrated path planning and dynamic steering control for autonomous vehicles, Proc. IEEE Int. Conf. Robot. Autom. San Francisco (1986) pp. 1664–1669

77.31 R.A. Brooks, C.L. Breazeal, M. Marjanović, B. Scassellati: The COG project: Building a humanoid robot, Lect. Notes Comput. Sci. **1562**, 52–87 (1999)

77.32 R.C. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1998)

77.33 R.C. Arkin, M. Fujita, T. Takagi, R. Hasegawa: An ethological and emotional basis for human-robot interaction, Robot. Auton. Syst. **42**(3/4), 191–201 (2003)

77.34 P. Dean, P. Redgrave, G.W.M. Westby: Event or emergency? Two response systems in the mammalian superior colliculus, Trends Neurosci. **12**, 138–147 (1989)

77.35 B.E. Stein, M.A. Meredith: *The Merging of the Senses* (MIT Press, Cambridge 1993)

77.36 T. Strosslin, C. Krebser, A. Arleo, W. Gerstner: Combining multimodal sensory input for spatial learning, artificial neural networks – ICANN 2002, Lect. Notes Comput. Sci. **2415**, 87–92 (2002)

77.37 J. O'Keefe, L. Nadel: *The Hippocampus as a Cognitive Map* (Clarendon, Oxford 1978)

77.38 V. Braitenberg: Taxis, kinesis, decussation, Progr. Brain Res. **17**, 210–222 (1965)

77.39 A. Guazzelli, F.J. Corbacho, M. Bota, M.A. Arbib: Affordances, motivation, and the world graph theory, Adapt. Behav. **6**, 435–471 (1998)

77.40 J.J. Gibson: *The Senses Considered as Perceptual Systems* (Allen and Unwin, London 1966)

77.41 I. Lieblich, M.A. Arbib: Multiple representations of space underlying behavior, Behav. Brain Sci. **5**, 627–659 (1982)

77.42 B. Girard, D. Filliat, J.A. Meyer, A. Berthoz, A. Guillot: Integration of navigation and action selection functionalities in a computational model of cortico-basal-ganglia-thalamo-cortical loops, Adapt. Behav. **13**(2), 115–130 (2005)

77.43 J.A. Meyer, A. Guillot, B. Girard, M. Khamassi, P. Pirim, A. Berthoz: The Psikharpax project: Towards building an artificial rat, Robot. Auton. Syst. **50**(4), 211–223 (2005)

77.44 G. Deco, E.T. Rolls: Attention and working memory: A dynamical model of neuronal activity in the prefrontal cortex, Eur. J. Neurosci. **18**(8), 2374–2390 (2003)

77.45 S.B. Choi, S.W. Ban, M. Lee: Biologically motivated visual attention system using bottom-up saliency map and top-down inhibition, Neural Inf. Proc. – Lett. Rev. **2**(1), 19–25 (2004)

77.46 L. Itti, C. Koch: A saliency-based search mechanism for overt and covert shifts of visual attention, Vis. Res. **40**, 1489–1506 (2000)

77.47 J.M. Wolfe: Guided search 2.0: A revised model of visual search, Psychon. Bull. Rev. **1**, 202–238 (1994)

77.48 A. Yarbus: *Eye Movements and Vision* (Plenum, New York 1967)

77.49 V. Navalpakkam, L. Itti: Modeling the influence of task on attention, Vis. Res. **45**, 205–231 (2005)

77.50 F. Orabona, G. Metta, G. Sandini: Object-based visual attention: A model for a behaving robot, Conf. Comput. Vis. Pattern Recogn. (2005) pp. 89–89

77.51 G. Sandini, V. Tagliasco: An anthropomorphic retina-like structure for scene analysis, Comput. Vis. Gr. Image Proc. **14**(3), 365–372 (1980)

77.52 G. Rizzolatti, L. Riggio, I. Dascola, C. Umiltá: Reorienting attention across the horizontal and vertical meridians: Evidence in favor of a premotor theory of attention, Neuropsychologia **25**, 31–40 (1987)

77.53 J.R. Flanagan, R.S. Johansson: Action plans used in action observation, Nature **424**, 769–771 (2003)

77.54 J.R. Flanagan, P. Vetter, R.S. Johansson, D.M. Wolpert: Prediction precedes control in motor learning, Curr. Biol. **13**, 146–150 (2003)

77.55 J.M. Mataric, M. Pomplun: Fixation behavior in observation and imitation of human movement, Brain Res. Cogn. Brain Res. **7**, 191–202 (1998)

77.56 E. Kandel, J. Schwartz, T. Jessell: *Principles of Neural Science*, 5th edn. (McGraw-Hill, Columbus 2000)

77.57 J. Fallon, R. Carr, D. Morgan: Stochastic resonance in muscle receptors, J. Neurophysiol. **91**, 2429–2436 (2004)

77.58 B. Edin: Quantitative analyses of dynamic strain sensitivity in human skin mechanoreceptors, J. Neurophysiol. **92**(6), 3233–3243 (2004)

77.59 K.O. Johnson: Closing in on the neural mechanisms of finger joint angle sense. Focus on quantitative analysis of dynamic strain sensitivity in human skin mechanoreceptors, J. Neurophysiol. **92**(6), 3167–3168 (2004)

77.60 B. Edin: Cutaneous afferents provide information about knee joint movements in humans, J. Physiol. **531**(1), 289–297 (2001)

77.61 B. Edin, J. Westlin: Independent control of human finger-tip forces at individual digits using precision lifting, J. Physiol. **450**, 547–567 (1992)

77.62 K. Doya: Complementary roles of basal ganglia and cerebellum in learning and motor control, Curr. Opin. Neurobiol. **10**(6), 732–739 (2000)

77.63 L.M. Bjursten, K. Norrsell, U. Norrsell: Behavioural repertory of cats without cerebral cortex from infancy, Exp. Brain Res. **25**(2), 115–130 (1976)

77.64 G. Holmes: The cerebellum of man, Brain **62**, 1–30 (1939)

77.65 R.C. Miall, D.J. Weir, D.M. Wolpert, J.F. Stein: Is the cerebellum a Smith predictor?, J. Motor Behav. **25**, 203–216 (1993)

77.66 J. Peters, P. van der Smagt: Searching a scalable approach to cerebellar based control, Appl. Intell. **17**, 11–33 (2002)

77.67 E.J. Nijhof, E. Kouwenhoven: Simulation of multijoint arm movements. In: *Biomechanics and Neural Control of Posture and Movement*, ed. by J.M. Winters, P.E. Crago (Springer, New York 2002) pp. 363–372

77.68 N. Bernstein: *The Coordination and Regulation of Movements* (Pergamon, Oxford 1967)

77.69 M.C. Tresch, V.C.K. Cheung, A. d'Avella: Matrix factorization algorithms for the identification of muscle synergies: Evaluation on simulated and experimental data sets, J. Neurophysiol. **95**(4), 2199–2212 (2006)

77.70 C. Castellini, P. van der Smagt: Evidence of muscle synergies during human grasping, Biol. Cybern. **107**(2), 233–245 (2013)

77.71 J.C. Eccles, M. Ito, J. Szentágothai: *The Cerebellum as a Neuronal Machine* (Springer, New York 1967)

77.72 M. Ito: *The Cerebellum and Neural Control* (Raven, New York 1984)

77.73 D.A. Marr: A theory of cerebellar cortex, J. Physiol. **202**, 437–470 (1969)

77.74 J.S. Albus: A theory of cerebellar function, Math. Biosci. **10**, 25–61 (1971)

77.75 P. van der Smagt: Cerebellar control of robot arms, Connect. Sci. **10**, 301–320 (1998)

77.76 P. van der Smagt: Benchmarking cerebellar control, Robot. Auton. Syst. **32**, 237–251 (2000)

77.77 J.S. Albus: Data storage in the cerebellar model articulation controller (CMAC), J. Dyn. Syst. Meas. Control ASME **3**, 228–233 (1975)

77.78 W.T. Miller: Real-time application of neural networks for sensor-based control of robots with vision, IEEE Trans. Syst. Man Cybern. **19**, 825–831 (1994)

77.79 C. Sabourin, O. Bruneau: Robustness of the dynamic walk of a biped robot subjected to disturbing external forces by using CMAC neural networks, Robot. Auton. Syst. **51**, 81–99 (2005)

77.80 J.C. Houk, J.T. Buckingham, A.G. Barto: Models of the cerebellum and motor learning, Behav. Brain Sci. **19**(3), 368–383 (1996)

77.81 M.A. Arbib, C.C. Boylls, P. Dev: Neural models of spatial perception and the control of movement. In: *Cybernetics and Bionics*, ed. by W.D. Keidel, W. Handles, M. Spreng (Oldenbourg, Munich 1974) pp. 216–231

77.82 N. Schweighofer: Computational Models of the Cerebellum in the Adaptive Control of Movements, Ph.D. Thesis (University of Southern California, Los Angeles 1995)

77.83 N. Schweighofer, M.A. Arbib, M. Kawato: Role of the cerebellum in reaching quickly and accurately: I. A functional anatomical model of dynamics control, Eur. J. Neurosci. **10**, 86–94 (1998)

77.84 N. Schweighofer, J. Spoelstra, M.A. Arbib, M. Kawato: Role of the cerebellum in reaching quickly and accurately: II. A neural model of the intermediate cerebellum, Eur. J. Neurosci. **10**, 95–105 (1998)

77.85 M.A. Arbib, N. Schweighofer, W.T. Thach: Modeling the cerebellum: From adaptation to coordination. In: *Motor Control and Sensory-Motor Integration: Issues and Directions*, ed. by D.J. Glencross, J.P. Piek (North-Holland Elsevier Science, Amsterdam 1995) pp. 11–36

77.86 D. Wolpert, M. Kawato: Multiple paired forward and inverse models for motor control, Neural Netw. **11**, 1317–1329 (1998)

77.87 O. Donchin, J.T. Francis, R. Shadmehr: Quantifying generalization from trial-by-trial behavior of adaptive systems that learn with basis functions: Theory and experiments in human motor control, J. Neurosci. **23**(27), 9032–9045 (2003)

77.88 R. Osu, H. Gomi: Multijoint muscle regulation mechanisms examined by measured human arm stiffness and EMG signals, J. Neurophysiol. **81**(4), 1458–1468 (1999)

77.89 H. Höppner, J. McIntyre, S.P. der van: Task dependency of grip stiffness – A study of human grip force and grip stiffness dependency during two different tasks with same grip forces, PLOS ONE **8**(12), e80889 (2013)

77.90 P. van der Smagt, F. Groen, K. Schulten: Analysis and control of a rubbertuator robot arm, Biol. Cybern. **75**(4), 433–440 (1996)

77.91 G. Rizzolatti, G. Luppino, M. Matelli: The organization of the cortical motor system: New concepts, Electroencephalogr. Clin. Neurophysiol. **106**, 283–296 (1998)

77.92 G. Rizzolatti, L. Fogassi, V. Gallese: Neurophysiological mechanisms underlying the understanding and imitation of action, Nat. Rev. Neurosci. **2**, 661–670 (2001)

77.93 M.A. Umiltà, E. Kohler, V. Gallese, L. Fogassi, L. Fadiga, C. Keysers, G. Rizzolatti: I know what

you are doing: A neurophysiological study, Neuron **31**(1), 155–165 (2001)
77.94 P.F. Ferrari, S. Rozzi, L. Fogassi: Mirror neurons responding to observation of actions made with tools in monkey ventral premotor cortex, J. Cogn. Neurosci. **17**, 212–226 (2005)
77.95 L. Fogassi, P.F. Ferrari, B. Gesierich, S. Rozzi, F. Chersi, G. Rizzolatti: Parietal lobe: From action organization to intention understanding, Science **308**(4), 662–667 (2005)
77.96 M. Taira, S. Mine, A.P. Georgopoulos, A. Murata, H. Sakata: Parietal cortex neurons of the monkey related to the visual guidance of hand movement, Exp. Brain Res. **83**, 29–36 (1990)
77.97 A.H. Fagg, M.A. Arbib: Modeling parietal–premotor interactions in primate control of grasping, Neural Netw. **11**, 1277–1303 (1998)
77.98 G. Rizzolatti, G. Luppino: Grasping movements: Visuomotor transformations. In: *The Handbook of Brain Theory and Neural Networks*, 2nd edn., ed. by M.A. Arbib (Bradford Books/MIT Press, Cambridge 2003) pp. 501–504
77.99 S.H. Johnson-Frey: The neural bases of complex tool use in humans, Trends Cogn. Sci. **8**, 71–78 (2004)
77.100 G. Rizzolatti, L. Fadiga, V. Gallese, L. Fogassi: Premotor cortex and the recognition of motor actions, Cogn. Brain Res. **3**, 131–141 (1995)
77.101 E. Oztop, M.A. Arbib: Schema design and implementation of the grasp-related mirror neuron system, Biol. Cybern. **87**(2), 116–140 (2002)
77.102 D.I. Perrett, J.K. Hietanen, M.W. Oram, P.J. Benson: Organization and functions of cells in the macaque temporal cortex, Philos. Trans. R. Soc. B **335**, 23–50 (1992)
77.103 D.P. Carey, D.I. Perrett, M.W. Oram: Recognizing, understanding and producing action. In: *Handbook of Neuropsychology: Action and Cognition*, Vol. 11, ed. by M. Jeannerod, J. Grafman (Elsevier, Amsterdam 1997) pp. 111–130
77.104 B. Bonaiuto, E. Rosta, M.A. Arbib: Extending the mirror neuron system model, I. Audible actions and invisible grasps, Biol. Cybern. **96**(1), 9–38 (2006)
77.105 Y. Demiris, M.H. Johnson: Distributed, predictive perception of actions: A biologically inspired robotics architecture for imitation and learning, Connect. Sci. **15**(4), 231–243 (2003)
77.106 M.A. Arbib, A. Billard, M. Iacoboni, E. Oztop: Synthetic brain imaging: Grasping, mirror neurons and imitation, Neural Netw. **13**(8/9), 975–997 (2000)
77.107 S. Schaal, A.J. Ijspeert, A. Billard: Computational approaches to motor learning by imitation, Philos. Trans. R. Soc. Biol. Sci. **358**(1431), 537–547 (2003)
77.108 E. Oztop, M. Kawato, M.A. Arbib: Mirror neurons and imitation: A computationally guided review, Neural Netw. **19**(3), 254–271 (2006)
77.109 J. Tani, M. Ito, Y. Sugita: Self-organization of distributedly represented multiple behavior schemata in a mirror system: Reviews of robot experiments using RNNPB, Neural Netw. **17**(8/9), 1273–1289 (2004)
77.110 Y. Kuniyoshi, Y. Yorozu, M. Inaba, H. Inoue: From visuomotor self learning to visual imitation – a neural architecture for humanoid learning, IEEE Int. Conf. Robot. Autom. (ICRA), Vol. 3 (2003) pp. 3132–3139
77.111 D.M. Wolpert, K. Doya, M. Kawato: A unifying computational framework for motor control and social interaction, Philos. Trans. R. Soc. Biol. Sci. **358**(1431), 593–602 (2003)
77.112 G. Metta, G. Sandini, L. Natale, L. Craighero, L. Fadiga: Understanding mirror neurons: A bio-robotic approach, Interact. Stud. **7**(2), 197–232 (2006) special issue on epigenetic robotics
77.113 D.M. Wolpert, Z. Ghahramani, R.J. Flanagan: Perspectives and problems in motor learning, Cogn. Sci. **5**(11), 487–494 (2001)
77.114 M.I. Jordan, D.E. Rumelhart: Forward models: Supervised learning with a distal teacher, Cogn. Sci. **16**(3), 307–354 (2006)
77.115 M. Kawato, K. Furukawa, R. Suzuki: A hierarchical neural network model for control and learning of voluntary movement, Biol. Cybern. **57**, 169–185 (1987)
77.116 M. Haruno, D.M. Wolpert, M. Kawato: MOSAIC model for sensorimotor learning and control, Neural Comput. **13**, 2201–2220 (2001)
77.117 E. Oztop, D.M. Wolpert, M. Kawato: Mental state inference using visual control parameters, Cogn. Brain Res. **22**, 129–151 (2005)
77.118 A.L. Woodward: Infant selectively encode the goal object of an actor's reach, Cognition **69**, 1–34 (1998)
77.119 M. Lopas, J. Santos–Victor: Visual learning by imitation with motor representations, IEEE Trans. Syst. Man Cybern. B **35**(3), 438–449 (2005)
77.120 A. Gijsberts, T. Tommasi, G. Metta, B. Caputo: Object recognition using visuo-affordance maps, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS2010) Taipei, Taiwan (2010) pp. 1572–1578
77.121 N. Noceti, B. Caputo, C. Castellini, L. Baldassarre, A. Barla, L. Rosasco, F. Odone, G. Sandini: Towards a theoretical framework for learning multi-modal patterns for embodied agents, ICIAP-09, 15th, Int. Conf. Image Anal. Process. (2009)
77.122 P. Fitzpatrick, G. Metta: Grounding vision through experimental manipulation, Philos. Trans. R. Soc. Math. Phys. Eng, Sci. **361**(1811), 2165–2185 (2003)
77.123 S.M. Khansari Zadeh, A. Billard: Learning stable non-linear dynamical systems with Gaussian mixture models, IEEE Trans. Robotics **27**(5), 943–957 (2011)
77.124 A. Ude, A. Gams, T. Asfour, J. Morimoto: Task-specific generalization of discrete and periodic dynamic movement primitives, IEEE Trans. Robotics **26**(5), 800–815 (2010)
77.125 E.A. Theodorou, J. Buchli, S. Schaal: A generalized path integral control approach to reinforcement learning, J. Mach. Learn. Res. **11**, 3137–3181 (2010)
77.126 C.L. Nehaniv: Nine billion correspondence problems. In: *Imitation and Social Learning in Robots*,

Humans, and Animals: Behavioural, Social and Communicative Dimensions, ed. by C.L. Nehaniv, K. Dautenhahn (Cambridge Univ. Press, Cambridge 2006)

77.127 C.L. Nehaniv, K. Dautenhahn: Mapping between dissimilar bodies: Affordances and the algebraic foundations of imitation, Eur. Workshop Learn. Robots (EWRL-7) Edinburgh (1998)

77.128 Y. Demiris, G. Hayes: Imitation as a dual-route process featuring predictive and learning components: A biologically-plausible computational model. In: *Imitation in Animals and Artifacts*, ed. by K. Dautenhahn, C. Nehaniv (MIT Press, Cambridge 2002)

77.129 M. Ito, K. Noda, Y. Hoshino, J. Tani: Dynamic and interactive generation of object handling behaviors by a small humanoid robot using a dynamic neural network model, Neural Netw **19**(3), 323–337 (2006)

77.130 E.L. Sauser, A. Billard: Parallel and distributed neural models of the ideomotor principle: An investigation of imitative cortical pathways, Neural Netw. **19**, 285–298 (2006)

77.131 A.M. Liberman, F.S. Cooper, D.P. Shankweiler, M. Studdert-Kennedy: Perception of the speech code, Psychol. Rev. **74**, 431–461 (1967)

77.132 A.M. Liberman, I.G. Mattingly: The motor theory of speech perception revised, Cognition **21**, 1–36 (1985)

77.133 G. Rizzolatti, M.A. Arbib: Language within our grasp, Trends Neurosci. **21**(5), 188–194 (1998)

77.134 L. Fadiga, L. Fogassi, G. Pavesi, G. Rizzolatti: Motor facilitation during action observation: A magnetic stimulation study, J. Neurophys. **73**(6), 2608–2611 (1995)

77.135 M. Gentilucci, L. Fogassi, G. Luppino, M. Matelli, R. Camarda, G. Rizzolatti: Functional organization of inferior area 6 in the macaque monkey. I. Somatotopy and the control of proximal movements, Exp. Brain Res. **71**(3), 475–490 (1988)

77.136 F. Pulvermüller, L. Fadiga: Active perception: sensorimotor circuits as a cortical basis for language, Nat. Rev. Neurosci. **11**(5), 351–360 (2010)

77.137 L. Fadiga, L. Craighero, G. Buccino, G. Rizzolatti: Speech listening specifically modulates the excitability of tongue muscles: A TMS study. Eur. J, Neurosci. **15**(2), 399–402 (2002)

77.138 J.-L. Schwartz, A. Basirat, L. Ménard, M. Sato: The Perception-for-Action-Control Theory (PACT): A perceptuo-motor theory of speech perception, J. Neurolinguist **25**(5), 336–354 (2012)

77.139 C. Moulin-Frier, M.A. Arbib: Recognizing speech in a novel accent: The motor theory of speech perception reframed, Biol Cybern. **107**(4), 421–447 (2013)

77.140 C. Castellini, L. Badino, G. Metta, G. Sandini, M. Tavella, M. Grimaldi, L. Fadiga: The use of phonetic motor invariants can improve automatic phoneme discrimination, PLoS ONE **6**(9), e24055 (2011)

77.141 A. D'Ausilio, F. Pulvermüller, P. Salmas, I. Bufalari, C. Begliomini, L. Fadiga: The motor somatotopy of speech perception, Curr. Biol. **19**(5), 381–385 (2009)

77.142 M. Grimaldi, F.B. Gili, F. Sigona, M. Tavella, P. Fitzpatrick, L. Craighero, L. Fadiga, G. Sandini, G. Metta: New technologies for simultaneous acquisition of speech articulatory data: Ultrasound, 3-D articulograph and electroglottograph, Poster, LangTech Conference, Roma (2008)

77.143 G.E. Hinton, V. Nair: Inferring motor programs from images of handwritten digits, Adv. Neural Inform. Proces. Syst., Vol. 18 (2004) pp. 515–522

77.144 L. Badino, C. Canevari, L. Fadiga, G. Metta: Deep-level acoustic-to-articulatory mapping for DBN-HMM based phone recognition, Spoken Language Technology Workshop (SLT) (2012) pp. 370–375

77.145 M.A. Arbib (Ed.): *From Action to Language Via the Mirror System* (Cambridge Univ. Press, Cambridge 2006)

77.146 C. Bell, P. Cordo, S. Harnad: Controversies in neuroscience IV: Motor learning and plasticity in the cerebellum, Behav. Brain Sci. **19**(3), v–vi (1996)

77.147 P. van der Smagt, D. Bullock: Applied intelligence, Scalable Appl. Neural Netw. Robotics **17**(1), 7–10 (2002)

77.148 V. Gallese, L. Fadiga, L. Fogassi, G. Rizzolatti: Action recognition in the premotor cortex, Brain **119**, 593–609 (1996)

77.149 L. Fadiga, L. Craighero, G. Buccino, G. Rizzolatti: Speech listening specifically modulates the excitability of tongue muscles: A TMS study, Eur. J. Neurosci. **15**(2), 399–402 (2002)

Multimedia Contents

78. Perceptual Robotics

Heinrich Bülthoff, Christian Wallraven, Martin A. Giese

Robots that share their environment with humans need to be able to recognize and manipulate objects and users, perform complex navigation tasks, and interpret and react to human emotional and communicative gestures. In all of these perceptual capabilities, the human brain, however, is still far ahead of robotic systems. Hence, taking clues from the way the human brain solves such complex perceptual tasks will help to design better robots. Similarly, once a robot interacts with humans, its behaviors and reactions will be judged by humans – movements of the robot, for example, should be fluid and graceful, and it should not evoke an *eerie* feeling when interacting with a user. In this chapter, we present Perceptual Robotics as the field of robotics that takes inspiration from perception research and neuroscience to, first, build better perceptual capabilities into robotic systems and, second, to validate the perceptual impact of robotic systems on the user.

78.1 **Perceptual Mechanisms of Object Representations** 2097
78.1.1 Perceptual and Computational Basis of Object Representations .. 2097
78.1.2 Neural Representations in Object Recognition 2100
78.1.3 Object Recognition: Lessons from Computer Vision 2101
78.1.4 Object Learning and Recognition for Perceptual Robotics 2102
78.2 **Perceptual Mechanisms of Action Representation**..... 2103
78.2.1 Recognition of Complex Movements and Actions in Primate Cortex 2103
78.2.2 Biological Principles with Relevance for Computer Vision and Robotics..... 2104
78.3 **Perceptual Validation of Robotics** 2107
78.3.1 Realistic Faces for Robots 2107
78.3.2 Perceptual and Neural Processing of Body Movements of Robots 2108
78.4 **Conclusion and Further Reading**..... 2108
Video-References..... 2109
References..... 2109

The technical realization of perceptual functions is a central problem for many applications in robotics. Robots require perception to navigate in space and to localize and recognize goal objects, e.g., for manipulation (Chaps. 7, 8, 32, 33, 36–38, 47, 67). Social interactive robots must be able to interpret gestures, actions, and even emotions (Chap. 69, 71, 72) in order to interact naturally with their users. One important approach for the programming of complex perceptual and behavioral functions, for example, needed for humanoid robots is imitation learning (Chaps. 75, 77). Imitation learning requires the robot to *perceive* complex actions that are executed by the user and to subsequently map them into an efficient representation that is suitable for the synthesis of the corresponding motor behavior on the available platform. This chapter focuses on important principles of the representation of complex shapes and movements, which can be derived from biological perception systems, and more specifically the basic functionality of the primate visual cortex. Such principles have interesting implications for the design of technical systems in robotics and computer vision for the recognition of objects, shapes and faces, and for the recognition and synthesis of complex movements and

actions. The limited space of the chapter forced us to focus mainly on visual perception and related technical applications. In the context of robotics many other aspects of perception are important, for example haptic perception (Chap. 41), auditory perception, sensory cue fusion (Chap. 35), and the interaction between the visual recognition of objects and actions and motor programs, e.g., during grasping (treated in Chap. 38).

In the following, we will first formulate several biological principles that are relevant for form and motion representations, specifically in the visual system. We will then, on the one hand, describe technical systems that implement these principles using neural mechanisms that are inspired by the basic architecture of the brain. On the other hand, we will discuss also implementations that are inspired by biological principles on a more abstract level, and which exploit instead of neural networks more efficient technical algorithms for the realization of biologically relevant functions. Many of these systems are derived in the field of computer vision and are based on the advantages and limitations of modern digital computers in order to more efficiently realize biological principles of information processing.

Our approach to establish relationships between biological perception and robotics systems at different levels reflects *David Marr*'s classical distinction of multiple levels of description, originally developed for the analysis of vision systems [78.1]: Robotics systems can be inspired by biological system at the level of implementation, i. e., one can try to build robots containing neural mechanisms that imitate the function of neurons in central nervous systems of biological organisms. This type of analogy between technical and biological systems coincides with the definition of *Neurorobotics* given in Chap. 77. A transfer of principles from biological perception systems to robots might also be accomplished at the more abstract levels of computational problems and algorithms. The computational level is defined by the abstract theoretical formulation of computational problems that have to be solved by perception systems. Examples are the identification or classification of goal object, or the recognition of human gestures. Marr's level of algorithms specifies the computational methods for the solution of such problems, independent of the underlying specific hardware or architecture. For example, an object might be represented by modeling its full 3-D structure, e.g., using a parametric 3-D shape model, or it might be represented in terms of two-dimensional example views. Example views, however, might be represented using neural networks, establishing an analogy with the human brain at the level of implementation, or using more efficient computational methods, e.g., as support vectors of a classifier that has been trained with appropriate images of the object and distractor patterns. In both the cases, the robot system realizes mechanisms that are derived form perception in biological systems.

Marr's distinction of levels is only one way to introduce description levels for complex systems. Other approaches, particularly relevant for robotics, are, for example the subsumption architecture and behavior-based approaches (Chap. 13) that decompose robotics system into a system of simpler behavioral modules. Another examples are dynamical systems approaches to robotics [78.2–4] that are based on the biologically motivated idea that behaviors can be mapped onto stable states of (nonlinear) dynamical systems or recurrent neural networks. Individual behaviors result by self-organization over the whole system as collectively stable modes, which can be described and analyzed by the introduction of appropriate collective variables. Interestingly, such robotics-inspired approaches have been quite successful in modeling human navigation behavior [78.5].

In the following, we will apply the term Perceptual Robotics to signify the design of robots based on principles that are derived from human perception on all three levels in the sense of Marr. This includes a realization in terms of specific neural circuits as well as the transfer of more abstract biologically inspired strategies for the solution of relevant computational problems. A direct interaction between robotics and perception research can be very fruitful for both disciplines. On the one hand, our current knowledge about the human perception and the underlying computational principles might help us to build more efficient robotics architectures that inherit properties from biological perception, e.g., very efficient and robust processing or complex dynamic flexibility. Such architectures will be a necessary pre-requisite for the creation of truly intelligent, cognitive robots (Chaps. 13, 71, 74, 75). On the other hand, perception science often uses robots as testbed for gaining a deeper understanding of computational processes, in particular, for testing the computational power of specific computational solutions under *real-world conditions*. How can a child, for example, learn how to handle new objects, and what allows us to learn the visual categorization of thousands of objects from just a few examples? *Perceptual robot platforms*, equipped with a variety of sensory inputs and operating in different types of artificially structured or real-world environments provide very helpful tools for the study of such questions.

Finally, perceptual robotics not only means to take inspiration from perception to build more efficient robots, but it also encapsulates the perceptual validation of robotic systems. As robots move into the human environment and are increasingly also interacting with

humans, it becomes important to evaluate and validate their effectiveness and efficacy with respect to human standards. Here, we do not refer to their social acceptance, but rather to the way that robots are judged by humans in terms of their appearance, movements, and interactive capabilities. If a robot displays jerky movements, for example, it may still successfully grasp and manipulate an object, but it would be immediately noticeable to a human observer and potentially disturbing to interact with. This *eeriness* or *weirdness* was already anticipated in the early 1970s in a famous paper about the *uncanny valley* by a Japanese roboticist [78.6]. Mori anticipated that as robots become more human-like, humans' familiarity with the robot would increase until at some point (when the robot looks or acts almost human-like), they would suddenly feel highly unfamiliar toward the robot. As the human likeness increases further, the robot would again be judged as familiar or appealing.

More specifically, Mori also postulated that this uncanny valley would not only hold for the robot's static appearance, but would in fact be increased for a moving or acting robot. With the increase in interest in developing humanoid robots over the past decades, being aware of the perceptual judgments of such humanoids becomes a critical component in their development. Since the evaluation of appearance and movements of a humanoid are driven by perceptual processes, it makes sense to also use protocols from perception research to evaluate and fine-tune their effectiveness. In such experiments, typically the robot's performance is evaluated with respect to measures such as general user acceptance, recognizability of expressions, smoothness of motions, ease of interaction, duration and quality of interaction, etc. It is important that the experiment should not only be about simply asking *how good is the robot*, but it should actually tests the robot in the intended task context or that whether it uses additional, indirect measures of effectiveness. As a tutorial on designing and analyzing perceptual experiments and user studies is beyond the scope of this chapter, we refer the reader to introductory texts such as [78.7, 8]. In this chapter, we will focus on two important topics related to humanoid perception in the context of perceptual robotics: facial animation and the perceptual processing of body movements.

78.1 Perceptual Mechanisms of Object Representations

Object recognition is a fundamental visual function that is critical for many applications in robotics. Manipulation and grasping (Chaps. 36–38) require exact knowledge about the shape of the goal object that is often derived from visual sensors. Also the imitation of goal-directed movements (Chap. 77) requires knowledge about target objects. Finally, social and collective robots require robust recognition of other agents and objects which are taking part in the present action (Chaps. 71, 72). The importance of object and shape recognition for many other applied robot systems, like construction and assembly robots or smart cars (Chap. 54) is immediately evident.

78.1.1 Perceptual and Computational Basis of Object Representations

The question of how humans learn, represent, and recognize objects under a wide variety of viewing conditions presents a great challenge to both neurophysiology and cognitive research. Frameworks for explaining the amazing robustness of human recognition processes and how humans represent objects can be broadly classified into two approaches: in the model-based representation, an image on the retina is analyzed to yield three-dimensional parts of an object based on geometric primitives (cf. also Chap. 32). These primitives are then matched to an internal, three-dimensional model of the object (Fig. 78.1, bottom). Exemplar-based representation approaches assume that the internal storage consists of, typically two-dimensional, snapshot-like representations of objects, which are directly compared to the visual input via simple image transformations. In the following, we will briefly describe the basic properties of these two approaches as well as perceptual evidence for their plausibility in explaining human recognition performance.

Structural Description Models

The basic idea of structural description models is that object recognition or categorization is based on a structural representation, which is defined as a configuration of elementary object parts that are regarded as shape primitives [78.9]. Structural description models aim at supplying abstract and propositional descriptions of objects, while at the same time disregarding irrelevant spatial information. Therefore, structural description models typically predict that recognition performance is invariant regarding spatial transformations. *Biederman*'s recognition-by-components (RBCs) or geon structural description (GSD) model can be regarded as the best developed example of the struc-

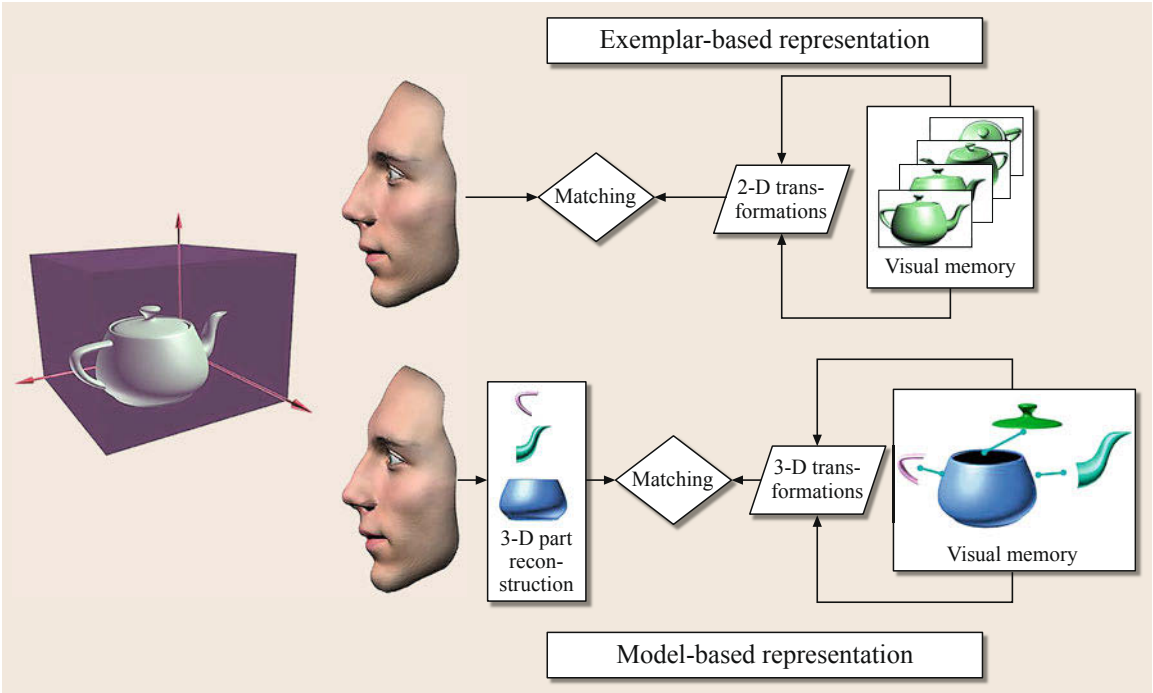


Fig. 78.1 Schematic drawing comparing exemplar-based with model-based representations. Object perception based on model-based representations assumes that the brain extracts 3-D parts from the visual image, which are then matched to an internally stored 3-D model of the teapot. Contrasting with this approach, object perception based on exemplar-based representations is accomplished by directly comparing stored templates or example images with the current picture of the teapot

tural description model type [78.10]. According to this model, objects are represented as configurations of elementary three-dimensional primitive parts, called geons. These geons are derived from nonaccidental properties (NAPs) in the image, i. e., from properties which unlikely arise by chance, and are more or less invariant over a wide range of views. For example, the properties straight vs. curved, symmetrical vs. asymmetrical, parallel vs. nonparallel are regarded as NAPs (NAPs were originally proposed within an image-based approach by *Lowe* [78.11]). According to the model, geons and their spatial configuration are combined into a structural representation, called GSD. The spatial relations between parts are described in a categorical way, using relations like above, below, etc. Like other structural description models, Biederman's model predicts invariance in relation to position and size and also in relation to orientation in depth, as long as no parts are occluded.

The question has to be raised whether objects can be decomposed into geons at all. It was argued that Biederman's RBC cannot be applied to a whole range of biological stimuli [78.12], or that biological shapes in general cannot be adequately described by structural description models [78.13]. This problem extends also to artifact categories like shoe, hat or backpack, which seem to exceed the scope of the geon model. Therefore it has to be doubted that object parts are necessarily represented as geons, or as similar geometrical primitives (further problems of RBC in [78.14, 15]). However, this does not mean that category representations do not have a part structure: in fact, it is not the notion of the part structure in object representations by itself which is problematic, but the use of parts and relations as a basis to derive invariant recognition performance [78.15].

Exemplar-Based Models

Over the last two decades, an increasing number of studies has demonstrated that recognition is not view-independent. Orientation-dependent recognition effects were found for novel objects [78.16, 17], and also for common, familiar objects [78.18, 19]. Orientation-dependent recognition performance has been shown not to be limited to individual objects, such as faces [78.20, 21], or to objects on the subordinate level of categorization [78.16, 22], but also was demonstrated for basic level recognition [78.19, 23].

Moreover, recognition performance is not only influenced by the orientation, but also by the size of the stimulus. Results are quite similar: reaction times (RTs) and error rates depend on the extent of transformation that is necessary to align memory and stimulus representation. RTs increase in a monotonic way with increasing change of (perceived) size (for a review *Ashbridge* and *Perrett* [78.24]). Several studies even show a systematic relationship between the amount of translation and recognition performance: Increasing displacement between two sequentially presented stimuli led to a deterioration of performance, both for novel objects [78.25] and familiar objects [78.26]. Overall, view-independent models are difficult to reconcile with these findings which indicate that recognition performance depends systematically on different spatial transformations.

In the following, we will briefly review three types of exemplar-based models, which – by virtue of different computational mechanisms and processes (including alignment, interpolation, and pooling/thresholding) – explain the transformation-dependent performance that was found in the psychophysical experiments.

In the class of alignment models, *Ullman*'s [78.12, 27] 3-D alignment model and *Lowe*'s [78.11] SCERPO model are probably the best-known examples. Both models work by storing 3-D models of objects, which are aligned to images by perspective projection of corresponding features (edges or feature points on the object). As an alternative to *Ullman*'s [78.27] model that relies on 3-D object representations, *Ullman* and *Basri* [78.28] suggested an alignment model on the basis of 2-D (two-dimensional) views. In this model, an internal object model is constructed by a linear combination of a small number of stored 2-D exemplar images. Thus, the alignment is not achieved by a spatial compensation process, but by linear combination of images. The intuition behind the linear combination approach can be explained in simple terms. Suppose that two views of the same three-dimensional object are stored, taken from somewhat different viewing directions. An intermediate view can then be described as a weighted sum of the views that are already stored. In this case, the representation is based on the two-dimensional positions of corresponding features in each view. Making the set of views closer results in an object representation that is equivalent to storing a 3-D model.

In the interpolation model, recognition is achieved by localization in a multidimensional representational space, which is spanned by stored views [78.29]. The interpolation model is based on the theory of approximation of multivariate functions and can be implemented with radial basis functions (RBFs). In this scheme, the whole viewing space of an object is approximated by the learned exemplar views through a limited number of series of so-called radial basis functions (such as Gaussian functions) each of which becomes activated within a limited region of the high-dimensional feature space. Object recognition then means to examine whether a new point corresponding to the actual stimulus can be approximated by the existing tuned set of basis functions. Thus, recognition does not occur by transformation or reconstruction of an internal image, but rather by interpolation or approximation of exemplars in a high-dimensional representational space.

At the end of the 1990s – and as an extension to the interpolation models – recognition models based on pooling and thresholding were developed [78.30–33]. Recognition is explained on the basis of the behaviour of cells that are selectively tuned to specific image features (fragments or whole shapes) in a view-dependent (and size-dependent) way. A hierarchical pooling of the outputs of view-specific cells provides generalization over viewing conditions [78.30]. A similar proposal was made by *Riesenhuber* and *Poggio* [78.31]. The threshold model [78.34] also accounts for the systematic relation between recognition latencies and the amount of rotation (and size-scaling). The speed of object recognition depends on the rate of accumulation of activity from neurons selective for the object, evoked by a particular viewing circumstance. For a familiar object, more tuned cells will be activated in the views most frequently presented, so that a given level of evidence (threshold) can be achieved fast. When the object is seen in an unusual view, fewer cells will respond, and activity among the population of cells selective for the object's appearance will accumulate more slowly. Consequently, these threshold models explain orientation-dependence without the need to postulate transformation or interpolation processes.

In a recent paper, an attempt has been made at view-dependent and view-independent approaches to object processing [78.35]. A careful study of the view-dependency of novel objects was designed by combining structural properties (number of parts) with metric properties (thickness, size of parts) has found that both view-dependent and view-independent processing seem to be combined in object recognition. Thus, instead of taking the extreme standpoints of view-based versus view-invariant processing, one might envisage a visual processing framework in which features are selected according to the current task, where the optimality, efficiency and thus the dependency on viewing parameters of the features depend on the amount of visual experience with this particular task.

Several computational models have been proposed that aim at modelling and explaining the dependence of human recognition performance on spatial transformations in its complexity. All of these models rely on storing exemplars – in the simplest form just 2-D views of objects – and matching the retinal image to these stored examples by different computational methods. The later models of recognition take their inspiration from recent findings from physiological studies concerning the functional building blocks of human vision in the brain. In the following, we will therefore briefly review the neural processing of visual information in the brain that underlies our ability to recognize objects.

78.1.2 Neural Representations in Object Recognition

Functionally, it has been shown that the flow of visual information in the brain can be divided into two major pathways: the dorsal pathway is believed to process motion and motor- or action-related visual information, whereas the ventral pathway usually is associated with the task of object recognition. The structure of the ventral pathway is hierarchically organized and consists of a series of interconnected stages that start from the retina, passing through the lateral geniculate nucleus (LGN) to the primary visual cortex (V1) and extrastriate visual areas V2, V4, and IT. The inferotemporal cortex (IT) provides input to the prefrontal cortex (PFC), which is believed to play an important role in identification and categorization of visual stimuli. Recordings in the parietal cortex [78.36] suggest, in addition, that specifically for grasping and object manipulation also dorsal regions might be centrally involved in the recognition of manipulable objects and their affordances (Chap. 77 for a more detailed discussion).

The seminal work of *Hubel* and *Wiesel* [78.37] in the cat (and later also in the macaque) visual cortex first established the idea of a hierarchical organization of visual processing. They found so-called simple cells in the early visual cortex (area V1) that responded best to bar-like stimuli at a particular orientation and position in the visual field. The response pattern of these cells could be modeled as a receptive field using Gabor-type functions. Later in the processing stream they found so-called complex cells which responded best to bar-like stimuli at a particular orientation nearly everywhere in the visual field – cells, which had become partially position invariant. This general idea of increasing invariance to stimulus properties with later stages of the processing stream has been verified in further physiological studies. In general, it has been found that the receptive field of the neurons increases and that the complexity of the stimulus it responds also increases. One of the key studies about the functional role of IT regions has investigated the responses of neurons to real-world objects in anesthetized monkeys ([78.38]; see also *Tanaka* [78.39] for a review). Although some neurons were found which responded maximally to simple bar-like stimuli, the majority of neurons in posterior inferotemporal cortex (PIT) preferred complex objects such as star shapes or circles with protruding elements. Interestingly, neurons were highly sensitive to minuscule changes to these objects such as the relative orientation or thickness of the elements. On the other hand, neurons were quite insensitive to stimulus variations such as size, contrast or retinal location. These findings were taken as evidence that one of the strategies for representing objects might be to use a number of moderately complex visual elements, whose pattern of co-activation encodes the visual appearance of the stimulus. In addition, *Wang* et al. [78.38] found neurons in anterior inferotemporal cortex (AIT), which responded maximally to images of whole objects such as faces or cars, indicating that already in IT object specific encodings might be present. Several other studies have also found neurons in this area which are tuned to faces, parts of faces, as well as body parts ([78.40] for a review).

In another set of experiments, *Logothetis* et al. [78.41] found AIT neurons, which showed a strong view-based behavior for the same stimuli that were used in the study of *Bülthoff* and *Edelman* [78.16], whereas they were invariant to size and location of the stimulus. Their findings provide strong evidence that a neural implementation of view-based object encoding is possible and indeed seems to be used for recognition. In addition to view-selectivity and size invariance, the investigated cells were also found to be maximally selective for the holistic stimulus rather than its constituent parts. This finding indicates that these cells might be encoding the pooled co-activation pattern of earlier PIT cells and thus form view-tuned units of recognition. It is important to stress in this context that an abstraction such as *grandmother* neurons, which specifically encode only one stimulus, does not seem plausible. Rather, the majority of neural responses in this and other experiments showed selectivity for a number of stimuli. A plausible explanation for this finding is that objects are encoded not by a single neuron but by a population code encompassing a number of neurons, which greatly increases the robustness of the representation [78.33].

The findings from this area of research can be summarized in a simple functional architecture: going from early stages to later stages of visual processing in a feedforward fashion, feature complexity increases from simple edge detectors toward view-tuned, com-

plex object cells and invariance to changes in the stimulus increases. This functional architecture is reflected not only in the object recognition framework discussed previously, but it also provides the motivation for computational vision systems that have been developed over the last few decades which will be discussed in the following.

78.1.3 Object Recognition: Lessons from Computer Vision

Computer vision started out as a subfield of artificial intelligence in the 1960s. Early work on scene understanding by *Roberts* [78.42] showed how computers could *parse* worlds consisting of simple, geometric objects such as cubes, pyramids, etc. The main thrust of computer vision systems in the following decades consisted of building algorithms for reconstructing a three-dimensional world from images – this development was further stimulated by *Marr*'s very influential theory of vision as 3-D reconstruction [78.1]. This theory was built on extracting geometric primitives from images that could be mathematically described as generalized cylinders. Although the mathematical rigor of such approaches was very appealing, computational implementations turned out to have strong limitations. Extracting robust features is a necessary prerequisite for building a 3-D reconstruction of the image, and finding these features proved to be hard under real-world conditions due to the enormous amount of variation in the image caused by changes in lighting, depth rotations, noise, occlusion, etc.

Parallel to the paradigm shift in human psychophysics and physiology, exemplar-based computational systems began to emerge, which for the first time showed good recognition performance under a larger range of viewing conditions. These recognition systems were based on – sometimes surprisingly simple – histograms of pixel values [78.43], local feature detectors [78.44, 45] or on a straightforward pixel representation of images using principle components analysis [78.46]. All of these recognition systems relied on a database of labeled example images, an algorithm for extracting features from these images, and a suitable classification method for comparing sets of image features.

Returning to the discussion of modeling human vision, in the following we provide an exemplary review of three neuromorphic recognition systems that are based on a functionally plausible, exemplar-based architecture: these are SpikeNET [78.48], LeNet [78.49], and a framework by *Serre* et al. [78.50]. The first system is motivated by the finding that humans are amazingly fast at categorizing images as containing an animal or a face [78.51]. Typical response times for this task are so small (on the order of 100 ms) that the visual signal has only time for one feedforward pass through the visual areas of the brain (Fig. 78.2) – any recurrent feedback processing would necessarily delay the decision and therefore result in longer response times. Based on this finding, a neural network architecture was designed [78.48] that exploits the timing of neuronal responses (spikes) to encode visual signals using a *who fires first* – strategy. This is different from traditional neural networks in that the timing is used rather than the firing strength. An object in this system will therefore be represented by an ensemble of neurons that represents a pattern of spike responses from earlier low-level, feature extraction neurons. In their implementation, these low-level neurons consist of standard Gabor-type receptive fields that are similar to the receptive fields found in the cat's visual cortex [78.37]. This spike time encoding allows for very fast processing of visual stimuli and has been shown to provide robust recognition results. The network ar-

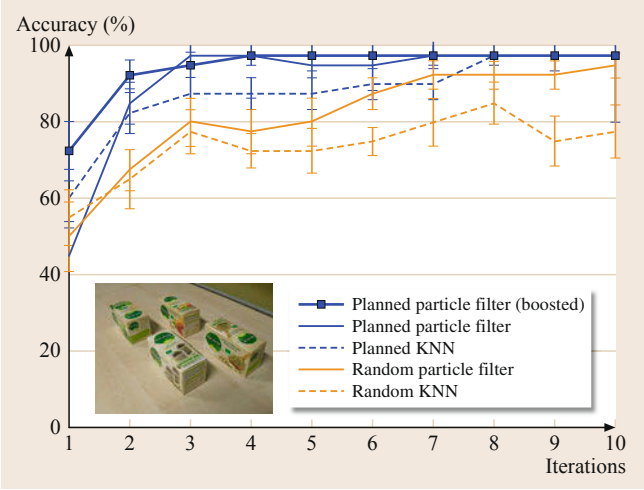


Fig. 78.2 Recognition performance of four highly similar object (*shown in the inset*) by an in-hand recognition system using active view selection (after [78.47]). The five methods compared in the plot contrast planned (*blue*) and unplanned (*orange*) exploration of the objects in the hand of the robot. The *x*-axis is the number of iterations, and the *y*-axis is the recognition accuracy in percent. As time (or iteration number) proceeds, the planned approaches surpass random exploration significantly. In addition, employing proabilistic methods for recognition of the objects using a particle filter also provides a recognition improvement. Finally, the *thick, solid blue line* shows performance in a system which boosts the likelihood of an object given the current visual evidence in the particle filter framework – this approach fares best overall. These results show that active view selection enhances the robot's ability to learn and recognize objects in real-world environments

chitecture LeNet [78.49] consists of a neural network that uses a hierarchy of layers of trainable convolutions and spatial subsampling, as well as nonlinear filtering to extract features of increasingly large receptive fields, increasing complexity, and increasing robustness. Using extensive, supervised training of the full hierarchy, such a network provides a very efficient, sparse set of features for many visual recognition tasks. Finally, the network architecture by *Serre* et al. [78.50] uses a very similar hierarchical structure of layers in which feature complexity and invariance are successively increased by linear and nonlinear pooling – its lower level feature detectors, however, are trained in an unsupervised fashion on a large database of natural images, yielding a large set of detectors that are optimally tuned to natural image statistics. Again, the performance of this model in recognition tasks has been shown to be very good – in addition, comparisons with physiological and psychophysical experiments have shown that this framework is also capable of modeling human results from these experiments.

Recent research has mainly focused on two topics: the automatic extraction of optimal visual features for efficient recognition and categorization, and the extension of the frameworks for providing invariance against changes in viewing conditions (such as rotations in depth, scaling, translation, illumination, and occlusion, for example, *DiCarlo* et al. [78.52], *Rolls* [78.53] for discussions of invariance in neuromorphic architectures). In a recent paper [78.54], these two issues have been addressed in a face recognition task conducted on a difficult database of faces taken in uncontrolled environments. The selection of optimal features was done by evaluating a large set of potential visual feature combinations using GPU-accelerated algorithms. The issue of invariance was addressed by using a hierarchical, multilayer model in which each layer includes linear and nonlinear pooling operations that encode the input image. The combined system was benchmarked against other standard feature-extraction methods and a *flat*, nonhierarchical one-layer model. Both properties resulted in increased recognition performance on the database outperforming other benchmarked state-of-the-art methods. In addition, the system also showed increased robustness against viewing variations, which included pose, position, scale, and background clutter.

In summary, neuromorphic architectures have now reached a stage of maturity that can put them even ahead of sophisticated, state-of-the-art computer vision frameworks. The ability to learn and adapt the feature set to viewing conditions and the increased robustness to viewing conditions makes such architectures good candidates for building the visual learning and recognition system for a perceptual robot.

78.1.4 Object Learning and Recognition for Perceptual Robotics

In general, it can be said that the success of perceptually inspired recognition systems can be seen as a strong indicator for the feasibility of a data-driven, exemplar-based approach to recognition. There are three issues, however, which so far have not been addressed in any of these vision systems and which will be important both for achieving human performance in generic recognition tasks in a perceptual robotics application – as well as for a full understanding of the processes in human object recognition.

First of all, all of the above-mentioned systems are feedforward – virtually no feedback, recurrent processing is implemented in their architecture, which makes them in a sense very similar to the simpler frog- or bee-like neural systems discussed in Chap. 77. Although there is evidence that humans solve some recognition tasks using very little feedback (see, e.g., *Thorpe* et al. [78.51]; *DiCarlo* et al. [78.52]), it nevertheless is a crucial component of visual processing driving, for example, attentional focus, context awareness, as well as memory and reasoning processes – basically everything that makes up visual *intelligence*. Some visual attention models that are relevant for robotics systems are reviewed in Chap. 77.

Secondly, a severe limitation of most of today's artificial recognition systems is that they solely focus on the static domain of object recognition. Visual input on the retina, however, consists of dynamic changes due to object- and self-motion, nonrigid deformations of objects, articulated object motion as well as scene changes such as variations in lighting, occluding, and re- and disappearing objects – where at any given point in time several of these changes can be interacting. Several psychophysical experiments, indeed suggest an important role for dynamic information, both in learning and recognition of objects [78.55–58]. These results ask for an extension of current object recognition frameworks with a temporal component in order to arrive at truly spatiotemporal object representations. Combining methods from computer vision, psychophysics, and machine learning, *Wallraven* and *Bülthoff* [78.59, 60], have developed a framework that fulfills this requirement and learns spatiotemporal, exemplar-based object representations from image sequences. More specifically, spatiotemporal characteristics of the visual input are integrated into a connected view-graph representation based on tracked local features. In order to provide robust classification performance, machine learning techniques are used to design efficient methods for combining support vector classification schemes with these local feature representations [78.61]. In sev-

eral studies it was shown that the framework achieved excellent recognition results on both highly controlled databases as well as on real-world data. The integration of spatiotemporal information provides characteristic information about dynamic visual input via the connection of views and the two-dimensional image motion of discriminative features. In addition to delivering good recognition performance, the framework was also able to model results from psychophysical experiments on face and object recognition. A similar model using a neuromorphic architecture integrating the temporal dimension was proposed by *Kietzmann* et al. [78.62].

A third issue that – in our view – will be essential for designing and implementing efficient perceptual robots consists of the multisensory nature of our perceptual system (see also the discussion of embodied robots in Chap. 13). As an example, there is a close coupling between the human visual and haptic system – touch can provide a wealth of complementary information about an object when it is manipulated, such as its texture, its shape, its position in space relative to our body, etc. In a series of psychophysical experiments [78.63], participants had to learn views of four simple, 3-D objects made of stacked toy-bricks either in the haptic modality (when they were blind-folded) or in the visual modality (without being able to touch them). Subsequently, they were tested both within the same modality as well as across modalities. Recognition results showed that cross-modal recognition is possible well above chance. Not surprisingly, recognition of rotated objects in the within-modality condition was severely affected by rotation in both modalities. This shows that not only visual recognition is highly view-dependent but also that haptic recognition performance is directly affected by different viewing parameters. The results from this experiment thus support the view that haptic recognition is also mediated by exemplar-based processes.

Taken together with the keyframe framework outlined above, this cross-modal transfer might be an important reason for the excellent visual performance of human object recognition – after all, it is known that infants learn extensively by actively grasping and touching objects, which thus could provide a *database* of object representations for visual recognition [78.64]. Using this basic perceptual principle as a motivation [78.60] have applied an extension of the keyframe framework in an online robotics scenario for efficient learning and recognition of multisensory object representations. More specifically, a framework was developed to integrate both proprioceptive information originating from haptic sensors in the robot's hands and visual information coming from the robot's cameras. For this, the robot would perform an exploratory movement with an object in its hand (such as turning it and looking at it from all angles) and from the resulting image sequence learn spatiotemporal, view-based representations using the keyframe framework. Each view of this representation, however, is also linked to the current proprioceptive state (i. e., the joint angles of the hand at that point in time) and therefore provides an anchor into a hand-centered, three-dimensional space. In this way, a representation is generated that links perception and action. The proprioceptive information can then be used as an additional constraint for both learning of objects and recognition of objects and was shown to provide increased robustness compared to visual matching alone. The framework was also used as the basis for recent work in which a humanoid robot (the iCub) performed active in-hand object recognition, searching for the optimal view that allowed it to disambiguate the object currently held from other, previously seen objects [78.47]. Again, linking the exploratory actions (turning the hand) with the visual data resulted in a much faster and more reliable object recognition performance. Sample data comparing unplanned and planned recognition of difficult objects is shown in Fig. 78.2 (VIDEO 569).

Such approaches pave the way for a view of recognition as an active, multisensory process in which rich, extensible object representations are formed and improved over the life-time of the robot.

78.2 Perceptual Mechanisms of Action Representation

The recognition of complex movements and actions is fundamental for many applications in robotics, such as imitation learning by observation. Interactive robots need to analyze their users' movements in order to respond in a natural way to their social and emotional behavior (Chap. 72). The following section reviews what is known about movement and action recognition in the brain and tries to highlight a few aspects that have or might be successfully transferred to biologically inspired applications in robotics and computer vision.

78.2.1 Recognition of Complex Movements and Actions in Primate Cortex

The recognition of complex movements and actions is a fundamental problem for higher animals and specif-

ically for primates. While simple movement patterns are sufficient for eliciting stereotypical prey catching behavior in simple vertebrates ([78.65]; see also Chap. 77), higher animals exploit more complex movement patters, e.g., for the recognition of conspecifics or predators, or for communication by facial movements, gestures, or body expressions. Human perception of body motion patterns is very efficient, even for extremely impoverished stimuli. This has been demonstrated in classical experiments by *Johansson* [78.66], who showed that complex dynamic actions can be recognized even from displays that consist only of a small number of dots moving like the joints of a human actor. Subsequent research has demonstrated that humans can extract highly specific information from such *point-light displays*, e.g., the gender or the identity of people. To our knowledge, no technical system for motion recognition has been proposed so far that accomplishes a comparable level of robustness. While much more research in neuroscience has been dedicated to object recognition (Sect. 78.1.2), some studies have tried to uncover neural [78.67–70], and computational principles [78.71–73] of visual movement recognition. Some of these principles have been transferred to the construction of systems in computer vision and robotics.

Neurophysiological and brain-imaging studies indicate that the recognition of facial and body movements involves the ventral and the dorsal visual pathway. This implies that likely form and optic flow information are integrated during the processing of action stimuli in visual cortex. The ventral pathway, which is specifically responsible for the processing of form information has been discussed already in Sect. 78.1.2. Like the ventral stream, also the dorsal pathway is hierarchically structured, and the size of the receptive fields of the neurons increases along the hierarchy. Some cortical areas that are part of the dorsal pathway are listed in Fig. 78.3. The *medial temporal area* (MT) contains neurons that are selective for simple local motion and coherent motion. On higher levels of the dorsal stream, e.g., in the *superior temporal sulcus* (STS), neurons that are selective for hand and body movements and for facial expressions have been found in monkeys [78.69], and similar structures are activated by these stimuli in the human brain. In addition, areas selective for human body shapes, such as the *extrastriate body part area* (EBA), likely to contribute to the recognition of actions [78.74], where information of form and motion features seems to be integrated on higher processing levels [78.73].

For the recognition of goal-directed actions, such as reaching or grasping, in addition cortical structures beyond the visual cortex, such as the parietal and premotor cortex, seem to play a critical role. The role of these structures for action recognition has been analyzed in particular in the context of the study of the *mirror neuron system* [78.75]. Mirror neurons are sensorimotor neurons that combine visual tuning during action observation as well as selective motor tuning. Areas in parietal cortex, such as the anterior might be specifically relevant for the recognition of action-related objects and their relationship to moving effectors [78.36]. Research about the mirror neuron system has influenced the construction of a whole generation of biologically inspired robots (Chap. 77). The guiding hypothesis has been that the visual recognition and *understanding* of actions is accomplished by mapping of observed body movements onto motor representations that are relevant for the execution of the same type of action.

78.2.2 Biological Principles with Relevance for Computer Vision and Robotics

We discuss in the following two major principles that have been derived from the analysis of action recognition in biological systems that have been transferred to technical applications in computer vision and robotics.

A first principle that seems to be implemented in movement recognition in primate cortex is a hierarchical architecture of feature detectors, which accomplishes action recognition by the detection of temporal sequences of relevant motion and form features. Such detection does not necessarily require the reconstruction of the three-dimensional facial or body shape, nor an exact simulation of the dynamics of the underlying movements. Instead it can be accomplished by much simpler computational mechanisms. Like object recognition, the recognition of complex motion patterns is strongly orientation- and view-dependent. This property has been observed at the level of individual neurons in the STS, and for the activation of biological motion-selective areas in human cortex [78.69] as well as for action-selective neurons in higher areas such as the premotor cortex [78.76]. View- and orientation dependence seem compatible with an encoding of visually perceived movements in terms of potentially learned example views, or *keyframes* (snap shots), and of instantaneous optic flow patterns that are characteristic for actions [78.67, 73]. While there might be some innate preferences for specific features [78.77], psychophysical and fMRI (functional magnetic resonance imaging) experiments suggest an important role of learning in visual movement recognition [78.78–80]. For example, subjects can learn easily to recognize individually – specific body and facial movements [78.81, 82]. Learning-based theoretical models, exploiting similar principles as neural object recognition models, account for a variety of experimental data on action

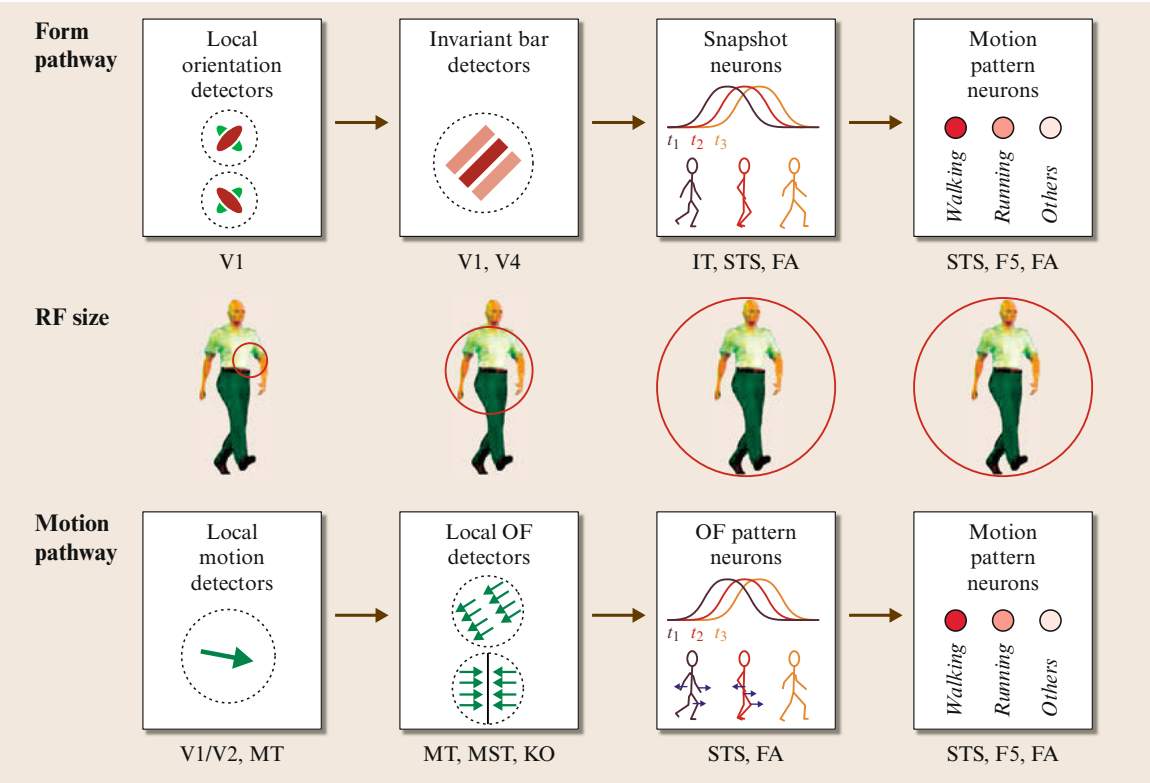


Fig. 78.3 Example-based neural model for the visual recognition of body movements that integrates the processing of form and motion features in the ventral and dorsal visual pathways (after *Giese* and *Poggio* [78.73])

recognition in biological systems [78.73, 83, 84], also supporting a central role of learning.

As an example of such a learning-based architexture, Fig. 78.3 illustrates a hierarchical model for the recognition of complex body movements [78.73]. It consists of two hierarchical streams modeling the ventral and the dorsal visual pathways, which contain detectors for action-specific motion and form features. The form pathway of this model is similar to the object recognition models described in Sect. 78.1.2. The motion pathway of the model contains detectors for action-specific optic flow features with different complexity. Like for the described object recognition models, position, and scale invariance is accomplished by appropriate nonlinear pooling of the responses of detectors with different spatial and scale selectivity along the hierarchy. In addition, the model contains recurrent neural circuits that make the responses of the recognition neurons selective for temporal order. In this way the model responds only to actions that are executed with the correct temporal order, and also with approximately correct speed. The underlying network dynamics can be interpreted as a neural implementation of a Markov model, where the present recognized pattern predicts possible future patterns (Chap. 68). A strong activity in the network emerges only when the stimulus sequence matches these predictions.

Similar hierarchical neural architectures inspired by the visual cortex have been used in the context of mirror-neuron robot systems [78.85, 86]. In addition, recent work in computer vision shows that such biologically inspired architectures can reach very high performance levels, comparable to state-of-the-art algorithms in computer vision [78.87–89].

A second principle of movement recognition, which has been discussed extensively as basis for the recognition of imitable actions, and as explanation of the function of the mirror neuron system [78.75] is the idea that action observation is based on an internal simulation of the observed motor behavior. A variety of computational models for action recognition by internal simulation have been proposed in the neuroscience literature, e.g., exploiting feedforward controllers [78.90], coupled forward and backward models [78.91], hierarchical Bayesian predictive models [78.92], or a free energy minimization framework [78.93]. (A further more extensive discussion about theoretical models for the mirror neuron system with relevance for robotics

can be found in Chaps. 68 and 77.) A main difficulty of the recognition of actions by internal simulation of associated motor behaviors is the accurate estimation of relevant geometrical quantities from image data, especially when no special depth sensors or even online motion capture are available. Many of the underlying motor control models are formulated in joint angle space, and the robust recognition of joint angles from monocular videos is known to be a difficult computer vision problem, which so far is solvable only for highly restricted classes of movements with strong learned priors, and at considerable computational cost [78.94, 95]. This raises the question about simpler computational approaches for the recognition of goal-directed actions, which explain biological data and might be interesting for technical applications.

A recently developed model for the visual recognition of goal-directed hand actions in cortex that follows these lines [78.96] is illustrated in Fig. 78.4. The underlying architecture is an extension of the form pathway of the model shown in Fig. 78.3, by the addition of neural circuits that process the spatial and temporal relationship between the observed effector (in this case the hand) and the recognized goal object (e.g., a grasped object). The model works, exploiting a purely exemplar-based approach (Sect. 78.1.1), without explicit reconstruction of the three-dimensional structure of the object or the effector. The model comprises three modules: The first module (A, in Fig. 78.4) recognizes shapes of the goal object and of the effector, implementing a shape recognition hierarchy similar to standard object recognition models as the ones described in Sect. 78.1.3. The analysis of the temporal deformation of the hand is based on the recognition of sequences of key shapes, like in the form pathway of the model in Fig. 78.3. Opposed to standard object recognition models, however, the highest level of this shape recognition hierarchy is not completely position-invariant. Rather, it retains coarse position selectivity by implementing multiple replica of the same shape detectors that are selective for different image positions. This makes it possible to further analyze the spatiotemporal relationship of the recognized goal objects and effector. This analysis is realized in the second module (B) whose core is formed by two-dimensional *relative position maps*. These are neural activity maps that represent the effector position as activity peak in a two-dimensional coordinate system that is centered on the object position in the image. These maps are computed by a gain fields [78.97] that multiply the output activities of shape selective neurons with selectivity for object and effector. The activity distribution in these neural map is analyzed by *affordance neurons* that are activated only when hand and object shape match and are in a spatial relationship that is suitable for a successful grip. By appropriate pooling of the responses of motion energy detectors that receive input from the relative position map *relative motion neurons* can be constructed, whose activity characterizes the relative motion between the effector and object

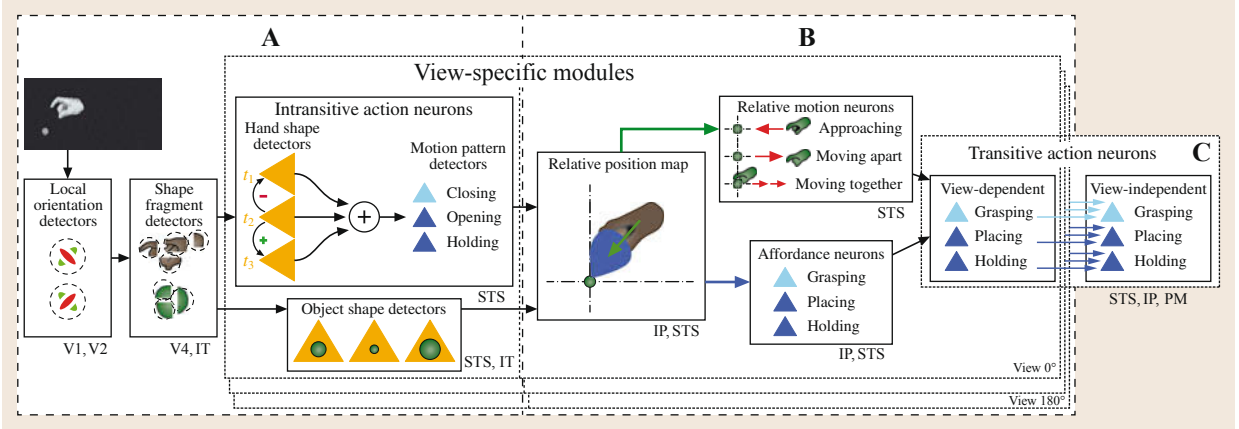


Fig.78.4a–c Physiologically inspired model for the recognition of goal-directed hand actions. The shape-recognition **(a)** recognizes the shape of goal objects and the shapes of individual hand postures, retaining some course position information. The module **(b)** associates the information of hand and object by computing maps that represent the relative positions of hand and object in image coordinates. From these maps the spatial matching hand and object and their relative motion can be computed. The highest level module **(c)** contains model neurons that are selective for different types of goal-directed actions. Up to this level the model recognizes actions in a view-dependent manner, and only at the highest level (view-independent transitive actions neurons) the model accomplishes view independence by pooling the outputs from view-specific modules (courtesy of *Fleischer* et al. [78.96])

(e.g., the hand approaching the object). The outputs of affordance and the relative motion neurons are integrated in the third module (C), which contains only neurons that are selective for visually observed goal-directed actions. This integration of information is first accomplished in a purely view-specific manner. View independence is not established until the very last hierarchy level of the model that contains view-independent action-selective neurons. The idea of establishing view-dependence only very late in the cortical hierarchy seems counter-intuitive, and is at odds with several established computational models for action recognition. However, this dominance of exemplar-based representations until very high levels of the cortical processing hierarchy has been observed in electrophysiological experiments studying mirror neurons in premotor cortex [78.76]. In this structure, which is traditionally associated with motor planning, the majority of mirror neurons is view-dependent and only a minority is view-independent. The discussed model can recognize hand actions from gray level videos. It could be augmented by integration of disparity or depth features, and by appropriate attentional control mechanisms that would make it more robust to cluttered scenes with multiple relevant objects. Whether similar architectures have advantages for the robust visual recognition of goal-directed actions in technical systems remains to be shown. Very recent work shows that such hierarchical deep architectures, which consist of learned feature detectors, outperform classical technical solutions on actual computer vision benchmarks for action detection [78.98].

78.3 Perceptual Validation of Robotics

Successful human–robot interaction is perhaps easiest when the robot offers interaction channels that are compatible to that of human–human interaction [78.99]. The most important interaction channels in this case are verbal and nonverbal communication with the face. Importantly, in human–human interaction, nonverbal communication using facial expressions, for example, constitutes up to 30% of the communicative content. Facial expressions are not only used to convey someone's mood and emotion [78.100], but are also used in communicative contexts to signal understanding (a nod of the head), to modify what is being said (a raise of the eye-brows), and to control the conversational flow (a look of confusion may signal to the speaker to repeat what has been said). Hence, many humanoids have incorporated more or less sophisticated heads capable of producing human-like facial expressions and movements. Traditionally, this has been achieved using mechatronic implementations in which actuators drive facial features directly (e.g., as in the MDS robot by *Lee* and *Breazeal* [78.101]), or – in more complex implementations of android robots – mimic human muscle movements that are then used to deform artificial skin [78.102–104]. Other systems have used LEDs for displaying simple, changeable facial features (e.g., as in the iCub platform [78.105].

78.3.1 Realistic Faces for Robots

With such a great variety of robot systems also comes the need for an investigation of their perceptual evaluation and their interaction capabilities ([78.106] for such a study in the context of facial animations in computer graphics). Indeed, one particular problem of android, human-like systems is that they easily could suffer from the uncanny valley effect as the actuators and/or the control framework cannot easily reproduce the smoothness of human facial expressions. A study of morphed images between a nonhuman robot face and a highly realistic android robot head, for example, clearly showed evidence of the uncanny valley effect [78.107] – a similar study for moving robot faces yielded more mixed results, but still showed that the most realistic robotic faces were clearly perceived as different from that of a human talking [78.108]. One solution for this is to change the robot's appearance such that it stays away from close human likeness; however, conveying the full breadth of human communicative signals with different facial features, or a different facial topology may also be problematic. A different solution consists of avoiding a mechanical solution and instead resorting to facial animation from computer graphics. One example of such a system was presented in *Delaunay* et al. [78.109] in which facial animations are projected onto a rigid face mask. Since facial animation techniques are in many ways much more advanced, such a system allows for a more realistic and flexible interaction in human–robot interaction. Subtle cues such as eye-gaze, wrinkles, and other nonrigid facial deformations could be displayed via projection of an advanced facial animation engine. Initial perceptual experiments with such systems [78.109, 110] have yielded promising results. However, more studies need to be done to assess the properties of human–robot interactions in these and other implementations of facial displays.

Fig. 78.5 Stimuli used in experiment investigating the fMRI correlates of the observation of human and robot movements. The neural responses to the movements by a real human are compared with the ones induced by a human-looking robot (Android), and by a nonhumanlike robot (Robot) (after *Saygin* et al. [78.111])

78.3.2 Perceptual and Neural Processing of Body Movements of Robots

Research in humanoid robotics finally aims at optimizing the perceived naturalness or *human-likeness* of generated robot movements, since this in the long run will increase the acceptance of humanoid robots in social contexts. However, the present humanoid platforms have typically substantial constraints that still prevent the realization of complex really human-like movements. This is even more the case for the realization of behaviors on bipedal robots, due to the difficult problem to maintain dynamic balance (Chap. 67). Therefore, most body movements realized by present humanoid robots still differ in many aspects from human movements. This makes the quantification of the degree of realism of such movements presently a less pressing topic than the field of computer graphics, where psychophysical studies for the validation of the realism and quality of computer animation methods are meanwhile a standard [78.112–114]. However, research in psychophysics and neuroscience has started to investigate the differences between the perceptual processing of human and robot movements, and interesting results have been obtained that localize cortical subsystems that might be essential to distinguish human and nonhuman robot movements. A typical question in these studies has been which critical properties determine whether visual stimuli produce *motor resonance*, or an activation of action-selective neural structures. The results of such studies have not been completely consistent, since some studies found decreased activation of action-selective networks for robot movements [78.115–118] while others found no such differences [78.119]. Primarily visual processing areas responded sometimes more for robot movements than for normal human [78.111, 117].

The problem of such studies is that many factors might influence the perception and neural signals in action-selective areas, such as form, kinematics, and optic flow patterns. Typically, it is very difficult to control these parameters separately for real robots. In addition, the learning experience of observers with the specific robot might play an important role [78.120]. A recent study by *Saygin* et al. [78.111] tried to separate at least the influences of the robot appearance (shape) and the motion kinematics by comparing the fMRI signals (using an adaptation paradigm) driven by three different stimuli (Fig. 78.5): a real person (that served as model for the building of the robot), the human-like looking robot (*android*), and the robot without skin and surface parts that made it look human-like (resulting in very similar motion as the full robot). In visual areas (e.g., the extrastriate body area) the human and the human-like robot stimulus result in very similar activity. This is not true for the parietal cortex, which is part of the *mirror neuron network*. This region shows large differences between the human-like robot and the other two conditions, potentially reflecting an increase of neural processing resources that are required to cope with the contradiction between the form and the kinematic information that is presented by this stimulus. Opposed to this, the not human-like robot makes it expected that the motion is also not human-like, potentially causing no such conflict. Future studies of similar type, controlling for the different information channels of action processing (Sect. 78.2.2) as well as for the predictibility of such stimuli dependent on previous learning, potentially combined with quantitative neural modeling, will be required to really understand how different factors are integrated in the neural processing of robot movements, and how this causes different levels of perceived *human-likeness*.

78.4 Conclusion and Further Reading

In this chapter, we have presented several principles derived from high-level cognitive processing in vision in the human brain that have been fruitful for the development of systems in robotics and computer vision. The recognition of shapes and complex movements and actions is an important problem for many

applications in robotics. We have discussed a variety of results from neuroscience that indicate that these brain functions are likely realized by example-based representations. We have discussed neural implementations of such representations which partially have been tested successfully in the context of technical applications, and which are strongly inspired by the real cortical neural architecture. In addition, we have presented some new computational principles that seem to emerge from recent experimental results on the representation of goal-directed actions. Finally, we have discussed work that tries to use psychophysical and neuroscience methods for the validation of the appearance and the movements of human-like robots, and for the investigation of underlying neural mechanisms.

Example-based mechanisms for object and motion recognition account for the invariant recognition of complex patterns. However, they do not automatically extract the metric information about the object geometry, position and the spatial parameters of complex trajectories in world coordinates. For some tasks in robotics, like grasping, manipulation, or obstacle avoidance, such information is required (Chaps. 36–38, 47). For such tasks, example-based recognition must be fused with methods for the extraction of the relevant metric information. In robotics such information can be extracted by stereo vision or using special sensors, like laser range finders. In the brain the fusion between such spatial information and information about objects occurs likely in parietal areas, like the anterior interparietal area (AIP) [78.121]. However, it is unclear whether the information about objects is only represented in terms of 2-D example views. Instead, it seems likely that also some form of 3-D information is encoded, potentially in an example-based manner. Also haptic and visual information about object shape might be merged in higher brain areas, e.g., in parietal and fusiform areas [78.122]. A further discussion about biologically inspired models for the extraction of action-relevant geometrical information in the context of grasping and manipulation is given in Chaps. 32 and 77.

The perceptual validation of the human-likeness and affective impact of humanoid robots likely will become increasingly important along with the further development of the technology that will increase the level of similarity between humanoid robots and humans. Likewise, it seems increasingly important to use quantitative methods from perception science to investigate the quality of the emotional and social interaction between robots and humans. We expect this to be a field where psychology can really contribute quantitative methods to engineering, reaching a level that goes beyond a qualitative and subjective comparison of *demos* which partially is still the standard in the field of humanoid robotics.

Video-References

VIDEO 569 Active in-hand object recognition
available from http://handbookofrobotics.org/view-chapter/78/videodetails/569

References

78.1 D. Marr: *Vision* (Freeman, San Francisco 1982)
78.2 J.A.S. Kelso: *Dynamic Patterns: The Self-Organization of Brain and Behaviour* (MIT, Cambridge 1995)
78.3 G. Schöner, M. Dose, C. Engels: Dynamics of behavior: Theory and applications for autonomous robot architectures, Robotics Auton. Syst. **16**, 213–245 (1997)
78.4 J. Tani, M. Ito: Self-organization of behavioral primitives as multiple attractor dynamics: A robot experiment, IEEE Trans. Syst. Man Cybern. A **33**(4), 481–488 (2003)
78.5 W.H. Warren: The dynamics of perception and action, Psychol. Rev. **113**, 358–389 (2006)
78.6 M. Mori: The uncanny valley, Energy **7**(4), 33–35 (1970), in Japanese
78.7 D.W. Cunningham, C. Wallraven: *Experimental Design: From User Studies to Psychophysics* (CRC, Boca Raton 2011)
78.8 A. Field, G. Hole: *How to Design and Report Experiments* (Sage, London 2011)
78.9 D. Marr, H. Nishihara: Representation and recognition of the spatial organization of three-dimensional shapes, Proc. R. Soc. B **200**, 269–294 (1978)
78.10 I. Biederman: Recognition-by-components: A theory of human image understanding, Psychol. Rev. **94**, 115–147 (1987)
78.11 D. Lowe: *Perceptual Organization and Visual Recognition* (Kluwer, Boston 1985)
78.12 S. Ullman: *High-Level Vision. Object Recognition and Visual Cognition* (MIT, Cambridge 1996)
78.13 M.A. Kurbat: Structural description theories: Is RBC/JIM a general-purpose theory of human entry-level object recognition?, Perception **23**, 1339–1368 (1994)
78.14 S. Edelman: *Representation and Recognition in Vision* (MIT, Cambridge 1999)

78.15 M. Graf, W. Schneider: Structural descriptions in HIT – A problematic commitment, Behav. Brain Sci. **24**, 483–484 (2001)

78.16 H.H. Bülthoff, S. Edelman: Psychophysical support for a two-dimensional view interpolation theory of object recognition, Proc. Natl. Acad. Sci. USA **89**, 60–64 (1992)

78.17 M.J. Tarr, S. Pinker: Mental orientation and orientation-dependence in shape recognition, Cogn. Psychol. **21**, 233–282 (1989)

78.18 W.G. Hayward, M.J. Tarr: Testing conditions for viewpoint invariance in object recognition, J. Exp. Psychol. **23**, 1511–1521 (1997)

78.19 S.E. Palmer, E. Rosch, P. Chase: Canonical perspective and the perception of objects. In: *Attention and Performance IX*, ed. by J. Long, A. Baddeley (Erlbaum, Hillsdale 1981) pp. 135–151

78.20 H. Hill, P.G. Schyns, S. Akamatsu: Information and viewpoint dependence in face recognition, Cognition **62**, 201–222 (1997)

78.21 C. Wallraven, A. Schwaninger, S. Schuhmacher, H.H. Bülthoff: View-based recognition of faces in man and machine: Re-visiting inter-extra-ortho, Lect. Notes Comput. Sci. **2525**, 651–660 (2002)

78.22 M.J. Tarr: Rotating objects to recognize them: A case study on the role of viewpoint dependency in the recognition of three-dimensional objects, Psychon. Bull. Rev. **2**, 55–82 (1995)

78.23 R. Lawson, G.W. Humphreys: View-specific effects of depth rotation and foreshortening on the initial recognition and priming of familiar objects, Percep. Psychophys. **60**, 1052–1066 (1998)

78.24 E. Ashbridge, D.I. Perrett: Generalizing across object orientation and size. In: *Perceptual Constancy. Why Things Look as They Do*, ed. by V. Walsh, J. Kulikowski (Cambridge Univ. Press, Cambridge 1998) pp. 192–209

78.25 M. Dill, S. Edelman: Imperfect invariance to object translation in the discrimination of complex shapes, Perception **30**, 707–724 (2001)

78.26 K.R. Cave, S. Pinker, L. Giorgi, C.E. Thomas, L.M. Heller, J.M. Wolfe, H. Lin: The representation of location in visual images, Cogn. Psychol. **26**, 1–32 (1994)

78.27 S. Ullman: Aligning pictorial descriptions: An approach to object recognition, Cognition **32**, 193–254 (1989)

78.28 S. Ullman, R. Basri: Recognition by linear combinations of models, IEEE Trans. Pattern Anal. Mach. Intell. **13**, 992–1006 (1991)

78.29 T. Poggio, S. Edelman: A network that learns to recognize three-dimensional objects, Nature **343**, 263–266 (1990)

78.30 D. Perrett, W.M. Oram: Visual recognition based on temporal cortex cells: Viewer-centred processing of pattern configurations, Z. Naturforsch. C **53**, 518–541 (1998)

78.31 M. Riesenhuber, T. Poggio: Hierarchical models of object recognition in cortex, Nat. Neurosci. **2**, 1019–1025 (1999)

78.32 E.T. Rolls, T. Milward: A model of invariant object recognition in the visual system: Learning rules, activation functions, lateral inhibition, and information-based performance measures, Neural Comput. **2**(11), 2547–2572 (2000)

78.33 G. Wallis, H.H. Bülthoff: Learning to recognize objects, Trends Cogn. Sci. **3**, 22–31 (1999)

78.34 D. Perrett, W.M. Oram, E. Ashbridge: Evidence accumulation in cell populations responsive to faces: An account of generalization of recognition without mental transformations, Cognition **67**, 111–145 (1998)

78.35 D.H. Foster, S.J. Gilson: Recognizing novel three-dimensional objects by summing signals from parts and views, Proc. R. Soc. B **269**, 1939–1947 (2002)

78.36 H. Sakata: The role of the parietal cortex in grasping, Adv. Neurol. **93**, 121–139 (2003)

78.37 D.H. Hubel, T.N. Wiesel: Receptive fields, binocular interaction and functional architecture in the cat's visual cortex, J. Physiol. (Lond.) **160**, 106–154 (1962)

78.38 G. Wang, M. Tanifuji, K. Tanaka: Functional architecture in monkey inferotemporal cortex revealed by in vivo optical imaging, Neurosci. Res. **32**, 33–46 (1998)

78.39 K. Tanaka: Representation of visual feature objects in the inferotemporal cortex, Neural Netw. **9**(8), 1459–1475 (1996)

78.40 K. Grill-Spector, R. Malach: The human visual cortex, Annu. Rev. Neurosci. **27**, 649–677 (2004)

78.41 N.K. Logothetis, J. Pauls, H.H. Bülthoff, T. Poggio: View-dependent object recognition by monkeys, Curr. Biol. **4**, 401–414 (1994)

78.42 L. Roberts: Machine perception of three-dimensional solids. In: *Optical and Electro-Optical Information Processing*, ed. by J.T. Tippet (MIT, Cambridge 1965) pp. 159–197

78.43 M. Swain, D. Ballard: Color indexing, Int. J. Comput. Vis. **7**, 11–32 (1991)

78.44 C. Schmid, R. Mohr: Local greyvalue invariants for image retrieval, IEEE Trans. Pattern Mach. Intell. **19**, 530–535 (1997)

78.45 D. Lowe: Distinctive image features from scale invariant keypoints, Int. J. Comput. Vis. **60**(2), 90–110 (2004)

78.46 M. Kirby, L. Sirovich: Applications of the Karhunen-Loeve procedure for the characterization of human faces, IEEE Trans. Pattern Mach. Intell. **12**, 103–108 (1990)

78.47 B. Browatzki, V. Tikhanoff, G. Metta, H.H. Bülthoff, C. Wallraven: Active in-hand object recognition on a humanoid robot, IEEE Trans. Robotics **30**(5), 1260–1269 (2014)

78.48 A. Delorme, S. Thorpe: SpikeNET: An event-driven simulation package for modeling large networks of spiking neurons, Netw. Comput. Neural Syst. **14**, 613–627 (2003)

78.49 Y. LeCun, F. Huang, L. Bottou: Learning methods for generic object recognition with invariance to pose and lighting, Proc. 2004 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn. (2004)

78.50 T. Serre, L. Wolf, T. Poggio: Object recognition with features inspired by visual cortex, Proc. 2005 IEEE

Comput. Soc. Conf. Comput. Vis. Pattern Recogn. (2005)
78.51 S. Thorpe, D. Fize, C. Marlot: Speed of processing in the human visual system, Nature **381**(6582), 520–522 (1996)
78.52 J.J. DiCarlo, D. Zoccolan, N.C. Rust: How does the brain solve visual object recognition?, Neuron **73**(3), 415–434 (2012)
78.53 E.T. Rolls: Invariant visual object and face recognition: Neural and computational bases, and a model, VisNet. Front. Comput. Neurosci. **6**(35), (2012)
78.54 N. Pinto, D. Cox: High-throughput-derived biologically-inspired features for unconstrained face recognition, Image Vis. Comput. **30**(3), 159–168 (2012)
78.55 G. Wallis, H.H. Bülthoff: Effects of temporal association on recognition memory, Proc. Natl. Acad. Sci. USA **98**, 4800–4804 (2001)
78.56 J.V. Stone: Object recognition using spatio-temporal signatures, Vis. Res. **38**(7), 947–951 (1998)
78.57 J.V. Stone: Object recognition: View-specificity and motion-specificity, Vis. Res. **39**(24), 4032–4044 (1999)
78.58 Q.C. Voung, M.J. Tarr: Rotation direction affects object recognition, Vis. Res. **44**(14), 1717–1730 (2004)
78.59 C. Wallraven, H.H. Bülthoff: Automatic acquisition of exemplar-based representations for recognition from image sequences, CVPR 2001 – Workshop Models vs. Ex. (2001)
78.60 C. Wallraven, H.H. Bülthoff: Object recognition in humans and machines. In: *Object Recognition, Attention and Action*, ed. by N. Osaka, I. Rentschler, I. Biederman (Springer, Tokyo 2007) pp. 89–104
78.61 C. Wallraven, B. Caputo, A.B.A. Graf: Recognition with local features: The kernel recipe, Proc. Int. Conf. Comput. Vis., Vol. 2 (2003) pp. 257–264
78.62 T.C. Kietzmann, S. Lange, M. Riedmiller: Computational object recognition: A biologically motivated approach, Biol. Cybern. **100**, 59–79 (2009)
78.63 F.N. Newell, M.O. Ernst, B.S. Tjan, H.H. Bülthoff: Viewpoint dependence in visual and haptic object recognition, Psychol. Sci. **12**, 37–42 (2001)
78.64 H. Lee, C. Wallraven: Exploiting object constancy: Effects of active exploration and shape morphing on similarity judgments of novel objects, Exp. Brain Res. **225**(2), 277–289 (2012)
78.65 J.-P. Ewert: Neural mechanisms of prey-catching and avoidance behavior in the toad *Bufo bufo* L, Brain Behav. Evol. **3**, 36–56 (1970)
78.66 G. Johansson: Visual perception of biological motion and a model for its analysis, Percept. Psychophys. **14**, 201–211 (1973)
78.67 K. Verfaillie: Perceiving human locomotion: Priming effects in direction discrimination, Brain Cogn. **44**, 192–213 (2000)
78.68 A.J. O'Toole, D.A. Roark, H.H. Abdi: Recognizing moving faces: A psychological and neural synthesis, Trends Cogn. Sci. **6**, 261–266 (2002)
78.69 D. Perrett, A. Puce: Electrophysiology and brain imaging of biological motion, Philos. Trans. R. Soc. B **358**, 435–445 (2003)
78.70 R. Blake, M. Shiffrar: Perception of human motion, Annu. Rev. Psychol. **58**, 47–73 (2007)
78.71 D.D. Hoffman, B.E. Flinchbaugh: The interpretation of biological motion, Biol. Cybern. **42**, 195–204 (1982)
78.72 J.A. Webb, J.K. Aggarwal: Structure from motion of rigid and jointed objects, Artif. Intell. **19**, 107–130 (1982)
78.73 M.A. Giese, T.T. Poggio: Neural mechanisms for the recognition of biological movements, Nat. Rev. Neurosci. **4**, 179–192 (2003)
78.74 J. Jastorff, G.A. Orban: Human functional magnetic resonance imaging reveals separation and integration of shape and motion cues in biological motion processing, J. Neurosci. **29**(22), 7315–7329 (2009)
78.75 G. Rizzolatti, L. Craighero: The mirror-neuron system, Annu. Rev. Neurosci. **27**, 169–192 (2004)
78.76 V. Caggiano, L. Fogassi, G. Rizzolatti, J.K. Pomper, P. Thier, M.A. Giese, A. Casile: View-based encoding of actions in mirror neurons of area f5 in macaque premotor cortex, Curr. Biol. **21**(2), 144–148 (2011)
78.77 F. Simion, E. Di Giorgio, I. Leo, L. Bardi: The processing of social stimuli in early infancy: From faces to biological motion perception, Prog. Brain Res. **189**, 173–193 (2011)
78.78 E.D. Grossman, R. Blake, C.Y. Kim: Learning to see biological motion: Brain activity parallels behavior, J. Cogn. Neurosci. **16**, 1669–1679 (2004)
78.79 J. Jastorff, Z. Kourtzi, M.A. Giese: Learning to discriminate complex movements: Biological versus artificial trajectories, J. Vis. **6**, 791–804 (2006)
78.80 J. Jastorff, Z. Kourtzi, M.A. Giese: Visual learning shapes the processing of complex movement stimuli in the human brain, J. Neurosci. **29**(44), 14026–14038 (2009)
78.81 H. Hill, F.E. Pollick: Exaggerating temporal differences enhances recognition of individuals from point light displays, Psychol. Sci. **11**, 223–228 (2000)
78.82 B. Knappmeyer, I.M. Thornton, H.H. Bülthoff: The use of facial motion and facial form during the processing of identity, Vis. Res. **43**, 1921–1936 (2003)
78.83 J. Lee, W. Wong: A stochastic model of coherent motion detection, Biol. Cybern. **91**, 306–314 (2004)
78.84 J. Lange, M. Lappe: A model of biological motion perception from configural form cues, J. Neurosci. **26**(11), 2894–2906 (2006)
78.85 G. Tessitore, F. Donnarumma, R. Prevete: An action-tuned neural network architecture for hand pose estimation, Proc. Int. Conf. Fuzzy Comput. Int. Conf. Neural Comput. Valencia (2010) pp. 358–363
78.86 G. Metta, G. Sandini, L. Natale, L. Craighero, L. Fadiga: Understanding mirror neurons – A bio-robotic approach, Interact. Stud. **7**, 197–232 (2006)

78.87 H. Jhuang, T. Serre, L. Wolf, T. Poggio: A biologically inspired system for action recognition, IEEE Int. Conf. Comput. Vis. (ICCV) (2007) pp. 1–18

78.88 M.J. Escobar, G.S. Masson, T. Vieville, P. Kornprobst: Action recognition using a bio-inspired feedforward spiking network, Int. J. Comput. Vis. **82**(3), 284–301 (2009)

78.89 H. Jhuang, E. Garrote, J. Mutch, T. Poggio, A. Steele, T. Serre: Automated home-cage behavioral phenotyping of mice, Nat. Commun. **1**(86), 1–9 (2010)

78.90 D.M. Wolpert, K. Doya, M. Kawato: A unifying computational framework for motor control and social interaction, Philos. Trans. R. Soc. B **358**, 593–602 (2003)

78.91 Y. Demiris, M. Johnson: Distributed, predictive perception of actions: A biologically inspired robotics architecture for imitation and learning, Connect. Sci. **15**(4), 231–243 (2003)

78.92 J.M. Kilner, K.J. Friston, C.D. Frith: The mirror-neuron system: A Bayesian perspective, Neuroreport **18**, 619–623 (2007)

78.93 K. Friston, J. Mattout, J. Kilner: Action understanding and active inference, Biol. Cybern. **104**(1/2), 137–160 (2011)

78.94 R. Li, T.P. Tian, S. Sclaroff, M.H. Yang: 3D human motion tracking with a coordinated mixture of factor analyzers, Int. J. Comput. Vis. **87**, 170–190 (2010)

78.95 D.R. Weinland, R. Ronfard, E. Boyer: A survey of vision-based methods for action representation. Segmentation and recognition, Comput. Vis. Image Underst. **115**(2), 224–241 (2011)

78.96 F. Fleischer, V. Caggiano, P. Thier, M.A. Giese: Physiologically inspired model for the visual recognition of transitive hand actions, J. Neurosci. **33**, 6563–6580 (2013)

78.97 E. Salinas, L.F. Abbott: Transfer of coded information from sensory to motor networks, J. Neurosci. **15**, 6461–6474 (1995)

78.98 A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, L. Fei-Fei: Large-scale video classification with convolutional neural networks, Proc. 2014 IEEE Conf. Computer Vision and Pattern Recognition, New York (2014) pp. 1725–1732

78.99 C. Breazeal: *Designing Sociable Robots* (MIT Press, Cambridge 2002)

78.100 M. Nusseck, D.W. Cunningham, C. Wallraven, H.H. Bülthoff: The contribution of different facial regions to the recognition of conversational expressions, J. Vis. **8**(8), 1 (2008)

78.101 J.K. Lee, C. Breazeal: Human social response toward humanoid robot's head and facial features, Proc. CHI 2010 (2010) pp. 4237–4242

78.102 D. Hanson: Exploring the aesthetic range for humanoid robots, CogSci-2006 Workshop: Toward Soc. Mech. Android Sci. (2006)

78.103 H. Ishiguro: Understanding humans by building androids, Proc. SIGDIAL Conf. (2010)

78.104 P. Jaeckel, N. Campbell, C. Melhuish: Facial behaviour mapping – From video footage to a robot head, Robotics Auton. Syst. **56**(12), 1042–1049 (2008)

78.105 G. Metta, G. Sandini, D. Vernon, L. Natale, F. Nori: The iCub humanoid robot: An open platform for research in embodied cognition, Proc. 8th Workshop Perform. Metr. Intell. Syst. (2008) pp. 50–56

78.106 C. Wallraven, M. Breidt, D.W. Cunningham, H.H. Bülthoff: Evaluating the perceptual realism of animated facial expressions, ACM Trans. Appl. Percept. **4**(4), 1–20 (2008)

78.107 K.F. MacDorman: Subjective ratings of robot video clips for human likeness, familiarity, and eeriness: An exploration of the uncanny valley, ICCS/CogSci-2006 Symp. Toward Soc. Mech. Android Sci. (2006) pp. 26–29

78.108 C. Ho, K.F. MacDorman, Z.A.D. Pramono: Human emotion and the uncanny valley: A GLM, MDS, and isomap analysis of robot video ratings, Proc. HRI 2008 (2008) pp. 169–176

78.109 F. Delaunay, J. de Greeff, T. Belpaeme: Towards retro-projected robot faces: An alternative to mechatronic and android faces, Proc. 18th IEEE Int. Symp. Robot Human Interact. Commun. RO-MAN (2009) pp. 306–311

78.110 T. Kuratate, M. Riley, B. Pierce, G. Cheng: Gender identification bias induced with texture images on a life size retro-projected face screen, Proc. 21st IEEE Int. Symp. Robot Human Interact. Commun. RO-MAN 2012 (2012) pp. 43–48

78.111 A.P. Saygin, T. Chaminade, H. Ishiguro, J. Driver, C. Frith: The thing that should not be: Predictive coding and the uncanny valley in perceiving human and humanoid robot actions, Soc. Cogn. Affect. Neurosci. **7**(4), 413–422 (2012)

78.112 P.S.A. Reitsma, N.S. Pollard: Perceptual metrics for character animation: Sensitivity to errors in ballistic motion, ACM SIGGRAPH 2003 Papers (SIGGRAPH '03) (ACM, New York 2003) pp. 537–542

78.113 T. Ezzat, G. Geiger, T. Poggio: Trainable videorealistic speech animation, Proc. 29th Annu. Conf. Comput. Gr. Interact. Techn. (SIGGRAPH '02) (ACM, New York 2002) pp. 388–398

78.114 J. Wang, B. Bodenheimer: Synthesis and evaluation of linear motion transitions, ACM Trans. Graph. **27**(1), Article 1 (2008)

78.115 M. Candidi, C. Urgesi, S. Ionta, S.M. Aglioti: Virtual lesion of ventral premotor cortex impairs visual perception of biomechanically possible but not impossible actions, Soc. Neurosci. **3**(3/4), 388–400 (2008)

78.116 T. Chaminade, J. Hodgins, M. Kawato: Anthropomorphism influences perception of computer-animated characters' actions, Soc. Cogn. Affect. Neurosci. **2**(3), 206–216 (2007)

78.117 T. Chaminade, M. Zecca, S.J. Blakemore, A. Takanishi, C.D. Frith, S. Micera, P. Dario, G. Rizzolatti, V. Gallese, M.A. Umiltà: Brain response to a humanoid robot in areas implicated in the perception of human emotional gestures, PLoS ONE **5**(7), e11577 (2010)

78.118 Y.F. Tai, C. Scherfler, D.J. Brooks, N. Sawamoto, U. Castiello: The human premotor cortex is 'mir-

ror' only for biological actions, Curr. Biol. **14**, 117–120 (2004)

78.119 L.M. Oberman, J.P. McCleery, V.S. Ramachandran, J.A. Pineda: EEG evidence for mirror neuron activity during the observation of human and robot actions: Toward an analysis of the human qualities of interactive robots, Neurocomput. **70**, 2194–2203 (2007)

78.120 C. Press, H. Gillmeister, C. Heyes: Sensorimotor experience enhances automatic imitation of robotic action, Proc. Biol. Sci. **274**(1625), 2509–2514 (2007)

78.121 C.L. Colby: Action-oriented spatial reference frames in cortex, Neuron **20**, 15–24 (1998)

78.122 A.R. Kilgour, R. Kitada, P. Servos, T.W. James, S.J. Lederman: Haptic face identification activates ventral occipital and temporal areas: An fMRI study, Brain Cogn. **59**, 246–257 (2005)

Multimedia Contents

79. Robotics for Education

David P. Miller, Illah Nourbakhsh

Educational robotics programs have become popular in most developed countries and are becoming more and more prevalent in the developing world as well. Robotics is used to teach problem solving, programming, design, physics, math and even music and art to students at all levels of their education. This chapter provides an overview of some of the major robotics programs along with the robot platforms and the programming environments commonly used. Like robot systems used in research, there is a constant development and upgrade of hardware and software – so this chapter provides a snapshot of the technologies being used at this time. The chapter concludes with a review of the assessment strategies that can be used to determine if a particular robotics program is benefitting students in the intended ways.

79.1 **The Role of Robots in Education** 2116
79.2 **Educational Robot Tournaments** 2117
79.2.1 Tournaments for Education 2117
79.2.2 Origins 2118
79.2.3 A Taxonomy of Robot Tournaments 2118
79.2.4 The Entertainment Link 2119
79.2.5 Tournaments for Education 2119
79.3 **Education Robot Platforms** 2120
79.3.1 Ground Robots 2121
79.3.2 Sea 2122
79.3.3 Air 2122
79.4 **Education Robot Controllers and Programming Environments** 2123
79.4.1 Robot Controllers 2123
79.4.2 Educational Robotics Programming Environments 2126
79.5 **Robotic Technologies for Student Learning** 2127
79.5.1 Cross-Disciplinary Integration of Robotics 2127
79.5.2 Robotics for Computational Thinking 2127
79.5.3 Early Childhood Robotics 2128
79.6 **Educational Evaluation of Robot Programs** 2129
79.6.1 Design-Time Assessment 2129
79.6.2 Formative and Summative Evaluation 2129
79.7 **Conclusions and Further Reading** 2131
Video-References 2131
References 2131

In recent years, robots have penetrated the education market, both as tools to motivate students to explore science technology engineering and mathematics (STEM) disciplines and as concrete deployments as curriculum materials for teaching content; typically in math, physics, computer science, or engineering. The newest education robots share the notable feature of human–robot interaction with the general public. In contrast to a space exploration robot or a nuclear cleanup robot, these newest, interactive robots do not function solely with the well-trained specialist. They interact with individuals and groups with the goal of inspiring learning amongst students of all ages.

This chapter provides an overview of the key ingredients that make successful education robots possible. One very popular outlet for public interaction with educational robots is the robot tournament. Section 79.2 provides a survey of the very popular world of robot-themed tournaments, which have already impacted tens

of thousands of students across diverse geographic and age group boundaries [79.1–5].

To make educational robots successful, a new level of technology robustness and standardization is required, and significant progress has been made on this front in the past decade. Educational robot devices consist of both hardware (preassembled or as kits or components) and software (both as source code and programming environments). Section 79.3 discusses physical robot platforms that have achieved notable success, while Sect. 79.4 describes both low-level controllers that interface those platforms to high-level computation, as well as the top-level programming environments themselves. The technology in both these areas is evolving rapidly pushed not only by educational needs but also by a burgeoning hobbyist market.

Finally, an important class of tool in the study and execution of educational robotic systems is the ability to formally evaluate the efficacy of a robot system in an educational context. Numerous tools from human–computer interaction, cognitive psychology, and education have demonstrated usefulness in this regard. Section 79.6 summarizes the manner in which conventional analytical tools may be used to evaluate unconventional educational programs that tap robotic technologies as learning tools across a variety of ages and in both formal and informal learning venues.

79.1 The Role of Robots in Education

Education is itself a broad area of application for robotics, with a rich history that started at least with notions of active learning disseminated by Seymour Papert and others [79.6]. Karel the Robot still serves as a metaphor for concrete activity in numerous Introductory Programming courses. Karel the Robot is merely an on-screen simulation of a two-dimensional robotic world. This first of three roles that robots can play in the educational setting is *robot as programming project*. The robotic system is a focus of problem set assignments, a black box to be programmed in order to create a concrete physical manifestation of the art of computer programming. Numerous studies that have shown that poor engagement and retention statistics in courses such as Introduction to Programming (CS1) often arise from student's inability to see how the skills they learn can have a concrete impact on what they care about: their physical world; their friends and family. Physical robot programming projects can cast coding problems into the real world, making those skills push back on one's world and thereby achieve a level of significance and engagement that, for instance, computation of the Fibonacci Sequence on a computer screen cannot inspire [79.7–15].

The second role for educational robotics is that of *robot as learning focus*. Mechatronics-style courses can focus on the creation and use of a physical robot as a goal in and of itself. Robots do this by stimulating general interest in science, technology and engineering, and in thereby showing students that they can have an active role in shaping technology in their future. Such an application-oriented class has great benefits in cases where students learn best through integrative, project-oriented learning, and such classes have demonstrated significant lifelong-learning results in areas such as teamwork, problem-solving, and self-identification with technology-focused careers [79.6, 16–20]. In the United States, the National Academy of Engineering has identified trends reporting dropping technology literacy as troubling phenomena throughout the nation. Programs that re-connect students to complex mechatronic systems, providing the self-empowerment for lifelong, active relationships to technology artifacts, serve as one possible avenue stemming this dangerous trend away from knowledge about the technology artifacts on which society depends so completely. Demonstrated learning themes in educational analyses include the following:

- *Interest in science and engineering*: The study of robotics can be sufficiently rewarding, intellectually and emotionally, as to generate general enthusiasm for the follow-on study of science and engineering. Some educational robotics programs have shown promise in increasing retention rates among female students [79.21, 22], who are under-represented in technology-focused fields.
- *Teamwork*: An important result of team-based, project-based robotic building experiences is that those with low self-esteem in their technology capability, teamwork skills, and communication skills can find measured improvement, leading to mitigation of fear toward working in teams on high-technology ventures [79.18, 20].
- *Problem-solving*: The study of a systems engineering phenomena – a complex robot system – provides lessons in hierarchy, risk mitigation, planning, and diagnosis that far outweighs the benefits of silo-driven, single-topic explorations of debugging and problem analysis. The resulting problem

solving and decomposition skills have lifelong applicability to any complex system [79.6].

The third role for educational robots is that of *robot as learning collaborator*. In this case, students are not designing robots, but rather conducting inquiry during which a high-functioning robot can serve as an all-season companion, aide, and even intellectual foil [79.23, 24]. Robots that have social capabilities are well-suited in this regard, and this is an area where the potential for application greatly outweighs what has, so far, been done. There are two reasons why such a social role for robots in education holds particular promise. First, because of their novelty, robots can inspire dedication in learning, just as it is possible in the case of the above two roles. Second, students have little preconceived expectations for the behavior of robots. In therapeutic cases, such as students with autism spectrum disorder (ASD), the robot can play an important role as studies have shown those students disproportionate interest in mechanical systems [79.23, 25]. In all cases, the interaction can itself be a form of discovery and engagement quite different from traditional, more staid educational methods, and thus this can even border on the application area of entertainment robotics.

79.2 Educational Robot Tournaments

Competition catches people's attention and makes them want to get involved. This is true in almost all fields of endeavor. In education, competitions are commonplace from spelling bees to science fairs. Over the past 20 years, robotic competitions have also come into their own; garnering the level of attention from students, teachers, and parents previously found almost exclusively in more traditional sports.

In this section, we provide a brief history of robot tournaments, look at some of the popular current tournaments, and overview how they are being used to supplement standard educational curricula.

79.2.1 Tournaments for Education

Professional organizations such as AAAI (Association for the Advancement of Artificial Intelligence), IEEE (Institute for Electrical and Electronics Engineers), and others often hold tournaments for their student members at their major conferences. For those organizations, the purpose of the tournament is to attract students to attend the conference and become active in the organizations. The activity of building the robot also sharpens the students' skills and gets them excited about technologies and businesses linked to that organization and motivates them to pursue relevant degrees [79.26]. Our focus in this chapter is on equipment and activities targeted at primary and secondary education students. While significant overlap exists, collegiate research oriented platforms and programs are covered in other chapters.

Many universities and colleges use robot tournaments as part of their curricula. Some create class tournaments (MIT 6.270) or use standard games such as *KIPR Open* (KISS Institute for Practical Robotics) [79.27] that are distributed for that purpose. Others use some of the tournaments mentioned in Table 79.1 as projects for their curricula [79.28].

But the bulk or robot tournaments are aimed at middle and high school students. Here the motivation is not so much to teach detailed skills as to spark creativity while at the same time emphasizing basic concepts and an appreciation for the fundamentals of science, technology, education, and math [79.29]. Some tournaments have added activities such as documentation, presentation, reports, and even tests [79.30] to reinforce and expand upon these fundamentals (Fig. 79.1). To assist educators in guiding the students, many tournaments

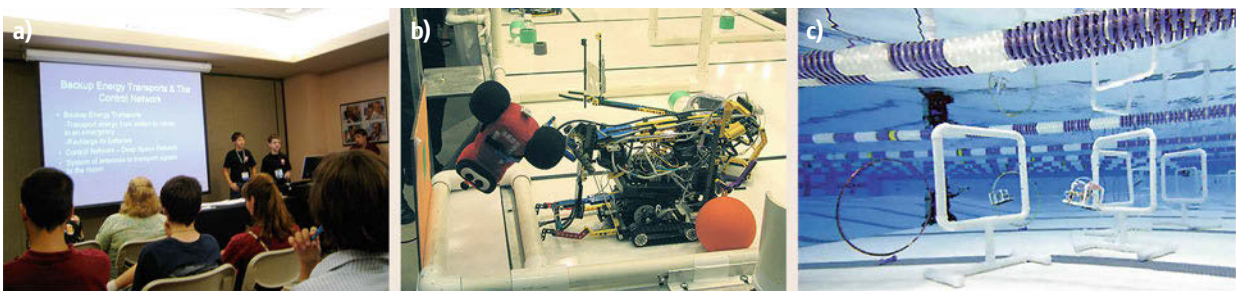


Fig. 79.1 (a) Middle-school students present their paper at the Global Conference on Educational Robotics and then (b) have their autonomous machine-vision guided Botball robots score points in the tournament. (c) The SeaPerch underwater robot contest

have curriculum materials linked to their activities. The National Aeronautics and Space Administration (NASA) has collected links to many of these materials and made them available on the internet [79.31]. Evaluating the effectiveness of these materials and programs in meeting their educational goals is the subject of Sect. 79.6.

79.2.2 Origins

In 1970, Professor H.H. Richardson adopted the Massachusetts Institute of Technology (IT)'s Introduction to Design Course (course 2.70), in the Mechanical Engineering department, and transformed it into a design contest. He had the students work in teams to design and create machines that would be operated by the students to compete against one another. As the course evolved over the years, and under new instructors, it became both popular and famous.

Several students in the Electrical Engineering and Computer Science department at MIT, envious of the mechanical engeneering (ME) course decided to create their own, more autonomous version of both the course and the contest. Since 1987, MIT students have held the 6.270 course during the intercession between the fall and spring semesters. While initially done in simulation, by 1991 the contest involved untethered autonomous robots in a head to head elimination tournament [79.32]. While only open to MIT students and, with the exception of the occasional documentary, poorly publicized, these two courses have become the models for many teleoperated machine contests and robot contests around the world.

79.2.3 A Taxonomy of Robot Tournaments

There are many possible ways to distinguish the multitude of activities often lumped together as robot contests. For our purposes, here we will use the following qualities.

Autonomy

The level of autonomous action that robots exhibit during a tournament varies widely. Some tournaments are completely radio controlled (e.g., boosting engineering science and technology (*BEST*) [79.33]) with no independent actions or decisions being undertaken by the *robots* in the tournament. In other contests, the machines are teleoperated – still under real-time control by their operators, but doing some processing on the commands, and sometimes mixing in feedback from onboard sensors (e.g., *FIRST* (for inspiration and recognition of science and technology) [79.5]). Autonomous robot contests (e.g., *Botball* [79.1]) have little or no interaction by the robot operators. But even in completely autonomous robot contests, there is a spectrum. Some autonomous robots simply replay a canned set of actions – operating completely open loop. Others control all of their actions based on feedback from onboard sensors, and still others *learn* as they go, modifying not only their actions, but their internal programming as well. One generalization that can be made with regards to robot tournaments and education: A contest that is more autonomous usually has more educational content in the areas of software/programming, while contests with little or no autonomy emphasize on the mechanical design and physical implementation aspects of robotics.

Performance Versus Opposition

Some robotics tournaments are based on ranking the absolute performance of the robot against the course. Other tournaments have contestants ranked based on their triumph over a series of specific opponents. In the latter case, it does not matter how well the victorious robot performs in a given round, as long as it performs better than its opponent for that round. Trinity College's *Firefighting Robot Contest* (TCFFHRC) [79.34] ranks teams by how quickly they can carry out the task (locating and extinguishing a candle) and is an example of a performance-based tournament. The scoring in *RoboCup Junior Dance* [79.3] is more subjective, but is still based on individual performance against an absolute scale. Sumo contests such as [79.35] are opposition-based tournaments sometimes known as head-to-head. The performance-based contests allow the robot designers to implement more intricate strategies, since the environment is more predictable than if there were an opposing robot running amok through the course. In oppositional tournaments, robots must be designed to cope with the actions of their opponents. Both types of tournaments have their merits and each allows detailed problem solving. In theory, most people find head-to-head tournaments more exciting and entertaining. However, as with traditional human sumo, head-to-head tournaments often involve head to head collisions followed by lengthy, and less exciting, periods of robots trying to push or disentangle themselves from their opponents.

Heritage: Fixed Game Versus New Game Every Year

Is it more educational for students to examine how others have tried to solve a problem and then build on that to create their own incremental solution? Or is it better for students to tackle something new, without extensive examination of precedent? The educational community does not have a definitive answer, but different robot tournaments can be used to support the

Table 79.1 Examples of different types of robot tournaments

Tournament	Web	Autonomy	Performance/Opposition	Heritage	Notes
BEST	[79.33]	radio controlled (RC)	Opp.	New	
Botball	[79.1]	Autonomous	Perf. & Opp.	New	Multiple robots per entry
FIRST	[79.5]	Mostly teleoperated	Opp.	New	Students teamed with engineers
Micro Maze	[79.36]	RC to autonomous	Perf.	Fixed	Cubic cm robots
Seaperch	[79.37]	RC	Perf.	New	Underwater remotely operated vehicles (ROVs)
Sumo	[79.35]	RC to autonomous	Opp.	Fixed	Divisions based on weight & autonomy
RoboCup	[79.38]	Autonomous	Opp.	Fixed	Teams of robots
RoboCup Jr. Dance	[79.3]	Autonomous	Perf.	N/A	Very free form
TCFFHRC	[79.34]	Autonomous	Perf.	Fixed	Random placement of fire
VEX	[79.39]	Mostly teleoperated	Opp.	New	

educational methodology of choice. Tournaments such as *Robocup* [79.38] are prime example of a high heritage tournament. In addition to the game and rules being well known (derived from soccer) and essentially remaining unchanged from year to year, there are a number of annual publications [79.40] that detail successful robots and techniques used in that year's contests. *Robocup* is a study in steady incremental improvement. Other contests, such as *Botball* Robot Tournament, use a different game every year. One of the motivations for doing this is so that established teams obtain the game rules at the same time as new teams – so whether experienced or not, everyone has to build a new system for the new task, and has the same amount of time to do so [79.4]. As with all of these dimensions, there is no clear advantage of one methodology over the other. Some educators prefer to have a new activity every year to help keep themselves and their students engaged. Others prefer having the same game every year, with the accompanying wealth of lessons-learned and how-to guides that process makes available.

Table 79.1 presents a number of established robot tournaments using the taxonomy. This table is not a complete listing. Readers interested in finding out more information about a particular tournament should consult the cited website. All of the tournaments below are for students – though the age eligibility and level of adult involvement varies widely between events.

79.2.4 The Entertainment Link

Tournaments are used (rather than a course assignment or demonstration) in part because of their entertainment value. A tournament is hopefully engaging both the participants and the audience. However, the entertainment value of a tournament may not be tied to the technical sophistication of the robots. *Battlebots* and its variations have been very successful as mass market entertainment. These robots typically are radio controlled, and while sometimes mechanically sophisticated, they are computationally simplistic. The *AAAI mobile robot contest* [79.41] usually has highly computationally sophisticated robots – however that sophistication usually has an accompanying decrease in audience entertainment value. AAAI robots strive to do what everyday humans easily do – go to specific places, manipulate household objects, give talks, and answer questions; but the robots tend to be much slower than humans performing these activities. *Battlebots* do what humans are not allowed to do – use power tools and blunt instruments to demolish their opponents. In robot tournaments as in the movie business, violence is often more entertaining than demonstrations of intellectual ability and human activity.

79.2.5 Tournaments for Education

All robotics tournaments can be educational – but that is not necessarily the emphasis of an event. Some programs are designed to be more inspirational than educational. FIRST [79.5] pairs schools with groups of professional engineers – but does not define how the work is to be divided between the professionals and the students. In some teams, the students are primarily observers getting to see professionals design professional quality machines. Other teams have the students play a more active or even leading role – though no distinction is made between these different approaches during the competition.

Other programs, e.g., Botball [79.1] and RoboCup Jr. [79.3], require that the design, building, and programming is all done by the students. In both these programs, it is what the students can do that matters. RoboCup Jr. consists of three different competitions: Rescue, Dance, and Soccer; teams pick in which of the three they will compete. All three of the competitions are open with regards to the specific technologies to be used. There are restrictions on size, power, and the acceptable level of off the shelf hardware and software, yet the potential options are still very large. This has

the benefits of allowing teams to use the equipment they already have at their schools or can easily afford to purchase and to program in the languages they have been learning in class. It does, however, make it more difficult to create a detailed curriculum or to even define what the learning objectives for the participants should be.

The Botball program specifies what equipment is to be used and provides a detailed curriculum for teams, especially new teams, to follow in order to make sure certain learning objectives are likely to be met. Specifying equipment requires teams to have a certain expense, but removes much of the advantage of teams that have stockpiled equipment from previous events or who have large financial resources – all participants are allowed to build their entries from the same set of hardware – extra pieces are not allowed. Because of the standardize kit, lesson plans and tutorials can be part of the program. Botball teams turn in documentation at specified dates during the build cycle to demonstrate certain skills and also help in keeping the teams on track for having a working entry.

The educational benefits of robotics tournaments can be greatly enhanced when the tournament is part of an educationally oriented conference. These can take many forms. The *RobotChallenge* [79.42] is an annual multiday international meeting that hosts 15 different robot competitions. While it does not contain a specific educational program, the diverse set of challenges combined with the many opportunities to interact wilt robotics enthusiasts in all areas of robotics makes this a beneficial event for all participants.

The Micro Robot Maze Contest (MAZE) [79.36] is an annual tournament held in Nagoya Japan, in parallel with the International Symposium on Micro Mechatronics and Human Science (MHS). Rounds of the tournament are interspersed with technical talks by conference attendees allowing students participating in the contest to be exposed to leading researchers in micro mechatronics.

The Global Conference on Educational Robotics (GCER) [79.43, 44], and its European specific counterpart ECER [79.45], are multiday conferences hosting global and regional (respectively) Botball tournaments, other robotics tournaments and a variety of technical events. Like the MAZE event, student attendees are exposed to leading researchers in robotics through a series of keynote talks. In addition, there are several sessions of technical talks presented by the student attendees about their own research related to or independent of the tournament events (Fig. 79.1). Session talks on educational issues and teaching techniques are given by teachers and education researchers. These events not only give students exposure to a wide variety of technical issues, but also a chance for the students to publish and present their own research in robotics.

The varieties of approach of the different tournaments and of the conference events that surround many of the tournaments presents the robotics student with a wealth of choices and opportunities. As with most things, students can get as much out of their robotics tournament experience as they are willing to put in.

79.3 Education Robot Platforms

Nonindustrial robotics platforms have gone through radical changes in the past decade. While many research institutions and universities continue to develop custom platforms (e.g., Honda's *Asimo* [79.46]) or low production quantity research platforms (e.g., the *iCub* [79.47] or Willow Garage's *PR2* [79.48]), the renaissance has been in the availability of relatively low cost kits and platforms whose capabilities challenge the expensive research robotics of just a few years old. These platforms and kits, a sampling of which is discussed below, are becoming widely used in primary, secondary, and higher level education throughout the world.

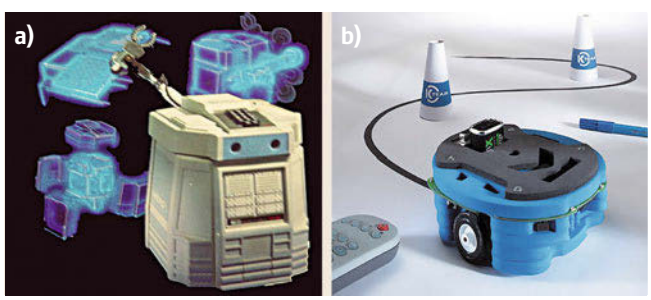


Fig. 79.2 (a) Hero-1, (b) Hemisson

Kit platforms are inexpensive, exist in large numbers, and are widely used in schools and by hobbyists. Most of the robot kits today either assume the addition of a separate processing element (either a PC (personal computer) or a specialty robot controller) or are built on one of two families of robot microcontrollers: Parallax Stamp/Propeller or the Arduino. The remainder of this section will discuss popular kits and mobility platforms currently in use. An exciting addition to edu-

cational robotics has been the recent addition of aerial and underwater robots in a sufficiently low-cost manner that opens the possibilities for large scale educational use. Some of the new platforms in these areas will also be discussed.

79.3.1 Ground Robots

Education robots really got their start in the early 1980s with the introduction of the Heathkit *Hero-1* (Fig. 79.2a). The Hero robots were sold as kits, both to keep the price down and to encourage users to learn how robots were made. However, while detailed assembly instructions were included, no information on the theory or principles behind the assembly were given; the kits, by themselves, did not provide a satisfactory educational experience. Unfortunately, neither the Hero robots, nor several other similarly sized and priced personal robots that came out in the 1980s never reached a level of capability and ease of use to attract an economically viable customer base. Heathkit and the other companies have since gone out of business.

K-Team is one of the few research robot companies making low-cost robot platforms specifically for the education market. The *Hemisson* (Fig. 79.2b) is a low-cost, less compact version of K-Team's Khepera research robot. It has reduced computational power and only few sensors when compared to most research robots, but is priced and designed for robotics classes in secondary and university classes.

In 2007, iRobot announced the Create, an educational platform based on their Roomba series of robot vacuum cleaners. The Create is a sturdy mobility platform that has TTL serial connections to allow easy control from an external processor and a small amount of memory to allow simple scripts to be downloaded and executed directly by the platform itself. The Create contains a differential drive system, with encoders on each wheel, two instrumented bumpers, four reflective cliff sensors, a side reflectance sensor, and numerous other internal sensors along with some digital and analog input output (I/O). The platform also contains some hard mounting points allowing other structures to be easily bolted on (Fig. 79.3).

Another educational platform, the Scribbler S2 [79.49] has similar functionality as the Create but in a smaller package. The S2 is not designed to be built upon but is more for initial experimentation and contains a pen hole that allows a drawing functionality reminiscent of the LOGO Turtle [79.50].

The Boe-Bot is a set of mobile robot kits that can be combined or switched to create wheeled, treaded and walking robots (Fig. 79.4). Boe-bot, like the S2, is based around the Parallax series of processors (Stamp and Propeller). However, the Arduino controller can also be used with Boe-Bot kits.

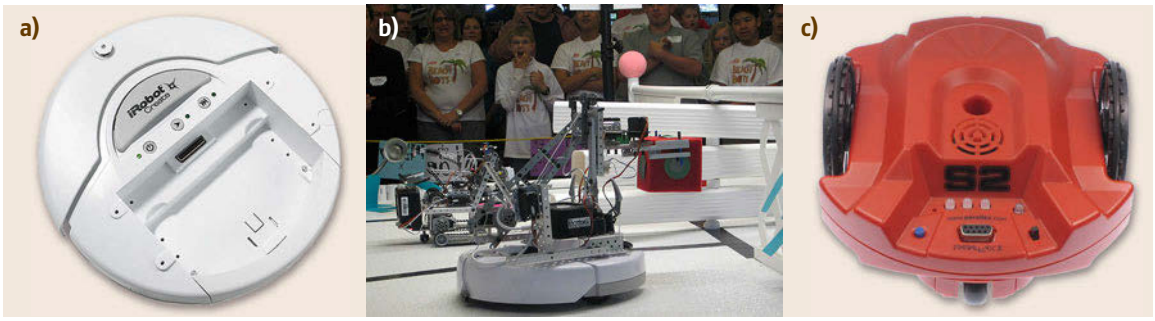


Fig. 79.3 (a) Create, (b) Create with a gripper mounted, (c) the Scribbler S2

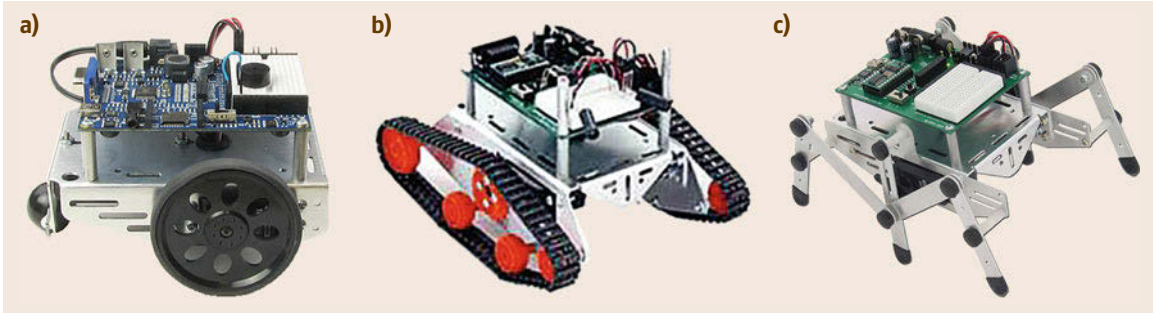


Fig.79.4a–c Different configurations for the Boe-bot series of robot kits (a) Wheeled, (b) tracked; and (c) legged

In addition to set robot kits and prebuilt chassis, the most common mobile robot platforms for education are built from sets of parts that originated as mechanical building systems. Manufacturers including VEX, KIPR and of course Lego create sets of metal and plastic pieces, along with motors and sensors allowing a wide range of mobile and fixed robot systems to be created. These manufacturers also produce specialize robot controllers, though most of the mechanical elements are independent of the controller being used.

Similarly, companies such as Lynxmotion, Dagu and Bioloid produce kit parts designed to link together small servo motors. The Lynxmotion and Dagu parts are designed particularly for the construction of manipulators. The Bioloid kits can be used to group large collections of servo motors together in order to form small but functional humanoid robots with a dozen or more degrees of freedom (Fig. 79.5).

79.3.2 Sea

While the vast majority of educational robots are land-based mobile and/or manipulator robot systems, underwater robots and ROVs are becoming a popular way to teach about robotics, and related STEM topics such as climate change and bio-diversity.

Most underwater robots are sophisticated research devices used to study marine biology [79.51], perform underwater archeology [79.52] or monitor ocean health [79.53].

However MBARI (the Monterey Bay Aquarium Research Institute) has released instructions for a build your own ROV [79.54]. The Seaperch organization [79.37] has also released an underwater ROV kit containing all the pieces necessary for building an ROV. These vehicles make use of low-cost materials such as polyvinylchloride (PVC) plumbing fixtures [79.55] as seen in Fig. 79.6. They also use a tether for control and power, simplifying the vehicle design. While lacking the sophistication of the research robots, these relatively simple ROVs can be used to teach many aspects of robotics and also issues of buoyancy and fluid flow – important physics topics are missing from the most these robotics education. These are the types of devices that are used in underwater robot contests produced by Seaperch and Mate [79.56].

79.3.3 Air

Radio controlled aircraft has been an active hobby for over 75 years [79.57]. A variety of contests such as Design Build Fly [79.58] and SpeedFest [79.59] have moved the hobby into the realm of education curricula. It can be debated as to whether or not a radio-controlled fixed wing aircraft is in fact a robot – most of them rely on the human operator to do all of the sensing, there

Fig.79.5a–c Robot arms and manipulators by Lynxmotion (a) and Dagu (b) and a humanoid robot constructed from a Bioloid kit (c)

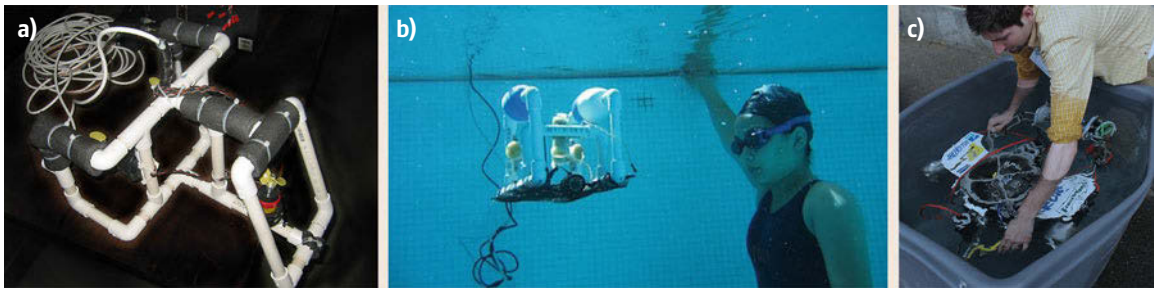


Fig.79.6a–c Underwater robots from kits/instructions by MBARI (a), Seaperch (b) MATE (c)

Fig.79.7a–c The Syma 107 gyro stabilized helicopter **(a)**, and the AR drone Wifi controlled quad rotor **(b)**. A drone knocks down scoring items where they can be gathered by ground robots in the 2012 KIPR Open **(c)**

is no automated control and communication is all one way. However, the same can also be said for some of the ground robotics contests and *robots* as well as many underwater shallow ROVs. Whether or not these aircraft are true robots, the RC (radio control) hobby has pioneered much of the technology now used in educational robots including small servo motors, radio communications, and many of the control interfaces.

In recent years, technology for semi-automated rotorcraft of a variety of configurations has become inexpensive enough to be considered for the educational market. These crafts range from traditional helicopter configurations to the now popular quad-rotor craft (Fig. 79.7b). The key technical advances have been in the integration of small gyros and accelerometers to allow much of the fine control to be done onboard allowing the operator to worry only about the crafts' gross motion.

These inertial sensors have been combined by researchers with precision positioning systems such as a Vicon system to allow aerial acrobatics that rival the capabilities of humming birds and insects [79.60, 61]. Even without the external precision positioning systems, manufacturers have been able to produce some inexpensive platforms capable of good performance (Fig. 79.7). The communications for some of these systems are down via standard Wifi packet or via infrared (IR) encoded commands. It is relatively easy to have computers talk to these platforms, and as a result packages are being put together to use these vehicles to encourage students to learn programming [79.62]. The AR Drone has already been used in autonomous flying vehicle contests organized by KIPR (Fig. 79.7c).

It is expected that aerial robotics platforms will become a major portion of educational robotics programs in the very near future.

79.4 Education Robot Controllers and Programming Environments

Education and entertainment activities in robotics sometimes run at cross purposes. For entertainment, we want robots that are active, very easy to use and to control. Products like the Lego Mindstorms provide both a robot controller and a programming environment that meets those criteria.

However, when it comes to robot education, it is important to remember that in most cases, we are using robots to inspire and motivate students to learn – learn many things of which robotics is just one. Robotics educators, especially at the K-12 level, are hoping to teach students general science, programming, math, and engineering techniques. The toy programming environments used in some products such as Mindstorms, may be inappropriate for this mission.

This section overviews some of the popular robot controllers and programming environments. Usually the two are linked, but there are some environments that work with several different controllers, and vice versa.

As with the robot tournaments described in Sect. 79.2, many of the robot controllers and their programming environments in use for tournaments and by hobbyists can trace their origins back to the MIT 6.270 course [79.32]. At an even more fundamental level, the origin of most robot controller starts with the Motorola 68HC11 processor.

79.4.1 Robot Controllers

The venerable HC11 is a single chip processor with small amounts of onboard flash and RAM (random-access memory). It also has a number of I/O and timing ports that make it relatively easy to interface to sensors and external devices. The 8-bit HC11 uses the same instruction set as the Motorola 6800 series of processors – which have been in existence in one form or another for more than three decades.

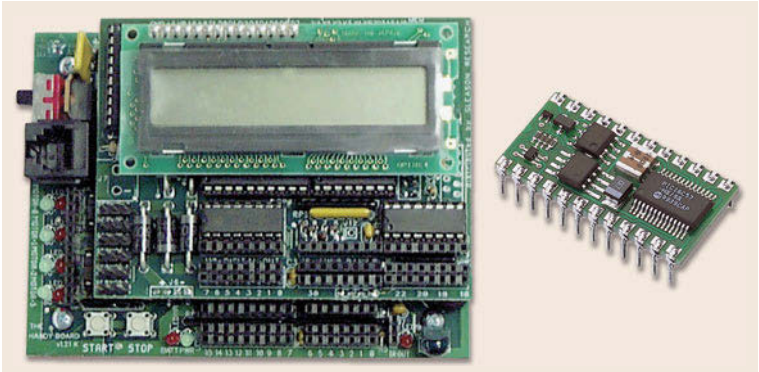


Fig. 79.8 The HandyBoard with its expansion board and the BASIC Stamp

The HC11 was used by Anita Flynn and Joe Jones in a board for the MIT robot talent show in 1991, which spawned the Book Board [79.63] the MIT 6.270 board and Fred Martin's Handyboard. The last of these became the *de-facto* high-end robot controller for hobbyists and autonomous robot contests for most of the 1990s.

The Handyboard (Fig. 79.8) can provide PWM control for four direct current (DC) motors, read 8 digital and 8 analog sensors, control a couple of light-emitting diodes (LEDs) and has a 32 character display. With the expansion board, 8 lines of DIO and control for 6 RC servos is also added. The Handyboard can be programmed using 6811 assembly language and several other languages including Java and C.

The other low-cost commercial robot controller in wide spread use throughout the 1990s was the BASIC Stamp [79.64] (Fig. 79.8). This PIC-based controller could be programmed in a special (often assembly language-like) version of the BASIC language. While very small in memory, and usually requiring supplementary electronics for most sensors and actuators, the Stamp was very popular because of its low cost and flexibility.

In the late 1990s, spurred on in large part by the growing interest in robot tournaments, a renaissance in robot controller hardware started making available a number of controllers ranging from advanced PICs to full blown PCs. Two approaches to robot controllers, and their environments seem to be dominant in the education market.

The first approach is to use a microcontroller that is programmed using a PC and has program compiled on the PC and then downloaded to the microcontroller which can run that program – and little else until new software is downloaded to it. This follows the model originally used in 6.270 and typified by the Handyboard but has been perfected by the Arduino family of robot processors.

Arduino is an open source project started in 2005 in Italy [79.65]. There are more than a dozen official Arduino boards available in a range of sizes and capabilities. The smaller Arduino's are about the size of a 1 Euro coin. Most are based on the ATMega series of processors, but some of the newest are using ARM processors [79.66]. Most of the Arduino boards contain some number of digital input and output pins as well as analog inputs and the capability to generate PWM signals for controlling dc motors or hobby servos. Arduino Shields are accessory boards that can be used to add a variety of capabilities and ale add breadboarding and experimental interfaces (Fig. 79.9).

In contrast to the Arduino family of controllers, the *KIPR Link* [79.67], and its immediate predeces-

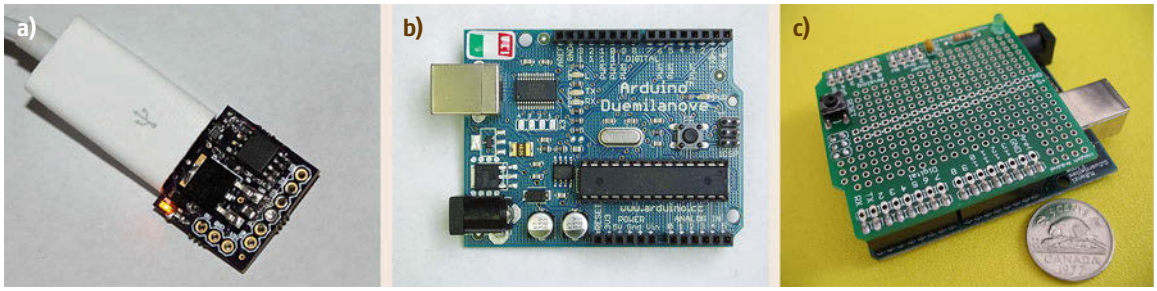


Fig.79.9a–c One of the smaller Arduinos: **(a)** the Digispark; **(b)** the Arduino Duemilanove; **(c)** a Shield compatible with the Duemilanove

sor the *CBC2* [79.68, 69], represent a different style of robot controllers. The Arduinos are inexpensive, small, and pretty barebones. The user needs to supply a power supply, case and often supplemental circuitry, depending on what kinds of mechatronics are being interfaced. Arduinos were originally targeted for the hobbyist and robot enthusiast and indeed their primary role in elementary and secondary education is for the self-motivated student who uses them in their extra-curricular projects or in kits put together by a third-party.

The Link and the CBC2, on the other hand are ready to use units that can be interfaced with a variety of hardware without the need of tools (Fig. 79.10). These controllers are in fact full fledged computers running Linux, but with a GUI (graphical user interface) front end and built-in touch screen allowing them to be used as stand-alone devices, if desired. These controllers have gigabytes of flash storage and host standard file systems. Multiple programs can be downloaded to the Link and the file browser can be used to select which program to run. In fact, the programs for these controllers are compiled and linked on the controller themselves using a standard Linux tool chain. This allows these controllers to run programs in a variety of languages, and for new languages to be added as desired. While the CBC2 and the Link can be connected via USB (universal serial bus) to a laptop or desktop computer as a device, they themselves can be the host to other USB devices including webcams. The Link uses standard libraries such as OpenCV to process the images generated by a web cam to do object recognition, color tracking, QR code decoding, and other standard computer vision processing.

The Link library and touchscreen and GUI allows the user to easily interface and explore robot sensors and effectors in ways not previously available.

Fig. 79.10 The CBC2 and KIPR Link Linux-based robot controllers

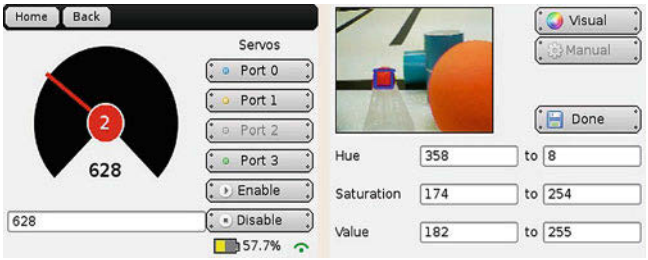


Fig. 79.11 Touch screen widgets on the KIPR link for manipulating a servo, or selecting the hue saturation and value for objects to be tracked using the camera (the numbers are automatically filled in when the image of the object, *red tip* of a badminton birdie, was touched on the screen)

Graphical widgets (Fig. 79.11) that let the user set servo positions, motor speeds, read sensor values using the built in oscilloscope mode, and select vision targets by touching the object in the image on the screen allows even the novice programmer to have their robot do sophisticated activities.

Both the Link and Arduino series of controllers owe much of their success and capabilities to the open source nature of both their hardware and software.

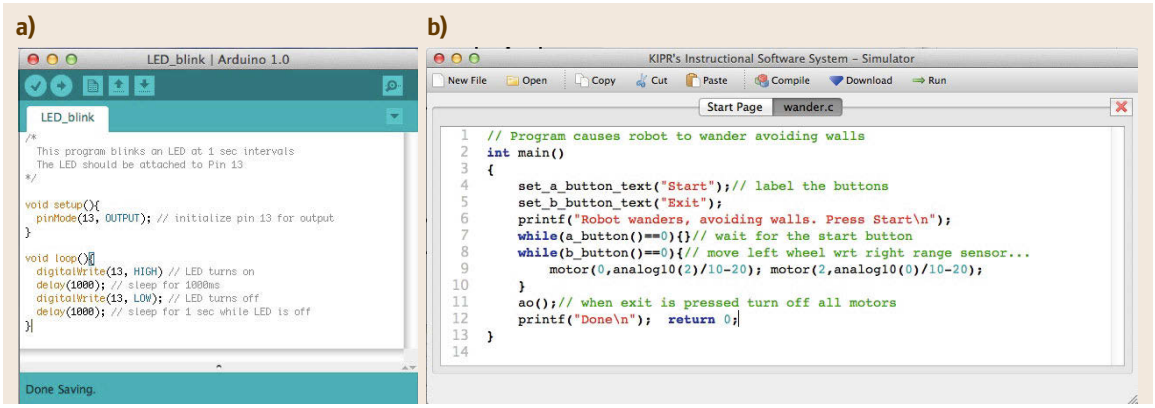


Fig. 79.12 **(a)** A simple program for an Arduino board to blink an LED. **(b)** A program in the KISS editor for a robot avoid walls

Complete schematics for the boards are available to anyone interested as is the code for the libraries and programming environments. There are now several producers of Arduino compatible boards, each with their unique capabilities. Much of the power of the Link system comes from its ability to exploit open source libraries such as OpenCV and also user community created libraries and tools.

79.4.2 Educational Robotics Programming Environments

Arduino boards are programmed using an IDE (integrated development environment), based on the *Processing* and *Wiring* IDEs [79.70, 71]. *Processing* was originally developed to assist artists doing graphical and other creative projects that required coding. The IDE is written in Java and runs on Windows, MacOS, and Linux platforms. The primary language for programming Arduinos is C++ and typical programs use two special functions `setup` and `loop`. The `setup` function is used to initialize variables and ports and the `loop` function is called repeatedly changing the output state of the board based on the current sensor inputs. This model works well for many robot applications (Fig. 79.12).

The Link robot controller is typically programmed using the KISS programming environment. Since the Link compiles its programs onboard, and since code can be loaded via network or via USB drive, or even typed in directly by connecting a keyboard to the Link, many other options for programming environments are possible. The KISS IDE consists of an editor (Fig. 79.12 right), target selector, and a variety of simulated and hardware targets. The editor is a general purpose language sensitive editor that helps to highlight and format code for the user's choice of language (C, C++, Java, Python, etc.). Each program can then be associated with one or more targets such as a particular Link, a Link simulator, or the computer running KISS. This last target choice makes KISS a general purpose IDE suitable for writing applications that can, with recompilation, work on MacOS, Windows and Linux machines and includes cross platform graphics, camera, and serial libraries. The Link simulator (Fig. 79.13) uses the same libraries as the Link itself, and so can be used for solving syntax errors and many logic errors. Calls to the vision routines on the Link activate the webcam on the laptop running the simulator allowing a more thorough testing of robot logic on the simulator than is commonly found without running large (and particularly novice user friendly) simulators such as Gazebo.

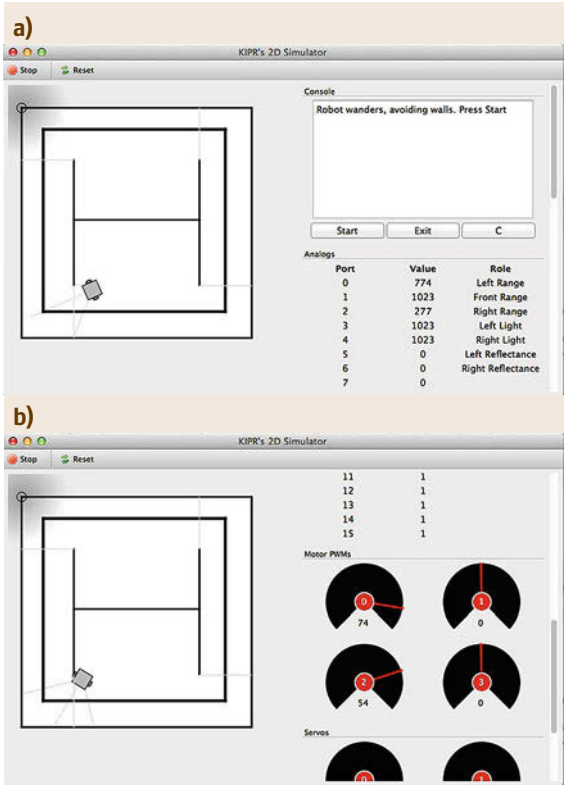


Fig.79.13a,b The Link simulator running the program shown in (Fig. 79.12b). **(a)** The simulator displaying the console and sensor values. **(b)** The simulator showing the motor status

Like their hardware counterparts, the Arduino and KISS programming platforms are open source. Libraries, development environments, and documentation are all improved by their ever growing user communities.

79.5 Robotic Technologies for Student Learning

Exclusive, robotics-oriented curriculum in formal learning faces a formidable challenge globally: while ICT curricula are recognized across countries as an important twenty-first century skill for all learners, continuing educational budget cuts have re-focused capacity building across educational institutions on teachers for the foundation subjects of mathematics and language arts, with scant resources remaining for computer science and even less for new directions such as educational robotics. Furthermore, a major issue of concern in numerous countries involves achievement gaps between diverse populations, and it is arguable that highly focused robotics courses will be particularly effective at engaging and retaining the interest of precisely the most precocious future engineers while failing to win over more left-brained students, further reinforcing and exacerbating an achievement gap through the natural process of self-selection.

79.5.1 Cross-Disciplinary Integration of Robotics

In combating these trends, traditional schools are reconsidering the role of cross-disciplinary and contextual learning, where lessons draw upon knowledge from multiple disciplinary specialties concurrently, teaching lateral thinking and, ideally, motivating the disciplinary learning process by presenting compelling pathways through which knowledge acquired will have meaningful value to a problem at hand [79.72].

A number of efforts in educational robotics concentrate now on this form of multidisciplinary curricula across all ages of learners, and these new directions are likely to be harbingers of many future, successful endeavours to incorporate educational, interactive robotics into schools in ways that pervade the educational experience rather than balkanize learning into ever more narrow categories.

Robotics is particularly apt for integrated learning because of its inherent diversity: robots have physical form and design considerations, computational aspects, sensors with which real-world values can be measured and visualized, effectors and mechanical considerations and even the ability to bridge the digital and physical worlds together. As such, robotics are an inherently cross-disciplinary medium, and this characteristic is being exploited by robotics researchers to design and pilot such integrative curriculum. ArtBotics is one of the many programs combining creative expression with robotics, enabling creative expression whilst using the advanced sensor and effector technologies that make robot-building a technically rewarding experience as well [79.73, 74]. This marriage of creativity and technology, often considered opposites in conventional educational thinking, has the potential to attract and encourage talent from both communities by demonstrating to artists how their creative energy can yield notable success in technology invention, and by showing technically minded students that their invention process can be as creative as it is rational.

Arts & Bots combines craft-based robot building with materials that are familiar to middle school students with a USB-driven robot control board that enables choreography of a physically built robotic puppet using an iconic programming language with appropriate computational complexity for sophisticated middle school coursework [79.75]. This program has demonstrated cross-curricular success in the sciences, for instance, enabling learners in an Anatomy course to build an elastic-band based arm model that can demonstrate the major degrees of freedom of the human arm using servomotors and sensors. The process of completely designing and implementing a working anatomical arm, in turn, both engages technically apt learners and drives inquiry to achieve a *worked-out solution* [79.76] that provides particularly valuable depth of learning both about the minutiae of anatomical musculature and about servomotor mechanisms.

The number of integrated robotics experiences is expanding rapidly, and this in turn guarantees that educational robotics will be widely woven into the fabric of general education, not by designing and implementing new coursework, but by augmenting existing learning experiences and by attracting a greater diversity of learners into every single discipline through the power of more authentic engagement.

79.5.2 Robotics for Computational Thinking

The ability to *think computationally* has become recognized over the past decade as a critical twenty-first century skill that prepares learners to effectively make creative and effective use of the technology that will doubtless suffuse society for the foreseeable future [79.77]. Computational thinking itself is a category of comprehension that includes understanding sequentiality, thresholds, systems behavior, digital values, and the process of decision-making and action so important in designing new ways to embed digital systems in process, whether in virtual or real domains. Computer programming and data visualization are two important and well-understood areas of inquiry that practice such

forms of thinking, and so one large movement has been to encourage the teaching of computer programming, whether using iconic or lexical languages, as early and as comprehensively as possible [79.78]. The languages that have been most successful include Scratch, Alice, Processing, Snap, and LabVIEW variants. All of these languages combine aspects of iconic programming with circuit semantics that can gain intuitive traction with learners as young as six year of age, and this has markedly increase the market *opportunity* for curricula dedicated to thinking computationally [79.79–81].

Robotics has particular value in this learning domain because robots have real-time action components; sensors that behave noisily in the real world; motors that can have nondeterministic effects; and processors that can be programmed directly. Furthermore, all of the most popular iconic programming languages have recently become so openly modifiable by the programming community that patches enabling direct programming and control of educational robotics platforms is both easy and under way. For instance, the Finch educational robot is now programmable in 16 separate languages, in many cases thanks to the efforts of unpaid volunteers [79.82, 83].

Robotics are ideally suited for active engagement around computational thinking, and as ever greater numbers of robot platforms become programmable via the best iconic languages in the market, it is likely that robotics will become the most powerful experimental domain for sensing, measuring, designing behavior, and designing computational processes from sensors to actions.

79.5.3 Early Childhood Robotics

An important area of change today in educational robotics and tangible interfaces is early childhood learning, defined as preprimary levels from three years of age through kindergarten and early primary schooling levels appropriate for nine year old. The early childhood experience has been changing dramatically due to the rapid onset of interactive screen media, and this in turn has led to major discussions on several continents about what the appropriate role might be for interactive media in the lives of the very young. A number of educationally chartered organizations have begun issuing significant guidelines and rationales regarding new media and early childhood, from the BBC in the United Kingdom to the most recent addition, a joint statement on digital media standards presented in the United States by the National Association for the Education of Young Children and the Fred Rogers Center for Early Childhood Education [79.84].

The current discourse on new media in the classroom concentrates on the role of interactive screen media, implying in many cases that interactive technologies are largely limited to the screens of traditional computers, Chrome books, and tablet-style devices. In this context, much of the research currently opines on the role of digital video games in learning, commenting on the onward rush of *gamification*, and its positive and negative consequences on learning [79.85].

In contrast to this general movement, numerous researchers have begun to explore the role of robotics and interactive electronics as tangible devices that can support the process of becoming technologically fluent rather than simply augmenting technical literacy in the very young. The basic argument along these lines is that teaching children to use designed applications on a tablet can make them confident consumers of technology artifacts, but in fact the craft-based and engineering design-based processes of inquiry, creation, and experimentation that can be strongly associated with robotics experiences have the potential to turn children into powerful, self-confident producers of technology rather than mere consumers.

One such tangible experiment is the Children's Innovation Project (CIP), led by Jeremy Boyle of Clarion University together with Melissa Butler of the Allegheny Traditional School District in Pittsburgh, Pennsylvania (www.cippgh.org). The CIP research project concentrates, not only on the creation of new artifacts but also on the process of deliberate exploration of technical artifacts, introducing tangible manipulatives that have both physical form and sensor- and motor-based behavior, together with curricular processes that privilege inquiry, repeated close examination, observation and documentation well above the creation of a finished artifact. The CIP trajectory even presents electronic toys from Goodwill stores (recycled old toys) to students, not as toys, but as containers of raw materials. Five year old students learn to open the plastic toys, remove and identify components, and then reuse these components together with custom tangible devices to create whole new behaviors.

New media for early childhood learning is certainly an area rife with funding and speculation already [79.86]. As the study of tangible devices for early childhood exploration takes hold, numerous projects such as CIP will change the relationship of educational robotics to the kindergarten classroom of the future, turning the discourse surrounding digital media into a conversation about interaction and exploration in ways that bridge the digital and physical realms more meaningfully. In turn, the students of the futures will have a fluency around technology artifacts that will be significantly the best prior generations.

79.6 Educational Evaluation of Robot Programs

An important aspect of assessment, beyond technical evaluation, is educational evaluation: how does human–robot interaction change models of learning and identities with respect to technology; and how is this done in educationally appropriate ways? We find several techniques to be applicable during each of the following phases of an educational project:

- Design-time assessment: the creation phase of an educational robot system
- Formative evaluation: during execution of an educational robot program
- Summative evaluation: following completion of an educational program.

This section will identify relevant tools for design, formative and summative evaluation that we borrow from the human–computer interaction community and the learning science community. These tools are well suited in helping researchers measure, quantify, and communicate how educational robots can impact learning in the classroom and, beyond the classroom, in informal learning venues such as museums and after-school workshops.

79.6.1 Design-Time Assessment

The field of human–computer interaction (HCI) offers formal and heuristic techniques for addressing interfaces between humans and technology [79.87]. Current HCI practice often focuses on user-centered design, which suggests that understanding the user and the entire task can help us to become better designers. One specific HCI technique, contextual inquiry and design, is especially useful for both informal and formal learning venues where robots will interact directly with students as users. Contextual inquiry guides the collection of background data concerning interactions taking place before inclusion of the new, to-be-designed robotic technology, and is often achieved through the collection and analysis of direct interviews and observations. The process includes identifying and polling various stakeholders in the educational environment, from the classroom as a whole to individual educators and mentors.

A second HCI activity, modeling, guides the creation of pictorial flow diagrams that identify the most important features of the environment in which the robotic educational technology is to be used. *Workflow* describes the communication necessary for the classroom activities. *Sequence* identifies actual sequences of action required for various activities in the classroom (or museum). *Artifact* captures the relationship of existing tools required for activities to be performed successfully. Finally, the *physical model* identifies how space is used, which can be notable in the case of the design and potential inclusion of mobile robots in spaces that were generally not designed for them in the first place, like conventional labs and, even worse, regular lecture halls.

In the tradition of learning science, one important technique to create the best possible curriculum is *aligned design*, in which both the educational curricula and its matching evaluation instruments are created simultaneously [79.88]. This design-time form of alignment is powerful because the student experiences and the ways in which learning will be measured can be suited to the classroom context and to one-another with far better results, as compared to the incorporation of standardized evaluation metrics as an afterthought. By the same token, in robotics we can go even further, aligning hardware, programming interfaces, curriculum, and evaluations all at the same time, thanks to the ever more rapid design-fabrication cycles in both the hardware prototyping world as well as the onset rapid software development tools. Educational robotics research has already demonstrated significant gains when robot hardware, software, classroom curriculum, and evaluation instruments are all designed and aligned simultaneously to yield the best possible educational robotics learning ecology [79.89].

Design-time assessment and joint design are principled ways of understanding and responding to existing environments. This, in turn, facilitates design-time judgments in regards to the best manner in which to add a new technology, such as interactive robots, while maximizing the chances that the technology is a real success. But it is the continual cycle of evaluations during the implementation phase, that is critical to a positive outcome. So in summary design-time data collection and assessment helps set the stage and control expectations. Formative evaluations and feedback into the continuing design process, as described below, helps keep the engineering on a path to success.

79.6.2 Formative and Summative Evaluation

Traditional school-based assessments of learning, for instance, course surveys or exam grades are useful as coarse-resolution data collection devices. However, these do not provide sufficient detail to build models of how learning is proceeding with or in spite

of technology changes, nor how further technology changes may improve learning. Furthermore, especially in informal settings such as museums, school-based assessments are even more inappropriate. In such settings, significant learning results from communication. As groups of visitors or teams of students use and talk about a museum exhibit or project challenge, they construct a shared understanding of the content and context of the challenge. There is an outstanding body of work in the education community that focuses on this form of conversation and learning. The following references are excellent starting points [79.90–93]. Applying this learning strategy to educational robotics yields wonderfully rich analyses in terms of how the robots change conversations in the classroom and in informal learning settings such as museums, clubs, and parks [79.94].

In education learning, tools include broad evaluation tools for qualitatively and quantitatively constructing a model of learning across the classroom and deep, focused tools for studying the specific communication-oriented mechanisms of learning that may inform patterns of change and exploration at the student and intra-team level. In terms of broad tools, student surveys are one useful data collection tool. A combination of linear, quantitative queries (called Likert scales) coupled with open-ended essay questions such as *What is the most important concept you learned this week?* invite both statistical analysis of raw results and thematic analysis of self-reported challenges and successes by the students themselves. Recorded interviews with students, individually or in teams, provides even richer information due to the ability to ask complex *What have you learned?* questions appropriate for open-ended analysis, again in terms of thematic content. A deep, focused tool that is expensive in terms of analyst resources but generates very useful information is the ethnography, whereby a trained observer spends, for instance, one week tracking all the activities of two teams in the classroom, including conversations within the team, among team members and other pupils, and between the team and the course instructors and teaching assistants [79.95, 96]. The ethnography is intended to detail the learning and problem-solving process that occurs in class on a minute-by-minute basis.

Taken as a whole, the above collected data can then be analyzed by thematic content, for themes such as those listed in Sect. 79.1. Statistical significance and correlation tests can be applied to the quantitatively derived results of coding all such information to yield information on how the frequency of learning themes appearing in conversation increases or decreases, and how specific themes become more refined or specific in student talk over the duration of the course.

However, assessment of the educational value of robotics programs such as those described in Table 79.1 is often trickier than evaluating programs in more traditional disciplines. There are widely accepted textbooks and curricula for many disciplines, but not yet for robotics. Additionally, the different educational robotics programs described earlier take a variety of approaches toward education.

Many programs (e.g., FIRST Lego League and RoboCup Jr.) are large programs, but are implemented at the local level, so the experience for students can vary tremendously from one locale to another. The more standardized programs (e.g., Botball and FIRST) have different educational models. FIRST uses an inspirational model where students work with professional engineers. Botball follows a more traditional high school model of education where the teachers are trained in the techniques and principles, so that they may guide the students at a high level, but all of the work (both design and implementation) is done only by the students.

Because robotics can serve as an unfamiliar tool for many students in such courses and programs, evaluation of how this unfamiliar, and potentially rich interactive tool impacts student learning and perceived interest levels in science and technology is a common focus of educational robotics assessment. This trend is exacerbated because these programs, which involve tens or even hundreds of thousands of students, are implemented by nonprofit companies and volunteers that lack specific funds and training for formal assessment. For these reasons, formal assessment of the programs tends to be based on student and/or teacher surveys rather than rigorous testing, and the assessment usually focuses on specific issues (e.g., gender issues in technology education) where assessment specific funding may be available.

As educational robotics becomes more mainstream, we can expect that evaluation will shift from efficacy on general enthusiasm about STEM learning to specific learning advantages derived from such robot-inclusive coursework for follow-on coursework in diverse disciplines such as biology, physics, and computer science. An important new trend in the United States offers hope for more broadly applicable educational robotics design. To date, very successful robotics curricula has often been hampered in terms of scaling due to the fact that learning standards in the US have been defined at the state level, making it nearly impossible to comprehensively justify new technology curricula in all 50 states. The new Common Core standards now published in the US have national reach across forty states, offering an opportunity for national consistency. Furthermore, the new Common Core includes major categories of learning that align exceptionally well with

project-focused, integrative learning experiences that are well-suited to educational robotics applications, e.g., learning precision, learning persistence, learning innovation, data interpretation, data representation, geometric measurements, use of statistics, and qualitative functional analysis.

79.7 Conclusions and Further Reading

This chapter has given an overview of many, but by no means all, of the educational robot hardware and programs currently available. Websites such as robots.net and robotevents.com attempt to maintain lists of the rapidly evolving field of educational robot tournaments. Keeping track of the latest hardware and software tools are more problematic. Some additional sources for these topics, and others covered in this chapter include: *Robots for Kids*, edited by Druin & Hendler [79.97]; *Robotic Explorations*, by Martin [79.98]; *Where the Action Is: Foundations of Embodied Interaction*, by Dourish [79.99]; *Piaget's Theory of Cognitive Development*, by Wadsworth [79.100]; and *Computer as Theater*, by Laurel [79.101].

Video-References

VIDEO 239 Hampton robotics club
available from http://handbookofrobotics.org/view-chapter/79/videodetails/239

VIDEO 240 Elementary robotics challenge: Soldier Creek Elementary
available from http://handbookofrobotics.org/view-chapter/79/videodetails/240

VIDEO 241 Global Conference on Educational Robotics & International Botball Tournament
available from http://handbookofrobotics.org/view-chapter/79/videodetails/241

VIDEO 633 Autonomous aerial vehicle carrier landing contest (2001)
available from http://handbookofrobotics.org/view-chapter/79/videodetails/633

VIDEO 634 SeaPerch Challenge 2014 'The Heist'
available from http://handbookofrobotics.org/view-chapter/79/videodetails/634

VIDEO 635 New Mexico elementary botball 2014 – Teagan's first ever run
available from http://handbookofrobotics.org/view-chapter/79/videodetails/635

VIDEO 636 Robotics summer camps – PRIA
available from http://handbookofrobotics.org/view-chapter/79/videodetails/636

VIDEO 637 World robot olympiad Japan 2014
available from http://handbookofrobotics.org/view-chapter/79/videodetails/637

References

79.1 KIPR: Botball robotics education, http://www.botball.org (2009)

79.2 R. Manseur: Hardware competitions in engineering education, Front. Educ. Conf., Vol. 2 (2000), pp. F3C/5–F3C/8

79.3 Robo Cup Junior: http://rcj.robocup.org (2013)

79.4 C. Stein: Botball: Autonomous students engineering autonomous robots, Comput. Educ. J. **13**(2), 72–80 (2003)

79.5 FIRST: FIRST, http://www.usfirst.org/ (2006)

79.6 R. Siegwart: Grasping the interdisciplinarity of mechatronics, IEEE Robotics Autom. Mag. **8**(2), 27–34 (2001)

79.7 R.D. Beer, H.J. Chiel, R.F. Drushel: Using autonomous robotics to teach science and engineering, Communication ACM **42**(6), 85–92 (1999)

79.8 A. Billard: Robota, clever toy and educational tool, Robotics Auton. Syst. **42**, 259–269 (2003)

79.9 P. Coppin: Eventscope: A telescience interface for internet-based education, Proc. SPIE Telemanipulator Telepresence Technol., Vol. 8 (2002)

79.10 B. Fagin: Ada/mindstorms 3.0, IEEE Robotics Autom. Mag. **10**(2), 19–24 (2003)

79.11 R.S. Hobson: The changing face of classroom instructional methods: service learning and design in a robotics course, IEEE Front. Educ. Conf., Kansas City, Vol. 2 (2000), pp. F3C/20–F3C/25

79.12 D. Kumar, L. Meeden: A robot laboratory for teaching artificial intelligence, ACM SIGCSE Bull. **30**(1), 341–344 (1998)

79.13 J. Schumacher, D. Welch, D. Raymod: Teaching introductory programming, problem solving and information technology with robots at West Point, Proc. IEEE 31st Front. Educ. Conf., Vol. 2 (2001)

79.14 E. Wang: Teaching freshmen design, creativity and programming with legos and labview, IEEE Front. Educ. Conf. (2001), pp. F3G–11–15

79.15 U. Wolz: Teaching design and project management with lego RCX robots, ACM SIGCSE Bull. **33**(1), 95–99 (2001)

79.16 J.K. Archibald, R.W. Beard: Goal! robot soccer for undergraduate students, IEEE Robotics Autom. Mag. **11**(1), 70–75 (2004)

79.17 A. Gage, R.R. Murphy: Principles and experiences in using legos to teach behavioural robotics, IEEE 33rd Front. Educ. Conf., Sarasota, Vol. 2 (2003)

79.18 E. Kolberg, N. Orlev: Robotics learning as a tool for integrating science technology curriculum in K-12 schools, IEEE Front. Educ. Conf., Vol. 1 (2001), pp. T2E–12–13

79.19 B.A. Maxwell, L. Meeden: Integrating robotics research with undergraduate education, IEEE Intell. Syst. **15**(6), 22–27 (2000)

79.20 A. Nagchaudhuri, G. Singh, M. Kaur, S. George: Lego robotics products boost student creativity in precollege programs at UMES, Front. Educ. Conf. (2002) p. 3,S4D–1–S4D–6

79.21 C. Stein, K. Nickerson: Botball robotics and gender differences in middle school teams, Proc. ASEE Annu. Conf., Salt Lake City (2004)

79.22 J.B. Weinberg, J.C. Pettibone, S.L. Thomas, M.L. Stephen, C. Stein: The impact of robot projects on girls' attitudes toward science and engineering, Proc. RSS Robotics Educ. Workshop, Atlanta (2007)

79.23 T. Fong, I. Nourbakhsh, K. Dautenhahn: A survey of socially interactive robots, Robotics Autom. Syst. **42**(3), 143–166 (2003)

79.24 F. Martin, B. Mikhak, M. Resnick, B. Silverman, R. Berg: To mindstorms and beyond: Evolution of a construction kit for magical machines. In: *Robots for Kids*, ed. by A. Druin, J. Hendler (Morgan Kaufmann, San Francisco 2000) pp. 10–32

79.25 M. Cooper, D. Keating, W. Harwin, K. Dautenhahn: *Robots in the Classroom – Tools for Accessible Education* (IOS, Amsterdam 1999)

79.26 C. Pomalaza-Raez, B.H. Groff: Retention 101: Where robots go . . . students follow, J. Eng. Educ. **92**(1), 85–90 (2003)

79.27 KIPR: Open autonomous robotics game, http://www.kipr.org/kipr-open (2013)

79.28 R.R. Murphy: Using robot competitions to promote intellectual development, AI Magazine **21**(1), 77–90 (2000)

79.29 D.P. Miller, C. Stein: So that's what Pi is for! And other Educational Epiphanies from Hands-on Robotics. In: *Robots for Kids*, ed. by A. Druin, J. Hendler (Morgan Kaufmann, San Francisco 2000) pp. 219–243

79.30 I. Vernor, D. Ahlgren, D.P. Miller: Olympiads: A new means to integrate theory and practice in robotics, Proc. ASEE Natl. Conf. (2006)

79.31 NASA Robotics Alliance Project: Education matrix, http://robotics.nasa.gov/edu/matrix.php (2006)

79.32 6.270 Organizers: The history of 6.270 – MIT's autonomous robot design competition, http://web.mit.edu/6.270/www/about/history.html (MIT, Cambridge 2005)

79.33 BEST Robotics: Middle and high school robotics competition, http://www.bestinc.org/MVC/ (BEST Robotics Inc., Auburn 2006)

79.34 D. Ahlgren: Trinity college fire fighting home robot contest, http://www.trincoll.edu/events/robot/ (Trinity College, Hartford 2006)

79.35 Fuji Soft ABC, Inc.: Fsi-all japan robot-sumo tournament, http://www.fsi.co.jp/sumo-e/ (2006)

79.36 IMRMC Organizers: International micro robot maze contest, http://imd.eng.kagawa-u.ac.jp/maze/ (2012)

79.37 Seaperch: http://www.seaperch.org/

79.38 RoboCup Federation: http://www.robocup.org/

79.39 Robotics Education & Competition Foundation: http://www.roboticseducation.org/vex-robotics-competitionvrc/ (2012)

79.40 D. Nardi, M. Riedmiller, C. Sammut, J. Santos-Victor (Eds.): *RoboCup 2004: Robot Soccer World Cup VIII*, Lecture Notes in Computer Science, Vol. 3276 (Springer, Berlin, Heidelberg 2005)

79.41 R. Tucker: Balch, Holly A. Yanco: Ten years of the AAAI mobile robot competition and exhibition, AI Magazine **23**(1), 13–22 (2002)

79.42 Robot Challenge: http://www.robotchallenge.org (2013)

79.43 KIPR: Global conference on educational robotics, http://kipr.org/gcer (2013)

79.44 D.P. Miller: Robot contests at GCER 2011, IEEE Robotics Autom. Mag. **18**(4), 10–12 (2011)

79.45 PRIA: European conference on educational robotics, http://ecer13.pria.at (2013)

79.46 Honda: Asimo: The world's most advanced humanoid robot, http://asimo.honda.com

79.47 iCub.org: An open source cognitive humanoid robotics platform, http://www.icub.org

79.48 Willow arage: Pr2 overview, http://www.willowgarage.com/pages/pr2/overview

79.49 Parallax Inc.: The scribbler 2, http://www.parallax.com/go/s2 (2012)

79.50 LOGO Foundation: A logo primer or what's with the turtle, http://el.media.mit.edu/logo-foundation/logo/turtle.html (2000)

79.51 V. Asper, W. Smith, C. Lee, J. Gobat, K. Heywood, B. Queste, M. Dinniman: Using gliders to study a phytoplankton bloom in the ross sea, antarctica, IEEE OCEANS (2011) pp. 1–7

79.52 C.M. Clark, C.S. Olstad, K. Buhagiar, T. Gambin: Archaeology via underwater robots: Mapping and localization within maltese cistern systems, IEEE 10th Int. Conf. Control Autom. Robotics Vis. (ICARCV) (2008) pp. 662–667

79.53 M. Theberge, G. Dudek: Gone swimmin' (seagoing robots), IEEE Spectrum **43**(6), 38–43 (2006)

79.54 MBARI: Build your own ROV, http://www.mbari.org/education/rov/

79.55 S.W. Moore, H. Bohm, V. Jensen: *Underwater Robotics: Science, Design & Fabrication* (MATEU, Monterey 2010)

79.56 Marine Tech: Marine tech – ROV competition, http://www.marinetech.org/rov-competition/ (2013)

79.57 F. Gudaitis: The first days of RC, Model Airplane News, April (2011)
79.58 AIAA: Student design-build-fly competition, http://www.aiaadbf.org (2013)
79.59 Speedfest: http://speedfest.okstate.edu (2013)
79.60 R. Mahony, V. Kumar, P. Corke: Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor, IEEE Robotics Autom. Mag. **19**(3), 20–32 (2012)
79.61 S. Lupashin, A. Schöllig, M. Sherback, R. D'Andrea: A simple learning strategy for high-speed quadrocopter multi-flips, IEEE Int. Conf. Robotics Autom. (ICRA) (2010), pp 1642 –1648
79.62 J. Meyer: A low cost, vision based micro helicopter system for education and control experiments, Master's Thesis (Univ. of Oklahoma School of Aerospace & Mechanical Engineering, Norman 2014)
79.63 J.L. Jones, A.M. Flynn: *Mobile Robots: Inspiration to Implementation* (Peters, Natick 1993)
79.64 Parallax Inc.: BASIC stamps, http://www.parallax.com/html_pages/products/basicstamps/basic_stamps.asp
79.65 J. Lahart: Taking an open-source approach to hardware, The Wall Street Journal, November 27 (2009)
79.66 C. Anderson: The future of arduino, http://diydrones.com/profiles/blogs/the-future-of-arduino
79.67 KIPR: Link, http://www.kipr.org/products/link (2013)
79.68 D.P. Miller, M. Oelke, M.J. Roman, J. Villatoro, C.N. Winton: The cbc: A linux-based low-cost mobile robot controller, Proc. IEEE Int. Conf. Robotics Autom. (ICRA) (2010)
79.69 G. Mitsuoka: CBC v2 controller: Brings ease of use as well as speed, power and vision to small robots, Robot Mag. **29**, 82–85 (2011)
79.70 D. Shiffman: Interview with Casey Reas and Ben Fry, http://rhizome.org/editorial/2009/sep/23/interview-with-casey-reas-and-ben-fry/ (2009)
79.71 H. Barragán, B. Hagman, A. Brevig: Wiring, http://wiring.org.co (2011)
79.72 B.J. Duch, S.E. Groh, D.E. Allen: Why problem-based learning. In: *The Power of Problem-Based Learning*, ed. by B.J. Duch, S.E. Groh, D.E. Allen (Wiley, New York 2011) pp. 3–11
79.73 H.A. Yanco, H.J. Kim, F. Martin, L. Silka: Artbotics: Combining art and robotics to broaden participation in computing, Proc. AAAI Spring Symp. Robots Robot Venues, Stanford (2007)
79.74 F. Martin, G. Greher, J. Heines, J. Jeffers, H.J. Kim, S. Kuhn, K. Roehr, N. Selleck, L. Silka, H. Yanco: Joining computing and the arts at a mid-size university, J. Comput. Sci. Coll. **24**(6), 87–94 (2009)
79.75 E. Hamner, T. Lauwers, D. Bernstein, I. Nourbakhsh, C. DiSalvo: Robot diaries: Broadening participation in the computer science pipeline through social technical exploration, AAAI Spring Symp. Using AI to Motiv. Gt. Particip. Comput. Sci. (2008)
79.76 A. Renkl: Worked-out examples: Instructional explanations support learning by self-explanations, Learn. Instr. **12**(5), 529–556 (2002)
79.77 J.M. Wing: Computational thinking, Communication ACM **49**(3), 33–35 (2006)
79.78 J. Countryman, A. Feldman, L. Kekelis, E. Spertus: Developing a hardware and programming curriculum for middle school girls, ACM SIGCSE Bull. **34**(2), 44–47 (2002)
79.79 M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, Y. Kafai: Scratch: Programming for all, Communication ACM **52**(11), 60–67 (2009)
79.80 J. Mönig, B. Harvey: *Build your own blocks* (Univ. of California, Berkeley 2010)
79.81 W.P. Dann, S. Cooper, R. Pausch: *Learning to Program with Alice* (Prentice Hall, Upper Saddle River 2011)
79.82 T. Lauwers, I. Nourbakhsh: Designing the finch: Creating a robot aligned to computer science concepts, AAAI 1st Symp. Educ. Adv. Artif. Intell. (2010)
79.83 T. Lauwers, E. Hamner, I. Nourbakhsh: A strategy for collaborative outreach: Lessons from the csbots project, Proc. 41st ACM Tech. Symp. Comput. Sci. Educ. (2010) pp. 315–319
79.84 E.A. Vandewater, V.J. Rideout, E.A. Wartella, X. Huang, J.H. Lee, M. Shim: Digital childhood: Electronic media and technology use among infants, toddlers, and preschoolers, Pediatrics **119**(5), e1006–e1015 (2007)
79.85 C. Renaud, B. Wagoner: The gamification of learning, Princ. Leadersh. **12**(1), 56–59 (2011)
79.86 R. Zevenbergen: Digital natives come to preschool: Implications for early childhood practice, Contemp. Issues Early Child. **8**(1), 19–29 (2007)
79.87 J. Nielsen: Heuristic evaluation, Usability Insp. Methods **24**, 413 (1994)
79.88 S.A. Cohen: Instructional alignment: Searching for a magic bullet, Educ. Res. **16**(8), 16–20 (1987)
79.89 T. Lauwwers: Aligning Capabilities of Interactive Educational Tools to Learner Goals, Ph.D. Thesis (Carnegie Mellon Univ., Pittsburgh 2010)
79.90 S. Allen: Looking for learning in visitor talk: A methodological exploration. In: *Learning Conversations in Museums*, ed. by G. Leinhardt, K. Crowley, K. Knutson (Lawrence Erlbaum, Mahwah 2002) pp. 59–303
79.91 K. Crowley, M. Callanan: Describing and supporting collaborative scientific thinking in parent-child interactions, J. Mus. Educ. **23**(1), 12–17 (1998)
79.92 K. Crowley, M.A. Callanan, J.L. Jipson, J. Galco, K. Topping, J. Shrager: Shared scientific thinking in everyday parent-child activity, Sci. Educ. **85**(6), 712–732 (2001)
79.93 G. Leinhardt, K. Crowley: Objects of learning, objects of talk: Changing minds in museums. In: *Perspectives on Object-Centered Learning in Museums*, ed. by S.G. Paris (Lawrence Erlbaum, Mahwah 2002) pp. 301–324

79.94 I. Nourbakhsh, E. Hamner, E. Ayoob, E. Porter, B. Dunlavey, D. Bernstein, K. Crowley, M. Lotter, S. Shelly, T. Hsiu, E. Porter, B. Dunlavey, D. Clancy: The personal exploration rover: Educational assessment of a robotic exhibit for informal learning venues, Int. J. Eng. Educ. **22**(4), 777–791 (2006)

79.95 M.U. Bers, A.B. Ettinger: Programming robots in kindergarten to express identity: An ethnographic analysis. In: *Robots in K-12 Education*, ed. by B.S. Barker (Information Science Refernce, Hershey 2012) p. 168

79.96 I.R. Nourbakhsh, K. Crowley, A. Bhave, E. Hamner, T. Hsiu, A. Perez-Bergquist, S. Richards, K. Wilkinson: The robotic autonomy mobile robotics course, Auton. Robots **18**(1), 103–127 (2005)

79.97 A. Druin, J. Hendler (Eds.): *Robots for Kids: Exploring New Technologies for Learning* (Morgan Kaufmann, Boston 2000)

79.98 F.G. Martin: *Robotic Explorations: A Hands-on Introduction to Engineering* (Prentice Hall, Upper Saddle River 2000)

79.99 P. Dourish: *Where the Action Is: The Foundations of Embodied Interaction* (MIT Press, Cambridge 2004)

79.100 B.J. Wadsworth: *Piaget's theory of Cognitive and Affective Development: Foundations of Constructivism* (Longman, New York 1996)

79.101 B. Laurel: *Computers as Theatre* (Addison-Wesley, Reading 2013)

Multimedia Contents

80. Roboethics: Social and Ethical Implications

Gianmarco Veruggio, Fiorella Operto, George Bekey

This chapter outlines the main developments of roboethics 9 years after a worldwide debate on the subject – that is, the applied ethics about ethical, legal, and societal aspects of robotics – opened up. Today, *roboethics* not only counts several thousands of voices on the Web, but is the issue of important literature relating to almost all robotics applications, and of hundreds of rich projects, workshops, and conferences. This increasing interest and sometimes even fierce debate expresses the perception and need of scientists, manufacturers, and users of professional guidelines and ethical indications about robotics in society.

Some of the issues presented in the chapter are well known to engineers, and less known or unknown to scholars of humanities, and vice versa. However, because the subject is transversal to many disciplines, complex, articulated, and often misrepresented, some of the fundamental concepts relating to ethics in science and technology are recalled and clarified.

A detailed taxonomy of sensitive areas is presented. It is based on a study of several years and referred to by scientists and scholars, the result of which is the Euron Roboethics Roadmap. This taxonomy identifies the most evident/urgent/sensitive ethical problems in the main applicative fields of robotics, leaving more in-depth research to further studies.

80.1 **A Methodological Note** 2137
80.2 **Specificity of Robotics** 2138
80.3 **Cultural Differences in the Acceptance of Robots** 2138
80.4 **Roboethics Foreshadowed in the Literature** 2139
80.5 **And Expressed in Real Robotics** 2139
80.6 **Ethics in Science and Technology** 2140
80.7 **Ethical Issues in an ICT Society** 2143
80.8 **Human Principles and Rights** 2144
80.9 **Legal Issues in Robotics** 2146
80.9.1 Responsibility 2146
80.9.2 Liability 2146
80.9.3 Privacy 2146
80.10 **Roboethics Taxonomy** 2147
80.10.1 Industrial Robotics 2147
80.10.2 Service Robots 2147
80.10.3 Humanoids 2148
80.10.4 Healthcare and Life Quality 2149
80.10.5 Distributed Robotic Systems 2151
80.10.6 Outdoor Robotics 2152
80.10.7 Military Robotics 2152
80.10.8 Educational Robot Kits 2154
80.10.9 Entertainment Robotics and Art 2155
80.11 **Roboethics Enforced: From Ideals to Rules** 2156
80.12 **Conclusions and Further Reading** 2157
Video-References 2158
References 2159

The philosophical issue of the relationship between humankind and *autonomous* machines appeared early in world literature, having been developed through legends and myths. In early mythology, the ancient peoples expressed their worries about the disrupting power of machines over the old societies: When these artificial creatures we have given birth to, will have learned everything from us, or understood that we are weaker than them, will they try to dominate us?

However, for almost 250 000 years humanity has relied on tools and machines, to increase its power by eliminating manual labor and needless drudgery. This factor has become one of the keys to successful economic progress, especially since the Industrial Revolution and the emergence of a mechanized economy, and even more so with the introduction of computers and automatic machines in the twentieth century [80.1]. Today, progress in the fields of computer science and telecommunications allows us to endow machines with enough intelligence so that they can act autonomously. The twenty-first century humanity will coexist with the first *alien* intelligence we have ever come into contact with – robots.

In a few years from now, many more fields of application will be robotized, because robotics will have occupied more territories. The 2013 McKinsey Report on the five disruptive technologies, *Advances that will transform life, business, and the global economy*, outlines advanced robotics as one of them.

Actually, we are confronted with the rapid development of research and applications in the service (professional and personal) robotics sector, marking the end of the robot segregation era. The field of social robots – although it does not yet show dramatic sale figures – now offers standardized market products for consumers. We can foresee that research and application will point in this direction, driven both by the market demand and by the challenges and richness of technological and scientific issues posed to the roboticist researchers by the human robotics interface. Our graying societies are open to – and demand – dependable and safe automatic autonomous machines to be employed in human cooperation and assistance.

Differently from the nineteenth century – when the novel *Frankenstein* appeared – today, the love and fear, pros and cons about robots are more grounded in science, technology, and in humanities. Along with other applied ethics (bioethics, computer ethics) ethical, legal, and societal aspect of robotics are debated in roboethics.

A worldwide debate on roboethics opened up 10 years ago, following the First International Symposium on Roboethics in 2004 [80.2]. Before that conference, the word *roboethics* did not exist, not in any encyclopedia, nor on the Web. Today, it counts several thousands of voices on the Web, an important literature relating to almost every robotics application, and hundreds of rich projects, workshops, and conferences. This increasing interest and sometimes even fierce debate expresses the perception and need of scientists, manufacturers, and users of professional guidelines and ethical indications about robotics in society.

Some major segments of our lives are already often dependent on sophisticated machines. However, in some critical instances (where robots are entrusted with human lives, such as in medicine, in human care, or in war theaters) we are approaching an ethical and regulatory gap, primarily due to the lack of legal guidelines to ascribe responsibility to software agents and learning machines. This lack of standards and ethical rules is related to the specificity of robot and robotics, and to all learning machines. That is, machines endowed with the learning capacity.

The recent thrust in research and applications programs aiming towards developing robots that are able to co-operate with humans – that is, software agents and robots entrusted with high learning and decision-making functions – presents many theoretical and practical cases in which moral and legal aspects of the responsibility attribution problem for learning robots may soon become an urgent issue [80.3].

Facing the development of ever more powerful computers and the variety of humanoid robots, some scholars and scientists have warned about the dangers of the unlimited use of technology, and especially about the hubristic endeavor to design and manufacture intelligent creatures [80.4, 5].

Their concern has been amplified by the harsh discussion around bioengineering and bioethics. Less dramatically, others have pointed out the need to introduce ethical rules in technological applications, especially regarding the behavior of intelligent machines. In this frame, the most matter-of-fact issue is: What will the cultural and social implication of the *robotics invasion* be? Could robots be dangerous to humankind in any way?

80.1 A Methodological Note

This chapter is of a somewhat different nature from – although complementary to – the totality of the book, because it deals not only with the scientific and technological issues inherent in the matter, but also with cultural and moral topics related to the introduction of robots in sensitive human areas.

It is an acknowledged feeling that robotics stimulates some of humanity's most fundamental questions. This means that we shall not expect that some simple answers be conclusive. The undertaking of discovering the essence and origin of human intelligence and self-consciousness is as tough and troubling as the challenge around the unification of physical forces, or the research on the origin of the universe. Simplistic answers could lead to gross mistakes and we cannot obtain correct answers if we pose the wrong questions. The problem is that robots are machines that are more similar to humans than anything we have ever built, and this makes it easier, on one hand, to adopt human terminology for machines and, on the other hand, to use machine terminology for humans. This said, the authors feel it necessary to state the following methodological premises:

- This Handbook – and in particular the present chapter – will be read by roboticists and also by non-roboticists, and by students of robotics as well as by students and scholars of ethics, the philosophy of science, sociology, laws, etc. Some of the issues presented here are well known to some and less known or unknown to others, and vice versa. Nonetheless, the authors deem it useful and important to recall and clarify some of the fundamental concepts relating ethics in science and technology, because the subject is complex, articulated, and often misrepresented.
- Roboethics is an applied ethics that refers to studies and works done in the field of science and ethics (S&TS – science and technology studies, science and technology and public policy, professional applied ethics), and whose main premises are derived from these studies. In fact, roboethics was not born without parents, but it derives its principles from the global guidelines of the universally adopted applied ethics. This is the reason that a relatively substantial part of this chapter is devoted to this matter, before a specific discussion of sensitive areas of roboethics.
- Many of the issues of roboethics are already covered by applied ethics such as computer ethics, bioethics, or neuroethics. For instance, problems – arising in roboethics – of dependability; of technological addiction; of digital divide; of the preservation of human identity and integrity; the applications of precautionary principle; of economic and social discrimination; and of the artificial system autonomy and accountability. As well as issues related to responsibilities for (possibly unintended) warfare applications; the nature and impact of human–machine cognitive and affective bonds on individuals and society; all these issues have already been matters of investigation by the computer ethics and bioethics.

Here, the specificity of robotics was underlined from a general point of view. Subsequently, in the taxonomy, the specific ethical issues related solely to robotics are carefully evaluated. The present taxonomy was not developed on the basis of the affinity of the techno-scientific or disciplinary areas – like the Index of the present book. In the roboethics taxonomy, the grid is based on the affinity of the application of robots, and on the specificity inherent in the human–robot interaction of these applications.

In terms of scope, we have taken into consideration – from the point of view of the ethical issues connected to robotics – a temporal range of two decades, in which frame we could reasonably locate and infer – on the basis of the current state-of-the-art in robotics – certain foreseeable developments in the field.

For this reason, we consider premature – and have only hinted at – problems related to the possible emergence of human qualities in robots: consciousness, free will, self-consciousness, and sense of dignity, emotions, and so on. Consequently, this is why we have not examined problems – debated in some other papers and essays – like the proposal to not behave with robots like with slaves, or the need to guarantee them the same respect, rights, and dignity we owe to human workers.

Likewise, and for the same reasons, the target of roboethics is not the robot and its artificial ethics, but the human ethics of the robots' designers, manufacturers, and users.

Although informed about the issues presented in some papers on the need and possibility to attribute moral values to robots' decisions and about the chance that in the future robots might be moral entities like – if not more than– human beings, the authors have chosen to examine the ethical issues of the human beings involved in the design, manufacturing, and use of robots.

We felt that problems like those connected to the application of robotics within the military and the possible use of military robots against some populations not provided with this sophisticated technology, as

well as problems of terrorism in robotics and problems connected with biorobotics, implantations, and augmentation, were pressing and serious enough to deserve a focused and tailor-made investigation. It is absolutely clear that without a deep rooting of roboethics in society, the premises for the implementation of an artificial ethics in the robots' control systems will be missing.

80.2 Specificity of Robotics

Robotics is a discipline originating from:

- Mechanics
- Automation
- Electronics
- Computer science
- Cybernetics
- Artificial intelligence.

However, it draws on from several other disciplines:

- Physics/mathematics
- Logic/linguistics
- Neuroscience/psychology
- Biology/physiology
- Anthropology/philosophy
- Art/industrial design.

Is robotics a new science?

On the one hand, robotics is often regarded only as a branch of engineering dealing with intelligent, autonomous machines. In this case, it shares experiences with other disciplines, and it is somehow the linear sum of all the knowledge. In fact, the keyword robotics seldom appears in institutional programs, mainly being hosted in the information and communication technology (ICT) cluster, or hidden under different initials.

Some of us, however, see robotics potentially as new science, in its early stage. Actually, it is the first time in history that humanity is approaching the challenge to develop an intelligent and autonomous entity. This extraordinary mission gives to robotics the special feature of being a platform where sciences and humanities – hard and soft sciences – converge: an experiment in itself [80.6]. Actually, it is not without some ground that we could forecast that robotics will emerge as a new science, with its own theory, principles, theorems, proofs, and mathematical language. We are just on the eve of the development of our science. It is hard to envisage its future. It could happen that robotics will incorporate other sciences; or that, like the giant red stars, it will explode into many other sciences, born from the intersections of adjoining fields.

80.3 Cultural Differences in the Acceptance of Robots

While analyzing the present and future roles of robots in our societies, we shall be aware of the underlying principles and paradigms which influence social groups and single individuals in their relationship with these machines.

Different cultures and religions regard the intervention in sensitive human fields like reproduction, neural therapies, implantations, and privacy differently. These differences originate from the cultural specificities towards the fundamental values regarding human reproduction, life, and death.

In different cultures, ethnic groups, and religions the very concepts of *life* and *human life* differ, first of all concerning the immanence or transcendence of human life. While in some cultures women and children have fewer rights than adult males (not even the *habeas corpus*), in others the ethical debate ranges from the legalization on a post-human status, to the rights of robots. Thus, the different approaches in roboethics concern the rights in diversity (gender, ethnicity, minorities) and the definition of human freedom and animal welfare. From these concepts all other ethical specificities such as privacy and the border between privacy and traceability of actions are derived. Cultural differences also emerge in the realm of *natural versus artificial*. Think of the attitude of different peoples towards surgical implants or organ implantation; how human enhancement could be viewed.

Bioethics has opened important discussions. How is the integrity of the person conceived? What is the perception of a human being? Last but not least, the very concept of *intelligence*, human and artificial, is subject to different interpretation. In the field of artificial intelligence AI and robotics alone, there is a terrain of dispute – let us imagine how harsh it could be outside of the inner circle of experts.

80.4 Roboethics Foreshadowed in the Literature

Literature is the instrument by which society expresses itself, free from rigid constraints, and by which it can *simulate* future social developments. Sometimes, by way of literature important and foresighted scientific issues have been dreamt of and anticipated.

The topic of the threat posed by artificial entities designed by man's ingenuity (legends like *the rebellions of automata*, the Frankenstein myth, the Golem) recurs in classical European literature, as well as the misuse or the evil use of the product of engineering (the myth of Dedalus). This is not the case in all world cultures. For instance, Japanese culture does not include such paradigms; on the contrary, machines (and, in general, human products) are always beneficial and friendly to humanity.

In 1942, the outstanding novelist *Asimov*, who coined the word *Robotics*, formulated the famous Asimov's three laws of robotics in his novel *Runaround* [80.7]:

Law 1: A robot may not injure a human being, or through inaction, allow a human being to come to harm.

Law 2: A robot must obey the orders given to it by human beings except where such orders would conflict with the first law.

Law 3: A robot must protect its own existence as long as such protection does not conflict with the first or second law.

Later on, in 1983, Asimov added the fourth law (known as the zeroth law):

Law 0: No robot may harm humanity or, through inaction, allow humanity to come to harm.

Over the last few decades, scientific and technological developments have brought forward the frontiers of robotics, so that those problems that years ago seemed only theoretical, or a matter of literature and science fiction, are becoming very practical, and even urgent. Some of these problems have alerted the robotics community to the need to open a discussion on the principles that should inspire the design, manufacturing, and use of robots.

Could these *laws* really become the ethics of robots? Of course not. The three laws are a beautiful message passed on to us by Asimov, but they are a literary expression of some general principles metaphorically applied to robotics – it is a work of art, and a source of inspiration. However, as such these laws are inadequate to encompass the complexity of robotics in all its branches and applications to society.

80.5 And Expressed in Real Robotics

In January 2004, in San Remo, Italy, the School of Robotics, in collaboration with roboticists, philosophers, and scholars of humanities, organized the First International Symposium on Roboethics, where the word *roboethics* was officially used for the first time (Fig. 80.1).

In this occasion the IEEE Robotics and Automation Society established a Technical Committee (TC) on Roboethics, with the aim of providing the IEEE Robotics and Automation Society with a framework for analyzing the ethical implications of robotics research, by promoting the discussion among researchers, philosophers, ethicists, and manufacturers, but also by supporting the establishment of shared tools for managing ethical issues in this context. The TC promoted many activities, workshops, and studies on roboethics.

In 2005, EURON (the European Robotics Research Network) funded a project called Euron Roboethics Atelier, with the aim of drawing the first roboethics roadmap. In 2006, in Genoa, Italy, scholars from humanities met with engineers and roboticists for 3 days to draw the lines of the Euron Roboethics Roadmap [80.8].

Following the First International Symposium on Roboethics, one of the first steps in roboethics was to select a methodology for the identification and analysis of techno-ethical issues in robotics. To our knowledge, three European projects (which also availed themselves of international experts, as well as background studies in information and computer ethics and in bioethics) represent the most structured efforts in this field. These are: the Euron Roboethics Atelier, 2006; Ethicbots, 2006–2008; and CARE, Coordination Action for Robotics in Europe, 2008.

As defined by its designer [80.2]:

> *Roboethics is an applied ethics whose objective is to develop scientific/cultural/technical tools that can be shared by different social groups and beliefs. These tools aim to promote and encourage the development of robotics for the advancement of human society and individuals, and to help preventing its misuse against humankind.*

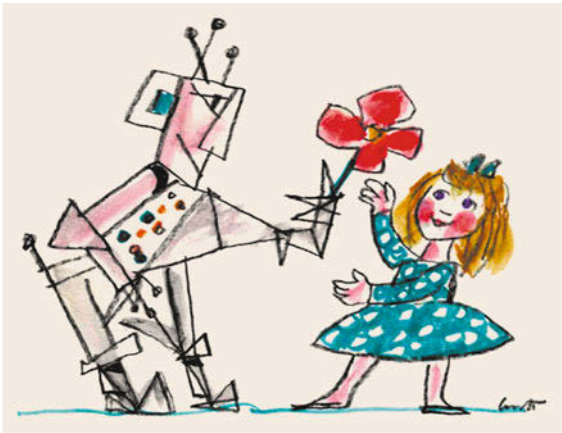


Fig. 80.1 The roboethics' logo, sketched by the renown Italian artist, Emanuele Luzzati (1920–2007), is represented by a smiling young girl receiving a flower from a chivalrous humanoid robot

According to the definition, roboethics is not the *ethics of robots*, nor any artificial ethics, but it is the human ethics of the robots' designers, manufacturers, and users.

Over the years since 2004, three different definition of roboethics emerged, according to the epistemological and moral conceptions of their authors. These three definitions correspond to three different words [80.9].

The *first definition* is represented by the adopted ethical theories, developed principally by the branch of philosophy called ethics or morality, which studies human conduct, moral assessments and the concepts of good and evil, right and wrong, justice and injustice, etc. In our case, a generic or fundamental ethical reflection is directly linked to the particular issues that are generated by the development of robotic applications and their diffusion in society. This is the proper concept of *roboethics*, meaning applied ethics that, similarly to bioethics, attempts to provide answers to new questions that are generated by the progress of a specific scientific and technical field. This level updates the various views on concepts such as dignity and integrity of the person and the fundamental rights of the individual, as well as the social, psychological, and legal aspects involved. Roboethics is developed for a) the identification and analysis of ethical issues that arise from current and upcoming robotics applications, and b) the possible definition of some general guidelines on the above. In a broader meaning of the word, roboethics not only applies to the prohibition and/or prevention of some actions, but identifies also those trends of robotics research and development that enhance what are viewed as human rights (the encouragement to some actions): robotics research and applications that promote and enhance human rights for well-being, education, and the right to medical care, food, housing, and employment.

The *second definition*, currently referred to as *robot ethics*, or *machine ethics*, regards the code of conduct that designers implement in the AI of robots. This means a sort of artificial ethics able to guarantee that autonomous robots will exhibit ethically acceptable behavior in all situations in which they interact with human beings or when their actions may have negative consequences on human beings or the environment. It is clear that the guidelines to define what is ethically acceptable and to enforce them are the product of the above-mentioned field of roboethics. Robots are, in fact, machines, meaning tools that are unaware of the choices made by their human creators, which, therefore, bear the moral responsibility for the actions, good or bad, of robots.

Finally, there is a *third meaning*, which we could perhaps define as *robots' ethics*, because it is the ethic born from the subjective morality of a hypothetical robot that is equipped with a conscience and freedom to choose its own actions on the basis of a full comprehension of their implications and consequences. It is only in this case that robots may be deemed as moral agents, and that one may refer to involving the responsibilities or rights of robots. This, obviously, is currently speculative and beyond the purposes of this handbook.

80.6 Ethics in Science and Technology

For the purposes of this study, we provide the various meanings of the term *ethics* within the context of modern debate. Traditionally, ethics is a philosophical or theological subject that studies human behavior and assessment criteria for human behaviors and choices. Modern ethics have developed from various points of view along classical philosophy. In the last decades, in our highly complex societies characterized by technoscientific development, the attribution of moral and practical responsibility is becoming more and more difficult, with regard to unintentional or collateral consequences of actions and operations that are produced by unidentifiable decisions (group decisions, complex administrative structure, distribution of responsibilities, computerized operations). Often, one finds it impossible to attribute final responsibility to a single person

or to a defined social entity. In the absence of a definition and of a precise analysis of the responsibility chain, technologically advanced societies have shifted the issue onto the concept of risk assessment, thereby attributing value to the damage produced by entities that are seemingly devoid of responsibility. In our case, the question is: Who is responsible for any damage that may be caused by an autonomous robot? Is it the designer, the manufacturer, the programmer, or the final user? Often it will be difficult to obtain easy answers to this question.

From the view point of ethical theories, we observe that individuals possess ethics of common sense, which provide them with the moral guidelines for decision making, from the small ones to important life decisions. This affects day to day life, as well as our actions in our close-knit social relations. Individuals often adopt different ethical theories from time to time: utilitarianists for some decisions, in other situations or moments they generally go by the terms of their up-bringing, customs, tradition, and religion (descriptive ethics). And this seems to be enough for them.

However, this line of thought reaches its limits when, as social figures, or in our profession, we are faced with complex problems, in which our actions may have multiple consequences that are difficult to follow and predict; and when our common sense is faced with problems we have never dealt with before – bioethical dilemmas for example. In these situations, the ethics of common sense leads to various paradoxes: we find ourselves without any conceptual resources and in the difficult position of having to *pass judgement*. In these cases, we need a logical-critical set of ethics (critical ethics) that: a) reveals the implicit, and perhaps never uncovered, assumptions in our ethics of common sense, and in our outdated ethical theoretical base; b) analyzes reasons, the pros and cons and identifies their origin. Inevitably – we may not even realize it – we resort to prescriptive ethics and ethical theories.

In practice, when faced with general and complex topics, we may refer to the fundamental, relevant values involved in our dilemmas; we may adhere to the more updated morality applied to issues close to ours; or we try to step up from the universally shared prescriptions towards new ethical frontiers. Prescriptive ethical theory, which develops and justifies the principles of moral actions, refers to ethical theories and the related guidelines. These, in turn, represent the general ideas that contain ethical principles, in order to reach an internal and systematic coherence. In defence of the ethical principle, the ethical philosophers either implicitly or explicitly refer to an ethical theory.

In the twentieth century, from the dissatisfaction raised by the issues that are deemed to be the limits to traditional ethical theory – utilitarianism and deontologism inspired by Kant – ethics became fragmented into several forms: rights ethics, virtue ethics, feminist ethics, and applied ethics [80.10]

According to virtue ethics – which have less to do with single actions, but more to do with different life styles, and ways of life – the most important moral obligation is our personal relationship with or actions or non-actions. In making or using robots, according to virtue ethics we would analyze the effect of robots in respect to the concept of good and happiness: Which robots could contribute most to the happiness, to the full and complete quality of life of human beings? [80.11] In another instance, considering the need to account for values that are part of a collective conscience, the so-called rights ethics sees human rights as the most relevant elements in common for a system of cultural and moral pluralism, and as being reasonably deemed to be the final manifestation of universal ethics.

One of the challenges to traditional ethical theories by some thinkers regarded the limitation of moral considerations to the problems of humans and of their relationship in human society, regardless of other living organisms and of the environment. Therefore ethical theories, and the new applied ethical theories in particular, were called upon to include non-human entities in their analysis (animal ethics, environmental ethics, and planetary ethics – known as the Gaia theory), as well as human products (bio-culture, computer ethics, roboethics). The various ethical theories applied are, in turn, connected and intertwined with other disciplines, including law, sociology (descriptive ethics), economics, and various scientific fields.

One of the central themes of applied ethics is the concept of responsibility, which is a moral notion. Legal responsibility determines the rules of the relevant prescriptive ethics, meaning the group of commands and prohibitions adopted by a society or a group, and that also defines professional ethics. For the purposes of this chapter, to analyze the human–robot relationship, we shall consider the two main meanings of the term responsibility:

- The analysis of the identity of the agent of the cause of certain actions and their effects (utilitarianism, consequentialism, or teleologism)
- An expression of motivations that leads an agent to act in a certain way (deontological ethics or Kantian ethics), according to which the individual assesses the consequences of his or her actions.

We know that in the last century, the question *What authority and what set of moral rules am I obliged to be accountable to? State law? God?* forced many answers

(utilitarian or deontological) to a crisis point. Often, the moral response of the individual as well as the moral evolution of our society have led to an opposition to the responsibility towards the nation, the church, or towards traditional roles of social institutions (*Weber*'s ethics of intentions or ethics of individual conscience) [80.12]

Some serious events related to World War Two changed the notion of responsibility and of differentiation of roles (i. e., engineers deal with engineering, doctors with medicine, soldiers obey orders from superiors, etc.); and also some famous legal cases, e.g., the Adolf Eichmann trial, during which Eichmann defended himself by stating that he was obeying orders under his administrative responsibilities and that he did not decide on, nor was he aware of, the entire project of Jewish extermination; or the debate that ensued after Hiroshima/Nagasaki in 1945.

Furthermore, there are different cases in which our personal morals clash with those adopted and imposed by the society we live in and whose law we live under. Such is the case, for example, with the death penalty (which is in force in various states in the USA, although contested by internal currents), animal rights, abortion, or euthanasia. In these cases, the implicit moral philosophy in state laws does not coincide with the feelings of many groups of the relevant societies, which leads to this sort of conflict.

Therefore, we observe that in contemporary societies, the notion of responsibility is not limited to the moral consideration of actions of the agent or cause, and it does not only deal with the necessary conformity of the action with a group of duties (therefore, the analysis of consequences is less decisive). In today's society, just the elements of complexity and of technological ruling determine the aspect that is known as heterogenesis of ends, according to which our actions may have consequences that are extremely difficult to estimate, which may even be opposite to our intentions. According to *Morin*'s ecology of action [80.13], once the action departs from the individual, it lives a life of its own, and combines itself with the environmental conditions (social models, actions and reactions of other agents), and the final result is beyond the agent's predictive abilities.

Faced with this vacancy in the attribution of individual responsibility, some researchers attempted to identify collectively shared responsibilities, if impossible or useless to identify individual responsibility. In the case of scientific research, S&TS expert *von Schomberg* [80.14] proposes to adopt an assessment system based on foresight and knowledge (foresight and knowledge assessment). The author sustains that, because the definition of responsibility is considerably more arduous to define in scientific fields, due to the unintentional consequences, uncertainty, or ignorance of results, instead of identifying the ethical responsibilities post hoc, it is necessary to establish the ethics of the overlap of knowledge between different areas beforehand (synergy: scientists, politicians, etc.) because the quality of knowledge shall determine the ethical value of the applications that will follow. At the same time, one must constantly ensure the maximum precision of predictions in order to identify both the wholesomeness of research and relevant applications, as well as the potential ethical problems.

Other authors have emphasized the need to avoid the overlap between ethical problems and technical solutions; among the latter, the expert of computer ethics, *Mowshowitz*. In his *Technology as excuse for questionable ethics* he states that [80.15]:

> *The seemingly eternal social problems are real enough, but to look for their cause in technique or autonomous technology is both mistaken and harmful. We should not blame technology for human failures. (...) Autonomous technology contributes to the belief in technological determinism, i. e., reinforces belief in the inability of people to make significant choices in their lives. It directs attention away from wielders of power to systems of reified collectivities. The law is smarter than the social sciences – it defines the corporation, for example, as a fictional person for purposes of assigning responsibility and does not absolve key actors of their responsibilities. Institutions should be seen as convenient fictions that help explain individual decision and action (...) Only the actions of human beings can be alienating or dehumanizing. Reification of technology allows for an illegitimate transference of responsibility from persons to a fictional social construct, and at the same time impedes our ability to come to grips with the very real ethical challenges posed by the uses of technology.*

In conclusion, we observe that in roboethics, the definition of moral responsibility, and the resulting notion of liability – which is central in the human–robot relationship – could differ according to the philosophical assumptions, which – knowingly or unknowingly – have been adopted.

80.7 Ethical Issues in an ICT Society

The importance of ethics in science and technology has been demonstrated by our recent history. Three of the front rank fields of science and technology: nuclear physics, bioengineering, and computer science, have already been forced to face the consequences of the applications of their research because of pressure caused by dramatic events, or because of the concern of the general public.

The introduction of intelligent machines in our daily life brings up global social and ethical problems, which are usually summarized as follows:

- Dual-use technology (every technology that can be used for civilian or military applications)
- Anthropomorphization of technological products (it is well known and documented that people attribute intentions, goals, emotions, and personalities to even the simplest of machines with life-like movement or form)
- Humanization of the human/machine relationship (cognitive and affective bonds toward machines)
- Technology addiction
- Digital divide, socio-technological gap (per ages, social layer, per world areas)
- Fair access to technological resources
- Effects of technology on the global distribution of wealth and power
- Environmental impact of technology.

Due to the interdisciplinarity of robotics, roboethics shares problems and solutions with other applied ethics: computer ethics, information ethics, bioethics, technoethics, and neuroethics.

Computer Ethics (CE), is a term that was coined by Walter Maner in the mid 1970s. It indicated the field of research that studies ethical problems *aggravated, transformed, or created by computer technology* [80.16]

Perhaps, the first contact between ethics and computer science took place in the 1940s, when Norbert Wiener, professor at the MIT and one of the founding fathers of computer science expressed his concern about the social effects of the technologies he himself contributed to develop. In 1948, in his book *Cybernetics: or Control and Communication in the Animal and the Machine* [80.17] and successively in his other book, *The Human Use of Human Beings*, he pointed out the dangers of nuclear war and the role of scientists in weapons development in 1947, shortly after Hiroshima [80.18]. Although he did not use the term *computer ethics*, he laid down a comprehensive foundation for computer ethics research and analysis. Wiener's foundation of computer ethics was far ahead of its time.

It was not before 1968 that Wiener's concern became actual practice, when Donn Parker, one of the most famous scientist of SRI (Stanford Research Institute), at Menlo Park, began to examine unethical and illegal uses of computers by computer professionals. According to *Bynum*, Park then said *that it seemed that when people entered the computer center, they left their ethics at the door* [80.19].

In 1968 *Parker* published *Rules of Ethics in Information Processing* [80.20], and he promoted the development of the first code of professional conduct of the Association for Computing Machinery (AMC), which was adopted by the ACM in 1973.

During the late 1960s, Joseph Weizenbaum, the designer of the computer program Eliza, shocked by the emotional involvement of psychiatric scholars towards his simple programs, expressed his concern that an *information processing model* of human beings was reinforcing an already growing tendency among scientists, and even among the general public, to see humans as mere machines. *Weizenbaum* wrote the book *Computer Power and Human Reason*, in which expressed his thoughtful ethical philosophy [80.21].

In the late 1970s, Walter Maner of the Virginia's Old Dominion University was the first to employ the lemma computer ethics to define the field of inquiry dealing with ethical problems aggravated, transformed, or created by computer technology. In 1985, *Moor* of Dartmouth College published his article *What is computer ethics?* [80.16] and *Johnson* of Rensselaer Polytechnic Institute published her book, *Computer Ethics* [80.22], the first textbook – and for more than a decade, the defining textbook – in the field. In 1983 at Palo Alto the CPSR (Computer Professional for Social Responsibility) CPSR was founded. This is a global organization promoting the responsible use of computer technology. Incorporated in 1983 (following discussions and organization that began in 1981), CPSR, is the first international association whose mission is to educate policymakers and the public on a wide range of issues. In 1991 computer ethics was officially added as subject to the programs in the computer science departments of the United States.

Information ethics (IE) was developed in the 1990s by several researchers (especially the Oxford team led by *Floridi*), when it was proposed that the core of the issues of CE did not lie in the specific technology, but in the raw material manipulated by it (data/information) [80.23].

Bioethics is the study of the ethical, social, legal, philosophical, and other related issues arising in

healthcare and in the biological sciences (International Association of Bioethics, IAB).

In 1970 Van Rensselaer Potter coined the word bioethics. He was an American biochemist, Professor of Oncology at the McArdle Laboratory for Cancer Research at the University of Wisconsin-Madison. The first appearance of the term was in his book *Bioethics, Bridge to the Future* [80.24]; he coined it after trying for many months to find the right words to express the need to balance the scientific orientation of medicine with human values.

Potter's original concept of bioethics comprised a global integration of biology and values designed to guide human survival, with a new bioethics as the bridge between science and humanities. Increasingly, he felt the need to link what he came to realize had become mainstream biomedical ethics with environmental ethics. During his career Potter continued to modify the term bioethics to differentiate his conceptions from the dominant view of biomedical ethics. He eventually selected the term global bioethics, and this became the title of his second book. In it, there is a new definition of the term bioethics [80.25]:

> *Biology combined with diverse humanistic knowledge forging a science that sets a system of medical and environmental priorities for acceptable survival.*

The field of bioethics is at a critical stage of evolution, having now passed the 30th year of the development of bioethics programs. It is in a phase of professionalization attending to both the ethical framework for clinical and industrial bioethical consultation and the creation of the next level of academic organizational success, namely, departments and PhD programs.

Technoethics is a recent definition, derived from Christian theology [80.26]:

> *As a sum total of ideas that bring into evidence a system of ethical reference that justifies that profound dimension of technology as a central element in the attainment of a finalized perfection of man.*

Neuroethics is concerned with ethical, legal, and social policy implications of neuroscience, and with aspects of neuroscience research itself [80.27]. Neuroethics encompasses a wide array of ethical issues emerging from different branches of clinical neuroscience (neurology, psychiatry, psychopharmacology) and basic neuroscience (cognitive neuroscience, affective neuroscience).

In conclusion, we can summarize the ethical issues in advanced robotics as derived from the studies of the applied ethics, as follows:

- Under what conditions a technology is to be considered acceptable? (*precautionary principle*)
- At what point the development of this technology is acceptable?
- How do we weigh the cost/benefits ratio of an advanced technology?
- How do we weigh any risk assessment?
- What is the rate of the ethics of functional compensation or repair vs. enhancement/augmentation?
- Are there cases where a particular type of technology itself should be considered unacceptable even though it has potential for compensation as well as enhancement?
- The issue of identifying cause, and assigning responsibility, should some harm result from the deployment of robotic technology [80.28].

80.8 Human Principles and Rights

The authors are inspired, in this study, by the thesis of the American moral and political philosopher *Rawls* and his reflective equilibrium [80.29]. According to Rawls, the charters and treaties, although advanced in an attempt to associate the highest number of participants, and although commendable in the attempt to widen the promotion of rights to as great a number of members and functions as possible, do not satisfy the possibility of positive progress, nor can they provide for the entire range of criminal uses of robots. In roboethics, scholars are heading off to an unknown land, where new and original solutions should be created. However, although searching for new solutions on its way, coherence with philosophy of law imposes that no new form of ethics can demean the rights acquired by the previous ones [80.30]:

> *Any new ethics must deal with the same substance as the old role responsibility ethics, namely with values and norms that restrict or delimit human action and thus enable or guide traditional decision making.*

This statement means that roboethics and robot ethics should be coherent with the universally shared sets of rules.

Universal sets of rules are expressed by internationally recognized institutions such as the United

Nations, The World Health Organization, the FAO, UNESCO/COMEST, the International Labor Organization (ILO), the World Medical Association, and the World Summit on the Information Society. Also, the European Union has identified general ethical principles, which have been adopted by most nations, cultures, and people of the world. Furthermore, the international scientific, juridical, economic, and regulatory communities have in many occasions proposed a harmonization (universally accepted, standardized) of world ethical principles applied to science and technology, especially in those cases where these principles involve sensitive issues such as life, human reproduction, human dignity, and freedom.

For instance, the Ethics of Science and Technology Programme, part of UNESCO's Division of Ethics of Science and Technology in the Social and Human Sciences Sector, and COMEST, an advisory body to UNESCO composed of 18 independent experts, has proposed in the field of bioethics to start a process towards a *declaration on universal norms on bioethics*. In Rio de Janeiro, December 2003, COMEST organized an international conference on the issue of a *Universal Ethical Oath for Scientists*. Because ethics in science and technology is not limited to deontology or professional ethics, but concerns a broader range of questions involving fundamental beliefs and moral principles, its results and conclusions become also guidelines for conduct in the professional daily life.

From the social and ethical standpoint, in deciding on the design, development, and application of a new technology, designers, manufacturers, and end users should follow rules, which are common to all human beings:

- Human dignity and human rights
- Equality, justice and equity
- Benefit and harm
- Respect for cultural diversity and pluralism
- Non-discrimination and non-stigmatization
- Autonomy and individual responsibility
- Informed consent
- Privacy and confidentiality
- Solidarity and cooperation
- Social responsibility
- Sharing of benefits
- Responsibility towards the biosphere
- Obligatory cost benefit analysis (whether ethical issues are to be considered as part of a proper cost-benefit analysis)
- Exploitation of potential for public discussion [80.31].

Computer and information ethics has developed a code of ethics called PAPA (privacy, accuracy, intellectual property, and access), which could be adopted by robotics:

- *Privacy*: What information about one's self or one's associations must a person reveal to others, under what conditions, and with what safeguards? What things can people keep to themselves and not be forced to reveal to others?
- *Accuracy*: Who is responsible for the authenticity, fidelity, and accuracy of information? Similarly, who is to be held accountable for errors in information and how is the injured party to be made whole?
- *Property*: Who owns information? What are the just and fair prices for its exchange? Who owns the channels, especially airways, through which information is transmitted? How should access to this scarce resource be allocated?
- *Accessibility*: What information does a person or an organization have the right or privilege to obtain, under what conditions, and with what safeguards?

Problems of the *delegation* and *accountability* to and within technology are daily life problems of all of us. Today, we give responsibility for crucial aspects of our security, health, and life-saving, and so on to machines.

Professionals are advised to apply, in performing sensitive technologies the *precautionary principle* [80.32]:

> *When an activity raises threats of harm to human health or the environment, precautionary measures should be taken even if some cause-and-effect relationships are not fully established scientifically.*

From the precautionary principle some other rules are derived, such as:

- Non-instrumentalization
- Non-discrimination
- Informed consent and equity
- Sense of reciprocity
- Data protection.

All over the world, associations and orders of engineers have adopted codes of ethics guiding to responsible conduct in research and practice. In this context, *security* and *reliability* are the most important ethical codes of conduct.

Among the other important recommendations are the following:

- Hold paramount the safety, health, and welfare of the public in the performance of their professional duties

- Perform services only in areas of their competence
- Issue public statements only in an objective and truthful manner
- Act in professional matters for each client as faithful agents or trustees
- Avoid improper solicitation of professional assignments [80.33].

80.9 Legal Issues in Robotics

Because of their increasing autonomy and anthropomorphic characteristics, robots are inherently different from other man-made devices and systems and hence require the development of new legal structures under which they operate (and will operate) in society. At present there is no clear legal framework on which to append issues relating to the behavior of autonomous, intelligent robots, and what distinguishes their behavior from that of human beings. Since robot intelligence is provided by their on-board computers, they too will require new legal frameworks. For example, about 15 years ago, *Kurzweil* wrote that by 2029 there will be *a growing discussion about the legal rights of computers and what constitutes being human*. Further, he states that *Machines will claim to be conscious and these claims will be largely accepted* [80.34]. Whether we accept this forecast or not, it illustrates the fact that our current legal structures have no provisions for dealing with autonomous machines [80.35]. Further, many of the ethical issues discussed earlier in this chapter will require legal support for their implementation, since society depends on law for the enforcement of its ethical foundations. Some of the major legal issues concerning robots in society include responsibility, liability, attribution, agency, and privacy. We consider these points briefly (and superficially) in this section. A major treatment of these issues has been provided by *Asaro* [80.36, 37], *Moon* and *Nass* [80.38], *Dennett* [80.39], *Calo* [80.40], *Denning* [80.41], *Sharkey* [80.42], and others.

80.9.1 Responsibility

In the present state of the art, we can say that a *dumb* or even a semi-autonomous robot that breaks the law does so because of bad design, bad programming, or bad construction. This implies that we need to attribute responsibility for improper or illegal behavior to the owners, designers, and/or builders of the machines. The question of responsibility becomes increasingly complex as the robots become more autonomous and capable of modifying their behavior through learning and experience, since their behavior will no longer be based entirely on their original design. Of course, one could attribute the learned behavior to the designers of the artificial intelligence programs; a second-order attribution. This is another issue for the writers of future laws. When human beings *break the law*, the consequences may be punishments, such as paying fines or serving time in jail. Such punishments do not apply to robots, unless their behavior is attributed to humans with evil intentions, who then face the consequences.

A related issue is that of agency, a legal term implying that one person acts on behalf of another; he/she is the agent of the second person. Thus, in real estate transactions, a salesman may be my agent and represent me in discussions with potential buyers of my house. If a robot can be considered an agent of a person, then the attribution of responsibility is clear and direct. Even so, there will be legal issues concerning the (mis)behavior of a robot owned by Mr. A, but acting as an agent for Mr. B. Such a situation may arise if B is a *hacker* and has gained control of the robot's computer.

80.9.2 Liability

Liability is the legal consequence of responsibility. If my robot breaks your property, even accidentally, I am responsible and liable. This is civil liability. If it can be proven that my robot harmed a person as a result of negligence on my part in the design, programming, or fabrication of the robot, I may also be criminally liable. Liability is a major cause of legal action in the United States, it provides income to many lawyers, and is too complex to deal with comprehensively in this chapter. However, we can say that sometimes liability may be apportioned, partially to the designer, partially to the manufacturer, and partially to the user, especially if the latter has modified the robot's hardware or software. Even toy robots like the RoboSapien (formerly manufactured by WowWee Toys in Hong Kong), have spawned Websites with guidelines for *hacking* the robot's software and modifying its behavior.

80.9.3 Privacy

The right to privacy is protected by a variety of laws. Yet, it is clear that recent developments in robotics will challenge both law and custom in this area. We know that drones (autonomous flying vehicles) carry

new cameras with such amazing resolution that they can discern objects 1 m in size from an altitude of several thousand meters [80.43]. This implies that unmanned flying vehicles could spy on citizens in their own backyards, producing pictures that could be used for blackmail or other illegal purposes. Even more serious are recent developments in very small flying robots, only a few centimeters in length, that could fly in and out of windows, perch in trees or on roofs of buildings, collecting information using vision and other sensors, including the ability to record both sounds and smells. The legal system is not currently equipped for handling such methods of surveillance. There is a still more subtle form of privacy violation, discussed by *Calo* [80.40] and *Denning* [80.41]. Consider the increasing variety of personal robots in our homes: vacuum cleaners, window washers, child companions, and assistants to the elderly and disabled. All these machines could also be spies, especially if *hacked* by third parties. Recently there was controversy in the United States over the discovery that some manufacturers of *smart phones* that include global positioning system (GPS) sensors had programmed their machines to send the owners' geographic location to a company's computer. This could be interpreted as another invasion of privacy.

It is evident from the brief summary in this section that current and future developments in robotics will pose greater and greater challenges to the legal systems in many countries. We believe that it is essential for representatives of governments, the legal profession, and robotics designers and manufacturers to begin dialogs on this subject now, before being forced to do so by major accidents or intentional violation of law.

80.10 Roboethics Taxonomy

Certainly, classifying the different branches of robotics is not an easy task. Likewise, it is a complex undertaking to organize a matrix of field of robotics/ethical issues. We have tried to classify these topics according to homogeneous fields from an applicative point of view. Furthermore, in the present taxonomy, we have chosen the *triage process* of identifying the most evident/urgent/sensitive ethical problems in robotics, leaving more complex problems to other times and further studies.

80.10.1 Industrial Robotics

An industrial robot is officially defined by the International Standards Organization (ISO) as an automatically controlled, reprogrammable, multipurpose manipulator. Typical applications of industrial robots include welding, painting, ironing, assembly, pick and place, palletizing, product inspection, and testing, all accomplished with high endurance, speed, and precision.

Complexity can vary from a simple single robot to very complex multi-robot systems:

- Robotic arms
- Robotic work cells
- Assembly lines.

From the social and economic standpoint, the benefits of these robots are extraordinary. They can relieve human beings of heavy work, dangerous workplaces, and routine and tedious activities. In the future, we can imagine robotic factories, completely managed by robots. In industrialized countries, which are facing looming labor shortages due to aging societies, robots in factories will cut costs. Industrial robots increase productivity (higher speed, better endurance); they increase quality (precision, cleanliness, endurance); they make highly miniaturized devices possible.

Social problems stemming from the introduction of robots in factories are, first of all, loss of jobs, and unemployment. On the other hand, while a welfare policy is to be implemented at national level to facilitate workers' reconversion, and educational programs to create new skills, it should also be said that robots have also created new jobs directly and can create wealth, leading to the development of new industries and workplaces.

80.10.2 Service Robots

Service robots are robots of several shapes and sizes (wheeled, legged, humanoids), equipped with different kinds of sensing systems (artificial vision systems, ultrasonic, radio) and manipulations (grippers, hands, tools, probes). Service robots support and back up human operators. These robots represent one of the fastest growing areas of robotics, and they raise a large number of ethical issues. According to the International Federation of Robotics (IFR) World Robotics Report, 2014 [80.44], the total number of professional service robots sold in 2012 went up to 6067 units; likewise, IFR projections for the period 2013–2016 forecast about 22 million units of service robots for personal use).

Consider first the robots used for personal service. In addition to the applications listed above, there are a large number of research projects worldwide to ex-

pand robot use in the home. During the coming decade many more robots will be present in our homes to assist us in tasks such as housekeeping, washing, ironing clothes, and child care. This trend to robots as *co-inhabitants* in our homes began with a vacuum cleaning robot introduced in United States in 2002. Since then millions of these units have been sold, and other vacuum cleaners have been developed in Sweden, South Korea, Japan, and other countries. The ability of service robots to share our living spaces depends to some extent on their communication skills and their ability to relate to human beings. Humanoids are a special category of service robots, which presents many different ethical, legal and societal (ELS) aspects from other service robots, to be dealt with separately.

80.10.3 Humanoids

One of the most ambitious aims of robotics is to design an autonomous robot that could reach – and even surpass – human intelligence and performance in partially unknown, changing, and unpredictable environments. Artificial intelligence will be able to lead the robot to fulfill the missions required by the end-users. To achieve this goal, over the past decades scientists have been working on AI techniques in many fields. Among these are:

a) Artificial vision
b) Perception and analysis of the environment
c) Natural language processing
d) Human interaction
e) Cognitive systems
f) Machine learning, behaviors
g) Neural networks.

In this context, one of the fundamental aspects of the robots is their capability to learn: to learn the characteristics of the surrounding environment, that is, a) the physical environment, but also b) the living beings that inhabit it. This means that robots working in a given environment have to distinguish human beings from other *objects*. In addition to learning about the environment, robots have to learn about their own behavior, through a self-reflective process. They have to learn from the experience, replicating somehow the natural processes of the evolution of intelligence in living beings (synthesis procedures, trial-and-error, learning by doing, and so on).

It is almost inevitable that human designers are inclined to replicate their own conception of intelligence in the intelligence of robots. In turn, the former becomes incorporated into the control algorithm of the robots. Robotics intelligence is a learned intelligence, fed by the world's models uploaded by the designers. It is a self-developed intelligence, evolved through the experience that robots have gained through the learned effects of their actions. Robotics intelligence also comprises the ability to evaluate what attributes a judgment to the actions carried out.

All these processes embodied in robots produce a kind of intelligent machine endowed with the capability of expressing a certain degree of autonomy. It follows that a robot can behave, in some situations, in a way that is unpredictable for their human designers. Basically, the increasing autonomy of the robots could give rise to unpredictable and non-predictable behaviors. So, without necessarily imagining some science fiction (sci-fi) scenarios, where robots are provided with consciousness, free will, and emotions, in a few years we are going to be cohabiting with robots endowed with self-knowledge and autonomy – in the engineering sense of these words.

Humanoids are robots whose body structure resembles the human one. They answer an old dream of humanity, and certainly do not spring only from rational, engineering, or utilitarian motivations, but also from psycho-anthropological ones. Humanoids are the expression of one of the demands of our European culture, which is that humankind was the creator of some mechanical being in the shape of a human. In Japanese culture, it is the demand to carefully replicate nature in all its forms. It is a very difficult and demanding enterprise, a project of the level of the mission to the Moon. However, precisely for its characteristic of being one of humanity's dreams, the investments are high and the progress speed very quick. It has been forecast that in the not-so-distant future we will cohabit with humanoids whose shape will be so similar to that of human beings that it will render it possible to get mixed up – in certain situations – with the latter. Humanoids will assist human operators in human environments, will replace human beings, and will cooperate with human beings in many ways.

Given the high cost and the delicacy of humanoids, they will probably be employed in tasks and in environments where the human shape would really be needed, that is, in all these situations where human–robot interaction is primary, compared to any other mission – human–robot interactions in healthcare, assistance with children/disabled people/elderly assistance, baby sitting, office clerks, museum guides, entertainers, sexual robots, and so on. Or, they will be employed as testimonials for commercial products. The special tasks humanoid robots can fulfill are manifold. Humanoids are robots that are so adaptable and flexible that will be rapidly used in many situations and circumstances. They can assist humans to perform very difficult tasks, and behave like true

and reliable companions in many ways. Their shape, and the sophisticated human–robot interaction, will be very useful for situations where a human shape is needed.

The research carried out in humanoids laboratories over the world will have as a side effects the development of a platform to study the human body, for training, haptic tests and training, with extraordinary outcomes on healthcare, education, edutainment, and so on. Faced with an aging population, Japanese society see humanoids robots as one way to enable people to continue to lead an active and productive life in their old age, without being a burden to other people.

From the point of view of the *safety* in the use of humanoids, and taking into account that in the not so far future they will be used as companions to human beings, humanoids can raise serious problems related to the reliability of the *internal evaluation systems* of the robots, and to the *unpredictability* of robots' behavior [80.41]. Thus, designers should guarantee the traceability of evaluation/actions procedures, and the identification of robots.

Concerning safety, it should be underlined that wrong action by humanoids can lead to dangerous situations for living beings and the environment. From the point of view of security, this is the case where the robot's autonomy is controlled by ill-intentioned people, who can modify the robot's behavior in a dangerous and fraudulent course. Because humanoids sum up almost all the characteristics of the entire spectrum of robots, their use implies the emergence of nearly all the problems that we examine below. In particular, their introduction into human environments, workplaces, homes, schools, hospitals, public places, offices, and so on, will deeply and dramatically modify our society.

There is already an important and well-documented literature on the implication of the coexistence between human beings and humanoids. The problems range from the replacement of human beings (economic problems, human unemployment, reliability, dependability, and so on) to psychological problems (deviations in human emotions, problems of attachment, disorganization in children, fears, panic, confusion between real and artificial, feeling of subordination towards robots [80.45]). Another set of questions arises around the shape of the humanoids. Is it right that robots can exhibit a *personality*? Is it right that robot can express *emotion*? The concern expressed by psychologists is that well before evolving to become conscious agents, humanoids can be an extraordinary tool used to control human beings.

Sophisticated humanoids raise a number of ethical issues, including the following:

- Loss of privacy for the human inhabitants if the robots are permitted free access to all rooms in a home.
- Ability of the robots to recognize commands that may lead to unethical behavior (e.g., to stealing a neighbor's camera or cell phone).
- Rights and responsibilities of the robots, e.g., should they be treated with respect as if they were human?
- Emotional relationships, e.g., how should a robot relate to human anger, say when the robot drops a dish of food on the floor? In other words, is it ethical to yell at a robot? Can and should robots be punished for misbehavior, and, if so, how?
- How should a robot react to multiple instructions from different humans, e.g., when a child calls for it to come and play while the mother calls for it to come and wash the dishes?
- Can the robot's computer be accessed by hackers, so it may stake out and send pictures from the home to potential burglars?

We believe these issues will need to be addressed before humanoid personal service robots come into common use.

80.10.4 Healthcare and Life Quality

Another area of service robot where robot-human interaction is developing rapidly is the field of healthcare, including nursing, surgery, physical therapy, and non-contact assistance during therapy and rehabilitation. These developments are becoming possible as the potential danger to humans from accidental robot activity decreases. This area of robotics is growing so rapidly that we can only indicate some typical applications.

From the social and ethical standpoint, this is one of the fields in robotics that bears the most sensitive safety and ethical problems. From a technical point of view, scientists in robot-surgery are working on the problems of reduced dexterity, workspace, and sensory input and possible fatal trouble, which could originate from a breakdown of surgical robot systems. Also issues of size, cost, and functionality should be addressed in surgery and haptic and assistive robotics.

In the context of assistive technology, some questions can be posed concerning the relationship between patients and health structures in which they are treated. Are we going to mechanize hospitals and to dehumanize our patients? Shall we improve our health structures, where human nurses can care about patients? Can we not develop new psychological and physical dependen-

cies? As a general principle of awareness, we should underline that the high cost of robotic systems in the medical field could widen the digital divide between developed and developing countries; and between layers of the same population.

The field of implantations raises concerns related to the fact that direct brain interfaces may at the same time pose ethical questions related to the enhancement of human function.

Assistive Robots

Assistive technology will help many people to conduct a more independent life.

Nursing care is typically a one-on-one relationship between a patient and a caregiver. Hence, it is an expensive part of healthcare, and a number of laboratories are developing robots that we may term *nurse's assistants*. One of the earliest of such robots was the wheeled HelpMate, currently marketed by a company named Pyxis. HelpMate assists nurses and other hospital personnel by smoothly transporting pharmaceuticals, laboratory specimens, equipment and supplies, meals, medical records, and radiology films back and forth between support departments, nursing floors, and patient rooms. HelpMate is able to navigate hospital corridors, avoid collisions with humans, summon the elevator, and locate a specific patient's room. Carnegie Mellon University and the University of Pittsburgh have developed a *nurse-bot* named Pearl [80.46] as an assistant who visits elderly patients in hospital rooms, provides information, reminds patients to take their medication, takes messages, and guides residents. Such robots are usually constructed as upright structures on wheels, with a somewhat human-appearing head containing cameras and voice synthesizing software for communication. They usually also have a digital display, on the head or chest, to display messages. In Europe, there have been (and are) a number of projects in this area, such as the Care-o-Bot developed in 2010 at the Fraunhofer Institute in Stuttgart, Germany. The Care-o-Bot also has an arm to assist in pick-and-place operations. Similar projects exist in Japan, South Korea, and other countries. A different approach to robots in healthcare involves the use of a mobile tele-presence robot under control of remote physician, which can travel throughout a hospital and visit patients. The physician's face appears on a large screen, allowing him or her to interact with the patient. Cameras on the robot allow the physician to see the patient, and verbal communication is possible. Such robots can decrease the need for doctors to visit patients in person, without decreasing the patient's ability to interact. Such a robot, named RP-VITA, was announced in 2012 by a joint venture of the robotics company iRobot and InTouch Health, a pioneer in remote presence robotics for medical applications.

A related set of projects involves *assistive robots*, which provide verbal guidance, encouragement, and interaction to people recovering from strokes, spinal injuries, as well as companionship to children with autism spectrum disorders. These robots do not make any physical contact with the subjects, but rather guide them through exercises and activities by voice and demonstration [80.47].

The humanoid Wakamaru, produced by Mitsubishi in 2010 was designed to co-inhabit living spaces with humans and was termed a *companion robot* for elderly and disabled persons. It is about 4 ft, or 120 cm, in height, has a head with large eyes, and movable arms but no legs; it moves on wheels, thus being restricted to relatively flat locations. It recognizes some 10 000 words, can place telephone calls, and communicate by internet. It carries a camera; if contacted by internet, it displays the camera image to the caller, and it recognizes 10 faces and can be programmed to react appropriately to each. It can read the owner's email and scan the news, passing the information on by voice. It communicates both by speech and gestures. Such robots are provided with binocular vision, hearing, touch, and other sensors, and can interact with humans by voice, gesture recognition, and safe navigation in the home. Mitsubishi originally made 100 of these robots for sale and then provided a rental service in Japan. It has not been available since 2012.

Among the potential ethical concerns in the use of such assistive robots and nurse-bots are the following:

- Patients may become emotionally attached to the robots, so that any attempt to withdraw them may cause significant distress.
- The robots will not be able to respond to patients' anger and frustration, except by calling for human help. For example, a patient may refuse to take the medication offered by the robot, throw it on the floor, and even attempt to strike the robot.
- A robot may be called by more than one user and not have the ability to prioritize the requests, thus causing anger and frustration.

More complete discussions of these topics may be found in the work of *Bekey* and *Abney* [80.48], and *Bekey* [80.49].

Surgical Robotics

Surgical robotics allows minimally invasive surgery, which can reduce patient recovery time. It could improve accuracy and precision. Robotics system increases the precision of microsurgery and enhances

the performances of complex therapies. Surgical robots can restore a surgeon's dexterity. Robo-surgery is applied to very delicate neurological procedures that are practically impossible to perform without robotic assistance.

The field of surgery is entering a time of great change, spurred on by remarkable recent advances in surgical and computer technology. Computer-controlled diagnostic instruments have been used in the operating theater for years to help provide vital information through ultrasound, computer-aided tomography (CAT), and other imaging technologies. Recently, robotic systems have made their way into the operating room as dexterity-enhancing surgical assistants and surgical planners, in answer to surgeons' demands for ways to overcome the surgical limitations of minimally invasive laparoscopic surgery, a technique developed in the 1980s. On July 11, 2000, FDA (Food and Drug Administration) approved the first completely robotic surgery device.

Typical applications are:

- Robotic telesurgical workstations
- Robotic devices for endoluminal surgery
- Robotic systems for diagnosis (CAT, NMR, PET)
- Robots for therapy (laser eye treatment, targeted nuclear therapy, ultrasonic surgery)
- Virtual environments for surgical training and augmentation
- Haptic interfaces for surgery/physiotherapy training.

Biorobotics and Biomechatronics

Biorobotics, while enhancing the quality of life after diseases or accidents, provides tools for studying biological behavior and brain functions, and is a test bed for the study and evaluation of biological algorithms and modeling.

According to *Dario* [80.50]:

> *Biorobotics is a new scientific and technological area with a unique interdisciplinary character. It derives its methodology mainly from the sectors of robotics and Biomedical Engineering, but also includes knowledge from, and provides useful applications to, many sectors of engineering, basic and applied sciences (medicine, neuroscience, economics, law, bio/nanotechnologies in particular), and even the humanities (philosophy, psychology, ethics). Biorobotics offers a new paradigm for engineers. The engineer no longer just cooperates with neuroscientists, but has also become a scientist in order to discover basic biological principles that make their job easier.*

Novel, high-performance bio-inspired machines and systems, for many different potential applications, are being designed and fabricated. This implies the development of (nano, micro, macro) novel devices for diagnosis, surgery, prosthetics, rehabilitation and personal assistance; and devices for biomedical applications (e.g., mini-invasive surgery and neuro-rehabilitation).

Biorobotics can be used for human prostheses for locomotion, manipulation, vision, sensing, and other functions:

- Artificial limbs (legs, arms)
- Artificial internal organs (heart, kidney)
- Artificial senses (eye, ears, etc.)
- Human augmentation (exoskeleton).

This field has important connection with neuroscience, to develop neural interfaces and sensory-motor coordination systems for the integration of these bionic devices into the human body/brain.

80.10.5 Distributed Robotic Systems

The fast growth of many wireless systems makes possible to link all robots to the Web, and among themselves. The development of cloud systems allows control of any addressable device, anywhere on the network, and every device on the network can connect to one another, creating intelligent robot groups.

Cloud, or network robotics will allow remote human–robot interaction for tele-operation and tele-presence; and, also, robot–robot interaction for data-sharing and cooperative working and learning. When Web speed will be comparable to that of the internal network of the robot, the machine will explode into a set of specialized systems distributed over the net.

Complex robotic systems will be developed, constituted by a team of co-operating robotic agents/components connected through ICT and cloud technologies:

- Networked knowledge system
- Networked intelligence systems
- Multi-robot systems.

Multi-robot systems are self-organizing robot teams consisting of a large number of heterogeneous team members. The organization in robot teams or squads is needed to perform specific tasks that require automatic task distribution and co-ordination at a global and local level; and when central control becomes impossible due to large distance and lack of local information, time of signal traveling.

A full-scale ecological robot team will be of tremendous value in a number of applications such as security, surveillance, monitoring, gardening, and pharmaceutical manufacturing. In addition, the co-ordination of heterogeneous teams of robots will also be of significant value in terms of planning, co-ordination, and use of advanced manufacturing systems. The benefits of the robot teams are manifold; e.g., an increase in the efficiency of the performance of complex tasks and in the capability to manage large-scale applications. They provide abundant and replaceable, interchangeable agents and they raise the reliability of the robots, because the group can perform even after losing most of its parts.

On the other hand, scientists should be aware of some of the risks in applying robot teams, for instance the increasing dependability of primary services on complex systems, and the unpredictability of robot team behavior. From a criminal point of view, the risks include the assignment of liability for misbehavior or crimes, vulnerability to hackers, and concerns about privacy.

80.10.6 Outdoor Robotics

Outdoor robots are intelligent machines that explore, develop, secure, and feed our world and worlds beyond land. Robots could be employed in dangerous operations, such as laying explosives, going underground after blasting to stabilize a mine roof, and mining in areas where it is impossible for humans to work or even survive.

They can work in the following environments:

- *Land*:
  - Mining (automated load-haul-dump trucks, robotic drilling and blasting devices)
  - Cargo handling (cranes and other automation technology for cargo lift on/lift off)
  - Agricultural (autonomous tractors, planters and harvesters, applicators for fertilizers and pest control)
  - Road vehicles (autonomous vehicles for humans or cargo transportation)
  - Rescue robotics (robots that support first response units in disaster missions)
  - Humanitarian demining (robots for detecting, localizing and neutralizing land mines)
  - Environmental protection (robots for pollution cleaning and decommissioning of dangerous plants).
- *Sea*:
  - Research (marine robots for oceanography, marine biology, geology)
  - Offshore (underwater robots for inspection, maintenance, repair and monitoring of oil and gas facilities in deep and ultra deep waters)
  - Search and rescue (underwater robots for first response intervention in casualties at sea, like submarines run aground).
- *Air*:
  - Unmanned aerial vehicles (UAV) (autonomous airplanes for weather forecasting, environmental monitoring, road traffic control, large area survey, patrolling).
- *Space*:
  - Space exploration (deep space vehicles, landing modules, rovers)
  - Space stations (autonomous laboratories, control and communication facilities)
  - Remote operation (autonomous or supervised dexterous arms and manipulators).

Mobile robots in particular can be highly valuable tools in urban rescue missions after catastrophes like earthquakes, bomb or gas-explosions, or everyday incidents like fires and road accidents involving hazardous materials. Robots can be used to inspect collapsed structures, to assess the situation, and to search and locate victims). Among the benefits of employing such robots is a rise in the efficiency of the exploitation of natural resources, to increase food production for the world population.

Concerning space robotics, it is obvious that, on the basis of current knowledge and technology, the robot can be our pioneer in space travels and mission of exploration of the far planets of the solar system and beyond.

On the social front, the unrestrained and planned use of outdoor robots can extend to its limit the excessive anthropization and exploitation of the planet, which can, in turn, become a threat to biodiversity and all the other forms of life on the planet. As for all other branches of robotics, this could lead to technology addiction. Furthermore, given the versatility of these robots, they can also be used the wrong way round, when reconverted from civilian use to devices for warfare and misuses (terrorism, pollution).

80.10.7 Military Robotics

After numerous science fiction stories about war-fighting robots during the twentieth century (and earlier), the stories have become reality in the twenty-first century. While military robots are now used extensively by the United States, a number of other countries are also developing robot weapons and, unfortunately, the world

has yet to come to terms with the ethical issues that arise from the use of such systems.

Humanity has been involved in war from the earliest days of homo sapiens on Earth. A famous psychologist published a book a few years ago entitled *The Terrible Love of War* [80.51]; this title summarizes the issue. The love of war has resulted in a love of weapons, as efficient ways to kill the enemy. Further, much of the development of weapons has been devoted to finding ways of inflicting death at a distance, to avoid injury to the attacker. Attacking an enemy with a club allows the victim to strike back, but attacking with bows and arrows allows the attacker to kill at a distance. Clearly, the development of muskets and guns allowed for greater and greater distances. This trend continues with military robots. Today there are many categories of robot vehicles equipped with weapons. Each of these systems can be controlled from a distance, allowing the owners of the vehicles to stay out of danger.

- *Intelligent weapons*: This field comprises all devices resulting from the development of traditional systems by robotics technology (automation, artificial intelligence, etc.):
  - Integrated defense systems: AI systems for intelligence and surveillance controlling weapons and aircraft capabilities
  - Unmanned ground vehicles (UGVs): autonomous tanks, armored vehicles carrying weapons and/or tactical payloads
  - Intelligent bombs and missiles
  - UAVs: also referred to as autonomous flying vehicles (AFVs). Unmanned spy planes and remotely piloted bombers
  - AUVs (autonomous underwater vehicles): intelligent torpedoes and autonomous submarines
- *Robot soldiers*: Eventually humanoids may be employed to substitute humans in performing *sensitive* tasks and missions in environments populated by humans. The main reasons for using humanoids are to permit a one-by-one substitution, without modifying the environment, or the engagement rules. This could be required where a safeguard of human life is considered a priority in many different scenarios:
  - Urban terrain combat
  - Indoor security operations
  - Patrol
  - Surveillance.

  Outdoor security robots could be able to make their night watch rounds and even chase criminals, directed by a remote control system via an internet connection or moving autonomously via their own artificial intelligence systems.
- *Superhuman*: There are several projects aimed at developing a superhuman soldier. Actually, the human body cannot perform tasks with the strength, speed, and fatigue resistance of the machines. *Augmentation* indicates the possibility of extending human's existing capabilities through wearable robot exoskeletons, to create superhuman strength, speed, and endurance:
  - Artificial sensor systems
  - Augmented reality
  - Exoskeletons.

Drones

Among the most visible of robotic vehicles capable of lethal force are semi-autonomous flying robots equipped with guided missiles, which are capable of flying over long distances to targets identified by GPS coordinates. These vehicles are known as *autonomous combat flying vehicles* (ACFVs), or more commonly as *drones*. While such vehicles are autonomous robots as far as flying is concerned (including takeoff and landing), they can fire lethal weapons only by human command. Drones flying over Afghanistan or various targets in Africa are controlled from Creech Air Force Base (near Las Vegas), or a base in New Mexico, thousands of kilometers away from the vehicles themselves. The drone carries extremely high-resolution cameras; the images are transmitted to the human controllers who can steer the aircraft so that a target is within the cross hairs of the on-board missiles. Human controllers activate and release the weapons. In the USA at the present time no flying robot is given the autonomy needed to release weapons without human supervision. Early 2013, there were an estimated 7500 drones in the US military arsenal, many flown in secret missions by the Central Intelligence Agency.

Drones (and other robot vehicles on land or under water) have been discussed in detail in recent books and other publications. Among the most influential authors are Sharkey, Singer, and Arkin. *Sharkey* from the University of Sheffield in the UK is a prolific writer on the subject [80.42, 52–54]. *Singer*, a researcher at the Brookings Institution in Washington, DC is a prolific author and has published a highly influential book entitled *Wired for War* [80.43]. *Arkin* from the Georgia Institute of Technology worked under a US Army contract to develop an *ethical governor* for killer robots, to be included in their software system [80.55]. A very complete description of military robots can be found in the book by *Ramo* entitled *Let Robots Do the Dying* [80.56]. While 10 years ago the USA was the

only country possessing autonomous flying vehicles, in 2012 it was estimated that 70 countries have AFV programs, but only about 15 of these have military drones.

The ethical issues arising from the use of drones include the following:

- Unintended injury or death of innocent non-combatants (usually known as *collateral damage*). While the missiles launched by a drone are extremely precise in steering to their intended targets, nearby civilians are frequently killed as well.
- The very fact that the human controllers who release the weapons are very far away, so that they do not see the blood and destruction directly but only from the drone's cameras means that for some of them such activities are more like a video game rather than the killing and destruction of human beings.
- Drones are used to attack suspected terrorists in countries that (officially) are not at war. Hence, they are used in undeclared wars. This may be a violation of international law, and it certainly raises ethical issues. It has been suggested that the availability of such military robots could make it easier for a country to start a war, since its own citizens would not be in danger.

Other Military Robots

Robot vehicles known as *Packbots* have been used in Iraq and Afghanistan for nearly a decade to locate and destroy improvised explosive devices (IEDs). Prior to the development and deployment of these robots (manufactured by iRobot, Inc. in the USA), IEDs hidden in bushes and behind rocks were the major cause of US soldier casualties in Iraq. Now, these small tank-like vehicles that can be carried by a soldier in a backpack are able to locate and destroy the mines.

Robots are currently being used as *mules* to assist soldiers in carrying heavy equipment, e.g., *Big Dog*, a four-legged robot built by Boston Dynamics to carry heavy weights over rough terrain [80.57]. The military also have robots designed to retrieve a wounded soldier and carry him or her to a safe location for medical help, and a variety of other special-purpose robots. Finally, there are programs in the USA and in Japan to provide an exoskeleton for a human soldier, enabling him or her to carry much larger loads than would be possible without assistance.

In conclusion. The benefits of military robots are as follows:

a) Tactical/operational strength superiority
b) Unemotional behavior
c) Limited loss of human lives in the robotized army
d) Better performances of superhuman versus human soldier.

Problems could arise from the following:

a) The inadequacy to manage the unstructured complexity of a hostile scenario
b) The unpredictability of machine behavior
c) The assignment of liability for misbehavior or crimes
d) The increased risk of starting a video game-like war, due to the decreased perception of its deadly effects.

Humans in mixed teams could face practical and psychological problems in having to distinguish humans from robots. Superhuman soldiers could also tackle problems of overstress and de-humanization.

It is clear that military robots are here and they have changed the nature of warfare dramatically. However, there are currently no international treaties or agreements governing their use, which raises serious ethical questions. The growing deployment of military robots (drones, mobile robots, exoskeletons) in many war theaters, and especially in the *war on terrorism* has caused the birth of a movement devoted to stopping *killing robots*. Likewise, some roboticists have firmly opposed the fact that autonomous robots have the *license to kill* humans because, in their judgment, the responsibility of a human life should always fall on another human being [80.58]. Military robotics should be thoroughly examined by specialized international organizations, as is the case with every type of military technology, to be regulated by international conventions or agreements.

80.10.8 Educational Robot Kits

The beneficial applications of robotics in education are known and documented. Robotics is a very good tool for teaching technology (and many other subjects) whilst, at the same time, always remaining very tightly anchored to reality. Robots are real, three-dimensional (3-D) objects that move in space and time, and can emulate human/animal behavior; but, unlike video games, they are real machines, true objects, and students learn much more quickly and easily if they can interact with concrete objects as opposed to formulas and abstract ideas.

In the age of electronics, computers and networks, it is necessary to modernize not only the contents and tools, but also the educational methods of traditional schools. It is also important to consider that the lifestyle of young people has changed, as well as the communication tools they use in their free time. Today, young

people communicate via the Internet and mobile telephones using e-mail, sms, and chat rooms, which allow them to be continually connected to a global community that has no limits regarding location and time.

Young people spend more time playing video games, playing with mobile phones, or downloading files from the Internet. These activities provide them with experiences, which are by now at the same standard as the most sophisticated technological systems. All this has accelerated the pace of life, so much so that fruition and consumption of experiences are both real and virtual. In fact, we are entering the age of cyberspace, which will not replace normal life relationships, but will certainly alter their characteristics.

In this context, we need to consider that traditional teaching and classical tools of support (books, documentaries) are at risk of becoming unsuitable when compared with the everyday possibilities offered to these young people by the world of mass media. Therefore, it is necessary begin to plan new ways to transmit knowledge which exploit the potential of new technology.

Learning about robotics is important not only for those students who want to become robotics engineers and scientists, but for every student, because this provides a strong method of reasoning and a powerful tool for grappling with the world. Robotics collects all the competencies needed for designing and constructing machines (mechanics, electrotechnics, electronics), computers, software, systems of communications, and networks. The special features of robotics boost student creativity, communication skills, cooperation, and teamwork. Learning about robotics promotes students' interest in and commitment to traditional basic disciplines (math, physics, technical drawing). Robotic construction kits, conjugating the physical building of artifacts with their programming, can foster the development of new ways of thinking that encourage new reflections on the relationship between the following:

a) Life and technology
b) Science and its experimental tool set
c) Robot design and values and identity [80.59].

80.10.9 Entertainment Robotics and Art

Entertaining robots are already used in featuring and advertising corporate logos, products, and events. These are marketing tools shown off by manufacturers on special occasions. Robot toys fall into the same category. They can be specifically designed to stimulate children's creativity and the development of their intellectual faculties. They can become children's companions, and – for the only child – could play the role of *friends*, *brothers*, or the traditional *imaginary fiend*. They could be used in the pedagogical assistance of autistic children.

On the negative side of technology, robot toys could cause psychological problems, such as:

- Lost touch with the real world
- Confusion between natural and artificial
- Confusion between real and imaginary
- Technology addiction.

Other entertainment robots will enable building real environments, which could be either perfect (or scaled) copies of some of existing environments, or reconstructions of settings that existed centuries/millennia ago, and which we can be populated by real or imaginary animals. Robots and robotics settings will make it possible to build up natural phenomena and biological processes, even cruel ones, but without involving living beings. In these settings, the users/audience could live interactive experiences, which are *real*, not only *virtual*. As extraordinary theatrical machines, robots will develop ever more *realistic* special effects.

In this frame, we should also consider sexual robots. These robots will be an important market field. They are used as sexual partners in many fields, from therapy to prostitution from the social standpoint. Some authors have proposed that their use could decrease the sexual exploitation of women and children [80.60]. They could also raise problems related to intimacy/attachments and safety and reliability.

The role of robotics in contemporary art, along with all the interactive artistic expressions (telecommunications, and interactive installations), is gaining importance and success. Artists are employing advanced technologies to create environments and works of art, utilizing the actuators and sensors to allow their robots to react and change in relation to viewers.

Robotic art will spread because of the following:

- It recalls (and is inspired by) the mythological traditions of various cultures. These traditions have created fantastic synthetic creatures.
- Robots exert a special fascination on the population at large.
- Robots can be used as tools in artwork and enable the building of artistic expression in shorter times, thus expanding the borders of human creativity.
- Robots can also perform actor's rules and allow playing living art.

The social and individual problems that can be produced by robotic art are, on the one hand, the dissemination of misinformation (wrong information on technology can be spread through art). On the other hand, technology can prevail over creativity.

80.11 Roboethics Enforced: From Ideals to Rules

At the conclusion of our taxonomy, and having analyzed the ethical issues in robotics outlining the main roboethics principles, we have to ask ourselves how roboethics conception and vision could become laws, rules, and standards. This question is more general: How have ethical principles been incorporated into laws and right, after having been discussed in transdisciplinary assemblies, expressed by warnings or the public's concern, suggested by religious personalities, theologians, and moral leaders, and/or forwarded by a community of concerned scientists? How is ethical thrust embodied in the R&D activity without imposing on it unjustified restrictions, so depriving the scientist of his/her own freedom of thought?

Along millennia of the history of science and technology, society has envisaged ways to express their ethical concern. These include the following. The professional *oath* is either a statement or a promise expressed by a new entry in professional careers to be faithful to the traditional values of the professional order he/she is entering into. The ancient *Hippocratic oath* is the recurrent example for other initiatives to develop and implement codes of conduct for scientists in general, and scientists in specific areas in particular.

A *manifesto* is a public declaration of intentions, opinions, objectives, or motives, often issued by a private organization or a government. For example, the Russell–Einstein Manifesto of 1955 is a public declaration against war and the further development of weapons of mass destruction.

A *statement* or a *declaration* can be employed to underline a given topic. As such, it can be either weakly or strongly prescriptive, morally or legally binding. For instance, during the World Robot Conference which took place in Fukuoka, Japan, the participants released a three-part list of *expectations for next-generation robots*, called the *World Robot Declaration* issued on February 25, 2004. It states that:

- Next-generation robots will be partners that coexist with human beings.
- Next-generation robots will assist human beings both physically and psychologically.
- Next-generation robots will contribute to the realization of a safe and peaceful society.

A *recommendation* serves to induce acceptance or favor. It is a prescription only in the weak sense of offering advice; a normative suggestion that is neither legally nor morally binding. More conclusive is the *appeal*, an earnest request for support: a petition, entreaty, or plea.

A *resolution* is a formal expression of opinion or intention made (usually after voting) by a formal organization, legislature, or other group.

In the last 50 years, many professional associations have adopted their *code*, a written text that offers a collection of laws, regulations, guidelines, rules, directives or principles for moral conduct. The *guiding principles* of the code of research ethics are *non-malfeasance* and *beneficence*, indicating a systematic regard for the rights and interests of others in the full range of academic relationships and activities. *Non-malfeasance* is the principle of doing, or permitting, no official misconduct. It is the principle of doing no harm in the widest sense. *Beneficence* is the requirement to serve the interests and well-being of others, including respect for their rights. It is the principle of doing well in the widest sense.

The *charter* is an ancient form of agreement. An example is the Charter of the United Nations. The charters have a legal character and are connected, in principle, to sanctions when not properly executed. Other means of implementing ethical concerns in science and technology are the *convention*, a form of agreement, or a contract, and also a practice established by general consent.

From a technical point of view, to ensure that robot and robotics devices adhere to the established applied ethical principles, *standards* have to be established. Generally, these standards are the results of the work of professional committees, which are expressed by professional and control bodies. At national level there are committees like ANSI (USA) or DIN (Germany). At the international level, ISO. Inside ISO is working a committed devoted to establishing technical standards for industrial and service robotics (TC184/SC2 robots and robotic devices). This committee is organized in several sub-committee groups: WG1 *vocabulary* to compile terms and definitions from all standards dealing with robots, WG3 *industrial safety* for the assurance of safety in design and construction of the robot and for the safeguarding of personnel during robot operations, WG 7 *personal care safety* on the development of safety standards for personal care robots, and WG 8 *service robots* on the development of safety standards for service robots.

Finally, in order that all previous instances and applied ethical principles be enforceable, they should become *national laws*. Each national government must incorporate those principles and standards in their own legislation, developing the required technical standards of implementations, and setting up adequate rules for violations.

80.12 Conclusions and Further Reading

We are witnessing a true, growing interest in robots in the general public, who are often more excited than the insiders, whose feelings swing between a position of cultural indifference to a behavior dictated by external pressures, be they political or industrial. We are also noticing the modern change – which already happened in the 1970s in the field of computer science – of the transformation of the robot from a research platform and a working tool to a consumer goods and an object of entertainment. It is a juvenile phenomenon, as shown by the increase of robotics contests among high school students. Today's young people who are getting their hands in the robotics kits will be the robotics professionals and the consumers of tomorrow.

A growing interest about the social effects of robotics is easy to observe among international professional associations and orders, stretching over the sister fields of computer ethics and bioethics. Certainly, roboethics is still far from being a well-established applied ethics, and by *well-established* we mean it has acquitted two qualities: on the one hand, roboethics should be universally accepted and standardized – or, at least, adopted by some communities, relevant in size and in political/economic/cultural influence. On the other, it should be embodied in the practice of the design, production and use of robots.

In fact, considering the history of the two widely applied and structured ethics which are extensively studied and which reach a certain organic unity, bioethics and computer and information ethics [80.61], we acknowledge that their development, which has been happening for over 30 years, came about with leaps and contradictions, chasms, and bends, and that they are far from being a suitable ethical standard shared by a plurality of subjects. Both these ethics were born in a policy and legislative vacuum, as technological changes have outpaced ethical developments, bringing about unanticipated problems [80.16].

The standardization of roboethics requires the accomplishment of some fundamental steps, both culturally and institutionally. From the general standpoint, it demands that the application of robotics to the human environment, especially to sensitive areas of human life, will be accepted by the quasi-totality of cultures, as has happened with other techno-scientific innovations such as electricity and computer systems. (In the case of free access to the Internet, the issue is still questionable in many of the world's nations.)

Should such a level be achieved, roboethics would have already passed the phase of being adjusted to fit different answers and situations, to being modified to the point of having acquired the capability of adapting to the different points of views. Different cultures and religions regard the intervention in sensitive fields like human reproduction, neural therapies, implantations, and privacy differently. These differences originate from the cultural specificities regarding fundamental issues. For example, the limit between a human and a cyborg, the separation between the natural and the artificial [80.62], the difference between human and artificial intelligence, the border between privacy and traceability of actions, the concept of integrity and the unity of the human being, the acceptance of diversity (in gender, ethnicity, minorities, etc.), the boundary between replacement and human enhancement, and so on. These are all milestones in defining the underlying paradigms, which, in turn, influence the day-by-day behavior of everyone.

From the experience provided by more than 30 years of discussions and disputes in the fields of bioethics and information ethics, we know that none of the achievements in the field of science and ethics are easy or negligible. For those who wish to thoroughly investigate some elements of the philosophy of science, the history of science and ethics, science and engineering ethics, the law applied to science and technology, we now suggest some fundamental steps.

In the field of the moral theories related to science and technology, we mention the considerable work of *Beauchamp* of the Kennedy Institute of Ethics [80.63] and of *Jasanoff* at Harvard University Program on Science and Technology Studies [80.64]. Two important annual gatherings of computer philosophy, CEPE (Computer Ethics Philosophical Enquiry) and IACAP (International Association for Computing and Philosophy) have recently added *roboethics* as one of their key topics.

Furthermore, it is very useful to follow the activity of the regulatory bodies entitled to deal with the issues of science and ethics. In accordance with what was said in Sects. 80.6–80.8 of this chapter, the person's interest should start from the general principles which are essentially accepted by most of the world's nations (at least nominally), and come down to the specific applications in our field.

The issue of the influence and the pressure given by the market on science and on R&D is handled by the applied ethics is called business ethics. In this frame, corporate social responsibility is one of the ways in which enterprises can affirm ethical principles and values. This state of view was introduced to the USA 15 years ago (especially in the field of healthcare) and is still running today. Also, training in the responsible conduct of research (RCR) has been adopted by the

USA, and is still continuing in the present time. The domain of RCR training does not include solely the ethical dimensions of research with human subjects, but every intricate dimension of responsible conduct in the planning, performance, analysis, and reporting of research. Difficulties here arise from the small amounts of resources allocated. See also the ethical paragraphs of the OECD (Organization for Economic Cooperation and Development) guidelines for multinational enterprises.

Concerning the philosophical and epistemological aspect of the issue in ethics and robotics, one of the main problems that people who are interested in roboethics will have to handle is the persistent confusion – ontologically, but especially linguistically – between human and artificial intelligence, as well as between other fundamental concepts of perception, consciousness, self-consciousness, emotions, and so on, as applied to humans and machines. It must be clarified that actual contemporary roboethics is human ethics as applied to robotics, which is considered non-human. A strong base of human roboethics is needed in order to construct the foundations to knowing how to responsibly manage the final problem, the one which, so far, no-one knows how to answer: can robots ever become *humans* [80.65]?

The need for serious and thorough work on the concepts of intelligence, knowledge, conscience, autonomy, freedom, free will, etc., is highlighted. Indeed, the heterogeneous composition of specialists often leads to incessant discussions about the meaning of words, rather than on the contents of the myriad of, often pressing, issues that need to be faced.

From all this, the necessity that the international community of robotics becomes the author of its own destiny is born, so as to directly face the task that needs defining, whilst collaborating with academics in the field of philosophy, of law, and generally experts of human sciences, the ethics and social aspects of their researches and applications of the former. Nor should they feel relegated to a merely techno-scientific role, delegating to others the task of reflecting and taking action on moral aspects. On the other hand, a closed-shop attitude would be damaging to the development of robotics, given the interdisciplinary nature of the amount of research undertaken in the field of robotics.

From this point of view, roboethics cannot fail to be beneficial to robotics, framing the research in close connection with the end-users and society, so avoiding many problems that other *sensitive* fields are now facing. All this, and more, will be wishful thinking if in the engineers' curriculum of study subjects such as: scientific philosophy, history of science, law and politics of science (as is already happening in some advanced polytechnics) are not included. Once again, we have to say that the deeper study into the history of science, for example, especially in the 1800s and 1900s, cannot but aid a better understanding of that complex scientific galaxy which is robotics. Even a restricted knowledge of cybernetics and of computer science, from Wiener, to von Neumann, to Weizenbaum, will immediately and directly demonstrate that these scientists immediately took care of ethical and social aspects of their discoveries and realizations, which marked the beginning of the field of computers and robotics.

As an example – in 2012 some young engineers started a project called the Open Roboethics Initiative (ORI) [80.66]. In their words:

> *Open Roboethics is the idea that, just like how the Web has allowed us to produce and maintain contents of Wikipedia and software designs of Linux by means of mass collaboration, maybe roboethics discussions and robot designs can benefit from the power of mass collaboration.*

They are creating a central Web space where policy makers, engineers/designers, and users and other stakeholders of the technology can freely share and access roboethics related contents; by this, they hope to accelerate roboethics discussions and inform robot designs.

At the same time, it is necessary that those not involved in robotics keep up to date on the field's real and scientifically predictable developments, in order to base the discussions on data supported by technical and scientific reality, and not on appearances or emotions generated by science fiction. In particular, apart from, obviously, this handbook, one must look to serious magazines published by recognized scientific associations, and not rely on headlines about ambiguous and scandalizing creations, which do not exist.

Ethics is a thousand-year old human science with an impressive literature. Its application to the field of science technology is no doubt more recent, even though precedents such as the Hippocratic oath suggest an extremely ancient origin. The research on robotics is casting light on a manifold of issues across science and humanities. It is no wonder that it will open new and unexpected field of studies and application also in ethics.

Video-References

VIDEO 773 Roboethics: Introduction
available from http://handbookofrobotics.org/view-chapter/80/videodetails/773

VIDEO 774 Roboethics: Prosthesis
available from http://handbookofrobotics.org/view-chapter/80/videodetails/774

VIDEO 775 Roboethics: Military robotics
available from http://handbookofrobotics.org/view-chapter/80/videodetails/775

References

80.1 D.S. Landes: *The Unbound Prometheus: Technological Change and Industrial Development in Western Europe from 1750 to the Present* (Cambridge Univ. Press, Cambridge 2003)

80.2 G. Veruggio: A proposal for a roboethics, 1st Int. Symp. Roboethics Ethics Soc. Humanit. Ecol. Asp. Robotics, Sanremo (2004)

80.3 M. Santoro, D. Marino, G. Tamburrini: Learning robots interacting with humans: From epistemic risk to responsibility, AI Society **22**(3), 301–314B (2008)

80.4 B. Joy: Why the future doesn't need us. In: *Nanoethics*, ed. by F. Allhoff, P. Lin, J. Moon, J. Weckert (Wiley, Hoboken 2000) pp. 17–39

80.5 J. Rotblat: A Hippocratic oath for scientists, Science **286**(5444), 1475 (1999)

80.6 G. Veruggio: The birth of roboethics, Proc. IEEE Int. Conf. Robotics Automation (ICRA), Workshop Roboethics, Barcelona (2005)

80.7 I. Asimov: Runaround. In: *I, Robot* (Bantam Dell, New York 2004)

80.8 G. Veruggio: The roboethics roadmap, Proc. IEEE Int. Conf. Robotics Autom. (ICRA), Workshop Roboethics, Rome (2007)

80.9 G. Veruggio, K. Abney: Roboethics: The applied ethics for a new science in robot ethics. In: *Robot Ethics: The Ethical and Social Implications of Robotics*, ed. by P. Lin, K. Abney, G.A. Bekey (MIT Press, Cambridge 2012) pp. 347–363

80.10 A. Fabris: *Ethicbots: Ethics and Robotics* (Edizioni ETS, Pisa 2007)

80.11 M. Nussbaum, A. Sen: *The Quality of Life* (Clarendon, Oxford 1993)

80.12 M. Weber: Politics as a vocation. In: *The Vocation Lectures*, ed. by D.S. Owen, T.B. Strong, R. Livingstone (Hackett, Indianapolis 2003)

80.13 E. Morin: *La Methode. Etique* (Edition Du Seuil, Paris 2004)

80.14 R. von Schomberg: *From the Ethics of Technology Towards an Ethics of Knowledge Policy and Knowledge Assessment* (Publ. Eur. Commun., Luxembourg 2007)

80.15 A. Mowshowitz: Technology as excuse for questionable ethics, AI Society **22**(23), 271–282 (2008)

80.16 J. Moor: What is computer ethics? In: *Computer and Ethics*, ed. by T.W. Bynum (Blackwell, Oxford 1985)

80.17 N. Wiener: *Cybernetics, or the Control and Communication in the Animal and the Machine* (MIT Press, Cambridge 1948)

80.18 N. Wiener: *The Human Use of Human Beings: Cybernetics and Society* (Doubleday, Garden City 1954)

80.19 T. W. Bynum: Computer ethics: Basic concepts and historical overview, The Stanford Encyclopedia of Philosophy (Winter 2001 Edition), http://plato.stanford.edu/archives/win2001/entries/ethics-computer/

80.20 D. Parker: Rules of ethics in information processes, Communication ACM **11**(3), 198–201 (1968)

80.21 J. Weizenbaum: *Computer Power and Human Reason: From Judgment to Calculation* (Freeman, San Francisco 1976)

80.22 D. Johnson: *Computer Ethics* (Prentice Hall, Upper Saddle River 2001)

80.23 L. Floridi: Information ethics: On the philosophical foundation of computer ethics, Ethics Inf. Technol. **1**(1), 33–52 (1999)

80.24 V.R. Potter: *Bioethics: Bridge to the Future* (Prentice Hall, Eaglewood Cliffs 1971)

80.25 V.R. Potter: *Global Bioethics. Building on the Leopold Legacy* (Michigan State Univ. Press, East Lansing 1988)

80.26 J.M. Galvan: On technoethics, IEEE/RAS Magazine **10**, 58–63 (2003)

80.27 J. Illes, S.J. Bird: Neuroethics: A modern context for ethics in neuroscience, Trends Neurosci. **29**(9), 511–517 (2006)

80.28 J.J. Wagner, D.M. Cannon, M.M. Van der Loos: Cross-cultural considerations in establishing roboethics for neuro-robot applications rehabilitation, IEEE Int. Conf. Rehabil. Robotics (ICORR) (2005)

80.29 J.J. Rawls: *A Theory of Justice* (Harvard Univ. Press, Cambridge 1971)

80.30 R. von Schomberg: *From the Ethics of Technology Towards an Ethics of Knowledge Policy: Implications for Robotics, AI and Society* (Springer, London 2008)

80.31 European Union: *Charter of Fundamental Rights of the European Union* (2000IC 364101) (EU, Brussels 2000)

80.32 Wingspread Conference on the Precautionary Principle: The Wingspread consensus statement on the precautionary principle, http://www.sehn.org/wing.html (1998)

80.33 American Council of Engineering Companies Ethical Guidelines: http://www.acec.org/about/ethics/

80.34 R. Kurzweil: *The Age of Spiritual Machines: When Computers Exceed Human Intelligence* (Viking, New York 1999)

80.35 G.A. Bekey: *Autonomous Robots, from Biological Inspiration to Implementation and Control* (MIT Press, Cambridge 2005)
80.36 P. Asaro: Remote-control crimes, IEEE Robotics Autom. Mag. **18**(1), 68–71 (2011)
80.37 P. Asaro: On banning autonomous weapon systems: Human rights, automation, and the dehumanization of lethal decision-making, Int. Rev. Red Cross **94**(886), 687–709 (2012)
80.38 Y. Moon, C. Nass: Are computers scapegoats? Attributions of responsibility in human–computer interaction, Int. J. Hum.-Comput. Stud. **49**(1), 79–94 (1998)
80.39 D. Denneh: When HAL kills, Who's to blame? In: *HAL's Legacy: Legacy: 2001's Computer as Dream and Reality*, ed. by A.C. Clarke (MIT Press, Cambridge 1997)
80.40 R. Calo: Robots and privacy. In: *Robot Ethics: The Ethical and Social Implications of Robotics*, ed. by P. Lin, K. Abney, G.A. Bekey (MIT Press, Cambridge 2012)
80.41 D.E. Denning: Barriers to entry: Are they lower for cyber warfare?, IO Journal **1**(1), 6–10 (2009)
80.42 N. Sharkey: Cassandra or false prophet of doom: AI robots and war, IEEE Intell. Syst. **23**(4), 14–17 (2008)
80.43 P.W. Singer: *Wired for War: The Robotics Revolution and Conflict in the 21st Century* (Penguin, New York 2009)
80.44 AAVV: *World Robotics report 2014*, (International Federation of Robotics, Frankfurt 2014)
80.45 B. Reeves, C. Nass: *The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places* (Cambridge Univ. Press, Cambridge 1966)
80.46 R. Calo: The drone as privacy catalyst, Stanf. Law Rev. Online **64**, 29–33 (2011)
80.47 D. Feil-Seifer, M.J. Matarić: Defining socially assistive robotics, IEEE Int. Conf. Rehabil. Robotics (ICORR) (2005) pp. 465–468
80.48 G. Bekey, L. Abney: Ethical implications of intelligent robots. In: *Neuromorphic and Brain-Based Robots: Trends and Perspectives*, ed. by J.L. Krichmar, H. Wagatsuma (Cambridge Univ. Press, Cambridge 2011)
80.49 G. Bekey: Current trends in robotics: Technology and ethics. In: *Robot Ethics, The Ethical and Social Implications of Robotics*, ed. by R. Lin, K. Abney, G.A. Bekey (MIT Press, Cambridge 2012) pp. 17–34
80.50 P. Dario: Biorobotics science and engineering: From bio-inspiration to bio-application, FET Conf. Sci. Beyond Fiction, Prague (2009)
80.51 J. Hillman: *A Terrible Love of War* (Penguin, New York 2004)
80.52 N. Sharkey: The ethical frontiers of robotics, Science **322**, 1800–1801 (2008)
80.53 N. Sharkey: Death strikes from the sky: The calculus of proportionality, IEEE Technol. Soc. Mag. **28**(1), 16–19 (2009)
80.54 N. Sharkey: Killing made easy. In: *Robot Ethics: The Ethical and Social Implications of Robotics*, ed. by P. Lin, K. Abney, G. Bekey (MIT Press, Cambridge 2012)
80.55 R.C. Arkin: *Governing Lethal Behavior in Autonomous Robots* (CR, Boca Raton 2009)
80.56 S. Ramo: *Let Robots Do the Dying* (Figueroa, Los Angeles 2001)
80.57 M. Raibert: BigDog, the rough-terrain quaduped robot, Proc. 17th World Congr. Autom. Control (2009)
80.58 G. Veruggio: Roboethics applied to military robotics, The 3rd Sci. Peace World Conf., Milan (2011)
80.59 E. Micheli, M. Avidano, F. Operto: Semantic and epistemological continuity in educational robots programming languages, Int. Conf. Simul. Model. Program. Auton. Robots (SIMPAR) (2008)
80.60 D. Levy: *Love and Sex with Robots: The Evolution of Human-Robot Relationships* (HarperCollins, New York 2007)
80.61 T.W. Bynum, S. Rogerson: *Computer Ethics and Professional Responsibility* (Blackwell, New York 2004)
80.62 G.O. Longo: Body and technology: Continuity or discontinuity? In: *Mediating the Human Body: Technology, Communication, and Fashion*, ed. by L. Fortunati, E. Katz, R. Riccini (Lawrence Erlbaum, Mahwah 2003) pp. 23–29
80.63 T. Beauchamp, J. Childress: *Principles of Biomedical Ethics* (Oxford Univ. Press, Oxford 2001)
80.64 S. Jasanoff: *Designs on Nature: Science and Democracy in Europe and the United States* (Princeton Univ. Press, Princeton 2005)
80.65 F. Operto: Ethics in advanced robotics, IEEE Robotics Autom. Mag. **18**(1), 72–78 (2008)
80.66 ORI, Open roboethics initiative: http://www.openroboethics.org/

Acknowledgements

B.21 Actuators for Soft Robotics
by Alin Albu-Schäffer, Antonio Bicchi

The authors of this chapter have used liberally of work done by a group of collaborators involved in the EU projects PHRIENDS, VIACTORS, and SAPHARI. We want to particularly thank Etienne Burdet, Federico Carpi, Manuel Catalano, Manolo Garabini, Giorgio Grioli, Sami Haddadin, Dominic Lacatos, Can zparpucu, Florian Petit, Joshua Schultz, Nikos Tsagarakis, Bram Vanderborght, and Sebastian Wolf for their substantial contributions to this chapter and the work behind it.

C.29 Inertial Sensing, GPS and Odometry
by Gregory Dudek, Michael Jenkin

We would like to thank Sarah Jenkin for her help with the figures.

D.36 Motion for Manipulation Tasks
by James Kuffner, Jing Xiao

We acknowledge the contribution that the authors of the first edition made to this chapter revision, particularly Sect. 36.1, Sect. 36.2, and Sect. 36.5.

E.52 Modeling and Control of Aerial Robots
by Robert Mahony, Randal Beard, Vijay Kumar

The authors would like to acknowledge the contributions of Peter Corke to the background material on quadrotor vehicles (R. Mahony, V. Kumar, P. Corke: Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor, Robotics Autom. Mag. 19(3), 20–32 (2013)) and Eric Feron and Eric Johnson (E. Feron, E. Johnson: Aerial robotics. In: Springer Handbook of Robotics, ed. by B. Siciliano, O. Khatib (Springer, Berlin, Heidelberg 2008)) to the introductory material in this chapter.

F.57 Robotics in Construction
by Kamel Saidi, Thomas Bock, Christos Georgoulas

Disclaimer: Although certain commercial construction equipment are included in this chapter, the inclusion of such information should in no way be construed as indicating that such products are endorsed by the National Institute of Standards and Technology (NIST) or are recommended by NIST or that they are necessarily the best equipment for the purposes described.

F.58 Robotics in Hazardous Applications
by James Trevelyan, William Hamel, Sung-Chul Kang

James Trevelyan acknowledges Surya Singh for detailed suggestions on the original draft, and would also like to thank the many unnamed mine clearance experts who have provided guidance and comments over many years, as well as Prof. S. Hirose, Scanjack, Way Industry, Japan Atomic Energy Agency, and Total Marine Systems for providing photographs.

William R. Hamel would like to acknowledge the US Department of Energy's Robotics Crosscutting Program and all of his colleagues at the national laboratories and universities for many years of dealing with remote hazardous operations, and all of his collaborators at the Field Robotics Center at Carnegie Mellon University, particularly James Osborn, who were pivotal in developing ideas for future telerobots.

Sungchul Kang acknowledges Changhyun Cho, Woosub Lee, Dongsuk Ryu at KIST (Korean Institute for Science and Technology), Korea for their providing valuable documents and pictures. He also appreciates Munsang Kim for his leading projects that have produced many of research achievements related to Sect. 58.3 enabling technologies.

All three authors acknowledge comments from Prof. Cdric Pradalier of Georgia Tech Lorraine and Mark Noakes of the Oak Ridge National Laboratory in USA.

F.59 Robotics in Mining
by Joshua Marshall, Adrian Bonchis, Eduardo Nebot, Steven Scheding

The authors would like to thank Elliot Duff at CSIRO for his work in mustering the team for this chapter project. Thanks to Andrew Crose and Michael Lewis and Modular Mining Systems for help in obtaining information about the Komatsu AHS. Thanks also to Johan Larsson, Ola Pettersson, and Oscar Lundhede for facilitating the use of some Atlas Copco AB images and video.

F.62 Intelligent Vehicles
by Alberto Broggi, Alex Zelinsky, Ümit Özgüner, Christian Laugier

The authors acknowledge the contributions of Chuck Thorpe and Michel Parent, who made significant contri-

butions to the first edition of the handbook chapter, and part of their contributions remain in the revised second edition. We also acknowledge the helpful contributions of Dizan Alejandro Vasquez.

G.70 Human–Robot Augmentation

by Massimo Bergamasco, Hugh Herr

The authors wish to thank Jared Markowitz, Elliott Rouse, Carlo Alberto Avizzano, Marco Fontana, and Rocco Vertechy for their helpful assistance and suggestions in writing this manuscript.

G.74 Learning from Humans

by Aude Billard, Sylvain Calinon, Rüdiger Dillmann

We warmly thank Daniel Grollman and Stefan Schaal for participation in drafting earlier versions of this chapter.

About the Authors

Markus W. Achtelik

ETH Zurich
Autonomous Systems Laboratory
Zurich, Switzerland
markus@achtelik.net

Chapter B.26

Markus Wilhelm Achtelik received his Diploma degree in Electrical Engineering from the TU München in 2009. He finished his PhD in 2014 at the Autonomous Systems Lab (ASL) at ETH Zurich, and currently works as Postdoc at ASL on control, state estimation and planning, with the goal of enabling autonomous manoeuvres for MAVs, using an IMU and onboard camera(s) as main sensors.

Alin Albu-Schäffer

DLR Institute of Robotics and Mechatronics
Wessling, Germany
alin.albu-schaeffer@dlr.de

Chapter B.21

Alin Albu-Schäffer graduated in Electrical Engineering from the Technical University of Timisoara, in 1993 and got the PhD from the TU Munich in 2002. Since 2012 he is the Head of the Institute of Robotics and Mechatronics at the German Aerospace Center, which he joined in 1995 as a PhD candidate. Moreover, he is a Professor at the TU Munich's Computer Science Department. His research interests include robot design, modeling and control, flexible joints, as well as bio-inspired robot design.

Kostas Alexis

ETH Zurich
Institute of Robotics and Intelligent Systems
Zurich, Switzerland
konstantinos.alexis@mavt.ethz.ch

Chapter B.26

Kostas Alexis obtained his PhD in the field of aerial robotics control and collaboration from the University of Patras, Greece in 2011. Since then he holds a senior researcher position at ASL – ETH Zurich. His research interests lie in control and optimization focusing on aerial systems navigation. He is the co-author of more than 40 technical publications.

Jorge Angeles

McGill University
Department of Mechanical Engineering and Centre for Intelligent Machines
Montreal, Canada
angeles@cim.mcgill.ca

Chapter B.16

Jorge Angeles graduated as an Electromechanical Engineer and obtained the MEng degree in Mechanical Engineering, both at Universidad Nacional Autónoma de México (UNAM), then received the PhD degree in Applied Mechanics from Stanford University. Research activities include robot kinematics, dynamics, design, and control as well as design theory and methodology. Angeles is a Fellow of ASME, CSME, IEEE and RSC, The Academies of Arts, Humanities, and Sciences of Canada, and Honorary Member of IFToMM, the International Federation for the Promotion of Mechanism and Machine Science.

Gianluca Antonelli Chapter E.51 For biographical details see "About the Multimedia Editors"

Fumihito Arai

Nagoya University
Department of Micro-Nano Systems Engineering
Nagoya, Japan
arai@mech.nagoya-u.ac.jp

Chapter B.27

Fumihito Arai received Master of Eng. degree from Tokyo University of Science in 1988. He received Dr. of Eng. from Nagoya University in 1993. Since 1994, he was Assistant Professor at Nagoya University, from 2005 Professor at Tohoku University. Since 2010 he has been Professor at Nagoya University mainly engaged in micro- and nano-robotics.

Michael A. Arbib

University of Southern California
Computer Science, Neuroscience and ABLE Project
Los Angeles, USA
arbib@usc.edu

Chapter G.77

Michael Arbib's group at the University of Massachusetts Amherst (UMass) introduced the notions of opposition space and affordances to the study of brain mechanisms for primate and control of robot hands. At the University of Southern California, his group developed new models of primate visuomotor coordination, including the first computational model of mirror neurons. He continues to develop the mirror system hypothesis for the evolution of the language-ready brain.

J. Andrew Bagnell

Carnegie Mellon University
Robotics Institute
Pittsburgh, USA
dbagnell@ri.cmu.edu

Chapter A.15

Dr. J. Andrew Bagnell is an Associate Research Professor at Carnegie Mellon University's Robotics Institute and National Robotics Engineering Center (NREC) and the Machine Learning Department. His research focuses on machine learning and perception, adaptive control, optimization and planning under uncertainty. He received a BSc in Electrical Engineering from the University of Florida in 1998. He joined the Robotics Institute at CMU in 2000 receiving an MS and PhD in Robotics in 2002 and 2004, respectively.

Randal W. Beard

Brigham Young University
Electrical and Computer Engineering
Provo, USA
beard@byu.edu

Chapter E.52

Randal Beard is a Professor in the Department of Electrical and Computer Engineering at Brigham Young University. He received his PhD from Rensselaer Polytechnic Institute. His research interests include guidance and control of teams of unmanned air vehicles. He is a Fellow of the IEEE and an Associate Fellow of AIAA.

Michael Beetz

University Bremen
Institute for Artificial Intelligence
Bremen, Germany
ai-office@cs.uni-bremen.de

Chapter A.14

Michael Beetz is a Professor for Computer Science at the Faculty for Informatics of the University Bremen and Head of the Institute for Artificial Intelligence. He received his diploma degree in Computer Science from the University of Kaiserslautern and his MSc, MPhil, and PhD degrees from Yale University in 1993, 1994, and 1996, respectively. His research interests include plan-based control of robotic agents, knowledge processing, integrated robot learning, and cognitive perception.

George Bekey

University of Southern California
Department of Computer Science
Arroyo Grande, USA
bekey@usc.edu

Chapter G.80

George A. Bekey is Professor Emeritus of Computer Science at the University of Southern California. He was President of the IEEE Robotics and Automation Society. His current research interest is in robot ethics and is co-editor of *Robot Ethics*. Professor Bekey is a member of the US National Academy of Engineering and a Fellow of IEEE and other societies.

Maren Bennewitz

University of Bonn
Institute for Computer Science VI
Bonn, Germany
maren@cs.uni-bonn.de

Chapter E.45

Maren Bennewitz is currently an Assistant Professor for Computer Science at the University of Freiburg and Head of the Humanoid Robots Laboratory. In the last few years, she has been developing novel solutions for humanoid robots navigating in multilevel and multi-room indoor environments containing articulated and movable objects. These include probabilistic methods for 3-D environment modelling as well as for manipulation and navigation.

Massimo Bergamasco

Sant'Anna School of Advanced Studies
Perceptual Robotics Laboratory
Pisa, Italy
m.bergamasco@sssup.it

Chapter G.70

Massimo Bergamasco is Professor of Theory of Mechanisms and Machines at the Scuola Superiore Sant'Anna, Pisa, Italy. His research deals with the study and development of haptic interfaces and wearable robots for the control of the interaction between humans and virtual environments. Presently he is focused on general aspects of perception and cognitive processes. He has been the Founder of the Perceptual Robotics Laboratory at Sant'Anna.

Marcel Bergerman

Carnegie Mellon University
Robotics Institute
Pittsburgh, USA
marcel@cmu.edu

Chapter F.56

Dr. Marcel Bergerman is a faculty at Carnegie Mellon University and co-founder of Near Earth Autonomy, in Pittsburgh, USA. His career is devoted to the design, development, and deployment of robotic systems for field applications, in particular agriculture and defense. Dr. Bergerman is a co-founder of GreenE Academy, teaching programming and robotics to children. He got his PhD from CMU in 1996.

Antonio Bicchi

University of Pisa
Interdepartmental Research Center "E. Piaggio"
Pisa, Italy
bicchi@ing.unipi.it

Chapter B.21

Antonio Bicchi is Professor of Robotics at the University of Pisa, Senior Scientist at the Italian Institute of Technology in Genoa, and an Adjunct Professor at Arizona State University. His main research interests are in robotics, haptics, and control systems. He has published more than 400 papers. In collaboration with Pietro Pietrini and Emiliano Ricciardi, University of Pisa, and Marco Santello, Arizona State University, he has set up a multidisciplinary group to capitalize on the interaction between neuroscience and robotics studying human perception and motor control.

Aude G. Billard

Swiss Federal Institute of Technology (EPFL)
School of Engineering
Lausanne, Switzerland
aude.billard@epfl.ch

Chapter G.74

Aude Billard is Professor of Mechanical Engineering at the Swiss Federal Institute of Technology in Lausanne (EPFL), Switzerland. She holds a BSc/MSc in Physics from EPFL and a PhD from the University of Edinburgh. A. Billard received the Intel Corporation Teaching Award, the Swiss NSF Career Award in. Her research interests include robot learning, humanoid robotics, human–robot interactions and computational neuroscience.

John Billingsley

University of Southern Queensland
Faculty of Engineering and Surveying
Toowoomba, Australia
john.billingsley@usq.edu.au

Chapter F.56

John Billingsley graduated in Mathematics and Electrical Engineering from Cambridge University in 1960, later gaining a PhD in control theory there. In 1992 he took up a Chair of Engineering at the University of Southern Queensland and supervises Technology Research in the National Centre for Engineering in Agriculture (NCEA). In 2006 he received an achievement medal from the IET, London.

Rainer Bischoff

KUKA Roboter GmbH
Technology Development
Augsburg, Germany
rainer.bischoff@kuka.com

Chapter F.54

Dr. Bischoff is Head of KUKA's Technology Development Department responsible for research and technology development projects preceding product development. Dr. Bischoff received his Dr.-Ing. degree from Karlsruhe Institute of Technology. He serves as Vice-President Industry of euRobotics AISBL – the European Robotics Association and helped to united European roboticists.

Thomas Bock

Technical University Munich
Department of Architecture
Munich, Germany
thomas.bock@br2.ar.tum.de

Chapter F.57

Prof. Dr.-Ing. T. Bock focuses on the automation and robotization in construction, from planning through construction production and use phase to rebuilding and restoration. In this area he has published almost 400 refereed publications in book chapters, international journals and conferences. Since 1997 he holds the Chair for Building Realization and Robotics, Technical University Munich, Germany.

Adrian Bonchis

CSIRO
Department of Autonomous Systems
Pullenvale, Australia
adrian.bonchis@csiro.au

Chapter F.59

Adrian Bonchis has been working in the robotics space in mining, manufacturing, and agriculture for over 25 years. He completed his PhD in Systems and Control at the University of Sydney in 2000. Since commencing with CSIRO in 2005, his activities focused mainly on automating heavy mining equipment, delivering a number of innovative solutions to industry clients.

Josh Bongard

University of Vermont
Department of Computer Science
Burlington, USA
josh.bongard@uvm.edu

Chapter G.76

Josh Bongard is an Associate Professor in Computer Science at the University of Vermont, USA. He currently serves as a vice chair of the UVM Complex Systems Spire, and is the co-author of the popular science book *How the Body Shapes the Way We Think: A New View of Intelligence*. His interests include evolutionary robotics, crowdsourcing and machine science.

Wayne J. Book

Georgia Institute of Technology
G. W. Woodruff School of Mechanical Engineering
Atlanta, USA
wayne.book@me.gatech.edu

Chapter A.11

Wayne J. Book, HUSCO/Ramirez Distinguished Professor of Fluid Power and Motion Control (emeritus), has taught at the Georgia Institute of Technology since 1974. A Fellow of ASME, IEEE and SME, he holds degrees from MIT (MS and PhD) and the University of Texas at Austin (BS) in Mechanical Engineering. He was awarded the 2013 Robert E. Koski Medal by ASME. His research focuses on dynamics of fluid power and flexible systems, robotics, and haptic human interfaces.

Cynthia Breazeal

MIT Media Lab
Personal Robots Group
Cambridge, USA
cynthiab@media.mit.edu

Chapter G.72

Cynthia Breazeal is Associate Professor at MIT, where she founded and directs the Personal Robots Group at the Media Lab since 2001. She holds degrees from MIT (MS, PhD) in Electrical Engineering and Computer Science with specialization in Artificial Intelligence and Robotics and from UC Santa Barbara (BS) in Electrical and Computer Engineering. She is a pioneer of Social Robotics and Human Robot Interaction with particular focus on personal robots.

Oliver Brock

Technical University Berlin
Robotics and Biology Laboratory
Berlin, Germany
oliver.brock@tu-berlin.de

Chapter D.40

Oliver Brock is the Alexander von Humboldt Professor of Robotics in the School of Electrical Engineering and Computer Science at the Technical Univerisity Berlin, Germany, since 2009. He received his Diploma in Computer Science in 1993 from the TU Berlin and his Master's and PhD in Computer Science from Stanford University in 1994 and 2000, respectively. Research of Brock's lab focuses on autonomous mobile manipulation, interactive perception, grasping, manipulation, soft hands, interactive learning, and motion generation.

Alberto Broggi

University of Parma
Department of Information Technology
Parma, Italy
broggi@ce.unipr.it

Chapter F.62

Prof. Alberto Broggi is full Professor at the Universita' di Parma and the President of VisLab, the Artificial Vision and Intelligent Systems Laboratory. As a pioneer in the use of machine vision for automotive applications and on driverless cars, he authored more than 200 publications. He served as the IEEE Intelligent Transportation Systems Society as President and is recipient of two ERC (European Research Council) prestigious grants.

Davide Brugali

University of Bergamo
Department of Computer Science and Mathematics
Dalmine, Italy
brugali@unibg.it

Chapter A.12

Davide Brugali received the PhD degree in Computer Science from Politecnico di Torino, Italy in 1998. He is currently Associate Professor at the University of Bergamo, Italy. Since 2005 he is Chair of the IEEE RAS Technical Commettee on Software Engineering for Robotics and Automation. He has founded the Journal of Software Engineering for Robotics (JOSER) in 2009.

Heinrich Bülthoff

Max-Planck-Institute for Biological Cybernetics
Human Perception, Cognition and Action
Tübingen, Germany
heinrich.buelthoff@tuebingen.mpg.de

Chapter G.78

Professor Heinrich Bülthoff is scientific member of the Max Planck Society and Managing Director of the Max Planck Institute for Biological Cybernetics. There, a group of 70 scientists investigates object and face recognition, human–robot interaction, social and spatial cognition, self-motion perception, and control. They developed unique motion simulators and novel virtual reality technologies for a cybernetics approach to study human cognition in a closed perception–action loop.

Joel W. Burdick

California Institute of Technology
Department of Mechanical Engineering
Pasadena, USA
jwb@robotics.caltech.edu

Chapter D.37

Joel Burdick received the PhD degree in Mechanical Engineering from Stanford University. He has been with the Department of Mechanical Engineering at the California Institute of Technology since May 1988, where he has been the recipient of the NSF Presidential Young Investigator Award, the Office of Naval Research Young Investigator Award, and the Feynman Fellowship. Professor Burdick's current research interests are in sensor-based robot motion planning, multifingered robotic hand manipulation, and the application of robotics to neural prosthetics.

Wolfram Burgard

University of Freiburg
Institute of Computer Science
Freiburg, Germany
burgard@informatik.uni-freiburg.de

Chapter E.45

Wolfram Burgard is a Professor for Computer Science at the University of Freiburg and head of the Research Laboratory for Autonomous Intelligent Systems. His research focuses on the development of probabilistic approaches to mobile robot navigation. Over the past years he and his group have developed a series of innovative probabilistic techniques for various mobile robot navigation problems including localization, map building, path planning, and exploration.

Fabrizio Caccavale

University of Basilicata
School of Engineering
Potenza, Italy
fabrizio.caccavale@unibas.it

Chapter D.39

Fabrizio Caccavale received the Laurea degree and the Research Doctorate degree in Electronic Engineering from the University of Naples Federico II in 1993 and 1997, respectively. He is currently Associate Professor at the School of Engineering of the University of Basilicata. His research interests include manipulator inverse kinematics techniques, cooperative robot manipulation, fault diagnosis, and nonlinear control of mechanical systems. He has published over 100 journal and conference papers and is co-author of 3 books.

Sylvain Calinon

Idiap Research Institute
Martigny, Switzerland
sylvain.calinon@idiap.ch

Chapter G.74

Dr. Sylvain Calinon is a Researcher at the Idiap Research Institute since May 2014 and a Visiting Researcher at the Learning Algorithms and Systems Lab (LASA), Ecole Polytechnique Federale de Lausanne (EPFL). He holds a PhD from EPFL (2007) awarded by Robotdalen, ABB and EPFL-Press awards. His research interests cover robot learning and human-robot interaction.

Raja Chatila Chapter A.14 For biographical profile, please see the section "About the Part Editors".

François Chaumette

Inria/Irisa
Lagadic Group
Rennes, France
francois.chaumette@inria.fr

Chapter C.34

François Chaumette graduated from Ecole Nationale Supérieure de Mécanique, Nantes, France, in 1987. He received the PhD degree in Computer Science from the University of Rennes I in 1990. He is an Inria research scientist at Irisa in Rennes, where he heads the Lagadic Group. His research interests include robotics and computer vision. He is Fellow of the IEEE.

I-Ming Chen

Nanyang Technological University
School of Mechanical and Aerospace Engineering
Singapore, Singapore
michen@ntu.edu.sg

Chapter B.22

I-Ming Chen (BS'86 National Taiwan University, MS'89 and PhD'94 California Institute of Technology, USA) is Professor in Nanyang Technological University, Singapore. His research interests are in wearable sensors, human-robot interaction, modular reconfigurable automation, and parallel kinematics machines (PKM). He is Fellow of IEEE and Fellow of ASME, General Chairman of 2017 IEEE International Conference on Robotics and Automation in Singapore.

Stefano Chiaverini

University of Cassino and Southern Lazio
Department of Electrical and Information Engineering
Cassino, Italy
chiaverini@unicas.it

Chapter A.10

Stefano Chiaverini is a Professor of Automatic Control at the University of Cassino where is the Head of the Electrical and Information Engineering Department. His research interests include inverse kinematics techniques, redundant manipulator control, cooperative robots, force/position control, underwater robotics, and mobile robotic systems. He has published more than 180 papers. He is a Fellow of the IEEE.

Gregory S. Chirikjian

John Hopkins University
Department of Mechanical Engineering
Baltimore, USA
gchirik1@jhu.edu

Chapter B.20

Gregory S. Chirikjian received undergraduate degrees from Johns Hopkins University in 1988 and the PhD degree from the California Institute of Technology, Pasadena, in 1992. Since 1992, he has beenon the faculty of the Department of Mechanical Engineering, Johns Hopkins University, where he has been a full professor since 2001. His research interests include robotics, applications of group theory, and the mechanics of biological macromolecules. In 2008 he became a Fellow of the ASME, and in 2010 he became a Fellow of the IEEE.

Kyu-Jin Cho

Seoul National University
Biorobotics Laboratory
Seoul, Korea
kjcho@sun.ac.kr

Chapter B.23

Kyu-Jin Cho received BS and MS degrees from Seoul National University, Seoul, Korea in 1998 and 2000, respectively, and a PhD degree from MIT in 2007. He was a post-doctoral fellow at Harvard Microrobotics Laboratory until 2008. He is now an Associate Professor of Mechanical and Aerospace Engineering and the Director of Biorobotics Laboratory at Seoul National University. His research interests include biologically inspired robotics, soft robotics, soft wearable robots, and rehabilitation/assistive robotics.

Hyun-Taek Choi

Korea Research Institute of Ships & Ocean Engineering (KRISO)
Ocean System Engineering Research Division
Daejeon, Korea
htchoiphd@gmail.com

Chapter B.25

Hyun-Taek Choi received his BS, MS, and PhD degrees from Hanyang University, Korea in 1991, 1993, and 2000, respectively. After working for Korea Telecom and ASL, University of Hawaii, in a postdoc position, he joined Korea Research Institute of Ships and Ocean Engineering (KRISO) in 2003. He has been leading several projects related to underwater robotic applications such as advanced control & navigation, recognition using optical camera and sonar.

Nak-Young Chong

Japan Advanced Institute of Science and Technology
Center for Intelligent Robotics
Ishikawa, Japan
nakyoung@jaist.ac.jp

Chapter D.44

Nak-Young Chong is a Professor for Research at the Japan Advanced Institute of Science and Technology School of Information Science. He received the PhD degree from Hanyang University, Seoul, Korea, in 1994. From 1994 to 2003, he was on the research staff at Daewoo Heavy Industries, Korea, and AIST, Japan. His research interests cover networked robots and cognitive and intelligent systems.

Howie Choset

Carnegie Mellon University
Robotics Institute
Pittsburgh, USA
choset@cs.cmu.edu

Chapter B.20

Howie Choset conducts research in path planning, motion control, estimation, and mechanism design with a focus on snake robots for medicine, search and rescue, archeology, and manufacturing. Choset's group develops multi-agent path planners for de-mining and coordinated manufacturing. Choset directs the Undergraduate Robotics Major at Carnegie Mellon. Choset is the co-founder of Medrobotics and co-author of Principles of Robot Motion.

Henrik I. Christensen Chapter A.5 For biographical profile, please see the section "About the Part Editors".

Wendell H. Chun

University of Denver
Department of Electrical and Computer Engineering
Denver, USA
wendell.chun@du.edu

Chapter F.61

Wendell Chun managed the Robotics Group at Lockheed Martin in Denver for the last twenty-five years. In addition to working in industry, he has been an Adjunct Faculty teaching robotics at the Colorado School of Mines since 1999 and the University of Denver since 2003. He holds over ten patents and was featured on the History Channel Modern Marvel's episode on Remotely Operated Vehicles.

Wan Kyun Chung

POSTECH
Robotics Laboratory
Pohang, Korea
wkchung@postech.ac.kr

Chapter A.8

Wankyun Chung received his BS degree from Seoul National University in 1981, his MS degree in 1983, and his PhD from KAIST in 1987. He is a Professor in the School of Mechanical Engineering, POSTECH and Director of the National Research Laboratory. He has written two books and published more than 100 journal papers. His research interests include medical robot and biomechatronics.

Woojin Chung

Korea University
Department of Mechanical Engineering
Seoul, Korea
smartrobot@korea.ac.kr

Chapter B.24

Woojin Chung received the BS degree from the Seoul National University and MS and PhD degrees from the University of Tokyo. He is a Professor in the Department of Mechanical Engineering, Korea University. His research interests include autonomous navigation of mobile robots. He received the King-Sun Fu Memorial Best Transactions Paper Award from the IEEE RAS in 2002.

Peter Corke

Queensland University of Technology
Department of Electrical Engineering and Computer Science
Brisbane, Australia
peter.corke@qut.edu.au

Chapter C.34

Peter Corke is a Professor of Robotics and Control at Queensland Institute of Technology, and Director of the ARC Centre of Excellence for Robotic Vision. His research is concerned with enabling robots to see, and the application of robots to mining, agriculture and environmental monitoring. He received his undergraduate degree in Electrical Engineering and PhD from the University of Melbourne.

Elizabeth Croft

University of British Columbia
Department of Mechanical Engineering
Vancouver, Canada
elizabeth.croft@ubc.ca

Chapter G.69

Elizabeth Croft (Honours BASc Mechanical Engineering UBC 1988, MASc Waterloo 1992, PhD Toronto 1995) PEng is Full Professor of Mechanical Engineering, Associate Dean of the Faculty of Applied Science, and Director of the CARIS Lab at the University of British Columbia. Her research spans human–robot interaction, industrial automation, controls, and mechatronics. She is a Fellow of the ASME and Engineers Canada.

Mark R. Cutkosky

Stanford University
Department of Mechanical Engineering
Stanford, USA
cutkosky@stanford.edu

Chapter C.28

Mark Cutkosky received the PhD in Robotics from Carnegie Mellon University in 1985. He became an Assistant Professor at Stanford University in 1985, an Associate Professor in 1991 and Professor in 1996. His research involves bio-inspired robots, mechanisms and sensors.

Kostas Daniilidis

University of Pennsylvania
Department of Computer and Information Science
Philadelphia, USA
kostas@upenn.edu

Chapter C.32

Kostas Daniilidis is Associate Professor of Computer and Information Science at the University of Pennsylvania, where he is a member of the interdisciplinary GRAS Plaboratory. He obtained the BSE and MSE degrees in Electrical Engineering from the National Technical University of Athens (1986) and his Dr. rer.nat. in Computer Science from the University of Karlsruhe (1992). His research interests are in space and motion perception with machines with applications to navigation, panoramic vision, and immersive environments.

Paolo Dario

Sant'Anna School of Advanced Studies
The BioRobotics Institute
Pisa, Italy
paolo.dario@sssup.it

Chapter F.63

Paolo Dario obtained the Dott. Eng. degree in Mechanical Engineering from the University of Pisa in 1977. He is Professor of Biomedical Robotics and Founder and Director of The BioRobotics Institute. His research interests involve robotics for surgery, rehabilitation and assisted living. He served as President of the IEEE Robotics and Automation Society in 2002–2003.

Kerstin Dautenhahn

University of Hertfordshire
School of Computer Science
Hatfield, UK
k.dautenhahn@herts.ac.uk

Chapter G.72

Kerstin Dautenhahn is Professor of Artificial Intelligence in the School of Computer Science at the University of Hertfordshire, U.K. She has pioneered research in social robotics, human-robot interaction, assistive technology and published more than 300 research articles. Prof. Dautenhahn is Editor in Chief of the Journal *of Interaction Studies,* as well as Associate Editor of several other international journals.

Alessandro De Luca

Sapienza University of Rome
Department of Computer, Control, and Management Engineering
Rome, Italy
deluca@diag.uniroma1.it

Chapter A.11

Alessandro De Luca received his PhD in Systems Engineering in 1987 and is Professor of Robotics at Sapienza University of Rome since 2000. He published over 180 articles on modelling and control of different robotic systems, in particular flexible manipulators. He was General Chair of ICRA 2007. He received the Helmholtz-Humboldt Research Award in 2005 and is an IEEE Fellow (class of 2007).

Joris De Schutter

University of Leuven (KU Leuven)
Department of Mechanical Engineering
Leuven-Heverlee, Belgium
joris.deschutter@kuleuven.be

Chapter A.9

Joris De Schutter received his PhD degree in Mechanical Engineering from KU Leuven in 1986. His research interests include sensor-based robot control and programming, human motion modeling, human-robot interaction, multi-sensor fusion and estimation for model building and real-time control, and optimization of mechanical and mechatronic drive systems. He is co-author of two volumes in the STAR book series.

Rüdiger Dillmann

Karlsruhe Institute of Technology
Institute for Technical Informatics
Karlsruhe, Germany
dillmann@ira.uka.de

Chapter G.74

Rüdiger Dillmann received his PhD from University of Karlsruhe in 1980. Since 1987 he has been Professor of the Department of Computer Science, where he became Director of the Research Center for Information Science (FZI). His research interest is in humanoid robotics with special emphasis on intelligent, autonomous and interactive robot behaviour based on machine learning methods and programming by demonstration. He is EiC of the Springer book series COSMOS.

Lixin Dong

Michigan State University
Department of Electrical and Computer Engineering
East Lansing, USA
ldong@egr.msu.edu

Chapter B.27

Lixin Dongreceived his PhD from Nagoya University in 2003. He became Research Associate, Lecturer, and Associate Professor at Xi'an University of Technology in 1992, 1995, and 1998, respectively, and Assistant Professor at Nagoya University in 2003. In 2004 he joined Swiss Federal Institute of Technology (ETH) Zurich in several positions. He is now Assistant Professor at Michigan State University, where his research is on nanorobotics.

Gregory Dudek

McGill University
Department of Computer Science
Montreal, Canada
dudek@cim.mcgill.ca

Chapter C.29

Gregory Dudek is a Professor in the McGill University School of Computer Science and a member of the McGill Research Centre for Intelligent Machines (CIM). He was named a William Dawson Scholar. He directs the McGill Mobile Robotics Laboratory, where his research interests include perception for mobile robotics, vision-based robotics, computational vision, and collaborative filtering.

Hugh Durrant-Whyte

University of Sydney
Australian Centre for Field Robotics (ACFR)
Sydney, Australia
hugh@acfr.usyd.edu.au

Chapter C.35

Hugh Durrant-Whyte received the MSE and PhD degrees, both in Systems Engineering, from the University of Pennsylvania, USA, in 1985 and 1986, respectively. Since 1995 he has been a Professor at the University of Sydney, where he leads the Australian Centre for Field Robotics (ACFR). His research work focuses on autonomous vehicle navigation and decentralized data fusion methods with applications including automation in cargo handling, mining, defence, and marine systems.

Roy Featherstone

The Australian National University
Department of Information Engineering
Canberra, Australia
roy.featherstone@anu.edu.au

Chapter A.3

Dr. Featherstone received the PhD degree from Edinburgh University in 1984. He is the inventor of the articulated-body algorithm, and the author of two books on rigid-body dynamics. He has 7 years experience in industry and was awarded an EPSRC Advanced Research Fellowship in 1992, which was held at Oxford University. He currently workson robot dynamics, control, simulation, and actuator technology.

Gabor Fichtinger

Queen's University
School of Computing
Kingston, Canada
gabor@cs.queensu.ca

Chapter F.63

Gabor Fichtinger received the doctorate degree in Computer Science from the Budapest Technical University, Hungary, in 1990. He is a Professor in Computer Science and Cancer Care Ontario Research Chair at Queen's University, Canada. He is the Director of the Laboratory for Percutaneous Surgery, specializing in computer-assisted interventions and medical robotics, primarily for the diagnosis and therapy of cancer.

Paolo Fiorini

University of Verona
Department of Computer Science
Verona, Italy
paolo.fiorini@univr.it

Chapter F.63

Paolo Fiorini received the Laurea degree in Electronic Engineering from the University of Padova, Italy, the MSEE from the University of California at Irvine, and the PhD in ME from UCLA. From 1985 to 2000, he was with NASA Jet Propulsion Laboratory. In 2001 he joined the Department of Computer Science of the University of Verona as a full Professor. His research focuses on teleoperation and surgical robotics. He is an IEEE Fellow (2009).

Paul Fitzpatrick

Italian Institute of Technology
Robotics, Brain, and Cognitive Sciences Department
Genoa, Italy
paul.fitzpatrick@iit.it

Chapter G.67

Paul Fitzpatrick received the PhD degree in Computer Science from MIT, on developmental approaches to machine perception. He worked at MIT as a postdoctoral lecturer, then held a postdoctoral position at LIRA-Lab at the University of Genoa, Italy. He is now working at the Italian Institute of Technology. His primary interest is the development of machine perception for object manipulation.

Luke Fletcher

Boeing Research & Technology Australia
Brisbane, Australia
luke.s.fletcher@gmail.com

Chapter F.66

Luke Fletcher is at Boeing Research and Technology – Australia working on perception driven robotics for future aircraft assembly. Previously, from 2007, Luke was a research scientist at CSAIL, MIT. He completed a PhD at the Australian National University (ANU) in 2006 on driver gaze and road scene computer vision. In 1996, Luke graduated from the University of Melbourne with BEHons(Elec) and BSc(Comp) degrees.

Dario Floreano

Swiss Federal Institute of Technology (EPFL)
Laboratory of Intelligent Systems
Lausanne, Switzerland
dario.floreano@epfl.ch

Chapter G.76

Dario Floreano is Professor of Intelligent Systems at EPFL and Director of the Swiss National Center of Competence in Robotics. He is interested in the convergence of artificial intelligence, robotics, and biology. He is co-founder of the International Society of Artificial Life Inc. and of the drone company senseFly Ltd. His activity has resulted in more than three hundred articles, four books, several patents, and three spin off companies.

Thor I. Fossen

Norwegian University of Science and Technology
Department of Engineering Cyberentics
Trondheim, Norway
fossen@ieee.org

Chapter E.51

Thor I. Fossen received the MSc degree in Naval Architecture and PhD degree in Engineering Cybernetics from NTNU. His research activities include guidance, navigation and control of marine craft and unmanned vehicles. He is one of the founders of the DNV-GL company Marine Cybernetics. He is the recipient of the Automatica Prize Paper Award in 2002 and the 2008 Arch T. Colwell Merit Award.

Li-Chen Fu

National Taiwan University
Department of Electrical Engineering
Taipei, Taiwan
lichen@ntu.edu.tw

Chapter A.8

Li-Chen Fu received the PhD degree (1987) from University of California at Berkeley, and then joined the Department of Electrical Engineering of the National Taiwan University. His research interests include robotics, smart home, visual detection and tracking, intelligent vehicle, production scheduling, and virtual reality. He was awarded Lifetime Distinguished Professorship from his university in 2007. He is an IEEE Fellow (2004).

Maxime Gautier

University of Nantes
IRCCyN, ECN
Nantes, France
maxime.gautier@irccyn.ec-nantes.fr

Chapter A.6

Maxime Gautier received the Doctorat d'Etat degree in Robotics and Control Engineering from the University of Nantes, France, in 1990. Since 1991, he has been a Professor of Automatic Control at the University of Nantes. He is a member of the Robotics Team, Institut de Recherche en Communication et Cybernétique de Nantes (IRCCyN, UMR-CNRS n6597). His research topics include modeling, identification, and control of robots.

Christos Georgoulas

Technical University Munich
Department of Architecture
Munich, Germany
christos.georgoulas@br2.ar.tum.de

Chapter F.57

Dr. Ing. Christos Georgoulas received his PhD degree on Stereo Vision from Democritus University of Thrace, Greece, in 2010. In 2011, he joined the Chair for Building Realization and Robotics, Technical University Munich, Germany. His research interests are mainly robotics, real-time systems design, embedded systems, intelligent machine vision systems, and image processing.

Martin A. Giese

University Clinic Tübingen
Department for Cognitive Neurology
Tübingen, Germany
martin.giese@uni-tuebingen.de

Chapter G.78

Martin A. Giese founded the Laboratory for Action Representation and Learning at the University Clinic Tübingen in 2001. After a position at Honda, Boston, and as Senior Lecturer at the University of Bangor (UK), he accepted a position as Professor for Computational Sensomotorics in the Hertie Institute for Clinical Brain Research and the University Clinic Tübingen, Germany in 2008. His research focuses on the role of learning in action recognition and control in biological systems, and technical applications.

Ken Goldberg

University of California at Berkeley
Department of Industrial Engineering and Operations Research
Berkeley, USA
goldberg@ieor.berkeley.edu

Chapter D.44

Ken Goldberg is Professor of Industrial Engineering and Operations Research at UC Berkeley, with joint appointments in Electrical Engineering and Computer Science and the School of Information. Goldberg was named IEEE Fellow in 2005 and is Founding Director of the UC CITRIS People and Robots Initiative. He serves as Editor-in-Chief of the IEEE Transactions on Automation Science and Engineering. He is Founding Chair of the IEEE Transactions on Automation Science and Engineering (T-ASE) Advisory Board. He is Co-Founder of the Berkeley Center for New Media (BCNM), the African Robotics Network (AFRON), the Center for Automation and Learning for Medical Robotics (CAL-MR), the CITRIS Data and Democracy Initiative (DDI), Hybrid Wisdom Labs, and Moxie Institute.

Clément Gosselin

Laval University
Department of Mechanical Engineering
Quebec, Canada
gosselin@gmc.ulaval.ca

Chapter B.18

Clément Gosselin received the PhD degree and the D.W. Ambridge Award from McGill University, Montreal, Quebec, Canada in 1988. Since 1997 he has been a Full Professor in the Department of Mechanical Engineering at University Laval, Quebec and a Canada Research Chair in Robotics and Mechatronics since January 2001. His research interests are kinematics, dynamics, and control of robotic mechanical systems with a particular emphasis on parallel mechanisms and robot hands. Dr. Gosselin is a Fellow of the ASME.

Eugenio Guglielmelli

University Campus Bio-Medico of Rome
Faculty Department of Engineering
Rome, Italy
e.guglielmelli@unicampus.it

Chapter F.64

Eugenio Guglielmelli is the Director of the Research Unit of Biomedical Robotics and Biomicrosystems at Campus Bio-Medico University of Rome, Italy. Previously, he was (1991–2004) with the ARTS Lab of the Scuola Superiore Sant'Anna, Pisa, Italy. His current research interests are in human-centered robotics, neurorobotics, biomechatronic design and bio-inspired control of robotic systems, and in their application to rehabilitation and personal assistance.

Sami Haddadin

Leibniz University Hannover
Electrical Engineering and Computer Science
Hannover, Germany
sami.haddadin@irt.uni-hannover.de

Chapter G.69

Sami Haddadin is full Professor and Director of IRT at the Leibniz University Hannover, Germany. His research topics include pHRI, robot control and learning, human motor control, and safe robotics. He is associate editor of TRO and published more than 100 scientific articles. He received several best paper and video awards, the George Giralt Award, and TRO best paper twice.

Martin Hägele

Fraunhofer IPA
Robot Systems
Stuttgart, Germany
mmh@ipa.fhg.de

Chapter F.54

Martin Hägele studied Mechanical Engineering and Engineering Science at the University of Stuttgart and the George Washington University. Since 1989 he is with Fraunhofer IPA and was promoted to Head of the Department Robot Systems in 1993. He received the Award of the German Association of Machine Tools Manufacturers (VDW) in 1990, the Fraunhofer IPA Innovation Award in 1995, the European Innovation and Emerging Technology Award in 1998, and the Engelberger Robotics Award in 2007.

Gregory D. Hager

Johns Hopkins University
Department of Computer Science
Baltimore, USA
hager@cs.jhu.edu

Chapter A.5

Gregory D. Hager is a Professor of Computer Science at Johns Hopkins University. Prof. Hager received the MS and PhD degrees in computer science from the University of Pennsylvania in 1985 and 1988, respectively. His current research interests include visual tracking, vision-based control, medical robotics, and human–computer interaction. He is a Fellow of the IEEE for his contributions in vision-based robotics.

William R. Hamel

University of Tennessee
Mechanical, Aerospace, and Biomedical Engineering
Knoxville, USA
whamel@utk.edu

Chapter F.58

William R. Hamel is a Professor of the Mechanical, Aerospace, and Biomedical Engineering Department at the University of Tennessee. He worked for 31 years at the Oak Ridge National Laboratory in robotics and remote systems for nuclear, military, and space applications. He is an IEEE Fellow and an ASME Fellow with ongoing research in telerobotics for hazardous environments.

Blake Hannaford

University of Washington
Department of Electrical Engineering
Seattle, USA
blake@ee.washington.edu

Chapter D.42

Blake Hannaford, PhD, is Professor of Electrical Engineering, Adjunct Professor of Bioengineering, Mechanical Engineering, and Surgery at the University of Washington, Seattle. He was awarded the National Science Foundation's Presidential Young Investigator Award, the Early Career Achievement Award from the IEEE Engineering in Medicine and Biology Society, and was named IEEE Fellow in 2005. His active interests include haptic displays on the Internet and surgical robotics.

Kensuke Harada

National Institute of Advanced Industrial Science and Technology
Intelligent Systems Research Institute
Tsukuba, Japan
kensuke.harada@aist.go.jp

Chapter G.67

Dr. Kensuke Harada is a Leader of Manipulation Research Group, Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology (AIST). He received PhD degree in Mechanical Engineering from Kyoto University in 1997. His main area of research includes planning and control of robotic systems such as robot manipulator, biped humanoid robot, and multi-fingered hand.

Martial Hebert

Carnegie Mellon University
The Robotics Institute
Pittsburgh, USA
hebert@ri.cmu.edu

Chapter E.45

Dr. Martial Hebert is Professor of Robotics at Carnegie Mellon University and Director of the Robotics Institute. His interests include computer vision, 3-D model building and perception for mobile robots. His group has developed approaches for object recognition in images and video sequences, techniques for recognizing 3-D shapes in 3-D point clouds and for building models of 3-D objects, and for modeling the environment of autonomous robots from sensor data.

Thomas C. Henderson

University of Utah
School of Computing
Salt Lake City, USA
tch@cs.utah.edu

Chapter C.35

Professor Thomas Henderson received the BS degree in Mathematics from Louisiana State University in 1973 and the PhD degree in Computer Science from The University of Texas at Austin in 1979. Since 1989 he has been Professor of Computer Science at the University of Utah. His current research interests concentrate on smart sensor networks, the simulation of accidental fires and explosions, on the DARPA Urban Challenge, and on technical drawing analysis.

Eldert van Henten

Wageningen University
Wageningen UR Greenhouse Horticulture
Wageningen, The Netherlands
eldert.vanhenten@wur.nl

Chapter F.56

Dr. Eldert van Henten is a Professor of Biosystems Engineering and Head of the Farm Technology Group at Wageningen University, The Netherlands. His research interests include sensing, modeling, design, and control of biosystems, biorobotics, and high-tech automation and company logistics. His robotics activities focus on applications to arable farming, livestock farming, and protected cultivation.

Hugh Herr

MIT Media Lab
Cambridge, USA
hherr@media.mit.edu

Chapter G.70

Hugh Herr is Associate Professor within MIT's Program of Media Arts and Sciences, and The Harvard-MIT Division of Health Sciences and Technology. His primary research objective is to apply principles of neuromechanics to the design of bionic limbs. He has employed cross bridge models of skeletal muscle to the design human-powered mechanisms that amplify endurance and built elastic shoes and powered leg exoskeletons. Herr is the (co-)author of over 150 peer-reviewed manuscripts and patents within biomechatronics.

Joachim Hertzberg

Osnabrück University
Institute for Computer Science
Osnabrück, Germany
joachim.hertzberg@uos.de

Chapter A.14

Joachim Hertzberg holds diploma and doctorate degrees in Informatics. He is a Full Professor for Informatics/Knowledge-Based Systems at Osnabrück University. He is also with DFKI's Robotics Innovation Center, heading its Branch Office in Osnabrück. His areas of research are AI, plan-based robot control, semantic mapping in robotics, and applications thereof.

Gerd Hirzinger

German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Wessling, Germany
gerd.hirzinger@dlr.de

Chapter F.55

Gerd Hirzinger received his doctor's degree from the Technical University of Munich in 1974. From 1992 to 2012 he was Director at the DLR Institute for Robotics and Mechatronics. The institute develops innovative robot systems for space and terrestrial applications, telerobotic concepts, man–machine interfaces, sensors and actuators, sensory feedback, ultra-lightweight robot arms, articulated hands, planetary rovers, and medical technology. He was prime investigator of ROTEX, the first remotely controlled robot in space, which flew onboard shuttle COLUMBIA in April 93. He has received numerous national and international awards.

John Hollerbach

University of Utah
School of Computing
Salt Lake City, USA
jmh@cs.utah.ledu

Chapter A.6

John M. Hollerbach is Professor of Computing, Research Professor of Mechanical Engineering, and Director of the Robotics Track at the University of Utah. Previously he held faculty positions at MIT and McGill University. He received his PhD from Massachusetts Institute of Technology in 1978. He is Editor of the International Journal of Robotics Research. His research interests include haptic interfaces and virtualreality.

Kaijen Hsiao

Robert Bosch LLC
Research and Technology Center, Palo Alto
Palo Alto, USA
kaijenhsiao@gmail.com

Chapter D.41

Kaijen Hsiao received her MS and PhD degrees in Computer Science from MIT. She spent four years as a Research Scientist working on grasping, manipulation, and shared-autonomous teleoperation at Willow Garage, and is now at Bosch Research and Technology Center in Palo Alto, where she works on personal robots.

Tian Huang

Tianjin University
Department of Mechanical Engineering
Tianjin, China
tianhuang@tju.edu.cn

Chapter B.18

Tian Huang received his BE, ME and PhD from Tianjin University, China in 1981, 1984 and 1990 all in Mechanical Engineering. Dr Huang is an expert in the field of robotics. His research interests cover kinematics, dynamics and control of parallel robots. He has published more than 200 papers. He has been selected as *Yangtze Chair Professor* in 2000 by the State Education Ministry of China.

Christoph Hürzeler

Alstom Power Thermal Services
Automation and Robotics R&D
Baden, Switzerland
christoph.huerzeler@power.alstom.com

Chapter B.26

Christoph Hürzeler received his Master of Science in Mechanical Engineering from the ETH Zurich in 2007. From 2008 to 2013 he joined the Autonomous Systems Lab under Prof. Roland Siegwart as a PhD student in the field of aerial robotics. Currently, he works at Alstom Switzerland as project leader and R&D engineer developing inspection robots for Alstom's Thermal Services Division.

Phil Husbands

University of Sussex
Department of Informatics
Brighton, UK
philh@sussex.ac.uk

Chapter G.76

Phil Husbands is Professor of Computer Science and Artificial Intelligence in the Department of Informatics and Co-director of the Centre for Computational Neuroscience and Robotics at the University of Sussex. His research interests include biologically inspired adaptive robotics, evolutionary systems, computational neuroscience, and creative systems.

Seth Hutchinson

University of Illinois
Department of Electrical and Computer Engineering
Urbana-Champaign, USA
seth@illinois.edu

Chapter C.34

After receiving his PhD degree from Purdue University, Seth Hutchinson joined the University of Illinois in Urbana-Champaign, where he is currently a Professor in the Department of Electrical and Computer Engineering. He has published more than 200 papers in the area of sensor-based robotics, and is co-author of the books *Principles of Robot Motion: Theory, Algorithms, and Implementations* and *Robot Modeling and Control*.

Karl Iagnemma

Massachusetts Institute of Technology
Laboratory for Manufacturing and Productivity
Cambridge, USA
kdi@mit.edu

Chapter B.24

Karl Iagnemma is a Principal Research Scientist at MIT, where he directs the Robotic Mobility Group. His interests include design, motion planning, and control of mobile robots in challenging environments. Dr. Iagnemma is the author of *Mobile Robots in Rough Terrain* (Springer, 2004) and has published nearly 150 research papers on robotic topics.

Fumiya Iida

University of Cambridge
Department of Engineering
Cambridge, UK
fumiya.iida@eng.cam.ac.uk

Chapter G.75

Fumiya Iida received his master degrees in Mechanical Engineering from Tokyo University of Science (1999), and Dr. sc. nat. in Informatics from the University of Zurich (2006). He is currently a University Lecturer at the University of Cambridge, and an Assistant Professor at ETH Zurich. His research interest includes dynamic legged locomotion, navigation of autonomous robots, human-machine interactions, and self-reconfigurable soft robots.

Auke Jan Ijspeert

Swiss Federal Institute of Technology (EPFL)
School of Engineering
Lausanne, Switzerland
auke.ijspeert@epfl.ch

Chapter G.75

Auke Ijspeert has a BSc/MSc in Physics from the EPFL, the Swiss Federal Institute of Technology at Lausanne, and a PhD in Artificial Intelligence from the University of Edinburgh. He is currently an Associate Professor at the EPFL, and Head of the Biorobotics Laboratory. His research interests are at the intersection between robotics, computational neuroscience, nonlinear dynamical systems, and applied machine learning.

Genya Ishigami

Keio University
Department of Mechanical Engineering
Yokohama, Japan
ishigami@mech.keio.ac.jp

Chapter E.50

Genya Ishigami is an Assistant Professor at Keio University, Japan. He received the PhD degree from Tohoku University, Japan in 2008 and was a Postdoctoral Associate at the MIT and a Research Associate at the JAXA. His research focuses on mobility analysis, vehicle–terrain interaction mechanics, perception, navigation and control, for application to planetary exploration rovers and field robots.

Michael Jenkin

York University
Department of Electrical Engineering and Computer Science
Toronto, Canada
jenkin@cse.yorku.ca

Chapter C.29

Michael Jenkin is a Professor of Computer Science and Engineering at York University, Canada. Working in the areas of virtual reality and visually guided autonomous robots, he is the co-author with G. Dudek of Computational Principles of Mobile. He has worked on a number of large robotics projects including the ARK and AQUA Robotic Systems and is also one of the designers of IVY – a six-sided immersive projective environment.

Shuuji Kajita

National Institute of Advanced Industrial Science and Technology (AIST)
Intelligent Systems Research Institute
Tsukuba, Japan
s.kajita@aist.go.jp

Chapter B.17

Shuuji Kajita received the Dr. degree in Control Engineering from the Tokyo Institute of Technology, Tokyo, Japan, in 1996. In 1985, he joined the Mechanical Engineering Laboratory, National Institute of Advanced Industrial Science and Technology (AIST). During 1996–1997, he was a Visiting Researcher at the California Institute of Technology. Since 2007, he has been the Leader of the Humanoid Robotics Group of Intelligent Systems Research Institute, AIST, Japan.

Takayuki Kanda

Advanced Telecommunications Research (ATR) Institute International
Intelligent Robotics and Communication Laboratories
Kyoto, Japan
kanda@atr.jp

Chapter G.72

Takayuki Kanda is a Group Leader of Human-Robot Interaction Group in ATR Intelligent Robotics and Communication Laboratories, Kyoto, Japan. He received his BEng, MEng, and PhD degrees in Computer Science from Kyoto University, Kyoto, Japan, in 1998, 2000, and 2003, respectively. His research focussses on human-robot interaction, social robotics, robots in real daily context, and intelligent robotics.

Makoto Kaneko Chapter B.19 For biographical profile, please see the section "About the Part Editors".

Sung-Chul Kang Chapter F.58

Korea Institute of Science and Technology
Center for Bionics
Seoul, Korea
kasch@kist.re.kr

Sung-Chul Kang is a Principal Research Scientist in the Center for Bionics of the Korea Institute of Science and Technology (KIST). His research interests are in mechanism design and control for dependable mobility and manipulation. He has been applying these mobility and manipulation technologies to service robotics area like medical and field robots. He has a PhD in Mechanical Design and Production Engineering from Seoul National University, Seoul, Korea.

Imin Kao Chapter D.37

Stony Brook University
Department of Mechanical Engineering
Stony Brook, USA
imin.kao@stonybrook.edu

Being the Director of the Manufacturing Automation Laboratory (MAL) at SUNY Stony Brook, Professor Kao conducts research in robotics and manufacturing, specialized in the modeling of contact interface. He integrates his study with MEMS transducers to study intelligent contact interface that augments the capability of robots and humans. The soft-contact theory he proposed, which correlates the growth of contact radius with normal force, subsumes the well-known Hertzian model.

Lydia E. Kavraki Chapter A.7

Rice University
Department of Computer Science
Houston, USA
kavraki@rice.edu

Lydia E. Kavraki is the Noah Harding Professor of Computer Science and Bioengineering at Rice University. She earned the PhD degree in Computer Science from Stanford University. Kavraki's research is in physical algorithms and their applications in robotics and structural bioinformatics. She is a Fellow of IEEE, ACM, AAAS, and AIMBE.She is also an elected member of the Institute of Medicine (IOM) of the National Academies of United States.

Charles C. Kemp Chapter G.67

Georgia Institute of Technology and
Emory University
Atlanta, USA
charlie.kemp@bme.gatech.edu

Charles C. Kemp is an Assistant Professor in the Department of Biomedical Engineering at Georgia Tech and Emory University. He is Director of the Center for Healthcare Robotics in the Health Systems Institute. He received a Doctorate in Electrical Engineering and Computer Science from MIT in 2005. His research focuses on autonomous robot manipulation and human–robot interaction for healthcare.

Wisama Khalil Chapter A.6

University of Nantes
IRCCyN, ECN
Nantes, France
wisama.khalil@irccyn.ec-nantes.fr

Wisama Khalil received the PhD and the Doctorat d'Etat degrees in Robotics and Control Engineering from Montpellier, France, in 1976 and 1978, respectively. Since 1983 he has been Professor of Robotics in the Ecole Centrale de Nantes. He carries out his research within the Robotics team of the IRCCyN (Institut de recherche en Communication et Cybernétique de Nantes-UMR-CNRS n6597). His research interest includes, kinematics, dynamics, calibration, identification, and control of robots.

Oussama Khatib Chapter 1 For biographical profile, please see the section "About the Editors".

Lindsay Kleeman Chapter C.30

Monash University
Department of Electrical and Computer
Systems Engineering
Melbourne, Australia
kleeman@eng.monash.edu.au

Lindsay Kleeman received the bachelor degrees in Electrical Engineering and Mathematics both with university medals, and, in 1986, the PhD degree from the University of Newcastle. He is currently an Associate Professor at Monash University, Australia with over 100 publications including a best paper award at IROS99. His research interests include mobile robotics, ultrasonic sensing, and digital systems.

Alexander Kleiner

Linköping University
Department of Computer Science
Linköping, Sweden
alexander.kleiner@liu.se

Chapter F.60

Alexander Kleiner is an Associate Professor of Computer Science at the Linköping University. He earned his PhD from the University of Freiburg and worked as postdoc at the Carnegie Mellon and La Sapienza University. He is member of the RoboCup executive committee and SSRR TC. His research focuses on collaborative search for stationary and moving targets in the real world.

Jens Kober

Delft University of Technology
Delft Center for Systems and Control
Delft, The Netherlands
j.kober@tudelft.nl

Chapter A.15

Jens Kober has recently been appointed Assistant Professor at TU Delft. Before joining TU Delft he has affiliated to the CoR-Lab of Universität Bielefeld and the Honda Research Institute Europe in Offenbach both in Germany. Jens has graduated in Spring 2012 with Doctor of Engineering from Technische Universität Darmstadt. His PhD thesis on *Learning Motor Skills – From Algorithms to Robot Experiments* has received the 2013 Georges Giralt PhD Award as the best Robotics PhD thesis in Europe in 2012.

Kurt Konolige

Google, Inc.
Mountain View, USA
konolige@gmail.com

Chapter C.31

Kurt Konolige is a Senior Staff Roboticist at Google, Inc. and a Fellow of AAAI. He has been a Consulting Professor in Computer Science and AI at Stanford University and one of the cofounders of Industrial Perception, Inc., Videre Design, and MobileRobotics LLC. He received his PhD from Stanford University in 1984. His recent research has concentrated on real-time perception and action for mobile robots and manipulators, and machine learning for robotics.

David Kortenkamp

TRACLabs Inc
Houston, USA
korten@traclabs.com

Chapter A.12

David Kortenkamp is a Senior Scientist with TRACLabs, Inc. supporting NASA Johnson Space Center's Automation, Robotics, and Simulation Division. He received the BS degree in Computer Science from the University of Minnesota and the MS and PhD degrees in Computer Science from the University of Michigan. His research at NASA focuses on intelligent control architectures, human–robot interaction, and autonomous execution systems.

Kazuhiro Kosuge

Tohoku University
System Robotics Laboratory
Sendai, Japan
kosuge@irs.mech.tohoku.ac.jp

Chapter F.65

Professor Kazuhiro Kosuge received the BE, ME, and PhD degrees in Control Engineering from the Tokyo Institute of Technology, in 1978, 1980, and 1988, respectively. After serving as a Research Associate at Tokyo Institute of Technology and as an Associate Professor at Nagoya University, he has been a Professor at Tohoku University since 1995.

Danica Kragic

Royal Institute of Technology (KTH)
Centre for Autonomous Systems
Stockholm, Sweden
dani@kth.se

Chapter C.32

Danica Kragic received the PhD from KTH Royal Institute of Technology, Stockholm, in 2001. She is currently a Professor at the School of Computer Science and Communication at KTH. She is a member of the Royal Swedish Academy of Sciences and the Young Academy of Sweden. Her research interests are robotics, computer vision and machine learning.

Torsten Kröger Chapter A.8 For biographical details see "About the Multimedia Editors"

Roman Kuc

Yale University
Department of Electrical Engineering
New Haven, USA
kuc@yale.edu

Chapter C.30

Dr. Kuc is Director of Educational Affairs in Enginineering and Director of the Intelligent Sensors Laboratory at Yale University. He received the BSEE degree from the Illinois Institute of Technology, Chicago, in 1968, and the PhD degree in Electrical Engineering from Columbia University, New York, in 1977. His interests cover digital signal processing with applications to bioengineering, robotics and telecommunications, inverse problems, intelligent sensors, modeling biological systems, as well as embedded systems.

James Kuffner

Carnegie Mellon University
The Robotics Institute
Pittsburgh, USA
kuffner@cs.cmu.edu

Chapter D.36

James Kuffner received the PhD degree in Computer Science from Stanford University in 1999. He was a Japan Society for the Promotion of Science Postdoctoral Research Fellow at the University of Tokyo until 2001. He joined the faculty at Carnegie Mellon University in 2002 and is currently an Assistant Professor. He has published over 100 technical papers and received the Okawa Foundation Award for Young Researchers in 2007.

Scott Kuindersma

Harvard University
Cambridge, USA
scottk@seas.harvard.edu

Chapter E.48

Scott Kuindersma is an Assistant Professor of Engineering and Computer Science at Harvard University. Prior to that he was a postdoc at MIT CSAIL and the Planning and Control Lead for MIT's DARPA Robotics Challenge team. He received his PhD in Computer Science from the University of Massachusetts Amherst in 2012. His research interests broadly encompass legged robotics, optimization, control, nonlinear systems, and machine learning.

Vijay Kumar

University of Pennsylvania
Department of Mechanical Engineering and Applied Mechanics
Philadelphia, USA
kumar@seas.upenn.edu

Chapter E.52

Vijay Kumar is the UPS Foundation Professor at the School of Engineering and Applied Science at the University of Pennsylvania. He received his PhD from the Ohio State University. He has won many awards including the NSF Presidential Young Investigator Award (1991), and recently a 2014 Engelberger Robotics Award. He was elected to the National Academy of Engineering in 2013.

Steven M. LaValle

University of Illinois
Department of Computer Science
Urbana, USA
lavalle@cs.uiuc.edu

Chapter A.7

Steve LaValle is an Associate Professor of Computer Science at the University of Illinois. His interests include motion planning, geometric algorithms, control theory, and sensing. He recently authored the book (free online) *Planning Algorithms*.

Florant Lamiraux

LAAS-CNRS
Toulouse, France
florent@laas.fr

Chapter E.47

Florent Lamiraux graduated from the Ecole Polytechnique Paris in 1993. He received the PhD in Computer Science from the Institut National Polytechnique de Toulouse in 1997 for his research on mobile robots and spent two years at Rice University as a Research Associate. He is currently Chargé de Recherche at LAAS-CNRS and has worked for ten years on nonholonomic mobile robots. He is now working in humanoid robots.

Roberto Lampariello

German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Wessling, Germany
roberto.lampariello@dlr.de

Chapter F.55

Roberto Lampariello received his BSe in Aerospace Engineering from Southampton University in 1990, MS in Aeroplane Aerodynamics from Cranfield University in 1991 and a specialization from the University of Rome *La Sapienza* in 1998. He is since then a Researcher with the Robotics and Mechatronics Center of DLR, Germany. Since 2012 he is Lecturer at the Chair for Aerospace Engineering of the TU Munich.

Christian Laugier

INRIA Grenoble Rhône-Alpes
Saint Ismier, France
christian.laugier@inria.fr

Chapter F.62

Dr. Christian Laugier is Research Director at INRIA and Scientific Leader of the *e-Motion* robotics team-project. He co-edited many books and recently published recognized scientific contributions and patented innovations to the field of perception and decision for autonomous robots. He is the recipient of several awards, including the IEEE/RSJ Harashima award 2012. He has also co-founded four start-up companies.

Jean-Paul Laumond

LAAS-CNRS
Toulouse, France
jpl@laas.fr

Chapter E.47

Jean-Paul Laumond, PhD 1984, IEEE Fellow, is Directeur de Recherche at LAAS-CNRS in Toulouse, France. His research is on robot motion planning and control. In 2001 and 2002 he created and managed a spin-off company Kineo CAM. From 2005 to 2008, he was Co-Director of the French–Japanese laboratory JRL. His current research explores anthropomorphic action via a multidisciplinary approach combining computer science, automatic control, robotics, and neuroscience.

Daniel D. Lee

University of Pennsylvania
Department of Electrical Systems Engineering
Philadelphia, USA
ddlee@seas.upenn.edu

Chapter A.15

Daniel Lee is the Evan C Thompson Term Chair and Professor at the University of Pennsylvania. He received his BA in Physics from Harvard in 1990 and his PhD in Condensed Matter Physics from MIT in 1995. Before coming to Penn, he was with AT&T and Lucent Bell Labs. He is a Fellow of the IEEE and a National Science Foundation CAREER Awardee. His group focuses on understanding general computational principles in biological systems, and on applying that knowledge to build autonomous systems.

Dongjun Lee Chapter D.43 For biographical details see "About the Multimedia Editors"

Roland Lenain

IRSTEA
Department of Ecotechnology
Aubiere, France
roland.lenain@irstea.fr

Chapter E.49

Roland Lenain obtained the PhD degree from Univ. Blaise Pascal, France, in 2005, and spent 6 months as a Postdoctoral Fellow in the Department of Automatic Control at Lund University, Sweden. Since 2006, he is Chargé de Recherche at Irstea, France. His research interests include modeling, observation and control of wheeled mobile robots moving under uncertain and harsh conditions.

David Lentink

Stanford University
Department of Mechanical Engineering
Stanford, USA
dlentink@stanford.edu

Chapter B.26

David Lentink is an Assistant Professor of Mechanical Engineering at Stanford University. He published papers on natural flight (Nature, Science), developed flying robots (DelFly, RoboSwift), won the Dutch Academic Year Prize (2010), is one of 40 scientists under 40 according to the World Economic Forum (2013), and is a member of the Young Academy of the Royal Netherlands Academy of Arts and Science.

John J. Leonard

Massachusetts Institute of Technology
Department of Mechanical Engineering
Cambridge, USA
jleonard@mit.edu

Chapter E.46

John J. Leonard is the Samuel C. Collins Professor of Mechanical and Ocean Engineering and Associate Department Head for Research in the MIT Department of Mechanical Engineering. He is also a member of the MIT CSAIL. His research addresses the problems of navigation and mapping for autonomous mobile robots. He holds the degrees of BS in Electrical Engineering and Science from the University of Pennsylvania (1987) and DPhil in Engineering Science from the University of Oxford (1994).

Aleš Leonardis

University of Birmingham
Department of Computer Science
Birmingham, UK
a.leonardis@cs.bham.ac.uk

Chapter C.33

Aleš Leonardis is Chair of Robotics and Co-Director of Computational Neuroscience and Cognitive Robotics Centre at the University of Birmingham. His research interests include object and scene recognition and categorization, statistical visual learning, and biologically motivated vision. He has been an associate editor of IEEE PAMI and an editor of the Springer book series Computational Imaging and Vision.

Stefan Leutenegger

Imperial College London
South Kensington Campus, Department of Computing
London, UK
s.leutenegger@imperial.ac.uk

Chapter B.26

Stefan Leutenegger has received his PhD in Robotics from ETH Zurich in 2014. Currently, he has a lecturer position at the Imperial College, London. His research interests range from unmanned solar airplane design to real-time vision-aided multi-sensor state estimation and mapping, as well as unmanned aerial system (UAS) modeling, control, and navigation close to structure and terrain.

Kevin M. Lynch

Northwestern University
Department of Mechanical Engineering
Evanston, USA
kmlynch@northwestern.edu

Chapter D.37

Kevin Lynch received the PhD degree in Robotics from Carnegie Mellon University in 1996. Since 1997 he has been faculty in the Mechanical Engineering Department of Northwestern University, where he is now Faculty Director. His research interests include robotic manipulation, motion planning, underactuated dynamic systems, human–robot interaction, and self-organizing systems.

Anthony A. Maciejewski

Colorado State University
Department of Electrical and Computer Engineering
Fort Collins, USA
aam@colostate.edu

Chapter A.10

Tony Maciejewski received BS, MS, and PhD degrees from The Ohio State University and served for 14 years on the Faculty of Purdue University, West Lafayette. He is currently a Professor and Head of the Department of Electrical and Computer Engineering at Colorado State University. He is a Fellow of the IEEE for his work on kinematically redundant robots.

Robert Mahony

Australian National University (ANU)
Research School of Engineering
Canberra, Australia
robert.mahony@anu.edu.au

Chapter E.52

Robert Mahony is a Professor in the Research School of Engineering at the Australian National University. He received his PhD in 1995 in Systems Engineering from the Australian National University after originally training in geophysics and working in marine seismology. His research interests are in nonlinear systems theory and optimization with applications in robotics and computer vision.

Joshua A. Marshall

Queen's University
The Robert M. Buchan Department of Mining
Kingston, Canada
joshua.marshall@queensu.ca

Chapter F.59

Dr. Joshua Marshall is an Assistant Professor of Engineering and Applied Science at Queen's University, Kingston, Canada, where he directs the multidisciplinary Mining Systems Laboratory. He holds an undergraduate degree in mining engineering, and graduate degrees in Mechanical and Electrical Engineering. Over his career, he has worked on many robotics and equipment automation R&D projects, focusing primarily on field robotics.

Maja J. Matarić

University of Southern California
Computer Science Department
Los Angeles, USA
mataric@usc.edu

Chapter G.73

Maja Matarić is Chan Soon-Shiong Professor of Computer Science, Neuroscience and Pediatrics at the University of Southern California and Founding Director of the USC Robotics and Autonomous Systems Center. She received the PhD and MS degrees from MIT in Computer Science and AI in 1994 and 1990. She is a Fellow of IEEE and AAAS and a Presidential Mentoring Awardee. Her research into socially assistive robotics enables robots to help users with special needs in rehabilitation.

Yoshio Matsumoto

National Institute of Advanced Industrial Science and Technology (AIST)
Robot Innovation Research Center
Tsukuba, Japan
yoshio.matsumoto@aist.go.jp

Chapter G.67

Dr. Yoshio Matsumoto is a Team Leader of Service Robotics Research Team, Robot Innovation Research Center, AIST. He received PhD degree in Engineering from the University of Tokyo in 1998. He has worked for the Australian National University, Nara Institute of Science and Technology, Osaka University, and joined AIST in 2009. His research interests include assistive robots, human–robot interaction, and real-time vision.

J. Michael McCarthy

University of California at Irvine
Department of Mechanical Engineering
Irvine, USA
jmmccart@uci.edu

Chapter A.4

J. Michael McCarthy received the PhD degree from Stanford Universityin 1979, and joined the faculty at the University of California, Irvinein 1986. He is the author of two books *Introduction to Theoretical Kinematics* and *Geometric Design of Linkages*, and has served as Editor of the ASME Journal of Mechanical Design and is the Founding Editor of the ASME Journal of Mechanisms and Robotics.

Claudio Melchiorri

University of Bologna
Laboratory of Automation and Robotics
Bologna, Italy
claudio.melchiorri@unibo.it

Chapter B.19

Claudio Melchiorri received the Laurea degree in Electronic Engineering from the University of Bologna, Italy, in 1985, and the PhD in 1990. He is Full Professor in Robotics since 2001, and is with the Department of Electrical, Electronic and Information Engineering of the University of Bologna. He has published about 280 papers, authored 5 books in control and trajectory planning and has filed 2 patents. His research interests include dexterous manipulation, haptics, and telemanipulation.

Arianna Menciassi

Sant'Anna School of Advanced Studies
The BioRobotics Institute
Pisa, Italy
a.menciassi@sssup.it

Chapter F.63

Arianna Menciassi obtained the MSc in Physics in 1995 and the PhD in Bioengineering in 1999. She is Professor of Biomedical Robotics and Area Leader of Surgical Robotics and Allied Technologies. Her research interests involve surgical robotics and microsystem technology, with a special synergy between robot-assisted therapy and bioengineering. She is co-chair of the IEEE Technical Committee on Surgical Robotics.

Jean-Pierre Merlet

INRIA Sophia-Antipolis
Sophia-Antipolis, France
jean-pierre.merlet@sophia.inria.fr

Chapter B.18

J.-P. Merlet is senior researcher at INRIA, where he works on interval analysis and robotics. He is the author of a book on parallel robots and of over 200 papers in robotics, mechanism theory, and numerical analysis. He is the chair of IFToMM France and has received the Altran Award and the *Micron d'Or* Award for his microrobot for surgical endoscopy.

Giorgio Metta

Italian Institute of Technology
iCub Facility
Genoa, Italy
giorgio.metta@iit.it

Chapter G.77

Giorgio Metta is director of the iCub Facility Department at the Istituto Italiano di Tecnologia (IIT), where he coordinates the development of the iCub humanoid robot. From 2001 to 2002 he was postdoctoral associate at the MIT AI-Lab. He is deputy Director of IIT delegate for international relations and external funding. He is author of approximately 250 peer-reviewed publications.

François Michaud

University of Sherbrooke
Department of Electrical Engineering and Computer Engineering
Sherbrooke, Canada
francois.michaud@usherbrooke.ca

Chapter A.13

François Michaud is Director of the Interdisciplinary Institute for Technological Innovation. He held the Canada Research Chair (2001–2011) in Mobile Robots and Autonomous Intelligent Systems, and received in 2003 the Young Engineer Achievement Award from the Canadian Council of Professional Engineers. His research interests are in decisional architectures, mobile robot design, human–robot interaction, robot learning, and intelligent systems.

David P. Miller

University of Oklahoma
School of Aerospace and Mechanical Engineering
Norman, USA
dpmiller@ou.edu

Chapter G.79

David P. Miller is the Wilkonson Chair Professor of Intelligent Systems at the University of Oklahoma. His teaching and research span the schools of Aerospace and Mechanical Engineering, Computer Science, and Bioengineering. His primary research areas are planetary rovers, assistive technology, and robots for STEM education. He is a co-founder of the KISS Institute for Practical Robotics.

Javier Minguez

University of Zaragoza
Department of Computer Science and Systems Engineering
Zaragoza, Spain
jminguez@unizar.es

Chapter E.47

Prof. Dr. Minguez is Professor of Computer Science and Founder of the Neurotechnolgy Group at the University of Zaragoza (Spain). He is Co-founder of the startup BitBrain Technologies, he raised funds for innovation and research from 50+ partners, published 90+ research publications and 5+ patents in neuroscience, neural engineering, human-computer interaction, cognitive and motor neurorehabilitation and intelligent robotics.

Pascal Morin

University Pierre and Marie Curie
Institute for Intelligent Systems and Robotics
Paris, France
morin@isir.upmc.fr

Chapter E.49

Pascal Morin obtained the PhD degree from Ecole des Mines de Paris, France, in 1996, and then spent one year as a postdoc at the CALTEC, Pasadena, USA. From 1997 to 2011, he was Chargé de Recherche at INRIA, France. Since 2011 he is in charge of the RTE-UPMC Research Chair on Autonomous UAVs. His research interests include control of nonlinear systems and its application to mechanical systems.

Mario E. Munich

iRobot Corp.
Pasadena, USA
mariomu@ieee.org

Chapter F.65

Dr. Mario E. Munich is Vice President of Advanced Systems Technologies at iRobot Corp. in Pasadena, where he currently manages the research and engineering development efforts. He received the degree of Electronic Engineer from the National University of Rosario, Argentina, and the MS and the PhD degrees in Electrical Engineering from CALTEC, Pasadena. His research interests include computer vision, sensors for robots, autonomous navigation, and human-robotic interaction.

Robin R. Murphy

Texas A&M University
Department of Computer Science and Engineering
College Station, USA
murphy@cse.tamu.edu

Chapter F.60

Robin Roberson Murphy is the Director of the Center for Robot-Assisted Search at Texas A&M. She has over 150 publications on artificial intelligence, human-robot interaction, and robotics including the *Introduction to AI Robotics* and *Disaster Robotics*. Her insertion of tactical ground, air, and marine robots at 16 disasters includes the 9/11 World Trade Center disaster, Hurricane Katrina, and the Fukushima Daiichi nuclear accident. She serves on several government and professional boards.

Bilge Mutlu

University of Wisconsin–Madison
Department of Computer Sciences
Madison, USA
bilge@cs.wisc.edu

Chapter G.71

Bilge Mutlu is an Assistant Professor of Computer Science at the University of Wisconsin–Madison. He received his PhD degree from Carnegie Mellon University's Human-Computer Interaction Institute in 2009. His research bridges human-computer interaction and robotics, focusing on the development of human-centered principles and methods for designing robotic technologies.

Keiji Nagatani

Tohoku University
Department of Aerospace Engineering,
Graduate School of Engineering
Sendai, Japan
keiji@ieee.org

Chapter E.50

Keiji Nagatani received his PhD degree from the University of Tsukuba, in 1997. Currently, he is an Associate Professor at Tohoku University, Japan. His research interest is field robotics, particularly, tele-operation of tracked vehicles for search and rescue missions, and development of mobile robots to explore volcanic areas. He is a Member of the RSJ, SICE, JSME, JSASS and IEEE.

Daniele Nardi

Sapienza University of Rome
Department of Computer, Control, and Management Engineering
Rome, Italy
nardi@dis.uniroma1.it

Chapter F.66

Daniele Nardi is Full Professor at the Faculty of Information Engineering, Informatics and Statistics, *Sapienza* Univ., Rome. He graduated at Turin Polytechnic (1981) and received a MS from Sapienza (1984). His research interests are in artificial intelligence and cognitive robotics. Trustee of the RoboCup Federation (President 2011–2014), he is leading the research laboratory Cognitive Robot Teams and formerly Intelligent Systems for Emergency and Civil Defense.

Eduardo Nebot

University of Sydney
Department of Aerospace, Mechanical and Mechatronic Engineering
Sydney, Australia
eduardo.nebot@sydney.edu.au

Chapter F.59

Eduardo Nebot holds a BE Electrical Engineering (UNS, Argentina), MS and PhD (Colorado State University, Fort Collins). He is a Professor at the School of AMME, University of Sydney and the Director of the Australian Centre for Field Robotics. His main research areas are in field robotics including autonomous system, navigation and safety with substantial impact in the mining and intelligent transport system sector.

Bradley J. Nelson

ETH Zurich
Institute of Robotics and Intelligent Systems
Zurich, Switzerland
bnelson@ethz.ch

Chapter B.27

Brad Nelson received the PhD in Robotics from Carnegie Mellon University in 1995. He became Assistant Professor at the University of Illinois-Chicago in 1995, Associate Professor at the University of Minnesota in 1998, and Professor of Robotics and Intelligent Systems at ETH Zurich in 2002. His research involves microrobotics and nanorobotics.

Duy Nguyen-Tuong

Robert Bosch GmbH
Corporate Research
Stuttgart, Germany
duy@robot-learning.de

Chapter A.15

Duy Nguyen-Tuong is a research scientist at Bosch Corporate Research in Schwieberdingen, Germany where he heads a project group on machine learning aplications for industrial robotics projects. Before joing Bosch, Duy Nguyen-Tuong has been a PhD student in the Robot Learning Lab at the Max Planck Institute for Biological Cybernetics. He graduated in Spring 2011 with a PhD from the University of Freiburg with a thesis on Model Learning in Robot Control.

Monica Nicolescu

University of Nevada
Department of Computer Science and Engineering
Reno, USA
monica@unr.edu

Chapter A.13

Dr. Monica Nicolescu is an Associate Professor with the Computer Science and Engineering Department at the University of Nevada, Reno and is the Director of the UNR Robotics Research Lab. In 2006 she received the NSF Early Career Development Award (CAREER) Award. Her research interests are in the areas of human-robot interaction, robot control, learning, and multi-robot systems.

Günter Niemeyer

Disney Research
Glendale, USA
gunter.niemeyer@email.disney.com

Chapter D.43

Dr. Günter Niemeyer is a Senior Research Scientist at Disney Research, Los Angeles. His research examines physical human-robotic interactions and interaction dynamics, force sensitivity and feedback, teleoperation with and without communication delays, and haptic interfaces. Dr. Niemeyer received his MS and PhD from MIT. In 2012 he joined Disney Research.

Klas Nilsson

Lund Institute of Technology
Department of Computer Science
Lund, Sweden
klas.nilsson@cs.lth.se

Chapter F.54

As an Associate Professor at the Department of Computer Science, where he is heading the group for Robotics and Semantic Systems, Dr. Nilsson has a leading role in several research projects. His research includes robot control systems and related software technologies. Recently, Dr. Nilsson founded the company Cognibotics AB that provides solutions for improved robot accuracy.

Stefano Nolfi

National Research Council (CNR)
Institute of Cognitive Sciences and Technologies
Rome, Italy
stefano.nolfi@istc.cnr.it

Chapter G.76

Dr. Stefano Nolfi is Director of the Laboratory of Autonomous Robots and Artificial Life within the Institute of Cognitive Sciences and Technologies of the Italian National Research Council (CNR-ISTC). His research activities focus on evolutionary robotics, collective robotics, adaptive behavior, complex systems, and embodied cognitive sciences.

Illah Nourbakhsh

Carnegie Mellon University
Robotics Institute
Pittsburgh, USA
illah@andrew.cmu.edu

Chapter G.79

Illah R. Nourbakhsh is Professor of Robotics and Head of the Robotics Master's Program in The Robotics Institute at Carnegie Mellon University. He is Director of the Community Robotics, Education and Technology Empowerment (CREATE) Lab. He is co-author of *Introduction to Autonomous Mobile Robots* and author of *Robot Futures*.

Andreas Nüchter

University of Würzburg
Informatics VII – Robotics and Telematics
Würzburg, Germany
andreas@nuechti.de

Chapter C.31

Andreas Nüchter is a Professor of Telematics at the University of Würzburg, Germany. Before he headed the Automation Group at Jacobs University Bremen as an Assistant Professor, was with the University of Osnabrück, Fraunhofer AIS, and the University of Bonn. Andreas' research interests include 3-D vision, and laser scanning technologies, resulting in fast algorithms for processing 3-D point clouds that enable robots to map their environment in 3-D.

Paul Y. Oh

University of Nevada
Department of Mechanical Engineering
Las Vegas, USA
paul@coe.drexel.edu

Chapter B.26

Paul Oh is the Lincy Professor of Unmanned Aerial Systems in the Mechanical Engineering Department at the University of Nevada, Las Vegas (UNLV). Formerly, from 2000 to 2014, he served as amechanical engineering professor at Drexel University in Philadelphia and both founded and directed the Drexel Autonomous Systems Laboratory (DASL). He received his Mechanical Engineering degrees from McGill (B.Eng. 1989), Seoul National (MSc 1992), and Columbia (PhD 1999).

Yoshito Okada

Tohoku University
Department of Aerospace Engineering,
Graduate School of Engineering
Sendai, Japan
okada@rm.is.tohoku.ac.jp

Chapter E.50

Yoshito Okada is an Assistant Professor at Tohoku University, Japan. He received his PhD from Tohoku University in 2012. His research interest is control, navigation, and mechanism of field robots, especially, tracked vehicles for search and rescue missions and aerial robots for structure inspection.

Allison M. Okamura

Stanford University
Department of Mechanical Engineering
Stanford, USA
aokamura@stanford.edu

Chapter D.42

Allison Okamura is a Professor at Stanford University in the Department of Mechanical Engineering, with a courtesy appointment in Computer Science. Her awards include the NSF CAREER Award and the IEEE Robotics and Automation Society Early Academic Career Award. She is an IEEE Fellow. Her interests include haptics, teleoperation, virtual environments, medical robotics, prosthetics, neuromechanics, rehabilitation, and engineering education.

Fiorella Operto

Scuola di Robotica
Genoa, Italy
operto@scuoladirobotica.it

Chapter G.80

Fiorella Operto educated in Philosophy, she collaborated with the Italian Research Council (IAN and IEIIT Institutes) on the social impact of advanced robotics applications. In 2000 she founded the School of Robotics Society and is its Chair today. Operto contributed to the definition and development of Roboethics. She is Member of the high-level Advisory Committee for European Centre for Women and Technology, consultant and member of *The Open Roboethics Initiative*, and member of the Ethics Committee of the Human Brain Project HBP.

David E. Orin Chapter A.3 For biographical profile, please see the section "About the Part Editors".

Giuseppe Oriolo

University of Rome "La Sapienza"
Department of Computer, Control, and Management Engineering
Rome, Italy
oriolo@diag.uniroma1.it

Chapter A.10

Giuseppe Oriolo is an Associate Professor of Automatic Control and Robotics at the Sapienza University of Rome and the coordinator of the DIAG Robotics Laboratory. His research interests are robot control; motion planning; redundant, underactuated and nonholonomic systems; localization and navigation for mobile robots; humanoid robots; visual servoing. He has been Associate Editor of the IEEE Transactions on Robotics from 2001 to 2005 and Senior Editor from 2009 to 2013.

Christian Ott

German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Wessling, Germany
christian.ott@dlr.de

Chapter B.17

Christian Ott received the Dr.-Ing. degree in Control Engineering from Saarland University, Germany, in 2005. In 2001, he joined the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). During 2007–2009, he was a Project Assistant Professor in the Department of Mechano-Informatics at the University of Tokyo, Japan. In 2014, he became Head of the Department for Analysis and Control of Advanced Robotic Systems at DLR.

Ümit Özgüner

Ohio State University
Department of Electrical and Computer Engineering
Columbus, USA
umit@ee.eng.ohio-state.edu

Chapter F.62

Ümit Özgüner is a Professor of Electrical and Computer Engineering at the Ohio State University, where he holds the TRC Inc. Chair on ITS and is the Director of the new *Crash Imminent Safety* University Transportation Center. He was the founding President of the IEEE ITS Council (now Society) in 1999 and is a Fellow of the IEEE.

Nikolaos Papanikolopoulos

University of Minnesota
Department of Computer Science and Engineering
Minneapolis, USA
npapas@cs.umn.edu

Chapter F.61

Papanikolopoulos is the Distinguished McKnight University Professor at the University of Minnesotaand Director of the Center for Distributed Robotics and SECTTRA. He holds a MS and PhD in Electrical and Computer Engineering from Carnegie Mellon University, and Diploma of Engineering, Electrical and Computer Engineering from the National Technical University of Athens, Greece (1987). His primary specialties are robotics, computer vision, sensor networks, and sensor-based control in transportation applications.

Frank C. Park Chapter B.16 For biographical profile, please see the section "About the Part Editors".

Jaeheung Park

Seoul National University
Department of Transdisciplinary Studies
Suwon, Korea
park73@snu.ac.kr

Chapter D.40

Jaeheung Park is Associate Professor at Seoul National University. He received the BS and MS degrees from Seoul National University, Korea, in 1995 and 1999, respectively, and the PhD degree from Stanford University in 2006. His research interests lie in the areas of humanoid robot, whole-body control framework, contact force control, robust haptic teleoperation, multi-contact control, and biomechanics.

Lynne E. Parker

University of Tennessee
Department of Electrical Engineering and Computer Science
Knoxville, USA
leparker@utk.edu

Chapter E.53

Lynne Parker is a Professor in the Department of Electrical Engineering and Computer Science at the University of Tennessee, Knoxville. She received the PhD degree from the Massachusetts Institute of Technology. She previously worked for several years as a full-time researcher at Oak Ridge National Laboratory. Her research focuses on distributed intelligent robotics, human–robot interaction, sensor networks, and machine learning.

Federico Pecora

University of Örebro
School of Science and Technology
Örebro, Sweden
federico.pecora@oru.se

Chapter A.14

Federico Pecora, PhD, is Associate Professor in Computer Science at Örebro University, Sweden. His research interests lie at the intersection of AI and robotics with a focus on constraint-based reasoning and planning algorithms. He has applied these techniques primarily in the development of domestic service robots, and in industrial automation domains with large autonomous vehicles.

Jan Peters

Technical University Darmstadt
Autonomous Systems Lab
Darmstadt, Germany
mail@jan-peters.net

Chapter A.15

Jan Peters is a Full Professor (W3) for Intelligent Autonomous Systems in the Computer Science Department of the Technical University Darmstadt and a Group Leader at the Max-Planck Institute for Intelligent Systems. He has received four Master's degrees in Computer Science, Electrical, Mechanical and Control Engineering from TU Munich and FernUni Hagen, National University of Singapore (NUS), and the University of Southern California (USC). He holds a Computer Science PhD from USC. Jan Peters has received the Dick Volz Best 2007 US PhD Thesis Runner-Up Award, the Robotics: Science & Systems – Early Career Spotlight, the INNS Young Investigator Award, and the IEEE Robotics & Automation Society's Early Career Award.

Anna Petrovskaya

Stanford University
Department of Computer Science
Stanford, USA
anya@cs.stanford.edu

Chapter D.41

Anna Petrovskaya received her PhD in Computer Science from Stanford University. Her expertise is in computer vision and robotics with focus on Bayesian methods for robotic and consumer applications. She developed efficient algorithms for autonomous vehicles, mobile manipulation, and tactile perception. Versed in both industry and academia, she has a strong interest in bringing latest research from the lab into everyday use.

J. Norberto Pires

University of Coimbra
Department of Mechanical Engineering
Coimbra, Portugal
norberto@uc.pt

Chapter F.54

J. Norberto Pires received the PhD degree in 1999 in Robotics and Automation from the University of Coimbra, Portugal. Since 1991 he has been with the Mechanical Engineering staff, where he currently serves as Associate Professor. His research interests include force control, industrial robotics, object-oriented and distributed programming, flexible manufacturing systems, and human–machine interfaces. He is an IEEE Senior Member, editor-in-chief of journal *Robotica*.

Paolo Pirjanian

iRobot Corp.
Bedford, USA
paolo.pirjanian@gmail.com

Chapter F.65

Dr. Paolo Pirjanian got a MSc in Computer Engineering and a PhD in Robotics from Aalborg University, Denmark. He is the CTO of iRobot and former CEO of Evolution Robotics with successful exit. He is a high-tech innovator, startup founder and entrepreneur. Dr. Pirjanian is strong strategist and innovator with operational experience in developing world-class teams and technologies from concept to market leadership while creating strong defensible IP portfolios and differentiated, competitive products.

Erwin Prassler

Bonn-Rhein-Sieg Univ. of Applied Sciences
Department of Computer Sciences
Sankt Augustin, Germany
erwin.prassler@h-brs.de

Chapter F.65

Erwin Prassler received a Master's degree in Computer Science from the Technical University of Munich in 1985. In 1989, he joined the Research Institute for Applied Knowledge Processing in Ulm, where he headed a research group working in the field of mobile robots and service robotics where he received his PhD in Computer Science in 1996. In 2004, Dr. Prassler was appointed Associate Professor at the University of Applied Science Bonn-Rhein-Sieg and serves as IEEE RAS Vice President for Industrial Activities in 2016–2017.

Domenico Prattichizzo

University of Siena
Department of Information Engineering
Siena, Italy
prattichizzo@ing.unisi.it

Chapter D.38

Domenico Prattichizzo received the MS degree in Electronic Engineering and the PhD degree in Robotics and Automation from the University of Pisa, Italy in 1991 and 1995, respectively. Since 2002 he is Associate Professor of Robotics at the University of Siena and since 2009 Scientific Consultant at the Italian Institute of Technology, Italy. His research interests are in robotic grasping, haptics, visual servoing, mobile robotics and geometric control.

Carsten Preusche

German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Wessling, Germany
carsten.preusche@dlr.de

Chapter D.43

Carsten Preusche became a Scientist at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR) in 1999. Since 2004 he is head of the telepresence and telerobotic group. His work is mainly focused on the control of time-delayed telerobotic systems and the design of multimodal human–system interfaces.

William Provancher

University of Utah
Department of Mechanical Engineering
Salt Lake City, USA
wil@mech.utah.edu

Chapter C.28

William R. Provancher earned a PhD from the Department of Mechanical Engineering at Stanford University in haptics, tactile sensing and feedback. He is now a tenured Associate Professor in the Department of Mechanical Engineering at the University of Utah. His current research involves haptics and tactile feedback. He has also founded Tactical Haptics to commercialize his lab's research related to tactile feedback.

John Reid

John Deere Co.
Moline Technology Innovation Center
Moline, USA
reidjohnf@johndeere.com

Chapter F.56

Dr. John Reid is the Director of Product Technology and Innovation at Deere and Co., where he is responsible for the company's worldwide network of R&D centers. Dr. Reid and his colleagues jointly orchestrate an accelerated innovation process and enterprise technology process to coordinate the delivery of technology innovations to the marketplace and to build capabilities to support future businesses.

David J. Reinkensmeyer

University of California at Irvine
Mechanical and Aerospace Engineering
and Anatomy and Neurobiology
Irvine, USA
dreinken@uci.edu

Chapter F.64

David Reinkensmeyer received the PhD degree in Electrical Engineering from the University of California at Berkeley in 1993, and was a Postdoctoral Fellow at the Rehabilitation Institute of Chicago. He is currently a Professor in the Department of Mechanical and Aerospace Engineering at the University of California, Irvine. His research interests are in neuromuscular control, motor learning, robotics, and neurorehabilitation.

Jonathan Roberts

Queensland University of Technology
Department of Electrical Engineering and
Computer Science
Brisbane, Australia
jonathan.roberts@qut.edu.au

Chapter F.66

Jonathan Roberts is Professor in Robotics at Queensland University of Technology (QUT). His main research interest is in the area of field robotics. Before joining QUT, he was Research Director of CSIRO's Autonomous Systems Laboratory. Jonathan is a Past President of the Australian Robotics & Automation Association Inc., and was Deputy Research Director of the Australian Research Centre for Aerospace Automation.

Nicholas Roy

Massachusetts Institute of Technology
Department of Aeronautics and
Astronautics
Cambridge, USA
nickroy@csail.mit.edu

Chapter G.71

Nicholas Roy is an Associate Professor in the Department of Aeronautics & Astronautics and a member of the Computer Science and Artificial Intelligence Laboratory at the MIT. He received his PhD in Robotics from Carnegie Mellon University in 2003. His research interests include autonomous unmanned vehicles, human-robot interaction, decision-making under uncertainty and machine learning.

Daniela Rus Chapter E.53 For biographical profile, please see the section "About the Part Editors".

Selma Šabanović Chapter G.71

Indiana University Bloomington
School of Informatics and Computing
Bloomington, USA
selmas@indiana.edu

Selma Šabanović is an Assistant Professor of Informatics and Cognitive Science at Indiana University Bloomington. She received her doctorate in Science and Technology Studies from Rensselear Polytechnic Institute in 2007. Her work combines the social study of interactive and assistive robots in various social and cultural contexts with research on human-robot interaction and social robot design.

Kamel S. Saidi Chapter F.57

National Institute of Standards and Technology
Building and Fire Research Laboratory
Gaitherbsurg, USA
kamel.saidi@nist.gov

Dr. Kamel S. Saidi is a Mechanical Engineer in the Intelligent Systems Division (ISD) at the National Institute of Standards and Technology (NIST). His research includes the development of metrics, test methods and ground-truth systems, safety, dexterity, mapping and autonomy for manufacturing and emergency response robots. He holds a PhD in Civil Engineering, a BS and MS in Mechanical Engineering from The University of Texas at Austin.

Claude Samson Chapter E.49

INRIA Sophia-Antipolis
Sophia-Antipolis, France
claude.samson@inria.fr

Claude Samson is Directeur de Recherche at the French research institute INRIA. Much of his research concerns robotics and automatic control, with a particular interest in the control of nonlinear mechanical systems. He has published the book *Robot Control. The Task-Function Approach* with his colleagues Bernard Espiau and Michel Leborgne.

Brian Scassellati Chapter G.73

Yale University
Computer Science, Cognitive Science, and Mechanical Engineering
New Haven, USA
scaz@cs.yale.edu

Brian Scassellati is a Professor of Computer Science, Cognitive Science, and Mechanical Engineering at Yale University and Director of the NSF Expedition on Socially Assistive Robotics. His research focuses on building embodied computational models of human social behavior, especially the developmental progression of early social skills. Dr. Scassellati received his PhD in Computer Science from the MIT in 2001.

Stefan Schaal Chapter A.15

University of Southern California
Depts. of Computer Science, Neuroscience, and Biomedical Engineering
Los Angeles, USA
sschaal@tuebingen.mpg.de

Stefan Schaal is Professor of Computer Science, Neuroscience, and Biomedical Engineering at the University of Southern California, and a Founding Director of the MPI for Intelligent Systems in Tübingen, Germany. Before joining USC, Dr. Schaal did research in the AI Lab at MIT, the ATR in Japan, at the Georgia Institute of Technology and in the Dep. Kinesiology of PennState University. His research interests include statistical and machine learning, neural networks, computational neuroscience, functional brain imaging, nonlinear control theory, and biomimetic robotics. He applies his research to problems of artificial and biological motor control and motor learning.

Steven Scheding Chapter F.59

University of Sydney
Rio Tinto Centre for Mine Automation
Sydney, Australia
steven.scheding@sydney.edu.au

Steven Scheding received the BE and PhD degrees in Mechatronic Engineering from the University of Sydney, Australia, in 1995 and 1998, respectively. He held the position of Senior Lecturer with the Australian Centre for Field Robotics, University of Sydney from 2000 to 2010. He is currently Director of the Rio Tinto Centre for Mine Automation at the University of Sydney. His research interests include the automation of large outdoor vehicles, fault detection and identification, and algorithms and sensors for perception in unstructured environments.

Victor Scheinman

Stanford University
Department of Mechanical Engineering
Stanford, USA
vds@stanford.edu

Chapter A.4

Victor Scheinman is a Consulting Professor in Mechanical Engineering at Stanford University. He has engineering degrees from MIT and Stanford. He was a founder of Automatix Inc. and Vicarm Inc. He designed and developed the Unimation PUMA robot, the Automatix/Yaskawa Robotworld, the Scheinman/Stanford Arm, and the Scheinman/MIT robot arms. He is a recipient of the ASME's Leonardo DaVinci Award for Design Engineering and the RIA's Joseph F. Engelberger Award for Robotics Technology.

Bernt Schiele

Saarland University
Department of Computer Science
Saarbrücken, Germany
schiele@mpi-inf.mpg.de

Chapter C.33

Bernt Schiele is Max Planck Director at MPI Informatics and Professor at Saarland University. Before coming to Saarbrücken, he was Full Professor at TU Darmstadt, Germany, Assistant Professor at ETH Zurich, Switzerland, and postdoctoral associate and Visiting Assistant Professor at MIT, USA. He holds a PhD from INPG, France. He is interested in computer vision, machine learning and ubiquitous computing.

James Schmiedeler

University of Notre Dame
Department of Aerospace and Mechanical Engineering
Notre Dame, USA
schmiedeler.4@nd.edu

Chapter A.2

James Schmiedeler is an Associate Professor in the Department of Aerospace and Mechanical Engineering at the University of Notre Dame. His current research activities involve biped robot locomotion, human motor coordination, robot-assisted rehabilitation, human biomechanics, and design of morphing systems. His research has been recognized with a National Science Foundation Presidential Early Career Award for Scientists and Engineers (PECASE), and he is a Fellow of the ASME.

Bruno Siciliano Chapter 1 For biographical profile, please see the section "About the Editors".

Roland Siegwart

ETH Zurich
Department of Mechanical Engineering
Zurich, Switzerland
rsiegwart@ethz.ch

Chapter B.26

Roland Siegwart is Full Professor for Robotics at ETH Zürich. He was formerly Professor at EPFL, R&D Director at MECOS, and held postdoc and visiting positions at Stanford University and NASA Ames. He was Vice President for Technical Activities of the IEEE Robotics and Automation Society and is author of the textbook *Autonomous Mobile Robots*. He is co-founder of several spin-off companies.

Reid Simmons

Carnegie Mellon University
The Robotics Institute
Pittsburgh, USA
reids@cs.cmu.edu

Chapter A.12

Reid Simmons received the PhD degree in Artificial Intelligence from MIT in 1988. His research focuses on autonomous, self-reliant robots, which involves issues ofrobot control architectures, probabilistic planning and reasoning, robot navigation,multirobot coordination, automated assembly, and human–robot social interaction. Dr. Simmons has developed over a dozen autonomous robots, receiving the Newell Awardfor Research Excellence in 2004.

Patrick van der Smagt

Technical University Munich
Department of Computer Science, BRML Labs
Munich, Germany
smagt@brml.org

Chapter G.77

Patrick van der Smagt is Professor for Computer Science at TUM and researcher at fortiss GmbH Munich, where he directs a joint lab on biomimetic robotics and machine learning. Besides numerous papers and patents on robotics, he has won various awards, including the Helmholtz-Association Erwin Schrödinger Award, the King-Sun Fu Memorial Award, and the Harvard Medical School/MGH Martin Research Prize. He is founding chairman of a non-for-profit organization for Assistive Robotics for Tetraplegics.

Dezhen Song Chapter D.44 For biographical details see "About the Multimedia Editors"

Jae-Bok Song

Korea University
Department of Mechanical Engineering
Seoul, Korea
jbsong@korea.ac.kr

Chapter A.4

Jae-Bok Song received the BS and MS degrees in mechanical engineering from Seoul National University and the PhD degree from MIT in 1992. He joined the faculty of the School of Mechanical Engineering, Korea University, in 1993. Dr. Song served as a President of the Korea Robotics Society and as an Editor-in-Chief of the International Journal of Control, Automation and Systems.

Cyrill Stachniss

University of Bonn
Institute for Geodesy and Geoinformation
Bonn, Germany
cyrill.stachniss@igg.uni-bonn.de

Chapter E.46

Cyrill Stachniss is a Full Professor for Photogrammetry at the University of Bonn, Germany. Before, he was a Lecturer at the University of Freiburg, a Senior Researcher at ETH Zurich, and a Guest Lecturer at the University of Zaragoza. In 2006, he finished his PhD thesis supervised by Wolfram Burgard in Freiburg.

Michael Stark

Max Planck Institute of Informatics
Department of Computer Vision and
Multimodal Computing
Saarbrücken, Germany
stark@mpi-inf.mpg.de

Chapter C.33

Michael Stark is a research group leader at MPI Informatics. Before coming to Saarbrücken, he was a Visiting Assistant Professor at Stanford University. He holds a PhD from TU Darmstadt, Germany. His research interests are in computer vision and machine learning.

Amanda K. Stowers

Stanford University
Department Mechanical Engineering
Stanford, USA
astowers@stanford.edu

Chapter B.26

Amanda Stowers is a Graduate Student in Mechanical Engineering at Stanford University. She studies the biomechanics of bird wing morphing and flapping as an inspiration for designing more effective flapping-winged robots.

Stefano Stramigioli Chapter D.43 For biographical details see "About the Multimedia Editors"

Gaurav S. Sukhatme

University of Southern California
Department of Computer Science
Los Angeles, USA
gaurav@usc.edu

Chapter E.53

Gaurav S. Sukhatme is Dean's Professor of Computer Science (CS) with joint appointment in Electrical Engineering at the USC. He received his undergraduate education at IIT Bombay in CS and Engineering, and MS and PhD degrees in CS from USC. He is the Director of the USC Robotic Embedded Systems Laboratory which he founded in 2000. His research interests are in multirobot systems and robot networks. He is a fellow of the IEEE and a recipient of the NSF CAREER award and the Okawa foundation research award.

Satoshi Tadokoro

Tohoku University
Graduate School of Information Sciences
Sendai, Japan
tadokoro@rm.is.tohoku.ac.jp

Chapter F.60

Satoshi Tadokoro is a Professor at the Graduate School of Information Sciences, and at the International Research Institute of Disaster Science, Tohoku University. He is President of the International Rescue System Institute since 2002, and was a Program Manager of Japan Government's DDT Project 2002–2007. He also is the Program Manager of the ImPACT Project since 2014. His research field is rescue robotics. He serves the IEEE Robotics and Automation Society as President in 2016–2017.

Wataru Takano

University of Tokyo
Department of Mechano-Informatics
Tokyo, Japan
takano@ynl.t.u-tokyo.ac.jp

Chapter G.68

Dr. Takano completed his master at the Department of Precision Engineering, Kyoto University in 2001 and his PhD in Information Engineering at the University of Tokyo in 2006. He became a Project Assistant Professor in 2006, an Assistant Professor in 2007, and then a Lecturer at the University of Tokyo in 2009. His research includes kinematics, dynamics, artificial intelligence of humanoid robots, and intelligent vehicles. He is an IEEE member co-chair of the Technical Committee of Robot Learning, IEEE RAS.

Russell H. Taylor

The Johns Hopkins University
Department of Computer Science
Baltimore, USA
rht@jhu.edu

Chapter F.63

Russell H. Taylor received a PhD in Computer Science from Stanford in 1976. He joined IBM Research in 1976, where he developed the AML programming language and managed the Computer-Assisted Surgery group before moving to Johns Hopkins University in 1995. He is the John C. Malone Professor of Computer Science with joint appointments in Mechanical Engineering, Radiology, and Surgery and Director of the Laboratory for Computational Sensing and Robotics.

Russ Tedrake

Massachusetts Institute of Technology
Computer Science and Artificial
Intelligence Laboratory (CSAIL)
Cambridge, USA
russt@csail.mit.edu

Chapter E.48

Russ is the X Consortium Associate Professor of Electrical Engineering and Computer Science and Aero/Astro at MIT, and the Director of the Center for Robotics at the Computer Science and Artificial Intelligence Lab. His research focuses on optimization, planning, and control for nonlinear dynamical systems including legged robots, robot manipulation, and unmanned aerial vehicles.

Sebastian Thrun

Udacity Inc.
Mountain View, USA
info@udacity.com

Chapter E.46

Sebastian Thrun is a Research Professor of Computer Science at Stanford University, Founder of Google X, and CEO of Udacity, Inc. Thrun directed the Stanford team that won the DARPA Grand Challenge. At Google, Thrun founded Google X, which was home to Google Glass, the Google self-driving car, project Loon, project Wing, medical contact lenses, new solutions to early cancer detection, and other projects.

Marc Toussaint

University of Stuttgart
Machine Learning and Robotics Lab
Stuttgart, Germany
marc.toussaint@ipvs.uni-stuttgart.de

Chapter D.40

Marc Toussaint is Full Professor of Machine Learning and Robotics at the University of Stuttgart since 2012. Before he was Assistant Professor at FU & TU Berlin, Germany. His research focuses on the combination of decision theory and machine learning, motivated by fundamental research questions in robotics. Specific interests are combining geometry, logic and probabilities in learning and reasoning.

James Trevelyan

The University of Western Australia
School of Mechanical and Chemical
Engineering
Crawley, Australia
james.trevelyan@uwa.edu.au

Chapter F.58

James Trevelyan pioneered sheep shearing robots between 1976 and 1993, then worked on remote access laboratories, landmine clearance, and understanding engineering practice. He received the Joseph Engelberger award for contributions to robotics science in 1993, two JIRA awards, and numerous awards for distinguished teaching. He is a Fellow of Engineers Australia and served 8 years as Elected Member of the IFToMM Executive Council from 2003.

Jeffrey C. Trinkle

Rensselaer Polytechnic Institute
Department of Computer Science
Troy, USA
trink@cs.rpi.edu

Chapter D.38

Jeffrey C. Trinkle received his BS in Physics (1979) and Engineering Science and Mechanics (1979) from Ursinus College and Georgia Institute of Technology, respectively. In 1987, he received his PhD from the University of Pennsylvania. He is now Professor and Chair of Computer Science at Rensselaer Polytechnic Institute in Troy, New York. His primary research interests are in the areas of robotic manipulation, multibody dynamics, and automated manufacturing.

Masaru Uchiyama

Tohoku University
Graduate School of Engineering
Sendai, Japan
uchiyama@space.mech.tohoku.ac.jp

Chapter D.39

Masaru Uchiyama received BE, ME, and PhD degrees from the University of Tokyo, Tokyo, Japan, in 1972, 1974, and 1977, respectively, all in Mechanical Engineering for Production. Since 1977, he has been with the School of Engineering, Tohoku University and currently is a Professor of the Design of Intelligent Machines in the Department of Mechanical Systems and Design. His research interest is design of intelligent mechanical systems.

H.F. Machiel Van der Loos

University of British Columbia
Department of Mechanical Engineering
Vancouver, Canada
vdl@mech.ubc.ca

Chapter F.64

Dr. Machiel Van der Loos is Adjunct Professor of Mechanical Engineering at UBC, Vancouver, Canada. He is Consulting Associate Professor of Mechanical Engineering at Stanford University, collaborating on design coaching and thesis advising. He received the Ingénieur Mécanicien degree from the Swiss Federal Institute of Technology, Lausanne (1979), and an Engineer's Degree (1984) and PhD (1992) from Stanford University.

Manuela Veloso

Carnegie Mellon University
Computer Science Department
Pittsburgh, USA
mmv@cs.cmu.edu

Chapter F.66

Manuela Veloso is Herbert A. Simon University Professor in Computer Science at Carnegie Mellon University. Veloso researches in artificial intelligence and robotics, perception, cognition, and learning for autonomous agents that collaborate, observe, reason, act, and learn. Veloso is a Fellow of IEEE, AAAS and AAAI, and the past President of AAAI and RoboCup. Veloso and her students have researched among others, soccer robots and service symbiotic autonomous mobile robots.

Gianmarco Veruggio

National Research Council (CNR)
Institute of Electronics, Computer and Telecommunication Engineering
Genoa, Italy
gianmarco@veruggio.it

Chapter G.80

Dr. Veruggio received the Degree in Electronic Engineering from Genoa University, Italy, in 1980. He is Director of Research at the Italian National Research Council and Responsible for the Operational Unit of Genoa of CNR-IEIIT. In 2000 he founded the Association *School of Robotics* and in 2002 he coined the word and proposed the concept of Roboethics, to which he is dedicating increasing resources. In 2009 he was awarded the title of Commander of the Order to the Merit of the Italian Republic.

Luigi Villani

University of Naples Federico II
Department of Electrical Engineering and Information Technology
Naples, Italy
luigi.villani@unina.it

Chapter A.9

Luigi Villani received the PhD degree in Electrical Engineering and Computer Science from the University of Naples in 1996, where he is Associate Professor of Automatic Control since 2002. His research interests include robot force/motion control, cooperative robots, visual servoing, safe physical human-robot interaction. He has co-authored five books and more than 150 journal and conference papers. He has been Associate Editor of the IEEE Transactions on Control Systems Technology and the IEEE Transactions on Robotics.

Kenneth J. Waldron

University of Technology Sydney
Centre of Mechatronics and Intelligent Systems
Ultimo, Australia
kenneth.waldron@uts.edu.au

Chapter A.2

Kenneth Waldron is Professor in the School of Electrical, Mechanical and Mechatronic Engineering of the University of Technology, Sydney and Professor Emeritus of Stanford University. He has previously occupied academic positions at the University of New South Wales, Houston and Ohio State University. He is past President of IFToMM, the International Federation for Promotion of Mechanism and Machine Science.

Ian D. Walker

Clemson University
Department of Electrical and Computer Engineering
Clemson, USA
ianw@ces.clemson.edu

Chapter B.20

Ian D. Walker received the BS from the University of Hull, England, in 1983 and the MS and PhD degrees from the University of Texas, Austin, in 1985 and 1989, respectively. He is currently a Professor at Clemson University. His research centers on robotics, particularly novel manipulators and manipulation. He conducts research in biologically inspired *trunk, tentacle, and worm* robots.

Christian Wallraven

Korea University
Department of Brain and Cognitive Engineering, Cognitive Systems Lab
Seoul, Korea
wallraven@korea.ac.kr

Chapter G.78

Christian Wallraven received his PhD in Physics in 2007 from the University of Tübingen. After two European projects in cognitive vision and cognitive systems, he moved to Korea University, where he founded the Cognitive Systems Lab in 2010. His work is set on the interface between machine learning & computer vision, robotics, and neuroscience. Recent work is on modeling of face perception, facial expressions for communication, as well as multisensory object representations in human and robots.

Pierre-Brice Wieber

INRIA Grenoble Rhône-Alpes
Grenoble, France
pierre-brice.wieber@inria.fr

Chapter E.48

Pierre-Brice Wieber graduated from Ecole Polytechnique in 1996 and received a PhD degree from Ecole des Mines de Paris in 2000. He has been Chargé de Recherche at INRIA Grenoble since 2001, except for a 2 year Marie Curie Fellowship at JRL in Tsukuba, Japan in 2008-2010. His research interests include walking robots and optimization based control.

Brian Wilcox

California Institute of Technology
Jet Propulsion Laboratory
Pasadena, USA
brian.h.wilcox@jpl.nasa.gov

Chapter F.55

Brian Wilcox is at the Jet Propulsion Laboratory and is the Principal Investigator for the ATHLETE robot. He was the Supervisor of the Robotic Vehicles Group at JPL from 1985–2005, leading the development of rover technology leading to the Sojourner rover in 1997 and subsequent Mars rovers. He holds six patents and has received the NASA Exceptional Engineering Achievement Medal.

Robert Wood

Harvard University
School of Engineering and Applied Sciences
Cambridge, USA
rjwood@seas.harvard.edu

Chapter B.23

Robert Wood is a Professor in Harvard's School of Engineering and Applied Sciences and founder of the Harvard Microrobotics Lab which leverages expertise in microfabrication. His current research interests include new micro- and meso-scale manufacturing techniques, fluid mechanics of low Reynolds number flapping wings, control of sensor-limited and computation-limited systems, active soft materials, wearable robots, and morphable soft-bodied robots. In 2012 he was selected for the Alan T. Waterman award of the National Science Foundation.

Jing Xiao

University of North Carolina
Department of Computer Science
Charlotte, USA
xiao@uncc.edu

Chapter D.36

Jing Xiao received the PhD degree from the University of Michigan in 1990. She is currently a Professor of Computer Science at the University of North Carolina, Charlotte. She served as the Director of the Robotics and Human Augmentation Program at the U.S. National Science Foundation from 1998 to 2000. Her research covers robotics and haptics, and she has about 100 publications.

Katsu Yamane

Disney Research
Pittsburgh, USA
kyamane@disneyresearch.com

Chapter G.68

Dr. Katsu Yamane received his BS, MS, and PhD degrees in Mechanical Engineering in 1997, 1999, and 2002, respectively, from the University of Tokyo, Japan. He is currently a Senior Research Scientist at Disney Research, Pittsburgh and an Adjunct Associate Professor at the Robotics Institute, Carnegie Mellon University. Prior to joining Disney, he was an Associate Professor at the University of Tokyo. His research interests include humanoid robot control and motion synthesis, character animation, and human motion simulation.

Mark Yim

University of Pennsylvania
Department of Mechanical Engineering
and Applied Mechanics
Philadelphia, USA
yim@seas.upenn.edu

Chapter B.22

Mark Yim is a professor at the University of Pennsylvania. His group studies modular self-reconfigurable robots that can transform into different shapes, jump, ride tricycles, climb stairs, poles and fences, manipulate objects and reassemble themselves after being kicked into pieces. His other research interests include product design, reactive art and architecture, low cost manipulation, snake locomotion, flying robots, and self-assembling floating structures.

Dana R. Yoerger

Woods Hole Oceanographic Institution
Applied Ocean Physics & Engineering
Woods Hole, USA
dyoerger@whoi.edu

Chapter E.51

Dana Yoerger has been a robotics researcher, system developer, and heavily involved in operations at the Woods Hole Oceanographic Institution since 1984. He holds SB, SM, and PhD degrees from the Massachusetts Institute of Technology. He has used underwater robots to explore the Mid-Ocean Ridge, ancient and modern shipwrecks, and to study the Deepwater Horizon oil spill.

Kazuhito Yokoi

AIST Tsukuba Central 2
Intelligent Systems Research Institute
Tsukuba, Ibaraki, Japan
kazuhito.yokoi@aist.go.jp

Chapter G.67

Kazuhito Yokoi received the BE in Mechanical Engineering from the Nagoya Institute of Technology in 1984, and the ME and PhD degrees from the Tokyo Institute of Technology in 1986 and 1994, respectively. In 1986, he joined the Mechanical Engineering Laboratory, Ministry of International Trade and Industry. He is currently a Deputy Director of Intelligent Systems Research Institute (IS), National Institute of Advanced Industrial Science and Technology (AIST), in Tsukuba, Japan.

Eiichi Yoshida

National Institute of Advanced Industrial
Science and Technology (AIST)
CNRS–AIST Joint Robotics Laboratory,
UMI3218/CRT
Tsukuba, Ibaraki, Japan
e.yoshida@aist.go.jp

Chapter G.67

Eiichi Yoshida received ME and PhD degrees from Graduate School of Engineering, University of Tokyo in 1993 and 1996, respectively. He is currently Co-Director of CNRS-AIST Joint Robotics Laboratory (JRL), UMI3218/CRT, at the National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan. His research interests include robot task and motion planning, human modeling, and humanoid robots.

Kazuya Yoshida

Tohoku University
Department of Aerospace Engineering
Sendai, Japan
yoshida@astro.mech.tohoku.ac.jp

Chapter F.55

Kazuya Yoshida received his Dr. Eng. from the Tokyo Institute of Technology in 1990. Since 2003 he has been Professor in the Department of Aerospace Engineering, Tohoku University, Japan. He is also serving as a Visiting Lecturer at the International Space University in Strasbourg, France, since 1998. His research activities include dynamics and control of space robots covering from free-flying robots to planetary exploration rovers.

Junku Yuh

Korea Institute of Science and Technology
National Agenda Research Division
Seoul, Korea
yuh.junku@gmail.com

Chapter B.25

Dr. Yuh, IEEE Fellow, is currently Endowed Chair Researcher at Korea Institute of Science and Technology (KIST). He served as the 5th and 6th President of the Korea Aerospace University, US National Science Foundation Program Director, Head of NSF Tokyo Regional Office, Professor of Mechanical Engineering and Information & Computer Science at the University of Hawaii.

Alex Zelinsky Chapter F.62 For biographical profile, please see the section "About the Part Editors".

Index

2-D flow 596
3-D computer aided design model 788
3-D point cloud data 1189
3-D printer 539
3-D query point 749
3-D reconstruction 626
3-tiered (3T) 337
6 DOF localization 987

A

a three layer architecture for navigating through intricate situations (ATLANTIS) 280
AAAI Robot Challenge 288
abstraction layer 1116
acceleration-resolved control 197, 207
accelerometer 1014
access and economics 1577
accuracy analysis 431
ACM-R snake robots 464
Acorn RISC machine architecture (ARM) 577
acoustic
– baseline 1212, 1216
– communication 577
– imaging 574
– modem 576, 578
– positioning system 575
– sensor 574
acoustic Doppler current profiler (ADCP) 574
Acrobot surgical robot 1532, 1533, 1536
acrylonitrile–butadiene–styrene 567
activation behavior-based 305, 307–309
activities for daily living 1552, 1718, 1781
activity theory 1754
actuating system 1213
actuation 1010
– architecture 451
– net 452
actuator 539
– dynamics 605
– effort sensor 677
– electromagnetic 78
– for manipulation 647
– hydraulic 80
– placement 449
– pneumatic 80
– vibrotactile 1014
– with mechanically adjustable series compliance (AMASC) 487
AdaBoost 785
added mass 1207
additive manufacturing 539
adjacency pair 1752
adjustable pattern generator (APG) 1896
admissible configuration 855
admittance 1007
– control 193, 199, 200
– matrix 861
adversarial environments 1282
aerial
– based precision agriculture 1382
– robot 623, 625
– robot control 1234
– robot guidance 1234
– robotic emerging application 625
– robotics 593, 1225
– vehicle 1485
aerodynamic
– center (AC) 607
– drag 613
– efficiency 606
– flight mechanics 595
– force 597, 606
– mechanism 617
– moment 606
Aesop surgical robot 1530
affordance
– detection 971
– learning 967
agent remote 281
agile
– manufacturing 1301
– robot development network (aRDnet) 292
agricultural robotics 1365
agriculture 626, 1366
agriculture and forestry (A&F) 1365
aircraft dynamics 603
airfoil 596
airplane 606
– static performance 606
– static stability 606
algorithmic singularity 229
altimeter 576
ambient assisted living (AAL) 1781
amplitude shift keying (ASK) 578
analytic Jacobian 219
anatomically correct testbed (ACT) 1649
android 1745
– robot 1767
angle of attack (AOA) 606, 1230
– matrix 1230
angular momentum 1127, 1143
animacy 1742
animal detection 1513
annealed particle filter (APF) 988
anthropomorphism 448, 1743, 1766
antipersonnel 647, 1413
anti-tank mine 1414
anti-vehicle 1414
arable farming 1369
arbitration 279
arc length 468
architecture 279, 300–302, 1255
– behavior-based 280, 310, 311
– component 282
– design 290
– implementation 291
– layered control 280
– subsumption 280
– three-layer 301, 302
Arduino 1939
area correlation stereo 744
arm guide 1561
articulated body 54
articulated object 791, 993
articulated robot 555
articulate-type modular robot 515
artifact 717

artificial
– bacterial flagella (ABF) 650
– constraint 205
– evolution 1871, 1879
– hand 1851
– intelligence (AI) 2, 8, 282, 1064, 1494, 1646, 1952
– intelligence system (AIS) 282
– muscle 486
– muscle system 482
– neural network (ANN) 789, 1823, 1872
artificial intelligence (AI) 319, 1768
ask-back 1776
assembled state 860
assembly 847
– incidence matrix (AIM) 514
– line 1306
– motion 859
– process 1309
assistive
– interaction dynamic 1804
– rehabilitation robot 1568
– robot 1567
– robot service manipulator (ARM) 1569
– technology 1006
atmosphere 596
attention
– focus 1771
– monitoring behavior 1770
– shared 1786
attitude
– and heading reference system 573, 699
– observer 1246
attitude control
– free-floating robot 1334, 1347
– reaction wheel 1347
attractor field 1673
augmentation 1575
augmented Jacobian 229
autism 1747, 1933
automated
– brick alignment 1397
– brick laying 1395
– bus rapid transit (ABRT) 1520
– guided vehicle (AGV) 1303
– highway 1520
– highway system (AHS) 1502
– reasoning 320
– reinforcement 1397
– synthesis of multirobot task solutions through software reconfiguration (ASyMTRe) 1275
– vehicle 1518
– welding robot 1397
automatic
– constructions building system (ACBS) 1404
– speech recognition (ASR) 1905
– spoken-language recognition (ASR) 1775
– target recognition 1483, 1489
autonomous 620
– aquatic vehicle (AUV) 1036
– combat flying vehicle (ACFV) 1967
– digging 1439
– dozer 1439
– excavation 1438
– exploration 993
– flight 621
– flying vehicle (AFV) 1967
– guided vehicle (AGV) 403
– mental development (AMD) 1649
– orchard vehicle 1373
– robot architecture (AuRA) 280, 302
– robotics research 1618
– space transport robotic operations (ASTRO) 1335
– subtrack control 1198
– tramming 1445
– underwater vehicle (AUV) 565, 566, 1204, 1466, 1485, 1967
– vehicle 1417, 1427, 1445
autonomous system 1450
– in mining 1450
autopilot 610
auxiliary surgical support robots 1530
average landmark vector 1850
average-reward criterion 362
aware home 1612

B

baby schema 1743
back-channel 1752
– response 1776
back-drivability 415, 1004, 1007
backstepping control 258
backward Euler algorithm 183
bag-of-word 780
balancer 435
ballast 568
bang–bang-control 144
baseball playing robot 1603
basic level object class 779
BASIC stamp 1939
basis behavior 1269
basis of object representation 1915
basket cell 1895
batch estimation 95
bathymetric survey 1435
battery 569
Battlebots 1935
Baumgarte stabilization 60
Bayes rule 325, 820
Bayesian
– belief 983
– filter 985
– framework 983
– inference 985
– network 324
– prior 985
– probability 324
– recursion 985
– state estimation 983
Bayesian dynamic network 325
beam
– bending 260
– sonar pattern 712
– torsion 260
bearing estimation 722
bee and robot 1889
behavior 279
– adaptive 1873
– control 284
– homeostasis 1769
– language 284
– mark-up language (BML) 1752
– primitive (BP) 308
behavior-based 303
– activation 302, 313
– basis behavior 304
– coordination 303
– fuzzy system 310
– history of behavior use 308
– interaction dynamics 301, 303, 305–307
– learning 306
– misconception 303
– multi-robot system 305–307, 310
– planning 304
– principles 302

– reasoning 304
– reinforcement learning 306
– representation 304
– robotics 279
– system 299
– vision system 311
belief 1753
– roughness 985
– set 1751
– space 990
– space search tree 990
– state 988
– system 1751
Bellman
– equation 326
– principle of optimality 363
bend propagation 536
Bernoulli equation 597
best practice in robotics 293
best-first-planner 1657
bi-articular muscle 534
bigram model 1673
bilateral control 1024, 1335
bi-manu-track 1561
bin-picking 1307
bio-inspired
– actuation 1847
– climbing 1850
– robotics 1842
– soft robot 1851
biological
– leg 1847
– model 1844
– principle 525
– society 1267
– system 499, 1844
biomimetic 731
– gait 526
– robot 524
bionic handling assistant 467
biped robot 1846, 1860
blind bulldozing 1280
blind-spot detection 1513
Bode
– diagram 248
– plot 457
body extender 1726
body pose 785
body-affect mapping 1770
boosting 782
botball 1934
boundary layer 171, 597
bounding sphere 1011
box pushing 1280
brachistochrone OC problem 501
bracing strategy 260
brain imaging 1563
brain-computer interface 1563
brickwork
– component 1395
– plant 1396
– positioning 1397
broad particle 987
broadcast 283
– of local eligibility (BLE) 310, 1277
Brockett's theorem 1145, 1169
browser 1049
Brunovsky canonical form 169
brush tire model 558
brushless DC electric motor 78
buckling 533
– gripper 536
building component recycling 1404
bulldozing resistance 1353
bundle adjustment 770
buoyancy 567, 1209
– center of 1209
bus rapid transit 1520

C

cable-driven parallel robot 240, 428
Canadian Scientific Submersile Facility (CSSF) 1204
CAN-bus 514
capacitive pressure sensing array 678
capstan 1009, 1010
capture
– and berting 1334
– point (CP) 1134
car body painting 1328
carbon fiber 537, 1328
– reinforced plastic 567
– reinforced prepreg 536
carbon nanotube 485, 652
cardinal direction calculus 331
car-like robot 553
carrot heading 1194
Cartesian
– impedance control 494
– manipulator 69
– space stiffness matrix 397
– stiffness 495
– stiffness control 494
cascaded control
– architecture 611
– structure 495, 622
caster wheel 549, 550
caterpillar 534, 1436, 1446
ceiling
– panel placement robot 1400
– vision SLAM 1609
cellular robotic system 513
Center for Robot-Assisted Search and Rescue 1468
center of
– buoyancy 1209
– compliance 195
– gravity 606, 1195, 1206, 1650
– mass 267, 414, 419, 995, 1127, 1556, 1656
– pressure (COP) 1128
– rotation 893
– stiffness 195
center of rotation (COR) 883
central pattern generation 528, 1143
centrifugal clutch assembly 1311
centroid
– contact 1015
– moment pivot 414
ceramic matrix composite 485
cerebellar
– control 1895
– model 1898
– model articulation controller 370, 1896
cerebellum 1887
cerebral palsy 1552
chain
– kinematic topology 388
– reconfiguration 517
chained-form system 1106, 1163
charge-coupled detector 93
charge-coupled device 637, 674, 1376, 1867
Chasles' theorem 17
chattering 170
Chebychev–Grübler–Kutzbach formula 429
chemical spill 1460
chemical vapor deposition 639
Christoffel symbol 46
circuit switching 1046
CKbot 516
clamping 1683
class detection 786
classical mobility formula 429

classification of customers 1594
classifier 782
Claytronics project 1272
cleaning
– robot 1402, 1592
– technology 1595
client/server 278, 1047
cliff sensor 1607
climb factor 606
climbing fiber 1895
clinical
– testing 1562
– use 1564
closed
– chain 427
– world assumption 327
closed kinematic
– chain 145
– loop 57, 114
closed-loop 122
– dynamic 499
– execution and recovery 282
– inverse kinematics 226
cluster bomb 1414
clustering 101, 781
clutching 1029
co-activation stiffness 490
coarse-acquisition 701
C-obstacle 857
cockroach inspired robot 525
codebook 780
coefficient 597
– of restitution 1350
cognition 299
– embodied 1747
– social 1747
cognitive
– aid 1572
– development 1746
– human–robot interaction 1741
– model 1744
– model of robot control 1757
– system 1741
cognizant failure 285
collaboration and teamwork 1772
collaborative control 1749
collaborative interaction 1681
collective 1254, 1268
– robot construction 1280
– transport 1280
collision 1011
– avoidance 1046
– avoidance technology 1443
– detection 1010, 1046, 1694
– isolation 1694
– mitigation 1512
– modeling 1683
– reaction 1694
– warning 1512
color camera with depth 998
command shaping 265
commercial off-the-shelf 832
common
– gateway interface 1047
– ground 1749, 1786
– object request broker architecture 279, 1056
– objects in context (COCO) 791
communication 622, 1257
– access for land mobiles 1508
– for control 1264
– for perception 1264
– graph 1263
– intelligence 1488
– interface 518
– protocol 518
– skill 1775
– system 1214
– theory 1674
compass 1212
– gait walking 1846
competing behavior 285
competition 1617
– and cooperation 1876
– challenge 1619
– scenario 1619
complementarity condition 1130
complementarity problem 887
complementary
metal-oxide-semiconductor 674, 715, 1539, 1607
compliance
– active 195
– center of 195
– control 193, 252
– frame 193, 205
– in grasping 954
– matrix 195, 204
compliant
– contact 880
– environment 204
– fin 532
– grasping 956
– hand 955
– humanoid platform 1689
– material 531
compliant motion
– active 862
– execution 865
– planning 862
component
– modeling 1211
– production 1395
composite material 567
composite-rigid-body algorithm 39, 55
compression 260
– criterion 1685
computational
– adequacy 321
– fluid dynamics 602, 1211
– model 1901
– neuroethology 1869
computed tomography 1528, 1537
computed-torque 255
– control 170, 173
computer ethic 1957
computer graphics 1654
computer integrated construction 1400
computer numerical control 513, 1302, 1305, 1394
computer-aided
– design 90, 788, 984, 1006, 1307, 1318, 1392, 1527, 1671
– drafting 1053
– engineering 388
– manufacturing 1313, 1395, 1527
– remote driving 1339
– tomography 1965
computer-aided design (CAD)
– model 1012
computer-integrated surgery 1528
computional brain 416
concentric tube manipulators 467
conceptual design 408
concrete
– distribution 1398
– finishing 1400
– leveling robot 1402
– production 1395
– pumping car 1469
– spraying robot 1399
concurrent execution 278
– mapping and localization 1216
– reactive plan 337
condition number 124, 126, 221
conditional factor graph 965
conditional random field 99, 965
conditioning kinematic 1007
confidence measure 125

configuration 859
– space 71, 136, 858, 1127
– space manipulation 854
– space obstacle 857
– space topology 856
connection
– event 1752
– system 1405
connectivity graph 47
conservative congruence transformation 435
constant curvature model 471
constellation model 783
constrained
– modes of a joint 21
– motion 192
constraint
– (selection) matrix 885
– artificial 205
– holonomic 25
– Jacobian 202
– kinematic 192, 201
– natural 204
– nonholonomic 25
– satisfaction problem (CSP) 329
– task Jacobian 229
construction 1392
– automation 1389
– elementary process 1391
– industry 1390
– machine 1394
– on-site 1399
– phase 1391
– process 1391
– project 1390
– robot 1390
– robotics 1392
– robotics categories of 1394
– site 1403
– stakeholder 1391
contact 1015
– angle of track 1196
– centroid 1015
– compliant 880
– configuration 860
– display 1015
– dynamics 1349
– estimation 994
– formation 863
– interface 879, 889
– kinematics 880
– location sensor 678
– manifold 988
– mode 882
– model 879
– sensor 1607
– stiffness 895
– vibration 1014
– virtual manipulator 206
– wrench sum 1658
contact state
– graph 863
– identification 864
– principal contact 860
contextual inquiry 1943
continuous
– activity scheduling, planning, execution and replanning 282
– hidden Markov model 1700
– operating reference station 704
– path 1303
– time 1013
– variable transmission 485
continuous-transmission frequency modulation 709, 723
– application 725
– range discrimination and resolution 724
– sonar 723
– transmission coding 723
continuum
– Jacobian 471
– kinematics 470
– limbed robot 468
– manipulation 858
– manipulator 466, 858
– robot modeling 463
contract net protocol 1277
contraction mapping 252
control 957, 1214, 1446
– acceleration-resolved 195
– adaptive 173, 258
– admittance 193
– basis 853
– biomimetic 1143
– command interpreter 834
– compliance 193
– damping 193
– decentralized 163
– feedback 1158
– for perception 1265
– force 1007, 1146
– impedance 193
– in mobile manipulation 960
– interaction 192
– law 499
– motion 160, 1317
– multiobjective 852
– null-space projection 852
– of nonholonomic systems 1163
– operational space 850
– optimal 239, 1137
– path-tracking 1446
– position and force 851
– redundant manipulator 851
– robot hand 459
– robust 1146
– shared 1028
– stiffness 193
– supervisory 1027
– synthesis 1270
– system 143
– task 1828
– task-level 850
– unit 647
– volume analysis 596
controllability
– capturability 1133
– capture point 1133
– small-space 1142
controlled
– floor-by-floor deconstruction 1404
– oscillatory dynamics 498
controller
– area network 411, 576, 841
– optimization 1675
controlling robot 1188
convergence tracking error 176
convolution 789
– neural network 783, 789
cooperative 1254
– adaptive cruise control 1505
– intelligent real-time control architecture 282
– interaction 1681
– manipulation 1280
– manipulation control 869
– manipulation planning 870
– manipulator 933
– multiarm system 943
– multirobot observation of multiple moving target 1281
– task space 939
– vehicle infrastructure system 1508
coordinate
– absolute frame 939
– generalized 25
– measurement machine 982
– spatial transform 41

coordination 1258
– behavior-based 303
– decentralized 1260
Coriolis and centrifugal force 160
correspondence 788
cost
– function 497
– of transport 1720
– pressure 1594
Coulomb 1011
– friction 462, 1129
counter-rotating thruster 572
counterweight 435
coupling
– momentum 1348
– stiffness 195
– virtual 1011, 1012
coverage 1266, 1369
– configuration protocol 1268
– strategy 1602
CPG-based controller 1848
Cramér–Rao lower bound 1264
crane automation 1379
crank-rocker 529
credit assignment 1278
– problem 1897
crop
– production cycle 1366
– yield estimation 1372
cross-sensor acoustic interference 576
crowd-sourcing 790
crush depth 567
C-space 136
cue 1744
– nonverbal 1746
cultural model 1744
curvature 469
– velocity method 1112
curvilinear abscissa 1161
customer expectation 1595
cutaneous (tactile) 1004
cyber physical system 1304
cyclicity 232
cylindrical
– algebraic decomposition 150
– joint 22
– shaped pressure vessel 567

D

da Vinci surgical robot 1530, 1538–1540
damaged building 1482
damped least squares 125
damper 485
damping 1009
– control 193
– factor 224
– ratio 199
DARPA
– challenge 1631
– grand challenge 1619, 1633
– Robotics Challenge 1687
– Software for Distributed Robotics 1262
– Urban Challenge 1633
data
– association 101
– communication 1426
– distribution service (DDS) 283
– gathering 993
– processing problem 1493
datcom 1208
Davis–Putnam algorithm 323
DC motor 79, 163
deactivation and decommissioning 1421
dead reckoning 1450
decentralized data fusion 836
decision-theoretic model 1754
decontamination 1469
decoupled architecture 389
decoupled robot 434
decoupling control 254
dedicated short-range communication 1507, 1521
deep
– belief network 1905
– reactive ion etching 642, 649
– sea remotely operated vehicle 571
Defense Sciences Office 1573
deformable
– object 994
– part model 780
– surface 1188
– terrain 556
– tire 557
degree
– of freedom (DOF) 23, 39, 69, 115, 195, 218, 292, 387, 409, 427, 487, 513, 530, 582, 621, 642, 680, 699, 756, 844, 886, 905, 958, 982, 985, 1007, 1029, 1093, 1162, 1197, 1208, 1332, 1376, 1440, 1561, 1689, 1784
degree of
– freedom (DOF) 465, 621
– mobility 551
– steerability 552
degree of freedom (DOF) 161, 854, 1303, 1649, 1669, 1715
deictic gesture 1776
deliberative system 300
delicate object 995
delta robot 388, 428, 1306
demining 1415–1418
Dempster–Shafer theory 830
Denavit–Hartenberg 49, 109, 112, 388, 514, 926, 1312
– convention 26
– parameter 112, 388
Denning ring 727
dense wave division multiplex 577
deoxyribonucleic acid 473, 641, 1045
deployment 1267
deposition 539
description logic 321
design 436, 1006
– for assembly 1303, 1405
– philosophy 388
– robotic architecture 290
detecting human activity 1494
detection
– and tracking algorithm 1489
– vehicle 1510
deterministic method 985
dexterity 448
– global 395
– index 392
– local 392
difference of Gaussian 780
differential
– dynamic programming 365
– elastic actuator 312
– flatness 1107
– global positioning system 703, 1442, 1504
– phase shift keying 578
dig
– control 1446
– planning 1446
digital
– analog converter 577
– map 1507
– signal processor (DSP) 184, 578, 727, 746
– terrain mapping 1442
digital-to-analog 183, 1009

dilution of precision 702
dimensional synthesis 429
dimensionless space 394
dip-pen nanolithography 641
direct
– control 1024
– marker tracking 1676
– methanol fuel cell 569
– modeling 351
– teleoperation 1029
directional stability 607
directivity pattern 712
Dirichlet process 964
disability 1566
disaster
– characteristics 1463
– response 624, 1248
– robotics 1460
disaster robot
– task 1461
– type 1462
discontinuous contact 1013
discrete
– cosine transform 1490
– element method 1191
– Fourier transform 724
– hidden Markov model 1700
discriminative
– function 966
– learning 787
– model 782
dispatch problem 1437
displacement group 429
display
– contact 1015
– slip 1015
– surface 1016
– tactile 1014
– vibration 1006
distance
– from singularities 221
– transform 787
distributed 1254
– cognitive system 1749
– control 1263
– field robot architecture 832
– localization 1263
– optimization 1266
– robot architecture 1255
domestic robotics 1591
Doppler velocity log 571, 573, 574, 1212
double-acting actuator 452
drag
– coefficient 603
– force 1209
– representation 597
dragline automation 1442
drawbar pull 1192, 1353
drilling 1439
driver assistance 1511
drone 624, 1412
dry adhesive 530
dual-arm telerobot 1025
Dubins path 1243
dutch-roll mode 1238
dynamic 37
– Bayesian network 325
– Bayesian system 983
– climbing 531
– covariance scaling 1092
– feedback linearization 256
– gait 409
– manipulability 1008
– matrix 398
– model 607, 1190
– movement primitive 967, 1673
– multisensor system control 835
– neural field 1699
– planning 1702
– programming-based method 363
– singularity 1345
– sinkage 1353
– state machine 1879
– system identification 492
– unconstrained 1682
– window approach 1112, 1114
dynamic passive walker 1860
dynamically consistent
– generalized inverse Jacobian 852
– inverse of the Jacobian matrix 234
dynamics 436
– accuracy 62
– assessment 616
– canonical equation 38
– closed-loop 57
– forward 38, 54
– inverse 38, 51, 197, 208, 245, 265
– inverse control 167
– model 983, 985
– real-time implementation 62
– software 63
– symbolic simplification 63

E

early device 1562
earth-centred, earth-fixed 703
earthmoving automation 1403, 1438
echo waveform
– coding 719
– processing 721
eddy current
– damper 486
– metal detector 1415
education 1006
educational
– evaluation 1943
– robotics 1932
effector-space 348
efficiency 792
ego-motion 92
elastic
– band 868, 1118
– joint 416
– joint torque 1688
– roadmap 868, 963
– strip 868, 963
elastostatic performance 396
electric support measure 1488
electrical
– discharge machining 641
– master–slave manipulator 1419
– stimulation 1561
electroactive polymer 81, 485
electroadhesion 531
electrocardiogram 1517
electroencephalography 1056, 1574
electrolytic corrosion 568
electromotive force 79
electromyography 1561, 1573, 1671, 1718, 1893
electron microscope (EM) 636
electron-beam 638
– induced deposition 653
electronic speed controller 1235
electronics controller unit 1335
electrooptical 1485
electrorheological 486
electrostatic 715
– actuator 483
electrostrictive 484
element momentum theory 600
elementary operator 1823
elevation model 756

embedded
– computer 1259
– intelligence 1575
embodiment 1745, 1801
emotion theory 1769
emotional
– empathy 1769
– support 1774
emotion-based interaction 1768
enabling technology 1566, 1606
encoder 1008
end of arm 261
end-effector
– based therapy robot 1552
– motion 163
endoluminal
– robot 1539
– surgery 1539
energy
– based function 966
– bounding algorithm 1034
– density 570
– dissipation 485
– efficiency 481, 1847
– leak 1013
– packet 1035
– stability margin 1189
– stability margin (ESM) 420
engagement generator 1752
enhanced horizon control 1447
enterprise resource planning 1396
entertainment 1006
entropy 989
– minimization 990
envelope display 1775
environment
– compliant 204, 211
– cultural 1747
– disturbance 1207
– model 91
– physical 1747
– social 1747
– virtual 1004, 1013
epipolar constraint 768
epistemological adequacy 321
equation of motion 160
– impulse 47
– joint-space 45
– Lagrangian 46
– operational-space 45
– rigid body 43
– spatial 43
error
– convergence 186
– correction 861
escapement cam 532
essential matrix 768
estimated inertia 174
estimation 89
– linear model 95
– process 94
ethernet 1046
ethical issues 1577, 1782
ethnography 1944
Euler
– angle 13, 984, 1205
– equation 597
European Conference on Educational Robotics 1936
European Octopus 468
European robotic arm 1333
event calculus 324
evidential reasoning 830
evolution
– artificial 1871
– Hebbian learning 1869
– learning rule 1875–1877
– of learning 1862
– robotics software platform 580
evolutionary
– algorithm 1857, 1871, 1880
– biology 1846
– robotics 1851, 1862
evolvable hardware 1879
evolving morphologies for human–robot symbiotic interaction 1565
excavator 1421
execution
– monitoring 287
– support language 279
exoskeleton 1004
– based therapy robot 1553
– for human performance augmentation 1724
expandable polystyrene 568
expanding role 1203
expectation maximization 94, 751, 781, 1069, 1674, 1898
exploratory procedure 1005, 1006
explosive
– disposal 1025
– ordnance disposal 1411, 1412
– vapor detector 1416
extensible
– hyper text markup language 1056
– markup language 283, 1056, 1452, 1754
external force 936
external velocity manipulability ellipsoid 942
exteroception 89
extrapolated center of mass 1134
extrastriate body part area 1921
extravehicular activity 1027, 1332, 1718
extreme locomotion 524

F

fabrication method 538
facade
– operation 1402
– painting robot 1402
facial recognition 1495
factor
– damping 224
– of safety 566
Fagg–Arbib–Rizzolatti–Sakata model 1899
false
– belief task 1771
– range 710
family of integrated rapid response equipment 1492
fast
– forward 338
– Fourier transform 263, 724
– simultaneous localization and mapping 1086
fault detection 1218, 1256
feasibility of compliant motion 865
feasible
– minimum buffering time 1048
– velocity 220
feature
– coding 782
– creep 516
– extraction 90, 780
– frame 206
– selection 790, 1867, 1868
feedback
– control 1158
– control and planning 866
– dynamic linearization 256
– haptic 1004, 1540
– linearizability 1107

– linearization 168, 253
– motion planning 866, 962
– of motor variable 249
– planning 866
– requirement 849
– vibration 1014
– vibrotactile 1014
feedforward 792
– feedback control 256
– motion 211
fiber to the home 1047
fiber-optic
– communication 576
– gyro 572, 696
fiber-reinforced
– plastics 411
– prepreg 537
field
– communication 1407
– of view 1752
– robotics 1392, 1453
fielded unmanned aerial vehicle 1484
field-emission SEM 652
field-programmable gate array 578, 728, 746, 1358, 1652, 1879
filtering 821
fine motion planning 862
fine-grained categorization 790
finger 1017
– pad 1015
– tip 996
finite element 1012
– analysis 402
– method 567, 1191
finite-state acceptor 280
fireproof coating 1402
firewall 1046
first-order predicate logic 319
Fisher vector 783
fish-like swimming motion 531
fixed-point stabilization 1159
fixture
– and contact 895
– virtual 1011
flapper sizing 618
flapping
– dynamics 615
– flight 1850
– frequency 618
– insect wing 617
– robot 617
– wing 528, 617
– wing unmanned aerial system 593
flash LIDAR 741
flat hierarchy 1894
fleet management 1437
flexible
– based manipulator system 1342
– link transmission 451
– manufacturing 1390
– manufacturing system 1303
– part model 783
– robot 503
FlexPicker 388
flexure-based transmission system 529
flight
– control-unit 623
– path matrix 1230
– performance analysis 603
floating-base 24, 47
– system 1669
flocking 1279
flow
– feature 786
– property 596
fluid dynamic 596
focus of attention 1752
foliage penetration (FOPEN) 1488
food handling 1305
foot rotation indicator 1135
foraging 1279
force 954
– based programming 1703
– calculation 1010
– closure 856, 1129
– control 1007
– hybrid motion control 193, 207
– position control 208
– reflecting teleoperation 1336
– reflection 1025, 1420
– scaling 1030
– sensing resistor 677
– sensor 1689
– sensor (fs) 435, 1009
– sensor information flow 680
– shading 1011
– spatial 40
– torque sensor 193, 674
– velocity control 208
foreground–background segmentation 784
forestry robotics 1365
forgetting factor 184
form
– closure 882, 954
– of human 1831
formal method 1270
formation 1279
– contact 863
formative evaluation 1943
forward
– dynamics 1670
– instantaneous kinematics 31
– kinematics 27, 430, 1010
– looking infrared 1486
– model 348, 349
free
– floating manipulator 1343
– floating robot differential flatness 1347
– modes of joint 21
– space voxel 752
– vibration 398
Frénet frame 1161
frequency
– converter 571
– modulation continuous wave 741
– shift keying 578
friction 201, 1008
– cone 884
– Coulomb 884
– damper 486
– limit surface 892
– model 1011
– modulation 1016
– stir welding 1307
frictional
– ambiguity 887
– inconsistency 887
Froude
– model 595
– number 421
fuel cell 569
full Bayesian model 783
full-state feedback 252
function approximation 367
functional
– electric stimulation 1553, 1714
– limitation 1567
– magnetic resonance imaging 1563, 1743, 1922
– neural stimulation 1552, 1575
funnel 962
fused deposition modeling 539
fusing air vehicle 841
fusion primitive 308

fuzzy
– logic 829
– system behavior-based 310, 311

G

gait 472
– sensitivity norm 420
– training robot 1564
galvanic corrosion 568
gantry 1306
gareki model 1189
gastrointestinal endoscopy 1539
Gator tech smart house 1612
Gaussian
– minimum shift keying 578
– mixture model 783
– noise 984
– process 966
– process clasification 966
Gauss–Markov estimate 128
Gauss–Newton nonlinear estimation 110
gaze 1744
– directed 1752
– following 1752
– mutual 1752
gecko
– adhesive 534
– adhesive system 530
– inspired adhesive 534
gender 1743
gendered interface 517
general
– contact model 868
– packet radio service 1508
generalized
– force 1207
– grounding graph 1748
– Jacobian matrix 1344
– Jacobian matrix (GJM) 1343
– least-squares estimate 128
– principal component analysis 94
generalized-inertia ellipsoid 394
generative model 781
generator of modules 281
genetic algorithm 515, 1858
geographic information system 626, 1484, 1493
geometric
– intersection data 1508
– Jacobian 219
– synthesis 436
geometrical reproduction 449
geon structural description 1916
geostationary Earth orbit 1334
glass-fiber reinforced plastic 567
glide ratio 606
global
– asymptotic stability 166
– conditioning index 395
– localization 985
– motion plan 868
– motion planner 962
– regression 355
– task planning 1702
– uncertainty 985
global navigation satellite system 703, 1212, 1367, 1442, 1506
global positioning system 4, 90, 571, 592, 670, 694, 756, 835, 1084, 1180, 1211, 1226, 1264, 1367, 1394, 1420, 1436, 1466, 1485, 1503, 1602, 1961
goal as parallel programs 280
goal-contact relaxation 863
goal-frame set 390
golgi
– cell 1895
– tendon organ 1005, 1893
Gough–Stewart platform 400, 427
GPS intelligent buoys 575
GRAB algorithm 988
gradient estimator 365
gradient-based win or learn fast 1278
grail 807
grammar 791
grand challenge 1557, 1566
graphical
– model 99, 325
– processing unit 752, 1094, 1541
– user interface 831, 1214, 1653, 1939
– user interface (GUI) 1653
grapple fixture 1332
grasping 858, 901, 953, 1483
– affordance 967
– compliant grasp 910
– contact model 904
– controllable force and velocity 907
– Coulomb friction model 917
– defective grasp 908
– design considerations 910
– desirable properties 909
– dynamic equation 906
– dynamics and equilibrium 906
– example 920
– force closure 917
– form closure 913
– friction model 917
– frictional form closure 918
– grasp classification 908
– grasp matrix and hand Jacobian 903
– hyperstatic 909
– indeterminate 908
– matrix 936
– model 902
– of a tumbling target 1336
– planar force closure test 920
– planar simplification of contact models 906
– planning method 988
– quasistatic equation 907
– redundant 908
– restraint analysis 912
– rigid body velocity kinematics 902
– Salisbury hand 902
– sequence 453
– velocity kinematics 903
Grassmann geometry 433
gravity 1209
– cancellation 251
– compensation 249, 435
– force 160
greenhouse 1368
grid filter 985
grid-based model 105
gripper 1305
Gröbner bases 430
ground 1008
– based augmentation system 703
– conditions 1188
– control station 1492
– detection 1198
– fault monitoring device 571
– moving target indicator 1488
– penetrating radar 1415
– vehicle 841
– vehicle (GV) 1485
group behavior 1259
guaranteed recursive adaptive bounding 988
guarded motion 988
guided exploration 993
gyroscope 1337

H

Haar measure 149
Hamilton–Jacobi–Bellman 177
Hamilton–Jacobi–Isaac 178
hand
– arm system 1689
– exoskeleton 1717
– eye locomotion 1339
handheld standoff mine detection system 1415
handling 1305
– remote 1412, 1420
hands-on cooperative control 1529, 1536
handyboard 1938, 1939
haptic 1003
– device 1004, 1017
– feedback 1004, 1025, 1540
– interaction point 1009
– interface 1003, 1426
– loop 1004
– paddle 1009
– rendering 1004, 1017
– temperature display 1017
harmonic drive 83, 240, 1303
Harvard micro fly 537
haul truck 1436
haulage 1436
Hayati parameter 113
hazard avoidance 1341
head injury criterion 1685
head-mounted display 1425
health care monitoring 1576
health monitoring 1612
– sensor 576
HeartLander 1539
Heathkit Hero-1 1936
heavy payload manipulator 1400
helical joint 22
helicopter-type UAV 614
Hemisson robot 1937
Hertzian contact 890
hexamethyldisilazane 639
hexapod robot 1846, 1859, 1880
hidden Markov model 717, 966, 1672, 1693, 1824, 1905
hierarchical 1255
– agglomerative clustering 781
– attentive multiple models for execution and recognition 1771
– communication model 1675
– hidden Markov model 1674
– task network 287, 338
high data rate digital subscriber line 1047
high definition 573, 1427, 1606
high safety goal 1116
high safety wide region 1116
high tech automotive system 1505
high-performance computing 1054
high-precision seeding 1371
high-resolution radar 1488
high-resolution transmission electron microscope 652
high-speed chronometer 740
high-speed gripper 536
highway, automated 1520
hinge offset 612
histogram 786
– filter 985
– intersection kernel 782
– of oriented features 967
– of oriented gradient 780
holonomic
– constraint 202
– manipulator 1162
home
– automation 1576
– based rehabilitation 1576
– point 983
– surface 983
homogeneous 1268
– space 394
– transformation 16
homography 809
Hooke's law 1011, 1012
horizontal brickwork panel 1396
hot zone 1463
hotel load 570
Hough transform 784, 1307
HRI operating system 1753
Hubble space telescope 1355
human 1567
– arm-like manipulator 218
– body pose estimation 784
– competition 1620
– control 1004
– culture 1648
– entertainment 1648
– grasping 955
– hand 448
– interface 1491
– motion segmentation 1672
– operator 841, 1013
– out of the loop control 1474
– skeleton model 1669
– social response 1773
– strategy 1667
– subject experiment 1786
– surrogate 1648
human motion data 1674
human–computer interaction 1006, 1612, 1742, 1809, 1943
human-inspired robot competition 1621
human-in-the-loop 1023
human–machine
– cooperative system 1533
– interaction 1380
– interface 1483, 1492
humanoid 48, 1645
– abstract task specification 1654
– automated motion planning 1654
– bipedal locomotion 1650
– body part 1649
– carrying of objects 1652
– coarse whole-body motion 1653
– dynamic balancing 1655
– dynamically stable motion 1654
– expressive behavior 1659
– expressive morphology 1659
– extension for complex tasks 1657
– force/moment controller 1651
– history 1648
– human environment 1646
– human example 1646
– human interaction 1647
– human-like walking 1651
– immitation 1649
– localization of obstacles 1652
– locomotion 1650
– mechanics 1649
– morphological communication 1659
– motion capture system 1653
– motion using GUI 1653
– multiple contacts 1657
– navigation among obstacles 1652
– pleasing mirror 1646
– pushing of objects 1652
– realtime walking pattern 1650
– robot 481, 1668
– running 1651
– sensor 1649
– understanding intelligence 1646
– whole-body activity 1653
– whole-body control 1658
humanoid communication
– auditory scene analysis 1660
– developmental robotics 1661
– expression 1660

– interpretation of human expression 1660
– locus of attention 1659
– multimodal perception 1660
– physical interaction 1661
– posture 1660
– saccade 1659
– smooth pursuit 1660
– speech recognition 1660
– vergence 1660
humanoid motion
– multiple contacts 1658
– stability 1658
humanoid robot project 409, 1647
human–robot
– augmentation 1714
– collaboration 1314
– communication 1776
– interaction 310, 311, 1024, 1460, 1474, 1675, 1741, 1765, 1799, 1831
– ratio 1474
hybrid
– assistive limb 1726
– behavior-based architecture 311
– chain-lattice 516
– dynamics 1130
– force/motion control 193, 207
– fuel/battery 570
– position/force control 944
– robot 556
– system 301, 310
– system manipulation planning 854
Hybrid III dummy 1689
hydrodynamic 1207
– damping 1208
– modeling 1210
hyper-redundant
– mechanism 464
– robot 218
– structure 468

I

ICP algorithm 747
identifiability 122
identifier 996
ideomotor principle 1903
illumination scenario 1511
illustrator cue 1776
image
– acquisition 624
– gradient 786
– pyramid 782
image-based visual servo 797
– control 796
– cylindrical coordinates 801
– interaction matrix 798
– stability analysis 800
– stereo cameras 801
ImageNet 789
imitation 1901
iMobot 516
impact 47
– dynamics 1349
– scenario 1681
impedance 488
– control 193, 198, 239, 258, 946, 1335, 1351
– controller 495
– matching 1351
implicit shape model 784
importance
– density 827
– sampling 98, 985
improvised explosive device 1414, 1968
incremental
– evolution 1876
– teaching method 1829
independent
– component analysis 1895
– joint control 163
– likelihood pool 820
indexed time table 281
indirectly actuated state 491
industrial
– description 1390
– manipulator 584
– robotics 1301
– standard architecture 576
inertia 1207
– articulated-body 54
– estimation 995
– matrix 160
– navigation system 573
– operational-space 56
– spatial 42
inertial
– measurement unit 92, 412, 571, 670, 698, 756, 813, 1179, 1195, 1212, 1226, 1246, 1449, 1507, 1611, 1809
– navigation 1506
– navigation system 571, 705, 1443
– parameter estimation 117
– sensor 572
– space 1343
inference 320, 787, 985
– Bayesian 325
inferior olive 1895
inferotemporal cortex 1899, 1917
inferring belief 1771
infinite impulse response 1490
information
– and communication technology 1396, 1941
– ethics 1957
– filter 826
informational
– support 1772
information-gathering grasp 991
infrared 648, 676, 746, 835, 1485, 1593, 1938
– detector 1415
inherent instability 621
inherently cooperative 1278
initialization 787
injury measure 1685
inner loop
– control 170
innovation 626, 824
input-output-to-state stability 166
input-to-state stability 166
insect-inspired robot 1849
insertion task 861
in-site actuation 451
in-situ resource utilization 1452
instability 1012
instantaneous
– allocation 1276
– center of curvature 694
– center of rotation 551
– forward kinematics 31
– inverse kinematics 32
instrumental
– support 1772
– variable 131
instrumented logical sensor system 833
integral control action 1167
integrated
– chip 291
– circuit 641
– development environment 1940
– factory 1390
– proximity model 984
– robotized 1393
– robotized construction site 1389
– services digital network 1047

integrating planning and execution 288
integration 620, 1283
intellectual property right 1612
intelligent
– assisting device 1730
– autonomous system 1714
– multimode transit system 1520
– wheelchair system 1570
intentional 1254
intention-based model 1753
interaction 954
– agent 1753
– control 192
– framework 1750
– modality 1757
– rhythm 1746
– unit 1750
interaction dynamics
– behavior-based 305–307
interaction matrix 796
– approximation 798
– direct estimation 803
– image-based visual servo 798
– pose-based visual servo 804
interactive perception 970
interaural
– amplitude difference 722
– time difference 722
interest operator 780
interface 848
– contact 889
– definition language 283
– design 514
– for demonstration 1825
– haptic 1003, 1426
interference rejection 726
interferometric fiber-optic gyro 572
interferometric SAR 1488
interior finishing robot 1400
intermediate haptic interaction point 1011
internal
– dynamics 247
– force 937
– friction angle 1353
– velocity manipulation ellipsoid 943
International Conference on Advanced Robotics 3
international space station 1027, 1332
international submarine engineering 583
International Symposium of Robotics Research 3
International Symposium on Micro Mechatronics and Human Science 1936
internet 1049
– communication 1034
– communications engine 283, 580
– engineering task force 1508
– protocol 1046, 1508
interoperability 1392, 1405
interpenetration 1012
interphalangeal 1718
interprocess communication 279, 289, 579
intersection operation 429
interval
– algebra 328
– analysis 430
– calculus 829
– programming 580
intrinsic tactile 455
invariance 202, 783
inverse
– instantaneous kinematics 32
– model 348, 349
– reinforcement learning 1831
– socially assistive robotics 1488
inverse differential kinematics
– least-squares solution 222
– weighted damped least-squares solution 234
inverse dynamics 51, 197, 208, 245, 265
– control 167, 357
inverse kinematics 29, 430, 857, 1317
– algorithm 225
– computation 1669
ionic polymer-metal composite 485, 532
isomorphic configuration 516
isotropic configuration 221
Istituto Italiano di Tecnologia 1689
i-swarm project in Karlsruhe 1261
iterated extended Kalman filter 98
iterative closest point algorithm 94, 747, 1092, 1449
iterative linear quadratic regulator 502
iteratively reweighted least square 100

J

jackknife effect 1163
Jacobian 31, 47, 71, 162, 851
– analytic 219
– condition number 393
– dynamically-consistent inverse 47
– ellipsoid 393
– extended 228
– geometric 219
– loop 59
– matrix 113, 935, 1010
– task 219
– time-derivative 396
jamming 888
Japanese experiment module remote manipulator system 1334
jigsaw positioning system 1442
jogging companion 1603
Johnson Space Center 1776
joint
– action 1749, 1785
– activity 1753
– architecture for unmanned systems 283
– attention 1752
– coordinate frame 48
– directors of laboratories 831
– drive gain 121
– elasticity 241
– intention theory 1750
– model 48
– parameter 26
– persistent goal 1753
– probability distribution 820
– reference 1770
– torque 119, 1671
– torque feedback 252
– torque sensor 193, 254, 454
joint-space 1010
– control 161
– inertia matrix 39, 55, 395
– stiffness matrix 397
– trajectory 180
jumping microrobot 533
just noticeable difference 1005
just-in-time 1397

K

Kalman filter 97, 129, 625, 823, 1070, 1188, 1610
– unscented 1445

Karel the robot 1932
key-poses 1654
Khepera 1858, 1871, 1937
Kilobot 516
Kinect 1669
kinematic 427, 1010
– chain topology 388
– conditioning 1007
– constraint 192, 549
– contact 880
– coupling 1029
– dissimilarity 1036
– duality 889
– equation 71
– inverse 29, 430, 857, 1317
– loop 39, 48
– redundancy 217
– singularity 220
– structure 993
– tree 39, 51, 121, 784
kinematic algorithms
– inverse 225
kinematic calibration 111
– closed loop 113
– index 116
– open loop 113
– sensor index 116
kinesthetic 1003
– display 1014
kinetic energy 43
– of motor 242
– recovery system 485
kinodynamic planning 143
k-mean 781
k-means clustering 101
k-nearest neighbor 749
knowledge
– assertional 323
– base update 324
– representation 320
– representation (KR) 320
– terminological 323
Koryu robot 464
Kullback–Leibler divergence 990

L

Lagrange
– dynamics 160
– equation 243
– formulation 46
– multiplier 202, 1011
Lagrangian 241
– dynamics 1127
– formulation 515
laminar damper module 487
landmine 1415, 1417, 1418
LANdroid 1472
lane
– changing 1516
– keeping 1515
– tracking 1509
Laplacian of Gaussian 745
LaSalle's theorem 165, 250
laser
– distance sensor 1607
– imaging radar 1488
– measurement system 742
– radar 740
– ranging 1341
– scanner 92
laser-based SLAM 1610
latent
– Dirichlet allocation 781
– support vector machine 787
lateral
– geniculate nucleus 1917
– intraparietal sulcus 1906
– slip 1188
lattice reconfiguration 517
lawn mowing 1602
layer 789
layered
– architecture 1257
– molding 536
lead lanthanum zirconate titanate 484
lead zirconate titanate 484, 636
leader–follower 943
leading edge vortex 617
leaf belief 991
learning 301, 787, 964, 1278, 1802
– and adaptation 1702
– applied to ground robots 1486
– architecture 351
– behavior-based 302
– by demonstration 308
– compound action 1828
– control 184
– control for assembly 861
– feature 967
– from demonstration 1824
– from history 308
– from human demonstration 1821
– in mobile manipulation 968
– individual motion 1827
– reinforced 306
learning from humans 1831
– algorithm 1827
– correspondence problem 1825
– evaluation metric 1824
– history 1822
– imitate 1824
– key issues 1824
least squares estimation 110
leg–arm hybrid robot 419
legged crawling 525
legged locomotion 959
– bio-inspired 1846
– model 1844
– stability and gait 1846
legged robot 1471
leg–wheel hybrid robot 418
Leonardo Da Vinci 1205
level of biomimicry 525
Lewellyn's absolute stability criterion 1013
LIDAR sensor 1198
Lie algebra rank condition 1104
Lie-group-algebraic method 429
life-cycle-costing 1304
life-like robot 525
lift force 1209
lifting line method 599
lift-to-drag 1485
light detection and ranging 670, 740, 1189, 1443
light-emitting diode 741, 1521, 1607, 1939
lighter-than-air system 593
light-weight
– communications and marshalling 579
– manipulator 240
– robot 240, 481, 1684
– structure 1687
likelihood function 820
Likert 1944
limb prosthetic device 1574
limbed system 408
limit cycle 1132, 1147
– walking 413
linear
– complementarity problem 1131
– constraint satisfaction program 887
– control design 256
– elasticity 241
– impedance 493

– program 915, 1146
– quadratic Gaussian 807
– quadratic regulator 271, 496, 611, 1140
– temporal logic 328
– variable differential transformer 117
linear inverted pendulum model 414
linearization feedback 168, 253
linearizing coordinates 254
line-of-sight 1492
link 1939
– equation 243
– parameter 26
linkage model
– virtual 945
liquid-crystal display 289, 1768
lithium
– polymer 624
– primary battery 569
livestock
– breeding 1380
– exploitation 1381
– harvesting 1381
– nurturing 1380
Llewellyn criterion 1032
load
– capacity 68
– distribution 941
– haul-dump 1435
– parameter 119
– sharing coefficient 941
local
– and small-time controllability 1103
– area network 514, 577, 1046, 1329, 1492
– autonomy 1027
– learning method 356
– minimum 866
– potential function 867
– product-of-exponential 514
– regression 356
– steering method 1105
locality constrained linear coding 782
localization 1216, 1265, 1445
– map-based 1445
– sound source 311
locomotion 470
– mechanism 1598
– performance 526
logic
– based reasoning 321
– description 321
– first order predicate 321
– propositional 322
logical sensor system 833
logistic regression 966
long-baseline 575
– system 1216
longitudinal stability margin 419
long-range cruising AUV 569
long-term
– behavior 1805
– depression 1896
– interaction 1766, 1779
– social interaction 1805
lookahead 991
loop
– closure 430
– closure constraint 58
– haptic 1004
loss function 788
low
– Earth orbit 1334
– level control 1702
– power consumption 1687
– uncertainty 985
low-cost sensor 1594
lower
– extremity nonanthropomorphic robot 1566
– extremity powered exoskeleton 1565
– pair 21
Lunokhod 1336
Lyapunov function 249–251

M

Mach model 595
machine
– interface 1491
– learning 1823
– safety standard 1314
– tool 439
– vision 1446, 1867
macro arm 1342
macro-micro manipulator 1334
magazining 1395
magnetic resonance imaging 1528, 1537, 1685
magnetoencephalography 1563
magnetorheological 486, 1426
magnetostrictive 484
Mahalanobis distance 753, 984
maintenance robot 1402
manage attention 1757
maneuverability 606
manifold
– learning 791
– particle filter 988
manipulability 1008, 1345
– dynamic 1008
– measure 221
manipulation 449, 470, 680, 847, 848, 953, 981, 1217
– admissible configuration 855
– cooperative 1280
– grasp configuration 856
– grasp planning 869
– inverse kinematics 857
– multiple part 870
– multiple robot 870
– nonprehensile 870
– pick-and-place task 854
– planning 145
– stable part configuration 856
– task specification 856
– three DOF manipulator example 856
– transit path 855
– workspace goal 858
manipulation aid 1568
– mobile autonomous system 1569
– wheelchair manipulator 1569
manipulator
– constraint 865
– cooperative 933
– Jacobian 495
– stiffness 396
manmade
– disaster 1463
– event 1467
man-packable UAV 1465
manual
– control 1024
– flight 621
ManuBuild 1392
manufacturing 1006, 1302
– and positioning 1398
– process 1302
map
– digital 1507
– geometric 1067
– grid 1067

mapping 756, 1216, 1608
– moving object 970
– passive 175
marine robotics 1203
Markov
– decision process 324, 361, 994, 1754
– decision process (MDP) 1270
– process 983
– random field 772
Marr–Albus model 1895
Mars environmental survey 1340
Mars exploration rover 1341
Mars Pathfinder 1340
Mars rover sample return 1339
Mary
– phase shift keying 578
– quadrature amplitude modulation 578
masonry wall erecting robot 1399
mass modal 263
Massachusetts Institute of Technology 1126
mass-spring system 994
master–slave
– manipulator 1419, 1425
– system 1024
material handling robotics 1403
maximally stable extremal region 781
maximum a posteriori 96, 785, 1491
maximum likelihood estimate 95, 721, 1265
max-pooling 783
M-blocks 516
McKibben muscle 1898
mean time between failures 85, 1304
mean-shift 784
measurement
– and signatures intelligence 1488
– model 983, 984
mechanical 488
– adjustable compliance and controllable equilibrium position actuator 487
– impedance adjuster 487
– interface 517
– motion capture 1668
mechanism 1776
– isotropy 1008
– passive 1424
– synthesis 429
– toothless gear 532
– used for HRI 1783
mechanization 1390
mechanoreception 674, 1016
mechatronic robot 1687
medial intraparietal sulcus 1906
medial temporal area 1921
medical
– image segmentation 1534
– robot 998, 1528
– simulation 1006
memory 1757
– alloy 484
mental inference 1773
mental model 1741, 1757
– robot 1742
mental perspective-taking 1771
mesencephalic locomotor 528
mesencephalic locomotor region 528
M-estimator 100
metal
– corrosion 567
– matrix composite 485, 567
metrics 1275
micro
– aerial vehicle 536
– arm 1342
– robot 1850
microcomplex 1895
microelectromechanical system (MEMS) 81, 265, 404, 483, 533, 572, 632, 675, 696, 716, 814, 1016, 1226, 1274, 1449, 1539
microfabrication 638
micromechanical flying insect 537
micromouse competition 1618
microrobot technology 525
Microsoft robotics developers studio 580
microspine array 530
microsurgery robot 1538
microsystem technology 647
microzone 1895
middle-size league 1624
middleware 283, 579
– for robot 580
millibot 518
millimeter/centimeter scale crawler 526
mindreading 1770, 1772
mine crawler robot 1482
mine-permissible robot 1482
minima-free potential function 867
minimally invasive surgery 1531
minimum
– description length 784
– mean-square error 96
– shift keying 578
mining 1433, 1447, 1452
– coal 1447
– disaster 1467
– dragline 1441
– ocean floor 1452
– open-pit 1441
– robotics 1433
– shovel 1440
– stages of 1434
– surface 1434
– system 1450
– underground 1434
mirror
– image motion enabler 1556
– image movement enhancer 1561
– neuron 1771, 1899
– neuron system 1900
– system 1771, 1887, 1903
misconception
– behavior-based 303
mission 625
– control system 1214
– oriented operating suite 580
mixed model 350
mixture model 787
mobile
– base system 1333
– beacon 1462
– detection assessment and response system 1491
– domestic robotic 1592
– manipulation 951
– manipulator 959
– navigation 1486
– platform 958
– repeater 1472
mobile robot 1253, 1310
– competition 1460
– nonholonomic 1159
– telerobotics 1036
mobility 58, 548
– and manipulation 852
– degree of 551
– equation 116
– restriction 552
mobility aid
– exoskeleton 1571

– walking assistance system 1571
– wheelchair navigation system 1570
modal
– mass 263
– stiffness 264
– vectors 398
mode 986
– contact 882
– identification and recovery 281
– shape 470
model
– based controller 471
– based method 346
– behavioral 1747
– checking 323
– cognitive 1747, 1749
– contact 879
– developmental 1747
– dialog-based 1749
– driven engineering 293
– environmental 91
– learning 1756
– of emotion 1768
– physical 1749
– predictive control 349, 622, 1137, 1236
– property 1210
– reference adaptive control 176, 349
– uncertainty 369, 1192
model learning 348
– architecture 351
– method 354
modeling 606, 1205
– actuator 487
– biological system 1844
– deformable terrain 1188
– first-order form closure 914
– of locomotion 472
– rubble 1189
– soft robot 490
– tracked vehicle 1195
modular
– block 1395
– manipulator 513
– mining system 1436
– robot configuration 514
modularity 1395
modulation friction 1016
modulation-based range sensor 741
molecular biology 1006
molten carbonate fuel cell 569
moment 606
– labeling 885
– representation 597
momentum
– conservation 596, 1344
– spatial 41, 42
– theory 600
monitor 288
monitoring system 1576
monolithic fabrication 538
Monte Carlo method 364, 986
moon buggy 1336
Moore–Penrose inverse 169
mossy fiber 1895
motion
– capture 1669
– category 1673
– constraint 849
– effect 728
– generation 960
– human-like style 1667
– instability 621
– primitive 1673
– requirement 961
– transmission 449
– whole-body 1144
motion control 160, 1157, 1317
– performance 1188
motion execution
– compliant 865
motion planning 470, 961, 1141
– compliant 862
– feedback 866
– feedback motion planning 866
– problem 434
motion-oriented operating system 1214
motivated behavioral architecture 311
motor
– equation 243
– evoked potential 1904
– feedback 249
– neuron 1671
mouse 1004
move value estimation for robot teams 310
movement therapy 1558
moving observation
– of a corner 729
– of a plane 729
– of an edge 730
moving plate 400
moving plate (MP) 389
moving target indicator 1487
multiappendage robotic system 403
multiarm system 943
multicellular organism 1843
multifingered manipulation 938
multifunctional satellite augmentation system 703
multihypothesis tracking 1511
multijoint
– bending 531
– model 164
multilateral telerobotics 1037
multilevel surface map 757
multimode transit 1520
multipass effect of wheels 1188
multiphalanx hand exoskeleton 1717
multiple
– beam scanning LIDAR 741
– input–multiple-output 173, 254, 610, 1237
– kernel learning 783
– layered composite 538
– master multiple-slave 1037
– master single-slave 1037
– material 536
– model switching adaptive estimator 271
– operator multiple robot 1044
– operator single robot 1044
– paired forward-inverse model 1897
– reflection 717
– resource host architecture 1492
– underwater vehicle 1219
– view registration 753
multipulse sonar 726
multipurpose system 1843
multiresolution mapping 970
multirobot 1254, 1281
– coordination 281
– path planning 1281
– system 304, 514
– system behavior-based 305–307
– task 1276
– task allocation 1276
multisensor
– data fusion 819
– environment modeling 835
– fusion architecture 831
– gripper 1334
– system control dynamic 835
multistep planning 990
multitarget observation 1281

multitask 1276
multitracked vehicle 1197
muscle
– length 1671
– like module 485
– spindle 1005
– tension 1671
musculoskeletal model 1670, 1671
musculoskeletal walking model 1720
mutual belief 1786

N

naive Bayes 781
nanoelectromechanical system 633
nanorobotic manipulator 651
NASA/NBS standard reference model 281
national livestock identification scheme 1380
national qualifying event 1633
national robotics initiative 1567
nationwide different GPS system 704
natural
– constraint 205
– disaster 1463
– frequency 199, 260, 398
– human cue 1757
– machine motion initiative 503
– muscle 482
– orifice transluminal surgery 1531
navigation 756, 1445, 1849
– and obstacle avoidance help 1570
– function 867
– in rat and robot 1890
– underground 1445
nearness diagram navigation 1115
nearness diagram navigation (ND) 1112
needle placement robot 1537
negative information 985
negotiation 1276
Nerd Herd 1255
Nereus 576
network
– data distribution service 283
– mobility 1508
– partitioning 1267
– real-time kinematic 704
– server 1048
– topology control 1267
network data distribution service 283
networked 1253
– infomechanical systems 1265
– mobile robot 1258
– robot 1258, 1259, 1472
neural interface 1574
neural network 789, 1858, 1879
– GasNet 1861, 1872
– neural oscillator 1860
– recurrent neural network 1866, 1867
– spiking neural network 1871
neurobiological system 1888
neurocontrol 1894
neuroethology 1887
neuromorphic engineering 1849
neuron 789
neuroprosthetics 1852
neurorobotics 1888
neutral buoyancy 1335
Newton–Euler
– equation 44, 117
– formulation 515
Newton–Raphson 431
– algorithm 431
N-gram model 1673
nickel metal hydride battery 412
NIST response robot evaluation exercises 1477
nonaccidental property 1916
nonacoustic sensor 572
noncontact therapy robot 1553
nonholonomic 1344
– constraint 1102
– mobile robot 1159
– planning 143
nonholonomy 1128
nonlinear
– dynamic inversion 611
– feedback 254
– force function 488
– mechanical system 488
– optimal control 177
– optimization convergence 1347
– programming problem 1346
nonmaxima suppression 787
nonminimum-phase 248
nonnegative matrix factorization 1895
nonneural element 1844
nonparametric method 985
nonprehensile manipulation 879, 896
non-RT thread (real-time thread) 581
nonslip condition 549
nontangential proper part 331
nonuniform rational B-spline 1012
nonverbal
– communication 1775
– cue 1752
norm
– communication 1744
– social 1744
normal distributions transform 970
normal vector 1012
normalized energy stability margin (NESM) 1197
normalized ESM 1189
nuclear
– decommissioning 1421
– magnetic resonance 1965
– operation 1025
– radiation 1412, 1421
null-space
– projection 851, 852
– velocity 221
numerical control 1313
nursery and greenhouse (N&G) 1377
nurturing 1380
Nyquist criteria 1013

O

Oberon 1212
object
– anchoring 321
– class detection 786
– class recognition 779
– detection 786
– identity resolution 339
– learning 1919
– localization 982
– part 791
– recognition 1917
– reorientation 991
– representation 1915, 1921
– request broker 283
– shape matching 996
– transportation 1280
objectness 789
Oblix/Mogura mechanism 464
observability index 111, 126

observation
– class ROV 571
– update 821
obstacle 758
– region 136
obstacle avoidance 470
– path modification 867
– problem 1111
obstacle restriction method 1113
obstacle restriction method (ORM) 1111
occupancy grid map 993
occupancy map 970
oceanic engineering 1203
octarm manipulator 467
octopod 439
octopus-arm 466
odometry 756, 1337, 1450
offline programming 1308
off-policy method 362
off-read locomotion 1338
off-road vehicle 1190
offsite robotics 1395
olive 1895
omnidirectional
– camera 92
– robot with steerable wheel 555
omnimobile robot 554
on board unit 1508
on real-time communication 1471
onboard autonomous science investigation system 282
onboard integration 623
onboard power source 569
one-step lookahead 989
one-way bearing 533
online
– gravity compensation 251
– programming 1317
onsite robotics 1399
on-the-fly target classification 726
ontological category 1744
ontology 323
– based unified robot knowledge 334
open
– agent architecture 1754
– dynamics engine 580
– platform for robotic service 580
– roboethics initiative 1972
open robot
– control architecture 580
– control software 292, 580
– controller computer aided design 279, 282
operating
– point 392
– system 1046
operational
– frame 850
– point 850
– therapy robot 1552
operational space trajectory 180
operational-space 45
– control 162, 358
– inertia matrix 39, 56
operator
– control unit 284
– model 348
optic flow 1871, 1889
optical
– coherence tomography 1530
– microscope 637
– microscope (OM) 652
– motion capture 1668
– quadrature encoder 1008
– underwater communication 579
optimal
– arbitrary time-delay 266
– control 500
– coverage path planning 1369
– design 436
– task assignment 1437
optimization 1008
optokinetic response 1659
orbital replacement unit 1333, 1335
orbital replacement unit (ORU) 1333
orchard 1367
ordinary differential
– equation 488, 1137, 1208
– inclusion 1128
orientation 13
– absolute 939
– quaternion 1198
– relative 939
oriented gradient 786
ornithopter 617
orthopaedic surgical robot 1536
Oswald
– coefficient 1231
– efficiency 599
out of field 1752
outer
– loop control 168
– poles 170
overfitting 789
over-the-horizon 1492
Oxford intelligent machine 1555

P

packaging line 1305
packet switching 1046
paddle
– haptic 1009
painting 1312, 1402
– robot 1312
palm tree spraying 1374
pan–tilt unit 289, 1440
pantograph 409
parallel
– fiber 1895
– force/position control 193, 211
– kinematic machine 400, 1303
– manipulator 427
– manipulator calibration 115
– mechanism 12
– robot 73, 122, 427, 1303
– tracking and mapping 1092
parameter
– drift 177
– estimate 173
– inertial estimation 117
parameterization of track 1195
parametric
– bias 1903
– design 430
– force 607
– model 607
parking assistance 1515
part-based model 780
partially observable Markov decision process 326, 963, 989, 1756, 1829
partial-order planning 338
particle 986
– filter 988, 1610
PASCAL visual object class 789
passenger protection 1518
passive
– action recognition 1257
– compliance 860
– dynamic walker 1860
– dynamic walking 412, 1126, 1147
– gripper 534
– mapping 175
– mechanism 1424

– set-position modulation 1034
– stiffness 435
– suspension dynamics 559
passivity 175, 210, 248, 1013, 1031
– based control 169
– controller 1034
– observer 1034
patchwork of primitives 992, 993
path 179
– consistency 329
– deformation 1118
– following 1159, 1162, 1193
– following algorithm 1194
– planning 179
Pathfinder 390
pattern generator 1848
pattern-based mixed-initiative 1750
payload 620
– delivery 624, 1249
– sensor 1486
pedestrian detection 1511
peg-in-hole 205
– problem 888
– task 848
PEIS Ecology project 334
people detection 784
perception 89, 299, 311, 969, 1262
– active 1266
– for action control theory 1904
– for off-road robotics 1486
– process 90
– via-manipulation 998
periodic motion
– control 497
– tracking 497
peripheral 1302
– component interconnect 576, 644
peristaltic waves 473
persistency of excitation 175
personal roving presence 1045
perspective taking 1750, 1772
– spatial 1752
– visual 1752
Petri net transducer 280
Phantom haptic device 1007
phase shift keying 578
Phoenix 1208
phosphoric acid fuel cell 569
physical
– damper 487
– embodiment 1773
– human–robot interaction 1642, 1679
– interaction 481
– vapor deposition 639
physically collocation 240
piano mover's problem 136
pick-and-place manipulation 854
pictorial structure 784
piecewise polynomial 180
piezoelectric 715, 1016
– actuator 484
pin in hole 1339
pinhole camera model 93
pipe clamp 887
place cell 1869, 1870
placing object 953
plan
– execution interchange language 279
– partial-order 338
planar joint 24
plan-based control 335
planetary exploration 1452
planner 288
planning 287, 300–302
– and control 847, 962
– behavior-based 304, 305, 307, 311
– closed kinematic chain 870
– configuration space obstacle 857
– feedback 866
– for hybrid system 854
– horizon 991
– manipulation 854
– movable obstacles 870
– nonholonomic 143
– self-collision 857
– trajectory 144
– under uncertainty 862
plant probing 1375
pleasure arousal dominance 1768
Plücker coordinates 20, 40
plug-and-play 512
pneumatic 483
– actuator 483
– artificial muscle 483
– continuum manipulator 467
– McKibben muscle 467
– network 539
Poincaré map 1126, 1147
point
– algebra 328
– contact 890
– estimation 95
– measurement 113
– of interest 1307, 1697
point cloud 993, 1196
– library 1092
point feature 749
– histogram 750
point-contact 904
pointing
– declarative 1752
– imperative 1752
point-to-point (PTP) 283
pole placement 255
policy gradient
– reinforcement learning 1810
– theorem 365
policy iteration 326
polygonal mesh 984, 992, 1011
polymer 484
– electrolyte fuel cell 569
– matrix composite 485
– MEMS 534
polymeric actuator 485
poly-mesh 992
polynomial trajectory 180
polyoxymethylene 567
polypod
– robot 464
– system 1271
polytope 396
polyvinyl chloride 567, 1937
polyvinylidene fluoride 484, 636, 675, 715
Pontryagin's minimum principle 500
pool cleaning
– robot 1600
pop-up book
– inspired design 538
– MEMS 538
port-based approach 1035
pose
– estimation 754
– normalization 790
– tracking 985
pose-based visual servo 803
– control 796
– interaction matrix 804
– stability analysis 804
poselet 784
position
– absolute 939
– control of subtrack 1199
– force architecture 1031
– localization 1506
– observer 1247
– relative 939

– sensing device 678
– velocity 164
positional dilution of precision 702
positioning accuracy 1308
position–position architecture 1030
position-sensitive-device 1607
positive photoresist 639
positron emission tomography 1528, 1965
possible
– failure 1218
– injury 1685
posterior
– distribution 983
– inferotemporal cortex 1918
pot handling 1377
potential
– damping 1208
– field 1890
– field method 1112
– function 285, 866
power
– consumption 1406
– data grapple fixture 1333
– loading 594
– scaling 1030
– source 647
– supply 620
– system 569
power-law equation 890, 895
power-to-weight ratio 1009
precise positioning system 700
precision
– forestry 1378
– irrigation 1372
predator and prey 1876
prediction step 821
predictive graphic display 1334
prefabrication 1389
prefrontal cortex 1899, 1917
preimage 862
prespective-n-point 765
pressure
– distribution 891
– hull 566
– sensor 573
– sinkage equation 1190
– vessel 566
pretectum 1890
principal
– behavior-based 302
– component analysis 749, 1694, 1895
– contact 860
– sonar 710
prioritized task behavior 851
prismatic joint 22, 76, 993
probabilistic
– graphical model 787
– grid 821
– latent semantic analysis 781, 998
– method 819
– roadmap 328, 961
– roadmap method 138, 1105
– roadmap method (PRM) 138
probability distribution 820
procedural reasoning system 281
procrustes 767
product modularity 512
production monitoring 1437
professional vocational assistive robot 1568
programmable
– construction machine 1394
– intelligent computer 1871
– logic controller 1302, 1308
– logic device 1879
– universal machine for assembly 1306, 1438
programming
– architecture 277
– by demonstration 1821
– environment 277
– language 287
– tool 277
projected gradient method 227
projection matrix 204, 207, 743
projective space 149
propeller
– force 613
– mechanical design 612
proportional–derivative 162, 194, 248, 350, 417, 933, 1031, 1143, 1235
– control 162, 248
proportional–integral–derivative 9, 159, 257, 284, 411, 643, 865, 1199, 1374, 1896
– control 159, 257
– gain tuning 166
– linear tuning 166
– square tuning 166
propositional integral 163
proprioception 89
propulsion 610
– load 570
prosthetics 1006
– and orthotic 1554
protected cultivation system 1368
proton exchange fuel cell 569
providing informational 1772
proxemic behavior 1745
proximity
– awareness technology 1443
– detection technology 1443
– measurement model 984
– sensor 1607
pseudo-amplitude scan 719
pseudo-amplitude scan (PAS) 720, 721
pseudo-inertia matrix 162
pseudoinverse 202
– Jacobian matrix 222
pseudolites 1442
pseudo-random noise 704
psychophysics 1005
pulleys 1009
pulse-width modulation 1235, 1374
Puma robot 389
puncture 1015
pure rolling condition 549
Purkinje cell 1895
pushing manipulation 889
push-up approach 1404
pyramid matching 782
pyramidal effect 397

Q

QR decomposition 123, 1090
quadratic
– optimal control 177
– programming 1146, 1655
quadrature
– amplitude modulation 578
– phase shift keying 578
quadro-copter 613
quality
– of life technology 1781
– of product 1594
– of service 292, 1046
– transmission 393
quantization 1013
quarter car model 559
quasistatic
– assumption 887
– constrained 1682
– motion 1126, 1135
– telerobotics 1049

quasi-two-dimensional laminate 537
quasi-zenith satellite system 700
quaternion 15, 149, 1206
quest for curiosity 403, 1652

R

radar 1444
radial
– basis function 1917
– basis function network 1824
– distortion 764
radiation contamination 1476
radiation-induced force 1207
radio
– communication 577
– control 1016, 1935
– frequency 92, 1047, 1487
– frequency identification 92, 1045, 1445, 1462, 1506, 1612
radiological survey 1469
random
– access memory 1609, 1938
– access memory system 1540
– forest 782
– loop generator 145
– measure 827
– motion pattern 1593
– sample consensus 94, 748
– walk 1593
randomization 790
randomized potential field 142
range
– data 737
– gated intensity 741
– image 738
– information 746
– scanner 738
– sensing 739
ranging module 718
ranging sonar 576
RANSAC algorithm 748
rapid prototyping 539
rapidly adapting lane position handler 1512
rapidly exploring dense tree 139
rapidly exploring random tree 139, 961, 1105, 1654
rate gyro 695
ratio damping 199
reaction
– null-space 1345, 1347
– time 1904
reactionless manipulation 1345
reactive
– action package 280
– obstacle avoidance 868
– system 300
reactive-ion etching 639
reactor building 1469
real-time 278, 581, 1012
– control system 281, 833
– innovation 283
– kinematics 704, 1180, 1370, 1442
– space link 1335
– system 282
– thread 581
– toolkit 292
– tracking 575
real-world
– desert environment 1850
– environment 952
– interface 289
– sample 368
reasoning 299–302
– behavior-based 304
– diagnosis 320
– envisioning 320
– prediction 320
– probabilistic 324
– query answering 320
– symbolic 319
receding mating feature 517
receiver
– autonomous integrity monitor 704
– operating curve 1005
receptive field 789
reciprocity 203
recognition
– by-component 1916
– of complex movement 1921
recognizing action 1771
reconfigurable 1271
– modular manipulator system 513
– robotic system workcell 513
reconfigure 1259
reconnaissance, surveillance, and target acquisition 1485
reconstructed motion data 1667
reconstruction of human motion 1667
rectangle algebra 332
recurrent neural network 1673
– with parametric bias 1903
recursive
– form of Bayes' rule 820
– Newton–Euler algorithm 39, 51
red–green–blue–depth 93, 1691
reduced instruction set computer 577
reduced-order model 941
reduction ratio 242
redundancy 409, 957, 1254
– kinematic 217
– of mobile manipulator 958
– resolution 958
– resolution via optimization 227
– resolution via task augmentation 228
redundant
– manipulator 395, 857
– robot 395
reference
– frame 12
– model 281
– resolution 1752
reference trajectory
– generation 179
– planning 179
referential
– communication 1750
– focus 1771
reflection force 1420
reflex reaction 1701
reflex-based approach 1848
region
– connection calculus 331
– of inevitable collision 143
– proposal 789
regional
– architecture 391
– structure 389
registered reflectance image 738
regolith 1338, 1351
regressor 173
– matrix 161
regular numbering 48
regulation 161
– control 247
regulatory cue 1775
rehabilitation 1006
– robotic 1554
– therapy and training robot 1557
reinforcement learning 306, 346, 367, 1700, 1821
– behavior-based 306

– method 361
– variants 1831
reinforcement manufacturing 1398
relational
– artifact 1779, 1781
– language 321
– Markov decision processes 994
– model 994
relative
– bundle adjustment 1093
– degree 253
– orientation 939
remote
– actuation 451
– agent 281
– center of compliance 192, 195, 860
– center of motion 1531, 1537
– control 1023, 1444
– handling 1412, 1420
– operations centre 1440
– presence virtual 1605
– procedure call 283
– sensing 624, 1248
remotely
– operated underwater vehicle 1452
– operated vehicle 565, 1204, 1412, 1466, 1935
remotely operated robot 998
rendering 788, 1011
– haptic 1004, 1017
rendezvous/docking 1334
repeatability 232
repetitive motion 184
representation
– behavior-based 302
– singularity 220
rescue 624
– adaptive shoring task 1462
– extrication and evacuation of casualties task 1462
– in-situ medical assessment and intervention task 1462
– logistics support task 1462
– reconnaissance and mapping task 1461
– rubble removal 1462
– search 1461
– structural inspection 1462
– surrogate 1462
rescue robot 1470, 1471, 1619
– evaluation 1478
– modalities 1462
– physical test bed 1477
– size 1462
– task 1461
– taxonomy 1462
– types of 1462
resilience 481
resistance temperature devices 676
resolve ambiguous referent 1772
resolved momentum control 1655
resource management 580
responsible conduct of research 1971
retraction method 141
retro-reflective marker 1668
return element 452
revolute joint 21, 76, 993
revolving rotor blade 601
Rex 280
rhythmic entrainment 1746
ridge regression 966
Riemannian
– curvature 395
– manifold 392
rig control system 1440
rigid
– body displacement 468
– body dynamic 607
– body model 47
– environment 201
– hull inflatable boat 1485
– link model 471
– link transmission 451
– motion 514
– object 791, 992
– terrain 556
– wheel 556
ring laser gyroscope 572
ring sonar 727
RiverNet 1261
road
– block 1557
– maintenance robot 1402
– scene understanding 1508
– side unit 1508
– sign detection 1510
RoboCup 1934
– competition 1622
– humanoid league 1625
– rescue league 1460
– rescue simulation project 1476
– simulation league 1622
Robodoc surgical robot 1531
Robonaut 460, 1355, 1772
robot 553, 1952
– application 1305
– architecture 277, 388
– assistant 1680
– behavior 1742
– behavior toolkit 1750
– classification 593
– cockroach inspired 525
– communication 278
– companion 1779
– configuration variable 550
– construction system for computer integrated construction 1400
– control 300, 1472
– controller 1313, 1938
– design 1743
– design process 388
– experiment 1334
– failure 1476
– four wheel 555
– hand 447
– insect-inspired 1849, 1850
– interaction 1831
– interface design 1567
– interior finishing 1400
– language 1317
– learning 1667
– leg–arm hybrid 419
– life-like 525
– locomotion 548
– microsurgery 1538
– mobility 551
– operating system 293, 580, 1055
– oriented design 1395
– platform 1936
– posture 388
– soccer 1622
– socially assistive 1800
– software 579
– standards 1477
– surgical 1532
– team coordination 1475
– technology 580
– test bed 1477
– topology 388
– tournament 1931
– workcell 513, 1302
robot learning
– approach 346
– main branch 346
robot-animal team 1475
robot-assisted play 1784
robot-ground contact 1195
robotic 1435
– agent 1801
– arm large and flexible 268

– assembly 1395
– baseball player 1603
– coach 1603
– competition 1618
– control 481
– crane 1403
– deconstruction 1404
– digging 1438
– dozing 1439
– excavation 1438, 1446
– exploration 1449
– explosive charging 1448
– explosive charging system 1448
– fin 531
– finger 450
– hand 955
– haulage 1437
– lawn mower 1602
– loading 1446
– mapping 1449
– movement 1566
– pool cleaner 1601
– recycling 1404
– roof field factory 1404
– running coach 1603
– sports companion 1603
– therapy 1562, 1563, 1565
– vacuum cleaner 1592
– vehicle 1435
– window cleaner 1597
robotic workstation 1333
robotics 1898
– and biology 1843
– behavior-based 279
– component verification on ISS 1027
– components verification on the ISS 1335
– construction 1395
robust
– estimation method 100
– feature 967
– grasping 956
– perception 972
robustness to failures 1260
rockbreaker 1440
rocker-bogie
– chassis 1338
– configuration 1188
Rodrigues angle 984
Rodrigues' equation 18
roll 1205
– control 616
– mode 1238
rolling
– condition 550
– contact joint 24
– locomotion 1850
roof field factory 1404
room positioning system 1598
root
– joint 1670
– mean square 127, 1610
rose harvesting 1376
Rossum's Universal Robots (R.U.R.) 1646
rotary vector 83
rotary-wing 594, 612
rotation
– center of 893
– matrix 13
– simple 16
rotational stiffness 195, 196
rotor
– and propeller performance 600
rotor force 613
rotorcraft
– dynamics 612
– modeling 611
rough terrain 1187, 1189
– navigation 758
– outdoor 757
round-trip 1335
route network definition file 1631
routing 1046
rover 1332
run-of-mine 1440

S

sacrificial anode 568
safe
– brachistochrone 501
– motion unit 1697
safety 1443, 1531–1533, 1577
– curve 1698
– evaluator 1700
– mechanism 1599
saliency map 1892
sample 986
– based filter 98
– measurement model 984
sampling 1013
– based motion planning 136
satellite servicing 1332
satellite-based augmentation 703
satisfiabiliy modulo theory 334
saturation 1009
SAUVIM project 1204
s-bot 1878
scalability 1255, 1283
scale effect 593
scaled vehicle 624
scale-invariant feature transform 750, 781, 1902
scaling
– analysis 618
– law 594
– series 987
scan matching 1119
scanning
– electron microscope 636, 679
– electron microscope (SEM) 637
– near-field optical microscopy 652
– probe microscope 651
– tunneling microscopy 633, 636, 637
SCARA standard task 391
scattering theory 1033
scene
– depth image 1607
– understanding 791
scheduler 288
Schönflies
– motion 429
– subgroup 391
science fiction 1962
scientific visualization 1006
screw transformation 17
Seaperch 1937
search and rescue 1249
– robot 1197
seawater battery 569
section space 234, 470
security identification 1495
segmentation method 1672
segmented backbone 467
selection matrix 203
selective
– availability 701
– compliance assembly robot arm 240
– compliance assembly robot arm (SCARA) 218, 387, 1303, 1306
– laser sintering 539
– search 789
self
– guided connector system 1406
self-as-simulator 1771
self-configurable system 515

self-excited tripodal dynamic robot 418
self-organization 1844
self-posture changeability 995
self-reconfigurable modular robot 515
self-reconfigurable robotics 1851
semantic
– map 1748
– natural language 1748
– part 785
semiactive suspension dynamics 561
semiautonomous navigation 1339
semi–fuel cable 571
semiglobal
– asymptotic stability 166
– matching 1358
– uniform ultimate boundedness 166
sense-plan-act 279
sensing 89, 1407
– active 865
– and motion 982
– and obstacle avoidance 1607
– graph 1263
– vibration 1014
sensor 302, 454, 674, 1008, 1473
– classification 92
– contact 676
– depth 1212
– depth resolution 739
– force 1009
– fusion 1506
– fusion effect 832
– LIDAR 1198
– low-cost 1594
– model 820
– network 1258
– position 454
– system 1211
– tactile 92
– velocity 454
sensor-based
– control 1181
– method 1189
sensorimotor control 982
sensorized environment for life 1612
sensory
– motor coordination 1849
– motor loop 1844
– substitution 1015
separation sound source 311
sequential
– manipulation 968
– Monte Carlo 827
– quadratic programming 1346
serial
– chain 12
– kinematic machines 1305
– link 531
– robot 72
serial digital interface 573
series
– elastic actuator 240
series elastic actuator 83, 416, 486, 960, 1565, 1680, 1720
serpentine
– motion 527
– robot 1470
Serret–Frenet frame 468
service robotics 1403
servo dynamic 605
seven-revolute 395
shading
– force 1011
shape
– deposition manufacturing 536, 954
– memory alloy 81, 484, 526, 646, 1016, 1271, 1539
– memory polymer 484
– model 784
– primitive 992
– reconstruction 992
shared
– attention 1770, 1786
– autonomy 1334
– control 1028
– goal 1749
– representation 1749
short baseline 575, 1216
short history 1554
short-period mode 1239
shuttle remote manipulator system 1332
side
– force 1192
– slip 1188
signal
– intelligence 1487
– phase and timing 1508
signal-to-noise ratio 741, 1497
Signorini graph 1130
similarity transformation 399
simple
– ranging module ring 727
– temporal problem 330
simplified aerodynamics 613
simulated annealing 515
simulation
– and active interfaces 1675
– and reality 1861, 1869
– and reality agent–environment relation 1862
– and reality minimal 1860, 1862
– and reality physics-based 1860
– based optimization 357
– experiment 989
– theory 1752
simultaneous localization and mapping (SLAM) 313, 622, 727, 753, 768, 1070, 1188, 1212, 1246, 1379, 1443, 1472, 1507, 1597, 1608
single
– acting actuator 452
– chip flash LIDAR 742
– electron transistor 654, 655
– input single-output 163, 610, 1216, 1237
– operator multiple robot 1044
– operator single robot 1044
– port laparoscopy 1531
single task robot 1393
single-master multiple-slave 1037
single-phalanx hand exoskeleton 1717
single-robot task 1276
singular
– configuration 220, 1008
– perturbation control 258
– perturbation model 244
– region 223
– value 220
– value decomposition 220, 359, 748, 766, 1345
singularity 428, 432
– avoidance 223
– kinematic 220
– representation 220
– robustness 222
sinusoidal locomotion 528
situated
– human–robot dialogue 1750
– interaction 1750
– learning 1750
– robotics 300, 313
situation awareness 1474

six-dimensional (6-D) 8, 20, 37, 212, 391, 607, 750, 772, 861, 936, 984, 1029, 1307, 1670
– robot 438
six-revolute 391
skeleton model 1669
skin
– drive mechanism 465
– friction 603, 1208
– stretch 1015
sky-hook controller 561
SLAM 1449
– 3-D 1449
sliding window 786
slip 1015
– angle 1352
– compensation 1193
– display 1015
– ratio 1191, 1352
slippage 1188
slipstream control 600
small
– and lightweight mechanism 1598
– and medium enterprises 1680
– unmanned ground vehicle 1158
smallest singular value 221
small-scale jumping 532
small-size league 1623
smart
– composite microstructure 537
– home 1611
– nursing 1576
– orthosis 1573
– prosthesis 1573
– soft composite 532
– tool 1471
smoothing and mapping 1092
smoothness requirement 246
snake
– arm robot 467
– like robot 527
– robot 463, 1470
soccer 1282
sociable robot 1766
social
– behavior 1744
– cognition 1744
– cognitive skill 1766
– emotional intelligence 1768
– entropy metric 1275
– influence 1801
– interaction 1782
– judgment 1773
– referencing 1771
– robot 1742, 1766, 1782
– stimulus 1773
– support 1774
– touch 1773
sociality 1742
socially assistive robotics (SAR) 1553, 1643, 1799
– Alzheimer's dementia 1812
– autism spectrum disorder therapy 1806
– cognitive rehabilitation 1812
– eldercare 1811
– ethics 1813
– rehabilitation 1808
socially intelligent 1766
socio-affective touch 1783
socio-cognitive skill 1770
socio-cultural cue 1743
soft
– actuator 483, 500
– contact 890
– finger 905
– finger (SF) 885
– gripper 536
– lithography 538
– robotics 481
soft robot 539
– design 482
– optimal control 500
software
– architecture 579
– component 282
– development kit 1048
– for distributed robotics 1265
– product line 293
– variability 293
soil
– contact model 1188
– parameter identification 1192
– penetrometer 1337
solid material assembly system 1395
solid oxide fuel cell 569
sonar 92, 1212
– beam pattern 712
– double refresh rate 728
– dual frequency imaging 574
– dual frequency technology 574
– imaging 574
– interferometric 574
– MLE TOF 721
– multibeam 574
– picton model 718
– principles 710
– profiling 574
– reflector model 716
– scanning 574
– side scan 574
– single-beam directional 574
sound source
– localization 311
– separation 311
– tracking 311
space 1006
– application 1027
– based space surveillance 1485
– joint 1010
– joint control 161
– operation 1025
– qualification 1335
– robotics 1453
– station remote manipulator system 1333
– sweeping 995
space based space surveillance 1485
Space Shuttle 1332
spallation neutron source 1422
sparse
– coding 782
– surface adjustment 1092
spatial
– cognition 1748
– discourse 1748
– dynamic voting 1052
– language 1748
– operator algebra 62
– pooling 783
– pyramid matching 782
– reasoning agent 1754
– referencing 1751
– relation 1748
– remote center compliance 861
– vector notation 39, 44
special purpose dexterous manipulator 1333
specific resistance 421
specification language for ICE 283
speech 1903
– recognition 1426
speed of sound 714
spherical
– joint 23
– shaped pressure vessel 567
– wheel 549
– wrist 389

spinal cord
– and muscle 1895
– injury 1554
spine-based gripper 534
spiral-dive mode 1238
spline 993
spoken dialog system 1750
sports robotics 1603
spray-painting robot 1312
spring 195
– function 493
– loaded inverted pendulum 415, 1847
– virtual 1011
spring stiffness 488
– function 494
stability 1010
– analysis 1189
– augmentation system 610
– criterion 1197
– evaluation 1199
– input-output 1135
– margin 419, 1135
– metastability 1134
– robust 1134
– stochastic 1134
– vehicle 1423
stabilization
– of fixed points 1159
– of trajectories 1159
– practical 1172
– trajectory 1159
stable contact 1012
staged evolution 1859
standard
– development kit 1007
– deviation 719
– end effector 1332
– platform 1622
– POMDP model 1756
– position system 700
Stanford arm 1303
star-model 787
state
– action-reward-state-action 364
– contact identification 864
– display cue 1776
– estimation 621
– observer 258
– of the art 954, 1576
– space model 263
static
– feedback linearization 253
– gait 409
– margin 610
– sinkage 1352
– stability 607
statics 435
statistical pattern recognition 997
steady hand robot 1538
steerability
– degree of 552
steering/driving maneuver 1193–1195
step–up/down transformer 571
stereo 743
– image geometry 743
– vision 742, 1341, 1608
stereolithography 539
Stewart platform 428
stiffness 435, 1011
– active 196, 197
– adaptation 497
– adjuster 491
– center of 195
– contact 895
– control 193, 495
– estimation 492, 995
– matrix 195, 396
– modal 264
– observer 493
– translational 195
stigmergy 1257, 1258
stochastic gradient descent 1092
stop-and-go 1515
strap down IMU 1246
strawberry harvesting 1376
straw-man task 399
stream-oriented messaging architecture 579
strictly proper 175
strip theory 1208
structural
– analysis 567
– connection system 1405
– damping 260
– inspection 1469
– synthesis 436
structure from motion (SFM) 746, 753, 1093
structured output prediction 788
structured-light distance sensor 1607
subordinate category 779
subsea system 571
substitute circle concept 558
subsumption 279
– architecture 832
suction cup 530
summative evaluation 1943
superadditive 1267
superior
– colliculus 1890
– temporal sulcus 1899, 1921
super-ordinate category 779
superpixel 789
superquadric 992
supervisory control 1024, 1428
support
– pattern 419
– polygon 419, 1132, 1197
– vector machine 782, 995
– vector regression 356
supportive interaction 1681
surface
– display 1016
– friction 995
– normal 790, 984
– power system 571
– texture 995
surgeon extender robot 1537
surgical
– assistance 1531
– assistance system 1529
– CAD/CAM 1529
– registration 1534
– robot 1532, 1540
– robotics 1006
– simulation 1012
– support robot 1531
surveillance 624, 1249
surveying 1443
surveyor lunar rover vehicle 1339
swarm 518, 1255, 1475
SWARM-BOT 1269, 1878
Swedish wheel 549
swimming mechanism 1851
symbol grounding 321
– problem 1748
symbolic
– planning 285
– task planning 992
symmetric formulation 936
synaptic plasticity 1848
synchronous-drive robot 554
synergy synthesis 1137
synthetic aperture
– radar 833, 1485
– sonar 574
system 1891
– behavior-based 299
– chain 516

– decomposition 278
– hybrid 301
– integration 1320
– lattice 516
– modeling 91
– polybot 516
systematic coverage 1593

T

tactical planning 1119
tactile
– array 996
– display 1014
– human–robot interaction 1783
– illusion 1017
– image 993
– interaction 1782
– interaction with social robots 1785
– localization 983
– object localization 982
– object recognition 997
– pattern display 1016
– sensing 673, 1004
– sensor 92, 455, 673
– sensor information flow 680
tactors 1015
tail volume coefficient 610
tangential proper part 331
tape measure 1407
target tracking 811
task 791
– allocation 1276
– compatibility 230
– consistent operational space dynamics 853
– constraint 850
– coordinator 582
– decomposition 286
– description 1316
– description language 282
– driven robot 515
– evaluator 1700
– execution 626
– frame 193
– Jacobian 219
– level programming 1318
– level simulation 1753
– model 1743
– oriented kinematics 218
– priority 230
– related dialogue 1750
– space 348
– space control 162
– variable scaling 127
task space
– cooperative 939
– retrieval using inverse optimal control 968
taxon 1891
– affordance model 1892
taxonomy of rehabilitation 1552
teaching force 1828
teamwork with robot partner 1766
technical
– committee 289, 1953
– problem 1405
– specification 1686
technology 1486, 1575, 1606
tectum 1890
tegmentum 1890
tele-care 1605
telemanipulation 1024
teleoperated
– robotic excavation 1402
– small emplacement excavator 1421
– system 1394
teleoperation 1004, 1024, 1043, 1394, 1413–1421, 1424–1426, 1471
– display 1426
– tactile display for 1015
teleoperator 1010
teleo-reactive executive 579
teleorobot 1043
telepresence 1335, 1394, 1411
– robot 1604
telerobotic servicer 1332
telerobotics 1036, 1445, 1472
telesensor programming 1027
telesurgery 1026, 1537
telesurveillance 1605
television 1661
template 786
– matching 721, 1307
– model 413
temporal
– action logic 324
– constraint satisfaction 329
– constraint satisfaction problem 329
– logic 338
tendon tension sensor 454
tendon-based
– design 1687
– robot 1687
– transmission 456
tension sensor 454
tension-differential type 455
tensor arm 466
terrain
– aided navigation 1216
– classification 756
– hazard 1341
– mapping 1189
terramechanics 556, 1188, 1351
– based approach 1190
– model 1191
test action pair 282
tether
– cable 571
– management system 565
– mechanism 1598
tethered walking robots 419
therapy approach 1566
thermal
– conductivity 996
– display 1017
– neutron detector 1415
– sensor 676
thermoelectric cooler 1017
three tiers 280
three-dimensional (3-D) 4, 20, 38, 113, 136, 391, 437, 456, 530, 573, 596, 641, 696, 716, 737, 763, 784, 833, 860, 895, 917, 959, 984, 1011, 1051, 1081, 1093, 1111, 1127, 1189, 1242, 1305, 1339, 1392, 1417, 1438, 1482, 1487, 1510, 1608, 1649, 1668, 1691, 1768, 1802, 1860, 1914, 1968
– normal distributions transform 1449
– representation 788
thruster 1213
– Kort nozzle 572
– Rice nozzle 572
tile placement robot 1400
time
– delay 1025
– domain passivity control 1335
– of arrival 575
– update 821
time-base generator 1729
time-extended assignment 1276
time-of-flight 710, 739, 740, 1375
– direct 740
– distance sensor 1607
– laser-based direct 740

– range sensor 740
– ranging 718
– sensor 1375
time-to-collision 1444
tire model 559
tolerance 1218, 1406
tool
– center point 494
– virtual 1010
tool-ground interaction 1438
topcoat 568
topic model 781
topological
– contact state 860
– localization and mapping 305
– map 106, 1445
topology 1262
torque
– computed 255
– control 170, 416
– controlled humanoid robot 1687
– controlled robot 1336
– controller 496
– joint feedback 252
– joint sensor 193
– reversal mechanism 533
torsion 469
torsional stiffness 397
total
– energy control system 611, 1237
– factor productivity 1366
touch 1004
– based interaction 1766
– sensor 1783
track drive 1601
tracked mechanism 1424
tracked vehicle 1187
tracking
– algorithm 1489
– sound source 311
– trajectory 161, 253
tractability 369
traffic control 1281
traffic-light detection 1510
trailer system 1167, 1181
training data 790
trajectory 179
– exciting 127
– generation 179
– modification 962
– planning 144, 179
– representation 180
– smoothness 246
– stabilization 1159
– tracking 161, 253
transcranial
– direct current stimulation 1563
– magnetic stimulation 1563, 1903
transducer technologies 715
transfer
– control protocol 292
– function 248
– learning 784
– matrix method 261
– mode 856
transformational planning 338
transit 1407
– mode 856
transition model 988
translate-turn module 514
translation
– model 1673
– simple 16
– stiffness 195
transmission 462, 1009
– control protocol 1046, 1258
– electron microscope 636
– quality 393
transparency 1032
transportation 624
transverse function 1172
traversability 1188
treadmill training 1563
tree
– fruit production 1373
– kinematic 51
– structure 122
triangulated surface 751
triangulation 739, 766
tricept robot 439
trigonometric trajectory 181
trim condition 605
tripod gait 526
truck spotting 1440
truss grasper 534
Tustin algorithm 183
twist 881
– end-effector 392
two-and-a-half-dimensional (2.5-D) 599, 671, 737, 795, 1066, 1609
two-dimensional (2-D) 136, 332, 537, 584, 596, 634, 678, 713, 738, 763, 783, 806, 833, 860, 967, 984, 1015, 1066, 1143, 1272, 1305, 1392, 1442, 1495, 1532, 1609, 1622, 1743, 1829, 1861, 1916
two-layer approach 1034
two-legged robot 530
two-point
– boundary value problem 231
– discrimination 1015
two-port network 1013
two-time scale (dynamics) 244
two-wheel differential-drive robot 554
type synthesis 429
types of rescue robot 1462

U

UAV challenge 1619
ultrahigh definition (UHD) 577
ultrahigh frequency 1487
ultrahigh-vacuum (UHV) 652
ultrashort baseline 575
ultrashort-baseline 1216
ultrasonic 1016
ultrasound imaging 1537
ultraviolet 638
ultrawide band 622
umbilical cable 571
uncanny valley 1743, 1767
uncertain environment 1843
uncertainty 324, 437
unconditional stability 1032
underactuatedness 409
underconstrained scenario 982
underground mine mapping 1449
under-modeling 369
underwater 1411
– computer vision system 573
– image 573
– intervention 584
– sensor 571
underwater actuator
– propeller design 572
– thruster 571
underwater manipulator 583
underwater robot 584
– design 565, 566
– fairing 566
– frame 566
– middleware 579
underwater vehicle 1214
– manipulator system 1217
undulatory motion 528
unexploded ordnance 1416–1418
unified modeling language 278
uniform
– completely observable 1236

– resource locator 1050
– ultimate boundedness 166
uniformly completely observable 1236
unit quaternion 196, 940
universal
– coordinated time 703
– joint 24
– plug and play 580
– serial bus 1939
unmanned
– aerial system 1226
– aerial vehicle 403, 592, 1037, 1056, 1226, 1261, 1300, 1382, 1412, 1462, 1966
– aircraft system 591
– construction machine 1482
– ground vehicle 1188, 1265, 1462, 1486, 1967
– marine vehicle 1462
– surface vehicle 1462, 1481, 1498
– underwater vehicle 565, 1211, 1462, 1485
unscented Kalman filter 1445
unstructured environment 952
unwinding algorithm 740
upper
– extremity 1559
– limb exoskeleton 1715
urban
– challenge event 1633
– search and rescue 1460, 1477
– terrain 756
use case 290
user data protocol 1046
user datagram protocol 292, 579, 1258
user-centered design 1943
users' mental model 1757
utility function 989

V

vacuum cup 1599
value
– function approach 363
– iteration 326
– iteraton (VI) 988
van der Waals 652
vapor detector explosive 1416
variable
– damping 487
– impedance actuator 480, 1680
– reluctance 79
– structure 170
variable stiffness 501, 518
– actuator 240, 480, 960, 1688
– joint 487
variable-geometry truss 472
vector
– modal 398
– quantization 782
– spatial 39
vector field
– histogram 1111
– SLAM 1608
vehicle
– automated 1518
– detection 1510
– formation control 1374
– stability 1423
vehicle-to-vehicle 1444
velocity
– analysis 431
– measurement 1009
– observer 1247
– obstacle 1112
– spatial 40
velocity-resolved control 201, 211
ventral intraparietal 1906
verbal communication 1775
vertebrate motor control 1893
vertical
– cell decomposition 141
– motion 573
– take-off and landing 625
viability 1133
vibration 1005
– contact 1014
– display 1006
– feedback 1014
– sensing 1014
– suppression control 1348
– waveform 1014
vibratory type gyro 572
vibrotactile
– feedback 1014
video
– cassette recorder 1306
– game 1006
virtual
– agent 1801
– constraint 1148
– contact manipulator 206
– coupling 1011
– embodiment 1773
– environment 1004, 1717
– fence 1602
– fixture 1011, 1529
– human 1675
– linkage model 945
– manipulator 1345
– object 711
– reality modeling language 1056
– spring 1011, 1012
– stick 936
– tool 1010
– wall 1011
virtual stick 936
viscous
– damping 1209
– effect 597
– friction 462
visibility graph 141
vision 1212, 1889
– active 1867
vision system 311
visual
– analog scale 1685
– attention 1892
– feature 796
– object class 788
– object recognition 997
– odometry 1188, 1354
– perspective 1772
– perspective taking 1786
– servo 795
– servoing 1448
– SLAM 1609
– word 780
visual servo control 795
– 2.5-D 806
– eye-in-hand systems 812
– feature trajectory planning 809
– hybrid approach 806
– image-based 797
– joint space control 812
– optimization 807
– partitioned approach 807
– pose-based 803
– switching approaches 808
visually guided behavior frog 1889
V-maneuver 1345
voice-coil motors 1014
volume analysis 597
volumetric grid map 993
vortex
– shedding damping 1209
– system 599
voting-based estimation 101

voxel 752, 1012
– grid map 993
VoxMap pointshell 1006

W

walking 1563
wall
– building 1280
– climbing robot 419, 530
– virtual 1011
Waseda bipedal humanoid 1655
Waseda robot 1648
watchmen tour 1266
water
– pressure-resistant design 567
– speed sensor 574
waterproof sealing design 567
wave
– drift damping 1208
– variable 1033
waveform 714
– vibration 1014
waypoint navigation 1338
wearable 1015
– monitoring device 1577
– therapy robot 1553
weather proof site 1404
web 1049
– ontology language 323
– server 1049
wedging 888
weed control 1370
weight sharing 789
weighted
– difference vegetation index 1382
– least squares 127
welding gun 1308
wheel
– offset 548
– sinkage 1352
– slippage 1191, 1338
– test bed 1191
– vehicle slippage 1193
wheel traction 1351
– characteristics 1191
– performance 1193
wheeled mechanism 1424
wheeled mobile robot 547, 1036, 1157, 1187
– modeling 1189
– structure 553
– suspension 559
wheel–terrain interaction 1189
– model 556
whisker array 1850
whole-arm
– grasping 859
– manipulation 467
– manipulator 481
whole-body sensing 994
wide-area
– augmentation system 702, 1442
– network 1046
Widrow–Hoff learning rule 1896
wing
– aerodynamics 599
– modeling design 617
winner-take-all 1890
wire-driven parallel robot 428
wireless
– access in vehicular environments 1508
– ad-hoc system for positioning 1443
– markup language 1056
– sensor network 1372
Wizard of Oz 1756
work envelope 68
workpiece localization 982
workspace 27, 68, 434, 957, 1008, 1302
– dexterous 72, 391
– mapping 1029
– reachable 72, 391
– shape design rule 390
– topology 391
– volume 391
world
– demographic 1553
– graph 1892
– wide web 1046
World Geodetic System 703
worm-like
– crawling 526
– robot 473, 1850
worst-case range 1685
wrench 33, 881
– cone 884
– transmission matrix 118
Wright brothers 1225
wrist center 389
– force-torque sensor 995

X

x-ray fluoroscopy 1537

Y

yaw 1205
yet another robot platform 292
Young's modulus 538

Z

zero
– moment point 355, 407, 1129, 1135, 1650, 1729
– order hold 1035
– pole 164
zero-dimensional (0-D) 634
zona pellucida 648
Z-transform 183